IoT 펌웨어 보안을 위한 FOTA 시스템 구성

김지수
고려대학교 컴퓨터정보통신대학원

kimjisu121@korea.ac.kr

FOTA System Configuration for IoT Firmware Security

Jisu-Kim
Korea University Graduate School of Computer & Information Technology

요 약

다양한 산업과 환경에서 IoT 기술이 사용되는 만큼 보안에 대한 위협도 증가하고 있다. IoT 서비스를 제공하는 디바이스와 시스템이 보안 공격을 당해 중단되는 경우 그에 따른 피해가 막대하기 때문에 IoT 의 보안 중요성은 날로 커지고 있다. IoT 디바이스 보안을 강화하기 위한 방법으로 FOTA 시스템를 통한 펌웨어 업데이트를 하는 것이 필요하다. 본 고에서는 IoT 디바이스 펌웨어 업데이트를 위해 필요한 FOTA 시스템에 대한 아키텍처와 구성 등을 제언한다.

1. 서론

IoT 기술은 모든 사람과 사물을 연결하는 초연결사회가 도래함에 따라 인터넷 기반으로 다양한 사물, 공간, 사람을 유기적으로 연결시키는 기술을 의미하고, 이는 공공서비스, 국가시설 및 다양한 산업에 활용되고 있다. [1] IoT 기술은 센싱-수집-관리(분석)목적으로 구축된 1 단계 연결형 IoT 기술로 시작되어, 2 단계 지능형 기술(1 단계 기술에 지능이 추가되어, 진단-예측이 가능한 기술), 3 단계 자율형 기술(클라우드 환경에서 수행하던 분석-진단-예측을 디바이스에서 수행)방향으로 지속적으로 진화하고 있다. [2] 여러 산업과 다양한 환경에서 IoT 기술이 사용되는 만큼 다양한 보안 위협도 증가하고 있다. IoT 서비스를 제공하는 디바이스와 시스템이 악성코드 감염 혹은 보안 공격을 당해 시스템이 중단되는 경우 그에 따른 피해는 막대하기 때문에, IoT 디바이스와 시스템에 대한 보안의 중요성은 커지고 있다. 따라서 IoT 장치의 안전하고 안정적인 작동을 보장하기 위해서 디바이스의 펌웨어를 업데이트하는 것이 중요해졌다.[3]
이러한 문제들을 해결하기 위해 원격에서 단말관리, 펌웨어 패치, 배포 관리를 효율적이고 안정적으로 할 수 있도록 글로벌 표준을 준용한 FOTA(Firmware Over The Air) 시스템이 필요하다.

2. 글로벌 표준 기반 분석

FOTA 시스템과 관련하여 아래와 같이 필요한 항목을 고려했다. IoT 디바이스의 경우 다양한 펌웨어/OS를 사용하기 때문에 FOTA 시스템의 개방성이 필요하고, 펌웨어 패치 프로세스의 높은 보안과 단말 상태를 확인할 수 있는 디바이스 관리 기능이 필수 항목이라 생각하였다. 이러한 부분을 고려하여 글로벌 표준인 OMA LWM2M(Open Mobile Alliance Light Weight Machine to Machine)을 기반으로 FOTA 시스템 구성요소를 분석 및 설계를 제언한다.

