차세대 IoT환경(ARMv7 Thumb Architecture) 침투테스트에 관한 연구

김시완* · 성기택*

*동명대학교

A Study on the Penetratrion Testing Next-generation IoT environment(ARMv7 Thumb Architecture)

Si-Wan Kim* · Ki-Taek Seong*

*Tongmyong University

E-mail : parca57@naver.com

요 약

IoT System의 특성상 IoT Device(혹은 Sensor)는 우리의 일상생활에 쉽게 노출(또는 장착)되어있고, IoT Device에 대한 직접적인 물리적 접근이 쉬우므로, 이로 인한 물리적 보안이 매우 취약하다. 본 연구에서는 IoT System의 ARMv7 Thumb Architecture 대상으로 직접적인 로컬 시스템 취약성을 공격하여 IoT System 속으로 침투하는 Zero-Day Attack을 구현하였다.

ABSTRACT

Due to the nature IoT Device (or Sensor) of the IoT System it may be susceptible (or attached) to our daily lives, so easy to direct physical access to IoT Device, which caused physical security is very weak. In this study, we implemented “Zero-Day Attack” on the ARMv7 Thumb Architecture as a direct target local system.

키워드

시스템 보안, IoT Systems, ARMv7 Thumb, 침투테스트, Zero-Day Attack, 로컬 시스템 해킹

Ⅰ. 서 론

최근 현대 사회의 인터넷 사용 확산에 따른 각종 정보보안의 사건·사고가 빈번히 발생하고 있다. 이에 따라 정부·공공기관 및 기업들은 ‘정보보안’이라는 사회적 이슈에 발맞춰 조직의 자산을 보호하기 위해 여러 방면으로 노력하고 있다. 이러한 정보보안 사건·사고의 유형에는 공통된 유사점이 있다. 바로 조직 내의 내부자(사람)에 의해 정보보안 사건·사고가 발생한다는 점이다. 즉, 조직 내부에 불순한 의도를 가지고 접근하는 악의적 내부자와 조직 구성원의 실수와 방심에 의한 부주의적 내부자 같은 유형으로 나타나며 전체 정보보안 사건·사고 중 80%이상이 이러한 유형의 내부자에 의해 발생된다는 점이 매우 흥미롭다. 특히, 내부자에 의한 정보보안 사건·사고는 E-Mail Phishing 또는 Spear Phishing과 같은 ‘Social Engineering’ 공격기법으로 인간의 심리적 성향과 신뢰성 관계를 공격하므로, 목표대상이 된 조직의 내부자는 속수무책으로 당하고 있는 것이 현실이다. 대표적인 사건으로 2011년 4월 12일에 발생한 ‘농협 전산망 마비 사태’와 2016년 7월 28일에 발생한 ‘㈜인터파크 기업의 개인정보 유출사건’은 조직 내부자에 대한 직접적 공격(로컬 시스템 공격)의 심각성을 보여주는 단편적인 지표이다. 우리는 여기서 한발 더 앞서 나아가, 최근 차세대 IT 환경으로서 각광받고 자리매김하고 있는 ‘IoT(Internet of Things) System’은 과연 “내부자에 의한 공격과 같은 로컬 시스템 공격의 위협으로부터 자유로울 수 있을까”라는 의문을 제기할 수 있다. 왜냐하면, IoT 시스템의

특성상 IoT 디바이스(혹은 Sensor)는 우리의 일상생활에 쉽게 노출(또는 장착)되어있고, IoT 디바이스에 대한 직접적인 물리적 접근이 쉬우므로, 이로 인한 '물리적 보안' 이 매우 취약하기 때문이다. 이에 따라 본 연구에서는 IoT 시스템에 대한 침투테스트(Penetration Testing)를 통하여, IoT 시스템의 ARMv7 Thumb Architecture 대상으로 직접적인 로컬 시스템 취약성을 공격하여 IoT 시스템 속으로 침투하는 Zero-Day Attack을 시연하였다. IoT 시스템의 자체적 로컬 시스템 또한 직접적으로 공격받을 수 있음을 실제 공격으로 확인하고, 그로 인한 파급효과로 IoT 시스템에 미치는 영향력이 어떠한지를 보여줌으로서 IoT 시스템 보안에 대한 경각심을 불러일으키기 위함이다.

