로봇청소기 제품군 Attack Surface 분석 방법

임준태, 석지원, 최원빈, 이재휴, 이경수, 안용호, 고영우, 이인형, 이상섭, 이원

한국정보기술연구원 차세대 보안리더 양성 프로그램(Best of the Best)

Robotic Vacuum Attack Surface Analysis Method

Jun Tae Im, Ji Won Seok, Won Bin Choi, Jae Hyu Lee, Gyeong Su Lee, Yong Ho An, Young Woo Go, In Hyeong Lee, Sang Sup Lee, Won Lee

KITRI 차세대 보안리더 양성 프로그램(Best of the Best)

요 약

최근 로봇청소기와 IoT 제품의 판매량이 증가함에 따라 매핑 기능, Wi-Fi 통신, 홈 뷰 기능 등 다양한 기능을 탑재한 로봇청소기들이 출시되고 가정에 보급되면서 이에 따른 공격벡터가 증가하였다. 로봇청소기 제품군에서 발생할 수 있는 보안 문제에 대한 Attack Surface를 기능 분석을 통해 청소기와 앱 간의 블루투스 통신, 청소기와 앱 간의 Wi-Fi 통신, 악성 앱 업데이트, 펌웨어 추출 및 기능 분석으로 나누어서 이에 따른 로봇청소기 Attack Surface 분석 방법을 제시한다.

I. 서론

최근 로봇청소기와 IoT 제품의 판매량이 증가함에 따라 매핑 기능, Wi-Fi 통신, 홈 뷰 기능 등 다양한 기능을 탑재한 로봇청소기들이 출시되면서 이에 따른 공격벡터가 증가하였다. 그리고 최근 카메라가 장착된 로봇청소기가 출시되기 시작하면서 해킹 공격으로 카메라를 장악할 수 있다면, 사생활 침해의 위험성도 높아진다. 따라서 로봇청소기 제품군의 보안성 향상에 대한 필요성이 증가하였다.

본 논문에서는 로봇청소기 제품군에서 발생할 수 있는 보안 문제에 대한 Attack Surface를 기능 분석을 통해 보안 문제가 발생할 수 있는 청소기와 앱 간의 블루투스 통신, Wi-Fi 통신, 악성 앱 업데이트, 펌웨어 추출 및 기능 분석으로 나누어서 이에 따른 로봇청소기 Attack Surface 분석 방법을 제시한다.

로봇청소기는 카메라 모듈을 탑재하고 있고 시장 점유율과 소비자 인지도가 높은 회사 5곳의 제품들을 분석 대상으로 선정하고 진행하였다.

II. 청소기와 앱 간의 블루투스 통신

[그림 1]은 블루투스 기능이 있는 로봇청소기의 통신 방식에 대한 구조도이다.

[그림 1] 로봇청소기 블루투스 통신 방식

블루투스 기능을 탑재한 로봇청소기 같은 경우, 블루투스 기반의 컨트롤 앱과 로봇청소기 간의 통신이 이루어지는 것을 파악을 하였고 앱 동작이 어떻게 이루어지는지 파악하기 위해서 앱 분석을 진행하였다.

그리고 [그림 2]와 같이 Frida를 이용하여 블루투스 통신 부분을 후킹해 블루투스 데이터를 확인할 수 있다.

```
[*] Connect... OK
[+] Execute script
[*] Bluetooth Hooked!
[*] [-86, 3, 1, 0, 10, -20]
[*] [-86, 3, 1, 0, 10, -20]
[*] [-86, 3, 1, 0, 11, -20]
[*] [-86, 3, 1, 0, 11, -20]
```

[그림 2] Frida 후킹한 블루투스 데이터

그리고 안드로이드의 블루투스 HCI 스누프 로그를 이용하여 앱과 청소기 간의 블루투스 패킷을 스니핑하고 Wireshark를 이용하여 앱 업데이트 및 버전 체크, 특정 URL 접속, 청소기 제어 블루투스 데이터 등의 패킷을 확인할 수 있다.