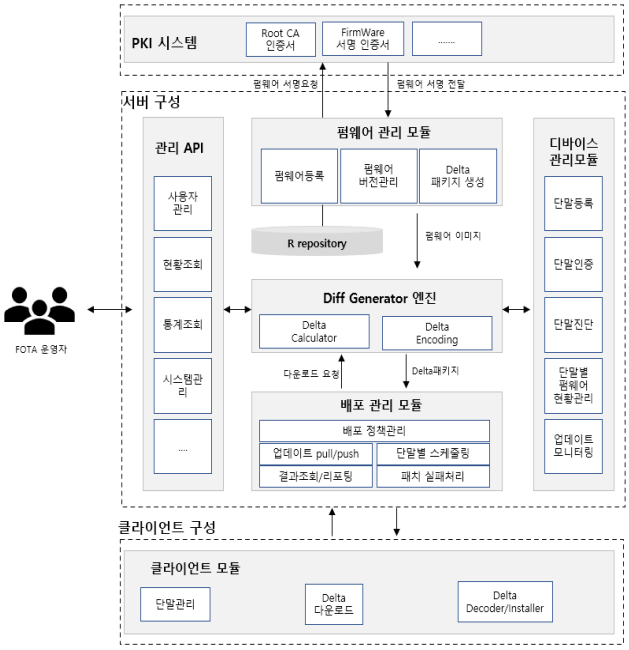
OMA LWM2M 은 OMA 에서 제정한 경량화된 IoT 서비스 표준(LWM2M)이다. LWM2M 의 경우 전력과 데이터소비를 절감할 수 있게 설계되어 경량, 저전력 IoT 디바이스에서도 사용 가능하며, 표준으로 Bootstrap, 단말인증, 장치관리, 메시지 인증 등 보안 필수적인 기능이 정의되어 안전한 시스템 구축이 가능하다. 또한 제조사, 플랫폼, 통신모델 등에 제한받지 않고 모니터링, 원격 장치, 업데이트 등에 적합한 표준 프로토콜이다.[4]

3. FOTA 시스템 구성

3.1 시스템 아키텍처

IoT 기기를 원격에서 관리하며 최신성을 확보하고 새로운 보안 위협과 위험에 대한 적시 대응을 위해서는 펌웨어 패치는 필수적이다. 이러한 펌웨어 패치는 디바이스 관리, 펌웨어 관리, 배포관리, 설치 관리등에 여러 복잡한 프로세스를 거치게 되는데 대규모 장비를 운영하기 위해서 이를 효율적으로 관리해주는 펌웨어 관리 서버가 필요하다. 또한 IoT 단말의 경우 메모리 여유분이 부족하여 펌웨어 업데이트 시 저장할 수 있는 패키지 사이즈에 제한이 있을 수도 있다. 따라서 클라이언트의 경우 저사양/저 전력 단말 내에서도 효율적이고 안정적으로 업데이트가 진행될 수

있게 해주는 클라이언트가 필요하다. FOTA 시스템을 이루는 요소는 아래와 같다.



(그림 1) FOTA 시스템 아키텍처

3.2 시스템 구성요소

3.2.1 FOTA 시스템 서버

펌웨어 패치를 기존에 배포된 또는 새롭게 제조될 장비에 안정적으로 적용하기 위해 펌웨어 패치 프로세스를 관리하는 서버이며, 크게 3 가지 모듈로 구성된다.

(1) 디바이스 관리 모듈 : 다양한 제조사, 모델의 단말 관리를 중앙 통제하기 위한 모듈이다. 단말에 대한 메타정보를 등록, 관리할 수 있고 단말의 현재 속성값 및 연결 상태, 이상여부를 모니터링이 가능하다. 세부기능으로는 단말 등록, 단말 인증, 단말별 펌웨어 버전 및 업데이트 상태 모니터링을 제공한다.

(2) 펌웨어 관리 모듈 : 다양한 제조사, 디바이스의 펌웨어를 버전별, 유형별로 통합 관리를 위한 모듈이다. 제공 기능으로는 펌웨어 등록, 펌웨어 서명 및 인증, 펌웨어 버전 관리, Delta 패키지 생성을 제공한다.

(3) 배포 관리 모듈 : 대규모 디바이스 대상으로 패치 업데이트 작업을 위한 패치 캠페인를 구성할 수 있다. 세부기능으로는 단말별 Delta 패키지 배포 스케쥴링, 업데이트 현황 관리 및 Push 발송, 통신 암호화, 패치 상태/결과 조회/ 실패처리 기능을 제공한다.