II. 본 론

2.1 IoT 시스템 운영체제 소개

IoT 시스템에서 구동되어지고 사용되는 운영체제들은 ARM Architecture로 포팅 되어 사용된다는 공통점이 있다. IoT 시스템 환경 특성상 소형화, 휴대성이 강조되는데 임베디드 환경에 특화되어 저전력을 사용하도록 설계된 ARM Architecture는 IoT 시스템 운영체제에 최적화되어 있기 때문이다. IoT 시스템의 운영체제는 보편적으로 'Windows 계열' 과 'Linux 계열' 의 두 분류로 구분할 수 있다. 'Windows 계열' 로 우리가 흔히 쉽게 접할 수 있는 Microsoft사의 Windows 10을 기반으로 IoT 시스템 환경에 최적화된 운영체제로 개발된 'Windows 10 IoT Core' 가 있다(그림 1).

그림 1. Windows 10 IoT Core

Windows 10 IoT Core 운영체제는 Microsoft사에서 전 세계의 IoT 시스템 환경에 대한 상용 촉진 목적으로 라이선스 비용을 무료하여 배포하고 있다. 또한, Microsoft 홈페이지의 다양한 포럼을 통해 IoT 시스템 관련 어플리케이션 개발에 대한 지원과 피드백을 받을 수 있다. 다양한 컨텐츠와 Visual Studio 2015 같은 개발자도구를 적극 지원하므로 앞으로의 IoT 시스템 시장에서 IoT 시스템 운영체제로서의 선두주자로 도약할 가능성이 높다. 다음으로 서버환경에서 자주 쓰이는 'Linux 계열' 의 IoT 시스템 운영체제들이다. 기본적으로 Linux 운영체제는 추구하는 이념과 같이 오픈소스로 무료로 이용가능하고, 전 세계적으로 다양하게 개발되고 있는 점이 큰 장점이다. 또한, 일반적인 PC 환경에서 사용되고 있는 Linux 운영체제와 비슷한 인터페이스와 구동방식으로 개발되어 거부감이 접근할 수 있다. 다만, IoT 디바이스의 특성상 플랫폼이 ARM Architecture인 점이 일반적인 PC 환경의 Linux 운영체제와 가장 큰 차이점이라고 볼 수 있다. Cent OS, Fedora, Ubuntu 등 대부분의 Linux 개발팀에서 ARM Architecture로 포팅하여 배포하고 있기 때문에 원하는 환경에서 요구되는 상황에 맞게 IoT 시스템을 이용할 수 있다.

위와 같이 Linux 계열 IoT 운영체제는 ARMv7 Architecture 기반 베이스로 포팅되어 개발되고, Windows 계열인 Windows 10 IoT Core은 ARMv7 Thumb Architecture로 포팅되어 개발되었다. 기술적으로 ARMv7 Architecture에 대한 침투테스트 관련 자료는 간혹 존재하나, ARMv7 Thumb Architecture에 대한 침투테스트 관련 자료는 없다. 본 연구에서는 ARMv7 Thumb Architecture 기반인 Windows 10 IoT Core를 대상으로, 로컬 시스템 취약점을 분석하여 Exploit 코드를 작성하고 대상 IoT System속으로 침투하는 Zero-Day Attack의 침투테스트를 구현한다.

2.2 Windows 10 IoT Core 취약점 분석

먼저, Windows 10 IoT Core는 Universal Windows Platform(이하, UWP) 인터페이스를 지원한다(그림 2). UWP은 서로 다른 다양한 장치에서도 동일한 어플리케이션을 실행시킬 수 있도록 개발된 기술이다. 즉, 모바일에서 실행 가능한 어플리케이션을 컴퓨터에서도 실행 시킬 수 있다는 점이다. 이러한 UWP 기술의 핵심은 바로 UWP API의 System Call이 지원된다는 점이다. PC, Moblie, IoT, Surface Hub에 해당하는 각각의 Windows 10 프로젝트 산하 운영체제 모두에서 Windows API를 호출 받아 사용할 수 있다는 말과 같다. 본 연구에서는 IoT 시스템에서도 Windows API 핸들링이 가능하다는 점을 적극 활용하여 대상 IoT 시스템 속으로 침투할 것이다.

그림 2. Universal Windows 플랫폼 구성도

2.3 ARMv7 Thumb Architecture Exploit Code 작성 및 구현

위 취약점 분석에 의한 Exploit Code를 작성하여 대상 IoT 시스템의 로컬 시스템 속으로 침투 할 수 있다. 이하, 작성하게 될 Exploit Code는 Session을 수립시키기 위해 TCP/IP 인터페이스를 지원하는 Socket 클래스 Windows Sockets 2 API를 핸들링 하여 IoT 시스템으로부터 역으로 세션을 수립시키는 Reverse_Tcp Connection을 유도한다(그림 3).