로봇청소기의 블루투스 기능을 분석한 결과 블루투스 명령 패킷은 [그림 3]처럼 0xAA로 시작하여 0xEC로 끝나는 것을 알 수 있다. 그리고 올바르지 않은 CMD 코드를 전송할 경우 조작이 불가능해지는 취약점이 발생하는 것을 확인할 수 있다.

구분	Start	CMD				End
전원	0xAA	0x04	0x01	0x00	0x07	0xEC
전원	0xAA	0x03	0x07		0x01	0xEC

[그림 3] 블루투스 명령 패킷

Ⅲ. 청소기와 앱 간의 Wi-Fi 통신

[그림 4]는 Wi-Fi 기능이 있는 로봇청소기의 통신 방식을 나타낸 구조도이다.

[그림 4] 로봇청소기 Wi-Fi 통신 방식

Wi-Fi 통신을 하는 로봇청소기 같은 경우 초기 등록 부분에서는 앱과 직접적인 통신이 이루어지고 그 이후부터는 클라우드 서버를 거쳐서 통신 한다는 것을 파악하였다. 그리고 서버와의 통신을 확인해본 결과, 평문이 아닌 SSL 통신을 하는 것을 확인했다.

앱 분석을 시작했을 때, 앱 상에서의 난독화나 루팅 탐지 등이 적용되어있는지 확인을 해본 결과, 분석을 진행한 로봇청소기 앱에서는 로컬에서 루팅을 탐지하는 루틴이 발견되었다. 이를 우회하기 위해 앱을 디패키징 하여 Smali 코드를 수정하고, 실시간으로 Frida를 이용한 동적 디버깅을 통해 루팅 탐지 로그를 확인할 수 있다.

JNI로 so파일 열어서 [그림 5]처럼 바이너리 패치를 하여 루팅 탐지 루틴을 우회를 할 수 있었다.

```
if ( v6[10] && !IsDebugMode(v8, v5) )
{
  v9 = 4;
  goto LABEL_7;
}
_android_log_print(3, "SecurityCheckerNat
_android_log_print(3, "SecurityCheckerNat
if ( v6[9] && !IsTampered(v6, v8, v5) )
```

[그림 5] 바이너리 패치를 수행한 네이티브 코드 일부

디패키징 한 앱 내부에서 피닝이 되지 않은 인증서와 Keystore파일들이 있었고 정적으로 인증서 비밀번호가 노출되어 있었다.

그리고 MITM Proxy를 사용하여 TLS MITM을 수행해 서버와 로봇청기와 서버간의 통신 URL과 파라미터를 중간자 공격을 통해 알아낼 수 있었다.

```
POST https://api.[redacted].com/v
     ← 200 application/json 5
POST https://kic-service.[redacted]
     =post&re··· HTTP/2.0
     ← 200 text/plain 601b 45
GET https://api.[redacted].com/v2
    ← 200 application/json 1.
POST https://api.[redacted].com/v
```

[그림 6] MITM Proxy를 사용해 TLS MITM을 수행하는 사진

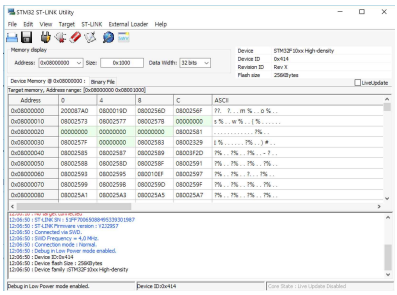
발견된 URL을 공격자의 서버로 변조 할 수 있다면, 공격자의 악성 펌웨어 업데이트 서버로 연결시키거나, 영상 데이터를 공격자의 서버로 유도하는 등 애플리케이션을 외부와 통신하는 부분을 통해 공격을 수행할 수도 있다.

그리고 앱에서 유출된 인증서 비밀번호와 인증서가 피닝되지 않은 부분을 이용하여, 변조한 인증서로 중간자 공격을 시도하여 계정을 탈취

하거나 영상을 유출하는 공격도 가능하다.

