Security Control Center

요약

최근 디바이스 보안 문제와 관련된 각종 IoT 보안 사고가 점차적으로 증가하고 있다. 이러한 보안 이슈를 해결하기 위해서는 IoT 환경 내 디바이스 보안 측면을 고려한 IoT 디바이스가 필수적으로 사용되어야 하고, 각종 사이버 위협에 적절히 대응할 수 있는 보안 관제 서비스가 운영되어야 한다. 본 논문에서는 선행 연구인 IoT 디바이스 보안 플랫폼의 각종 보안 기능을 기반으로 보안 관제 수행에 필요한 기능을 정의하였고, 정의된 기능을 바탕으로 보안 관제 시스템을 설계 및 구현하였다.

1 서론

1.1 연구 개요

IoT 디바이스의 안전한 상태를 주기적으로 확인하고 유지하기 위해서는 보안 관제 시스템이 필요하다. 본 연구에서는 앞서 개발된 디바이스 보안 플랫폼을 관리하기 위한 보안 관제 시스템을 구축하였다. 그리고 이를 통해, 다양한 디바이스 보안 플랫폼들을 통합적으로 관리 할 수 있는 보안 관제 시스템(SCC; Security Control Center)을 개발하였다.

SCC가 제공하는 기능은 ①Secure Key Storage & Management Monitoring, ②Secure Boot Monitoring, ③Secure Firmware Update Management, ④Remote Attestation Monitoring, ⑤File System Integrity Monitoring, ⑥File System Encryption Monitoring, ⑦Login Information Monitoring, ⑧IP/Port Information Monitoring과 같다. 이러한 기능들을 통해 SCC 관리자는 수 많은 IoT 디바이스들을 효율적으로 침해 대응 및 관제 할 수 있으며, 실시간 감시를 통해 다양한 불법적 주체에 대한 접근을 차단할 수 있다.

1.2 IoT 보안 관제 시스템 설계를 위한 요구 사항

본 장에서는 IoT 보안 관제 시스템의 보안 요소기술과 수행하는 기능을 설명한다. 타겟 보안 플랫폼은 COTS IoT 디바이스인 Raspberry Pi 2 (Model B)에 Atmel TPM을 장착한 Secure Pi를 사용하였다.

1.2.1 디바이스의 암호화 키 데이터 가용성 보장

IoT 디바이스의 암호화 키는 다양한 보안 기술에서 활용되며 필요할 때마다 즉시 사용할 수 있는 상태를 유지해야 한다. 대부분의 COTS IoT 디바이스의 경우, 중요한 정보는 Read/Write가 가능한 저장 메모리에 보관하여 해커에 의해 쉽게 누출될 위험이 존재한다. 그러나 Secure Pi는 TPM에서 제공하는 기술을 사용하여 안전한 저장소(Secure Storage), 안전한 키 관리(Secure Key Management)가 가능하다. 만약, 이 암호화 키가 사용될 수 없을 경우 다른 기술들의 정상적인 동작을 보장 할 수 없다. 따라서, Secure Pi를 주기적으로 모니터링 하여 Secure Key Storage & Management를 제대로 수행하고

있는지를 확인할 수 있어야 한다.

1.2.2 디바이스의 펌웨어 무결성 보장(Secure Boot)

IoT 디바이스의 암호화 키는 다양한 보안 기술에서 활용되며 필요할 때마다 즉시 사용할 수 있는 상태를 유지해야 한다. 대부분의 COTS IoT 디바이스의 경우, 중요한 정보는 Read/Write가 가능한 저장 메모리에 보관하여 해커에 의해 쉽게 누출될 위험이 존재한다. 그러나 Secure Pi는 TPM에서 제공하는 기술을 사용하여 안전한 저장소(Secure Storage), 안전한 키 관리(Secure Key Management)가 가능하다. 만약, 이 암호화 키가 사용될 수 없을 경우 다른 기술들의 정상적인 동작을 보장 할 수 없다. 따라서, Secure Pi를 주기적으로 모니터링 하여 Secure Key Storage & Management를 제대로 수행하고 있는지를 확인할 수 있어야 한다.