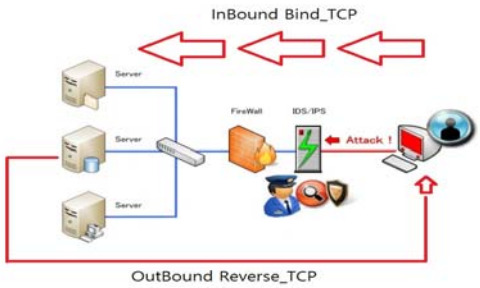

그림 3. In/Out bound Session 수립

대상 IoT 시스템의 Exploit에 사용되는 코드를 분석하면, 먼저 'winsock2.h' 헤더파일이 사용된다. 이 winsock2.h 헤더파일은 Windows 소켓 헤더 파일로 다양한 Instructions 구조체를 포함하고 있다. 한 가지 주의해야 할 점으로 winsock2.h 헤더를 사용하기 위해선 반드시 'ws2_32.lib' 라이브러리 파일이 링크 되어야만 한다. 또한, 'SOCKADDR_IN' 구조체도 사용된다. 이는 Windows 소켓이 소켓을 연결할 로컬 또는 원격의 끝점 주소를 지정하는 데 사용된다.

다음으로, Winsock2를 사용하기 위해선, 먼저 'WS2_32.DLL' 을 초기화 시켜줘야 하는데, 관련 Library를 초기화 시켜주기 위해 WSAStartup 함수가 사용된다. 또한, WSAStartup 함수 사용에 유의점으로 프로그램의 마지막부분에 Winsock 관련 Library를 해제할 때 사용하는 'WSACleanup' 함수를 WSAStartup 함수의 호출 수와 동일한 개수로 반드시 같이 써줘야 한다는 점이다. 더불어, 트랜스포트 서비스 Provider에 바인드 된 소켓을 생성하는 WSASocket과 숫자와 점으로 이루어진 IP 문자열을 Unsigned long(데이터 형식 범위)형의 숫자 IP 주소로 변환시켜주는 inet_addr 함수, Host System에서 Network로 Short 형의 메모리 값 데이터를 보낼 때 Host Byte 순서에서 Network Byte 순서로 Byte Order를 바꿔주는 Htons 함수, 다른 어플리케이션의 소켓으로 접속을 시도하는 WSAConnect 함수, 대상 시스템으로부터의 Cmd Shell Process를 Return 받기 위해 사용하는 Create Process 함수, 이 모두를 종합하여 대상 IoT 시스템으로부터 Cmd Shell을 획득할 수 있는 Exploit Code를 작성한다.

2.4 Cross Compile 구현

위의 작성된 Exploit Code를 IoT 시스템에서 동작 가능하도록 Exploit Backdoor로 컴파일하면 된다. 우리가 침투테스트 하게 되는 대상인 IoT 시스템은 ARMv7 Thumb Architecture 이기 때문에 작성된 Exploit Code를 일반적인 컴파일러로 컴파일하게 되면 Exploit Backdoor가 IoT 시스템에서 동작하지 않는다. 반드시 컴파일러가 실행되는 플랫폼이 아닌 다른 플랫폼에서 실행 가능한 코드를 생성할 수 있는 컴파일러인 Cross Compiler를 사용하여 작성된 Exploit Code가 ARMv7 Thumb Architecture에 포팅 되도록 컴파일 해야 한다(그림 4).

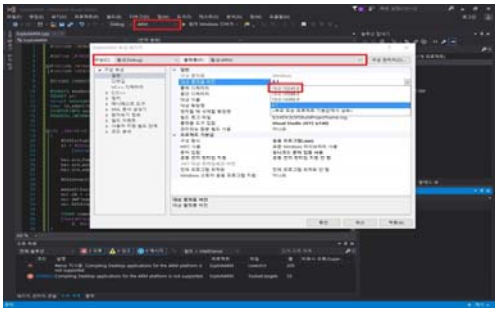

그림 4. 크로스 컴파일 구현

2.5 Session Handler Setting

Cross Compiler로 컴파일 된 Exploit Backdoor로부터 역방향으로 요청되어지는 Reverse_tcp Session을 수립해주고 컨트롤하기 위한 Session Handler를 준비한다. 먼저 첫 번째 방법으로, 윈도우 운영체제와 Linux 운영체제 모두에서 사용할 수 있는 Netcat Handler를 이용하는 방법이다. Netcat Handler를 Listen 모드 상태로 설정하는 -l 옵션과 세션을 수립하기 위한 포트를 지정해주는 -p 옵션을 사용하여 연결대기를 한다. 두 번째 방법으로는 다중 Session 수립을 지원하는 Linux 환경의 Multi Handler Framework를 이용하는 방법이다. Multi Handler Framework는 Resource Script File을 작성하여 Handler 옵션값을 미리 Setting 할 수 있으며, Session 수립뿐만 아니라 Framework로서의 이후 Post Exploit 단계에 대한 다양한 침투테스트 기능을 제공한다.