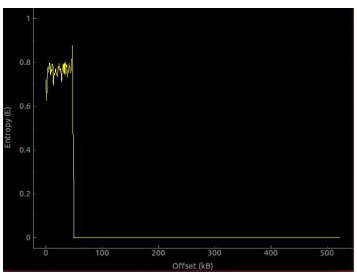
Ⅳ. 펌웨어 추출 및 기능 분석

펌웨어를 추출을 위해 우선 칩 전수조사를 해서 메인 MCU와 UART, JTAG 등의 디버깅 포트를 식별한다. 디버깅 포트 식별이 어려울 경우 메인 MCU의 데이터시트, 특히 핀 맵을 참고해 UART, JTAG 등의 디버깅 포트를 오실로스코프나 멀티미터기 등을 이용하여 식별할 수 있다. 식별한 디버깅 포트에 알맞은 디버깅 도구, 진행한 로봇청소기 같은 경우 STM32F103 MCU를 사용하므로 해당하는 JTAG 디버거인 ST-LINK/V2를 이용해 [그림 7]과 같이 펌웨어를 추출하였다.



[그림 7] ST-LINK를 이용한 펌웨어 추출

추출한 펌웨어 바이너리 파일에 binwalk를 사용하여 아키텍처, 파일 시스템, 엔트로피 분석([그림 8] 참고) 등 다양한 방면으로 펌웨어 분석을 할 수 있다. 특히, 펌웨어가 암호화가 되어 있을 경우 binwalk -E 옵션과 binvis 등으로 분석한 엔트로피 결과가 비정상적으로 나올 것이고, 이를 토대로 펌웨어 암호화 여부도 판단 할 수 있다.



[그림 8] binwalk를 이용한 엔트로피 분석

binwalk를 통해 얻은 정보를 토대로 FMK를 사용하여 펌웨어내의 파일을 추출할 수 있다. 이후, 추출한 펌웨어를 QEMU를 이용해 가상머신에 추출한 펌웨어를 구동시켜 분석을 진행할 수 있다.

하지만, 필자의 경우처럼 파일 시스템이 없고 펌웨어가 단일 파일로 동작하는 경우도 있다. 이런 경우는 binwalk를 이용하기 어려우므로 추출한 바이너리를 바로 IDA를 이용해 분석한다. 이때, IDA 분석 기능을 이용하면 만들어진 함수로 펌웨어를 분석 할 수 있다. 그러나 파일 시스템이 없는 경우에서 보통 바이너리의 심볼이 없을 확률이 높아서 분석을 위한 main을 찾기가 힘들어질 수 있다. 이 경우, IDA의 기능중 하나인 문자열 추출 기능을 사용하면 효율적으로 정적 분석이 가능하다.

ROM:00002F6C	0000000B	C	Booting OK
ROM:00002F78	0000000C	C	App State :
ROM:00002F84	0000000C	C	Advertising
ROM:00002F90	0000001A	C	Already Advertising state
ROM:00002FAC	00000019	C	Already Device Connected
ROM:00002FC8	00000014	C	Advertising Stopped
ROM:00002FDC	0000000E	C	Disconnecting
ROM:00002FEC	0000000D	C	Disconnected
ROM:00002FFC	0000001D	C	Device Name Setting Complete
ROM:0000301C	0000000A	C	Connected
ROM:00003028	0000000F	C	Connect Failed
ROM:00003038	0000000E	C	Not Connected
ROM:00003048	00000008	C	Success
ROM:00003050	00000009	C	LBD_Addr
ROM:00003060	00000009	C	FW Ver.
ROM:00003AC0	00000008	C	IRONMAN
ROM:00005A0C	00000020	C	SIGPVFN: Pure virtual fn called
ROM:00005BC0	0000001D	C	SIGRTMEM: Out of heap memory
ROM:00005BE0	00000018	C	: Heap memory corrupted
ROM:00005C4C	0000001E	C	SIGABRT: Abnormal termination

[그림 9] IDA를 이용하여 추출된 펌웨어 내부 문자열

[그림 9]과 같이 문자열 추출 데이터를 토대로 분석을 진행 한다면, 어떤 동작을 수행하는 함수인지 식별하기 수월해진다. 예를 들어 문자열 데이터 중 “Connected” 같은 경우, 블루투스 연결을 진행하는 함수와 로봇청소기를 제어하는 함수가 존재했다. 또한, “Out of heap memory” 같은 경우는 추적 결과, 힙 메모리 OOB가 발생했을 경우 디버그 모드로 전환되는 함수가 존재했다. 함수를 호출하는 부분이 어디인지 확인하려면 IDA의 xrefs 기능을 통해 역추적이 가능하다.