1.2.3 디바이스의 안전한 펌웨어 업데이트 보장

IoT 디바이스는 펌웨어 교체 공격을 방지할 수 있어야 한다. 펌웨어 교체 공격이란 부트로더나 커널 이미지를 변조하고 교체 및 실행하여 시스템 제어권을 탈취하는 공격이다. 이러한 보안 위협을 고려하지 않은 Raspberry Pi는 BIOS ROM에서 부트로더를 검증하지 않기 때문에 취약점이 존재하지만, Secure Pi는 TPM을 사용하여, 각 부트 단계마다 미리 생성한 서명과 각 부트 과정에서 생성한 서명의 일치 여부를 판단하여 취약점을 막을 수 있다. 따라서, Secure Pi 부팅 시 펌웨어 교체 공격을 막을 수 있는 Secure Boot가 정상 작동하는지를 확인할 수 있어야 한다.

1.2.4 디바이스의 펌웨어 무결성 보장(Remote Attestation

IoT 디바이스는 새로운 펌웨어로 업데이트할 시 펌웨어가 안전한지 확인할 수 있어야 한다. 만약 취약점이 발견된 적법한 버전의 이전 펌웨어를 IoT 디바이스에 설치하면, Secure Boot를 제공하는 디바이스에서도 무결성 검증을 통과할 수 있는 취약점이 있다. 따라서 Secure Pi에서는 TPM을 사용하여 이전 버전의 펌웨어 설치를 방지하는 Secure Firmware Update를 제공한다. 그러므로 IoT 보안 관제 시스템은 Secure Firmware Update를 할 수 있는 업데이트 서버 역할을 해야 하며, 업데이트 과정에 이상 유무를 확인할 수 있어야 한다.

1.2.5 디바이스 파일 시스템 내 파일의 무결성 보장

IoT 디바이스의 파일 시스템 내 파일들이 무결성과 일관성을 유지하고 있는지를 판단할 수 있어야 한다. Secure Pi는 TPM 내 암호화 키로 보완한 IMA(Integrity Measurement Architecture) / EVM(Extended Verification Module)을 사용하여 File System Integrity를 수행한다. 따라서, IoT 보안 관제 시스템은 Secure Pi의 IMA/EVM이 정상 작동하는지를 주기적으로 확인할 수 있어야 한다.

1.2.6 디바이스 파일 시스템 내 파일의 기밀성 보장

IoT 디바이스의 파일 시스템 내 기밀 파일을 암호화/복호화 하여 기밀성을 유지하고 있는지를 판단할 수 있어야 한다. Secure Pi는 파일 시스템 내 파일 기밀성을 제공하기 위해 eCryptFS라는 소프트웨어를 TPM과 연동하여 사용한다. 따라서 IoT 보안 관제 시스템은 eCryptFS 데몬이 정상작동 하는지를 주기적으로 확인할 수 있어야 한다.

1.2.7 디바이스 로그인 시도 감지

IoT 디바이스에 로그인 한 유저 정보를 파악할 수 있어야 한다. IoT 디바이스는 특정한 일을 수행하기 때문에 별도의 사용자 로그인이 필요하지 않다. 따라서 IoT 디바이스에 로그인 한 기록이 있다면 보안 문제가 발생했을 가능성이 있다. 리눅스 기반의 COTS IoT 디바이스인 Secure Pi는 /var/log/auth.log 파일에 로그인 기록을 남긴다. IoT 보안 관제 시스템은 이 로그 파일을 주기적으로 검사하여 로그인 성공 유무, 로그인 방법, 로그인 시간, 로그인 시도한 ID와 IP/Port의 정보를 확인할 수 있어야 한다.

1.2.8 디바이스 허용/거절 패킷 감지

IoT 디바이스에 특정 패킷이 전송되었을 때, 어떠한 패킷을 허용/거절하는 지를 파악할 수 있어야 한다. 리눅스 기반의 COTS IoT 디바이스인 Secure Pi는 Iptables를 방화벽으로 사용한다. 따라서, IoT 보안 관제 시스템은 주기적으로 Secure Pi의 Iptables가 허용/거절하는 패킷에 대한 정보를 확인할 수 있어야 한다.

