하드웨어 보안 칩과 SSL을 활용한 저성능 IoT 디바이스 기반의 Remote Attestation

이기웅 ○ 이기영 김병선 조진성

경희대학교 컴퓨터공학과

lkw1023@khu.ac.kr, lkysecurity@khu.ac.kr, ykbs0903@khu.ac.kr, chojs@khu.ac.kr

Remote attestation based on low-performance IoT device using hardware security chip and SSL

Kiwoong Lee○ Kiyeong Lee Byoungseon Kim Jinsung Cho

Department of Computer Science and Engineering

KyungHee University

요 약

유, 무선으로 인터넷에 연결되는 저성능 IoT 디바이스는 초경량, 저전력을 특징으로 하여 IoT 디바이스의 무결성을 검증하기 위한 소프트웨어 탑재에 어려움이 있다. 이에 IoT 디바이스의 무결성을 외부의 검증된 서버로부터 검증하는 Remote Attestation이 제안되었다. 하지만, Remote Attestation 간 사용되는 정보는 디바이스 내에 저장되는 경우가 있어 잠재적인 보안 위협이 발생할 수 있다. 이러한 문제를 해결하고자 본 논문에서는 SSL 기반의 Remote Attestation을 제시한다. 제시하는 방안은 하드웨어 보안 칩인 SE를 통해 Remote Attestation 간 사용되는 정보를 안전하게 보호하고, SSL 통신을 사용하여 외부의 검증 서버와 IoT 디바이스 간 안전한 데이터 전송을 한다.

1. 서 론

저성능 IoT 디바이스는 유, 무선으로 인터넷에 연결이 가능하며, 이를 통해 각종 센서 등의 값을 읽거나 제어가 가능하다. 이러한 저성능 IoT 디바이스는 초경량, 저전력을 특징으로 제작되어 저성능 IoT 디바이스의 무결성을 검증하기 위한 소프트웨어를 탑재하는데 어려움이 있다[1]. 이러한 문제점을 보완하기 위해 외부의 검증된 서버를 통해 디바이스의 무결성을 검증하는 Remote Attestation이 제안되었다. 하지만 Remote Attestation은 외부의 검증된 서버와 디바이스 간의 데이터 교환 및 생성되는 검증 값의 기밀성에 대한 문제점이 존재한다.

한편, 이러한 저성능 IoT 디바이스에서의 문제점을 개선하고자 여러 보안 단체에서는 보안 기능을 하드웨어적으로 구현한 보안 칩을 개발하였다. 보안 칩은 암호화 기능이 탑재되어 데이터의 기밀성을 유지할 수 있으며, 암호화에 사용되는 암호화 키는 보안 칩 내부에 구현된 안전한 저장소에 저장되어 외부의 공격으로부터 보호된다.

본 논문에서는 저성능 IoT 디바이스의 안전한 무결성 검증을 위해 하드웨어 보안 칩인 SE(Secure Element)를 활용한 Remote Attestation을 제안한다. 제안하는 방안은 SE가 장착된 저성능 IoT 디바이스에서 생성된 안전한 검증 값을 외부의 검증된 서버가 검증하고, 검증 결과에 따라 저성능 IoT 디바이스는 탑재된 Application의 실행 여부를 결정한다. 추가적으로 SSL 통신을 사용하여 외부의 검증된 서버와 저성능 IoT 디바이스 간의 안전한 데이터 교환을 진행한다.

논문의 목차는 다음과 같다. 2장에서는 SSL 및 Remote Attestation에 대해 설명하고, 3장에서는 제안하는 시스템에 대해 설명한다. 4장에서는 결론을 서술하였다.

"본 연구는 삼성전자의 지원으로 수행되었음."

2. 관련 연구

2.1 Remote Attestation

Remote Attestation은 검증된 원격 Server에서 Client의 하드웨어 및 소프트웨어 구성을 인증하는 방법이다[2]. Remote Attestation의 목표는 원격 시스템(Server)이 다른 시스템(Client)의 플랫폼 무결성에 대한 신뢰 수준을 결정할 수 있게 하는 것이다. 이러한 Remote Attestation은 저성능 IoT 디바이스에서 동작하는 SMART[3]가 있다. SMART는 메모리 관리 또는 보호 기능이 없는 Low-end Microcontroller에서 사용하도록 구현되어 있다.