3.2.2 FOTA 시스템 클라이언트

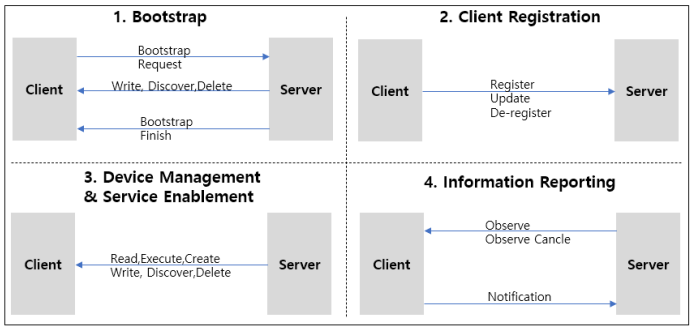
IoT 장비 내에서 구동되는 클라이언트 프로그램으로써, FOTA 시스템 서버로부터 전달되는 제어명령에 따라 펌웨어 패치 등을 수행한다. 주요 모듈은 다음과 같다.

(1) 디바이스 제어 모듈 : 관리서버로부터 받은 관리 명령을 받아서 단말 내에서 수행하는 모듈이다. 주요 수행 기능은 단말의 현재 속성값 및 연결 상태 확인 및 결과 전송, 단말진단 작업, 설정정보 변경, 주기적 패치정보 Pulling, 관리서버로부터 Delta 패키지 다운로드, Delta 패키지 서명 검증을 진행한다.

(2) Diff Generator 엔진 : 기존 소프트웨어 버전과 신규 버전 간의 차이점만 추출하여 패치파일로 패키징하여 사이즈를 최소화 시켜 IoT 단말 패치작업의 효율화를 위해 필요한 기능이다.

3.3 주요 Interface 기능

서버와 클라이언트 간의 Interface 는 1. 서버와 클라이언트 간의 인증정보를 주고 받는 프로세스인 bootstrap, 2. 부트스트랩 절차 완료 후 클라이언트가 서버에 등록하는 작업, 3. 인증된 장치 관리 및 서비스 활성화를 확인하는 기능, 4. 클라이언트 현황 및 장치관리 현황 등을 Reportiong 하는 기능으로 구성되어 있다. 각 구성요소에 대한 자세한 사항은 아래와 같다. [5]



(그림 2) Inferface 기능

(1) Bootstrap : 등록하려는 IoT 단말에 증명서, 키, 클라이언트 이름 및 Server 관련 정보 구성등을 디바이스에 주입하여 디바이스와 Server가 안전하게 연결할 수 있는 프로세스를 의미한다. 클라이언트에서 Bootstrap 진행 요청을 하게 되면 Client에 대한 연결을 포함하여 데이터 모델을 초기화 시킨다. 이후 서버에서 Bootstrap 관련 정보(인증정보, 개인키, 서버정보 등)를 전달 후 해당 기능을 종료한다.

(2) Client Registration : 서버에 클라이언트 및 리소스 등을 등록할 때 사용되며, Register/Register Update/De-Register 3개의 작업을 수행한다. Register의 경우 부트스트랩 절차 완료 후 클라이언트에 저장된 정보와 맞는 서버에 자신을 등록시키는 작업을 말하며, Register Update는 지속적으로 클라이언트 정보를 서버에 업데이트하는 작업, De-Register는 서버와의 연결을 종료하거나, 사용을 중

단하는 작업을 수행한다.

(3) Device Management & Service Enablement : 서버가 클라이언트 인스턴스 및 자원을 모니터링하거나 제어할 때 사용되며, 이미 정의되어 있는 명령어를 통해 클라이언트 관리, 서비스 활성 여부 등을 파악할 수 있다.

(4) Information Reporting : 서버가 클라이언트의 인스턴스 또는 리소스등으로 새로운 데이터를 받기 위해 사용하는 기능이다. 필요 시 서버는 클라이언트 자원의 변화를 관찰하기 위해 연결하며 클라이언트는 변화가 생길 경우 서버한테 데이터를 전달한다.