2 관련연구

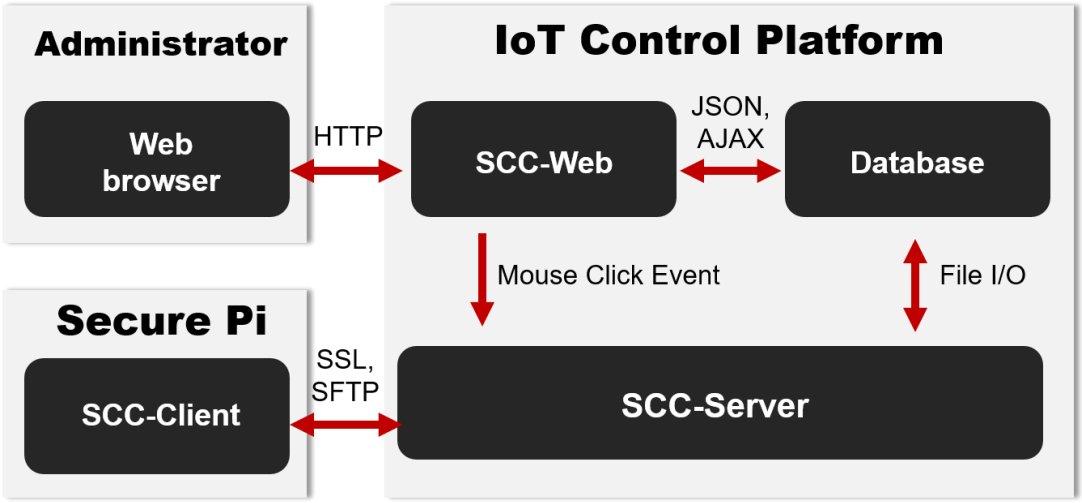
기존 연구로써 개발된 IoT 디바이스 보안 플랫폼(Secure Pi)은 oneM2M 보안 요구사항을 만족함과 동시에 하드웨어 보안 칩인 TPM을 기반으로 아래와 같은 시스템 보안 기술을 제공한다.

- **Secure Key Storage & Management**: 암호화 키를 안전하게 보관 및 관리하기 위한 기술
- **Secure Boot**: 펌웨어 무결성을 보장하기 위한 기술
- **Secure F/W Update**: 안전한 펌웨어 업데이트 지원 기술
- **Remote Attestation**: 디바이스의 신뢰 상태 증명 기술
- **Secure Communication**: 디바이스 간 안전한 통신
- **Mandatory Access Control**: 보안 레벨 기반의 접근 제어
- **Filesystem Integrity**: 파일 시스템 무결성 보장 기술
- **Filesystem Encryption**: 파일 시스템 기밀성 보장 기술

3 IoT 보안 관제 플랫폼

3.1 시스템 아키텍쳐

1.2장에서 분석한 요구사항에 대응하기 위해, Secure Pi의 상태를 모니터링 할 수 있는 보안 관제 시스템 SCC(Security Control Center)를 개발하였다. SCC는 [그림 1]과 같이 SCC-Client, SCC-Server, SCC-Web, Database로 구성되어 있으며 SCC 관리자는 웹 브라우저를 통해 Secure Pi를 모니터링 한다.



[그림 1] SCC System Architecture

SCC-Server, SCC-Web, Database가 있는 IoT Control Platform은 Ubuntu PC(16.04 LTS)를 사용하였고, SCC-Client가 있는 Secure Pi는 Raspberry Pi 2 (Model 2)에 Atmel TPM을 장착하였다.

SCC-Client는 수집한 Secure Pi의 데이터를 SCC-Server에 안전하게 전달하기 위해 SSL(Secure Sockets Layer)을 사용하였으며, SCC-Server는 SFTP(SSH File Transfer Protocol)를 이용하여 Secure Pi의 펌웨어 업데이트를 진행한다. SCC-Server가 파일 입출력을 통해 Database에 저장한 데이터는 SCC-Web이 JSON(JavaScript Object Notation)과 AJAX(Asynchronous JavaScript and XML)을 사용하여 웹 페이지의 형태로 나타낸다.