또한 ST-LINK/V2는 디버거 자체적으로 동적 디버깅을 지원하기 때문에 디버거를 이용한 동적 디버깅을 수행할 수도 있다.

펌웨어 분석을 통해 신뢰되지 않은 입력을

받아 힙 메모리를 오염시킬 수 있는 루틴이 있거나, 민감한 데이터가 정적으로 있거나, 커맨드 인젝션을 통해 비정상적인 동작을 수행하도록 유도하는 등 다양한 취약점이 발생할 수 있다.

Ⅴ. 결론

본 논문은 점차 다양한 기능을 탑재되고 출시되는 로봇청소기 제품군의 Attack Surface 분석 방법에 대해서 로봇청소기와 앱 간의 블루투스 통신, 청소기와 앱간의 Wi-Fi 통신에서 발생할 수 있는 보안 취약점을 분석하는 방법, 앱 위변조를 통해 분석 환경 구축 및 애플리케이션과 클라우드 또는 청소기 간의 패킷을 스니핑할 수 있었다. 또한 펌웨어를 추출해 파일시스템의 유무에 따라 QEMU를 이용해 가상머신에 펌웨어를 올려 분석하거나, IDA를 사용해 문자열 추출을 통해 main함수를 찾거나, 블루투스 통신을 수행하는 함수를 찾는 등의 로우 레벨단에서의 분석을 통해 신뢰되지 않은 입력을 받는 지점을 함수 역추적을 이용해 찾아낼 수 있었다.

위 분석을 토대로 블루투스 부분에서는 올바르지 않은 CMD 코드로 청소기 조작을 불가능하게 만드는 공격이 있었다. Wi-Fi 부분에서는 디패키징 한 앱 내부에 피닝되지 않은 인증서와 Keystore파일들, 그리고 인증서 비밀번호가 정적으로 노출되어있는 문제점을 이용해 중간자 공격을 시도하여 계정 탈취와 영상을 유출시키는 공격이 가능하다는 것을 알 수 있었다. 펌웨어 부분에서는 민감한 데이터를 유출하거나 비정상적인 동작을 수행하도록 하는 커맨드 인젝션 공격 등의 취약점이 발생할 가능성을 확인 할 수 있었다.

언급한 Attack Surface를 통해서 로봇청소기 제품의 보안 측면에서 참고 자료나 제품 보안 패치시 체크리스트를 만드는 등 로봇청소기의 보안성 향상을 기대할 수 있다.

[참고문헌]

[1] Koopman, Philip. "Embedded system security." Computer 37.7 (2004): 95-97.

[2] Parameswaran, Sri, and Tilman Wolf. "Embedded systems security－an overview." Design Automation for Embedded Systems 12.3 (2008): 173-183.

[3] Ravi, Srivaths, et al. "Security in embedded systems: Design challenges." ACM Transactions on Embedded Computing Systems (TECS) 3.3 (2004): 461-491.

[4] Chen, Daming D., et al. "Towards Automated Dynamic Analysis for Linux-based Embedded Firmware." NDSS. 2016.

[5] F. Bellard, “QEMU, a fast and portable dynamic translator,” in Proceedings of the USENIX 2005 Annual Technical Conference. USENIX, 2005, pp. 41 - 46. [Online]. Available: https://www.usenix.org/legacy/publications/library/proceedings/usenix05/tech/freenix/bellard.html

[6] A. Bessey, K. Block, B. Chelf, A. Chou, B. Fulton, S. Hallem, C. Henri-Gros, A. Kamsky, S. McPeak, and D. Engler, “A few billion lines of code later,” Communications of the ACM, vol. 53, no. 2, pp. 66 - 75, 2010.