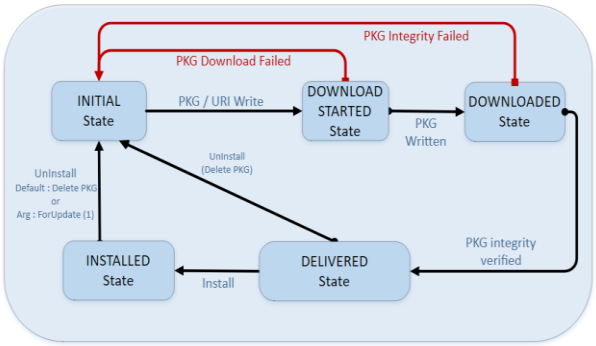
4. Firmware Update 방법

FOTA 시스템을 통해 진행되는 펌웨어 업데이트 패키지 생성 방법은 크게 Full Image 방식과 Delta firmware Update 방식으로 구분된다. Full Image 방식의 경우 신규 펌웨어 전체를 사용하는 방식이며, Delta Update 는 기존 펌웨어와 신규 펌웨어를 계산해서 변경된 펌웨어의 일부만 다운로드하여 업데이트를 진행하는 방식이다. Full Image 방식보다는 Delta Update 방식을 사용하는 경우 업데이트 패키지 파일 생성을 더 작게 만들수 있고 시간과 대역폭 사용의 효율적이다. 그렇기 때문에 Delta Update 방식으로 업데이트를 하는 것이 속도 및 효율성 등 측면에서 더욱 효과적인 방법이라 생각하여 택하게 되었다.

펌웨어 업데이트 생성은 기존 Firmware 와 신규 Firmware 의 차이를 계산해주는 부분인 델타계산기를 통해 변경된 부분(Delta)를 계산하고, 변경된 부분을 패키징해주는 델타 생성기를 통해 최종 업데이트 패키지가 생성되는 방식이다. [6][7]

4.1 Firmware Update 상태

Client 에서 Firmware 업데이트 진행 시 상태를 나타내는 항목은 크게 5 가지로 구분된다.

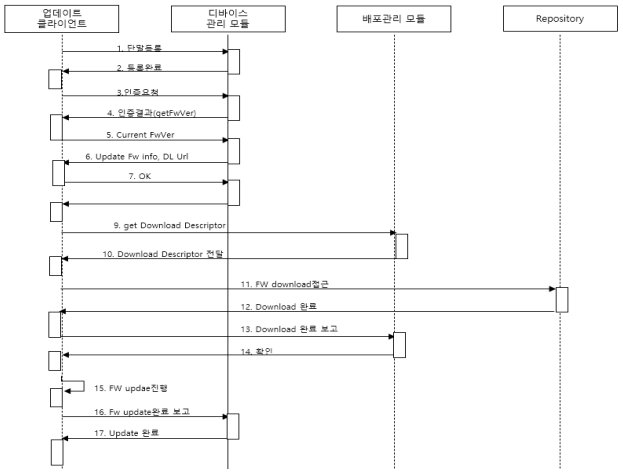


(그림 3) Update 상태

Initial 상태에서 Download PKG/URI Write 를 통해 다운로드 프로세스를 시작한다. 만약 다운로드 중 오류가 발생하면 오류 상태를 띄우고 초기상태로 돌아간다. Download Package 가 완료되었다면 초기상태에서 Download Started 상태를 거쳐 Download 상태로 변환된다. 다운로드 받은 PKG 의 무결성을 확인하여 양호한 PKG 의 파일인 경우 다운로드가 계속 진행되고 Installed 상태로 변환된다. 만약 PKG 의 무결성이 양호하지 않은 경우 Fail 처리로 변환 후 초기 상태로 돌아간다. PKG 가 무결성 파일이지만 중간에 Install 이 제대로 되지 않은 경우 Delivered 상태로 변환하여 설치가 제대로 되지 않았다는 오류 값을 전달 후 Initial 상태로 변환된다.[8]