3.2 SCC-Client

SCC-Client는 Secure Pi의 보안 요소기술들이 동작하면서 생성한 로그 파일 및 에러메시지를 SCC-Server에 전달하며, 앞서 설명한 IoT 보안 관제 시스템 설계를 위한 요구사항과 대응하는 기술이다.

3.2.1 TSS Daemon

TSS(TCG Software Stack)는 Secure Pi를 사용하기 위해 필요한 TPM 드라이버와 소프트웨어의 요구사항을 충족시켜 주는 소프트웨어 프레임워크이다. TSS를 사용하면 Secure Pi의 보안 요소기술들에

사용하는 키가 안전하게 생성되고 저장되었다는 것을 확인할 수 있다. SCC-Client는 이를 통해 Secure Key Storage & Management Monitoring을 수행하고 결과값을 SCC-Server로 전달한다.

3.2.2 Secure Boot Daemon

Secure Pi 부팅 시, TPM이 생성한 키를 사용하여 Secure Boot를 정상적으로 작동했는지를 판단하고 결과값을 SCC-Server로 전달한다.

3.2.3 Secure Firmware Update Daemon

SCC-Server로부터 펌웨어 업데이트를 진행하라는 신호와 펌웨어를 받으면 전송 받은 펌웨어에 대한 무결성 검증을 실시한다. 무결성 검증을 통과하면 업데이트를 진행하고, Secure Pi를 재 부팅 후 결과값을 SCC-Server로 전달한다.

3.2.4 Remote Attestation Daemon

Secure Pi 부팅 시, 주요 프로그램(부트로더, 커널, Secure Boot Daemon, 기타 프로그램)에 대한 무결성 검증을 위해 TPM을 사용하여 시그니처를 만들고 SCC-Server로 전달한다.

3.2.5 IMA/EVM

IMA 해쉬 결과는 공격자도 생성할 수 있는 문제점이 있기 때문에 EVM이라는 기술을 함께 활용하여 IMA의 무결성을 강화한다. 그러나 IMA/EVM을 설정 및 조작할 수 있는 공격자에게 암호화 키가 노출되었을 경우, 공격자는 EVM을 생성할 수 있다. 따라서 Secure Pi는 TPM을 통한 암호화 키를 사용하여 IMA/EVM의 문제점을 해결했다. SCC-Client는 파일 시스템의 무결성 검증을 위하여 주기적으로 IMA/EVM이 정상 작동하는지를 확인하고 결과값을 SCC-Server로 전달한다.

3.2.6 eCryptFS

eCryptFS는 파일 시스템 내 파일을 암호화하여 기밀성을 제공할 수 있는 암호화 소프트웨어다. 그러나 공격자에게 FEKEK(File Encryption Key Encryption Key)가 노출되면 암호화 파일을 복호화 할 수 있다. 따라서 Secure Pi는 TPM을 통해 FEKEK 유출을 방지한다. SCC-Client는 주기적으로 eCryptFS가 정상 작동하는지를 확인하고 결과값을 SCC-Server로 전달한다.