2.6 IoT System Exploit

앞서 제작한 Exploit Backdoor를 통하여 IoT 시스템의 로컬 시스템 상으로 직접적으로 침투하는 과정이다. 본 연구에서는 공격자 측 환경으로 Linux 운영체제를 사용하여 Exploit을 진행하였다. 먼저, Session 수립과 컨트롤을 위한 Netcat Handler(그림 6) 또는 Linux 환경의 Multi Handler를 구동하여 Listening 대기시킨다(그림 5). 이후, Cross Compiler로 ARMv7 Thumb Architecture에

맞게 포팅 시킨 Exploit Backdoor를 침투대상인 IoT 시스템의 Windows 10 IoT Core에 Upload 시킨다. Upload 된 Exploit Backdoor를 Windows 10 IoT Core에서 실행시키면, Exploit Code가 정상작동 되고, IoT 시스템 측에서 공격자 측으로 Reverse_Tcp Connection Session 수립요청을 하게 된다. 이후, Listening 대기 중인 공격자 측 Handler는 IoT 시스템 측에서 요청한 Session 수립을 정상적으로 공격자 측으로 수립시킨다. 위의 Exploit Code 작성단계에서 Session 수립 중간과정에 Exploit Code의 CreateProcess 함수에 의해 Cmd Shell을 실행 받아 오도록 코딩하였으므로, 이로서 공격자 측은 IoT 시스템의 Cmd Shell을 획득하게 되고 IoT 시스템 로컬 시스템에 대한 Zero-Day Attack Exploit 침투테스트는 성공적으로 완료된다.

그림 5. Multi Handler를 통한 Exploit 구현

그림 6. Netcat Handler를 통한 Exploit 구현

2.7 Zero-Day Attack Exploit Reporting

Zero-Day 공격에 사용된 Exploit Backdoor에 대한 악성 탐지여부를 조사하였다. 검증에는 전 세계의 각기 다른 백신프로그램을 사용하여 파일의 악성 여부를 분석해주는 VirusTotal 사이트를 이용하였다. 그 결과, 본 연구에서 작성한 Exploit Backdoor의 악성 탐지 비율은 전 세계의 주요 백신 프로그램의 46개 중 단 하나의 백신 프로그램도 악성 탐지를 하지 못한 수치인 '악성 시그니처 탐지비율 0%' 를 기록하였다(그림 7 참조).

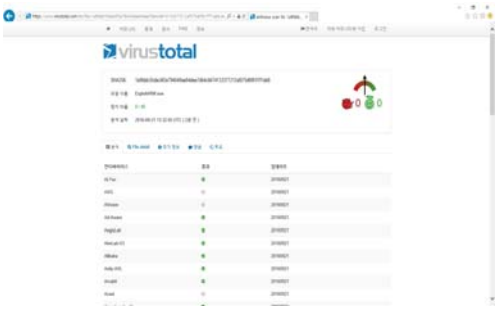

그림 7. VirusTotal 통한 유효성 확인

이로서, IoT 시스템의 ARMv7 Thumb Architecture에 대한 Zero-Day Exploit으로서의 사실이 검증된다.

Ⅲ. 결 론

IoT 시스템의 운영체제인 Windows 10 IoT Core에 대한 Zero-Day Exploit 공격으로 Cmd Shell을 획득하여, 대상 IoT 시스템의 로컬 시스템 속으로 침투를 하였다. 이로 인한 파급효과는 매우 클 것으로 예상된다. 예를 들면, 침투한 대상 IoT 시스템 속에 있는 주요 기밀문서와 같은 중요한 자료를 유출시키거나 파괴시킬 수 있다. 또한, 대상 IoT 시스템 자체를 무력화 시켜 System Down을 시킬 수도 있다. 그러므로 이에 대해, 보안적인 관점에서의 대응방안을 세워야한다.

먼저 첫 번째, 물리적 보안을 강화하는 것이다. 서론에서 언급한 바와 같이 IoT 시스템의 특성상 IoT 디바이스에 대한 직접적인 물리적 접근이 쉬우므로, 물리적 보안에 대한 정책을 강화하여 내부적 침투경로를 원천차단 하는 것이다. 두 번째로, Windows 10 IoT Core용 백신프로그램을 개발하거나 본 연구에서 작성된 Exploit Code에 대한 Signature를 탐지할 수 있는 보안솔루션을 개발하는 것이다.

참고문헌

[1] Microsoft MSDN 기술문서
[2] https://developer.microsoft.com/ko-kr/windows/iot
[3] http://www.pentest-standard.org/index.php/Main_Page