5. FOTA Update 진행 Process

FOTA 시스템을 사용해서 Firmware를 업데이트 하는 경우 아래와 같이 진행된다. 최초로 클라이언트를 등록하게 되는 경우 서버와 클라이언트 간의 Bootstrap을 진행하여 단말 등록을 진행해야된다. Bootstrap 완료 후 서버에 클라이언트 및 리소스 등록 진행인 Client Register를 진행해야된다. 클라이언트에 저장된 정보와 일치하는 서버에 단말 인증 요청 후 인증 결과를 전달받게 된다. 이 때 인증이 된 경우에는 단말에서 현재 Firmware 버전 상태를 전달하여 해당 버전이 구버전인 경우 서버에서는 Update 정보를 제공하고, 최신 버전의 경우 버전 확인 완료 후 통신을 중단한다. 서버에서 Update 정보 전달 시 FW 버전정보, Update 파일을 받을 수 있는 Download Url이 전달된다. 해당 정보를 받게 되는 경우 Client는 관리 서버 내 배포관리 모듈에 해당 Download Descriptor를 요청 후 전달 받은 다음 Firmware가 저장되어 있는 Repository에서 다운로드를 진행한다. 다운로드 완료 후 디바이스 상태는 서버 내 배포관리 모듈에 디바이스 상태를 보고한다. 다운로드 상태가 fail이거나 error인 경우 기존 펌웨어로 roll-back하거나, 아니면 다운로드 재 시작을 진행하게 된다. 다운로드 상태가 정상적이면 디바이스 내에서 해당 펌웨어 업데이트를 진행하고 디바이스 모듈에 업데이트 정보를 보고한 뒤 펌웨어 업데이트가 완료된다.



(그림 4) Firmware Update 흐름도

6. 결론

사물인터넷 환경에 존재하는 다양한 보안 위협 중 디바이스 펌웨어 업데이트 부재로 인한 보안 취약점 발생 및 보안 위협이 발생이 되고 있다. 펌웨어에 존재하는 보안 취약점을 이용해 비인가자 혹은 공격자가 관리자 권한을 부여받아 중요한 데이터 탈취 및 시스템 장애를 유발하며 기업 및 사회적으로 큰 피해를 입힐 수도 있다. 이러한 문제를 해결하고 IoT 디바이스의 펌웨어 업데이트가 중요해짐에 따라 OMA LWM2M 표준을 기반한 FOTA 시스템의 구성요소를 분석하고 제언했다. FOTA 시스템을 통한 빠르고 안정적인 펌웨어 업데이트 기능을 제공해 보안 취약점에 대응할 수 있으며 더 많은 산업에서 활용할 수 있을 것으로 예상된다. 다만, 해당 논문에서는 시스템의 구성요소의 전체적인 부분에 대해서만 내용을 다뤘고 Server-Client 간의 상태 모니터링 등과 같이 상세한 부분에 대해서는 논하지 못하였다. 추가적인 연구를 통해 이러한 부분은 보안한다면 실제로 해당 시스템을 개발하고 검증하여 상세한 결과를 제공할 수 있을 것이라 생각한다.

참고문헌

[1] 김호원 민경식 박진상, [KISA Insight Vol.05] 지능형 IoT 사회의 보안이슈 분석, 2022

[2] 정보통신기획평가원, ICT R&D 기술로드맵 2025 통신전파보고서, 2020

[3] https://www.keyfactor.com/blog/firmware-security-iot-devices/

[4] https://www.avsystem.com/blog/lwm2m-vs-mqtt-comparison/

[5] https://www.openmobilealliance.org/release/LightweightM2M/V1_2-20201110-A/HTML-Version/OMA-TS-LightweightM2M_Core-V1_2-20201110-A.html#Table-512-2-lessNOTIFICATIONgreater-class-Attributes

[6] elec4.co.kr/article/articleView.asp?idx=18469

[7] https://developer.nordicsemi.com/nRF_Connect_SDK/doc/latest/nrfxlib/nrf_modem/doc/delta_dfu.html#

[8] https://www.openmobilealliance.org/release/LWM2M_SWMGMT/V1_0_2-20210119-A/HTML-Version/OMA-TS-LWM2M_SwMgmt-V1_0_2-20210119-A.html#5-1-0-51-The-Package-Installation-State-Machine-and-operations