3.2.7 Login Checker

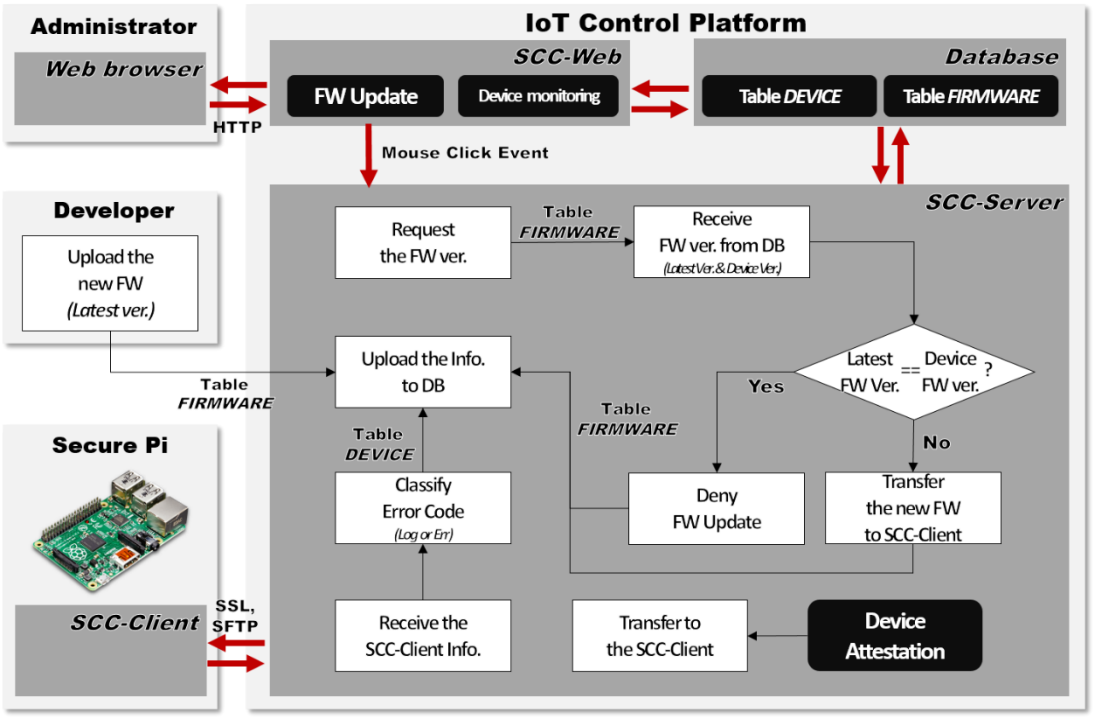
리눅스 기반의 플랫폼은 모든 로그인 시도가 /var/log/auth.log 파일에 기록된다. SCC-Client는 주기적으로 Secure Pi의 auth.log 파일을 확인하고 로그인 기록 결과값을 SCC-Server로 전달한다.

3.2.8 Packet Checker

Iptables를 사용하면 플랫폼에 들어오는 패킷에 대한 정책을 설정할 수 있고 Iptables가 허용/거절한 패킷에 대한 로그를 기록할 수 있는 방화벽이다. SCC-Client는 주기적으로 Secure Pi의 Iptables의 패킷 허용/정책과 로그를 확인하고 결과값을 SCC-Server로 전달한다.

3.3 SCC-Server

[그림 2]는 SCC-Server의 동작 과정을 그린 Flow Chart이다. SCC-Server는 다음과 같은 세가지 기능을 수행한다. 첫째, SCC-Client가 8가지 보안 기술의 결과 값을 전달하면 이를 지정된 에러 코드에 맞게 분류하여 Database DEVICE 테이블에 저장한다. 둘째, SCC-Web에서 펌웨어 업데이트를 하기 위한 마우스 클릭 이벤트가 발생하면, Database FIRMWARE 테이블을 확인하고 현재 Secure Pi의 펌웨어 버전과 최신 펌웨어 버전을 비교한다. 만약 두 버전이 일치하지 않으면 SCC-Client에 신호를 보내어 Secure Firmware Update를 실행시키고, FIRMWARE 테이블에 업데이트 동작을 기록한다. 두 버전이 일치할 때에도 FIRMWARE 테이블의 업데이트 동작 요청이 들어왔다는 것을 기록한다. 셋째, Secure Pi가 부팅되어 SSL 연결이 되면, SCC-Client에 신호를 보내어 Attestation을 실행시킨다.



[그림 2] SCC-Server의 Flow Chart

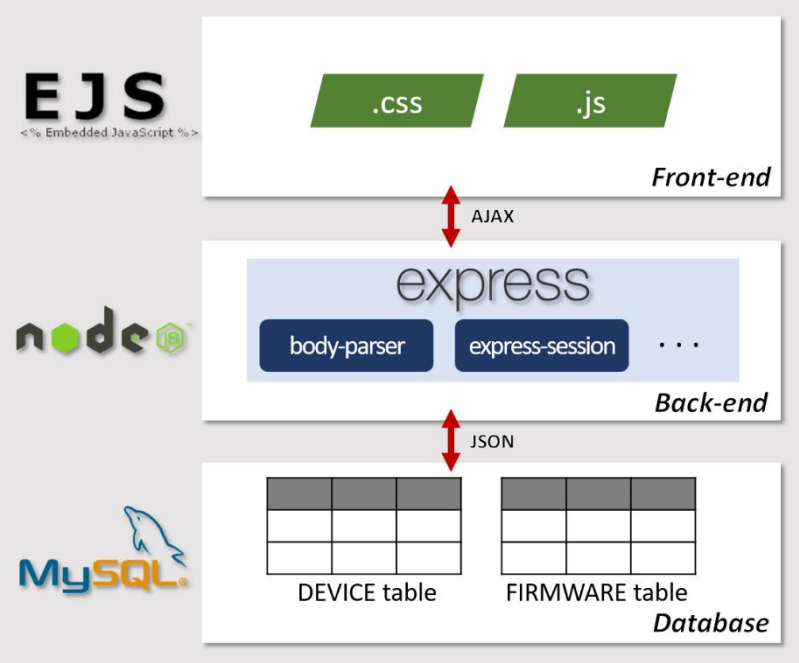
3.4 SCC-Web and Database

SCC는 관리자가 웹 브라우저를 통해 Secure Pi를 모니터링 할 수 있다. 이를 위해 SCC에는 웹 서버로 동작하는 SCC-Web과 Secure Pi의 정보를 저장하는 데이터베이스가 존재하며, 이에 대한 구조도는 [그림 3]과 같다.

MySQL의 DEVICE 테이블과 FIRMWARE 테이블에는 Secure Pi의 SCC-Client로부터 받은 디바이스 정보와 펌웨어 정보가 저장되어 있다. 이 데이터는 JSON 형태로 저장되어 있으며, Node.js와 Express로 작동하는 웹 서버가 사용한다.

Node.js는 Chrome V8 Javascript 엔진으로 빌드된 언어로서 서버 개발에 사용되며, Express는 Node.js의 핵심 모듈인 http와 Connect 컴포넌트를 기반으로 하는 웹 프레임워크다. SCC에서는 Express 모듈로 body-parser, express-session 등을 사용하여 CRUD(Create, Read, Update, Delete) API를 만든다. SCC의 웹 화면은 EJS 템플릿 엔진을 사용하여 동적으로 생성된다.

EJS 템플릿 엔진은 특정 형식의 문자열을 HTML 형식의 문자열로 렌더링하여 웹 페이지를 동적으로 생성시킬 수 있다. 이를 통해 실시간으로 변화는 디바이스들의 정보를 동적으로 브라우징 할 수 있으며, 문제가 발생한 디바이스에 대한 알림을 줄 수 있다.



[그림 3] SCC-Web과 Database의 System Architecture

4 결론

IoT 디바이스의 보안을 위해 디바이스 수준을 고려한 보안 플랫폼과, 이를 모니터링 하는 보안 관제 시스템이 필요하다. 본 논문에서는 IoT 디바이스 보안 관제 시스템의 핵심 요소기술을 정의하였고, 이전 연구를 통해 개발된 IoT 디바이스 보안 플랫폼을 기반으로, 보안 관제 시스템을 제한하였다. 이를 통해 보안 관제 시스템의 관리자는 IoT 디바이스들을 효율적으로 침해 대응 및 관제 할 수 있으며, 실시간 감시를 통해 다양한 불법적 주체에 대한 접근을 차단할 수 있다. 즉, oneM2M에서 제안하는 IoT 디바이스의 보안 진단 및 컨설팅이 가능하다. 향후 연구로 RTOS/펌웨어 기반의 저사양 IoT 디바이스 플랫폼의 설계를 완료하여 구현 중에 있으며, 이 또한 모니터링 할 수 있도록 고려하여 보안 관제 시스템의 품질을 향상시킬 계획이다.