SMART는 Verifier(Server)와 Prover(Client)로 구성되어 있으며 Verifier가 검증하고자 하는 코드의 정보를 Prover로 전달함으로써 시작되며, Verifier의 검증 요청을 받은 Prover는 검증 값을 생성한다. 검증 값은 Prover 제조과정에서 Data Memory Space의 안전한 영역에 삽입된 Protected Key를 이용하여 생성된 HMAC 값이며, 생성된 검증 값은 Verifier에게 전달된다. Verifier는 전달받은 검증 값을 통해 Prover를 검증한다. 하지만 SMART는 HMAC을 생성하는데 필요한 Protected Key가 디바이스 내부에 저장되어 있어 공격자에 의해 Protected Key가 탈취될 수 있는 취약점이 존재한다. Protected Key가 탈취되는 경우 공격자는 저성능 IoT 디바이스의 HMAC 값을 생성할 수 있으며, 이를 Verifier에게 전달하여 변조된 저성능 IoT 디바이스가 검증되는 취약점이 발생할 수 있다.

2.2 OpenSSL

SSL은 Secure Socket Layer의 약자로 Netscape에서 개발된 프로토콜이다. 현재 인터넷 사용자들에게 안전한 개인 정보를 교환하기 위한 표준 프로토콜로 인정되어 많은 온라인 상거래에 사용되고 있다. SSL은 다양한 장점을 지닌 암호화 기법들을 사용해 세계 각국에서 사용되는 대부분의 암호화 기법들을 지원할 수 있다. SSL은 크게 3가지 기능을 제공하며, 공개되어 있는 인터넷상에서 일어나는 트랜잭션의 기밀성을 보장한다[4].

사이트 인증

사용자가 선택한 Web Site를 인증한다. 즉, 사용자가 인터넷 뱅킹이나 인터넷 쇼핑몰을 사용할 경우 이용하는 Web Site가 실제로 신뢰할 수 있는 은행이나 쇼핑몰의 웹 서버인지를 인증하는 것이다.

데이터 기밀성

전달되는 데이터가 도중에 누군가에 의해 판독되지 않음을 보장한다. SSL은 다양한 암호화 알고리즘을 사용하여 인터넷을 통해 전송되는 개인의 사적인 정보를 외부로부터의 불법적인 판독으로부터 막는다.

데이터 무결성

사용자의 브라우저로부터 웹 서버까지 전달되는 동안 데이터가 도중에 누군가에 의해 변경되지 않도록 보장한다.

2.3 SE (Secure Element)

소프트웨어 보안 솔루션들은 일반적으로 고성능 PC 환경을 대상으로 개발되어 상대적으로 저성능 IoT 디바이스에 설치하는 것은 많은 어려움이 있다. 이러한 소프트웨어 보안 솔루션들의 어려움을 극복하고자 여러 보안 단체에서는 각종 보안 기능을 하드웨어적으로 구현한 보안 칩들을 개발하였다. 여러 보안 칩 중 SE는 다음과 같은 특징을 갖는다. 먼저 디바이스 인증을 제공하고, 개인정보 및 데이터를 보호하기 위한 암호화 기능을 제공하며, 암호화에 사용되는 암호화 키는 보안 칩 내부에 구현된 안전한 저장소에 보관되어 소프트웨어 기반 악성 프로그램으로부터 보호하는 기능을 제공한다[5]. 또한, 칩 내부에는 JavaCard OS와 같은 운영체제가 탑재되어 있어 기존 시스템과 통합하기 쉽고, 시스템의 요구사항에 맞춰 칩 내부에 개별적인 보안 솔루션을 프로그래밍 할 수 있다는 장점이 있다.

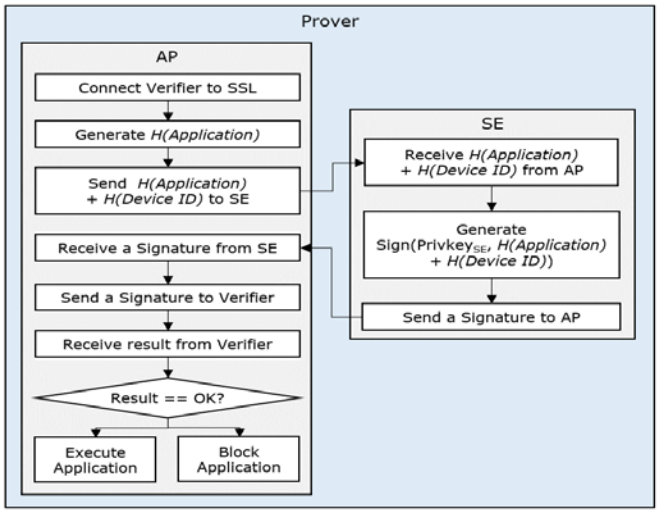
3. 제안하는 시스템

제안하는 시스템의 구성은 다음과 같다. 크게 Verifier와 Prover로 구성되며 Prover는 AP와 SE로 구성된다. AP란 저성능 IoT 디바이스를 나타내며, SE는 2.3에서 서술한 하드웨어 보안 칩을 나타낸다. 제안하는 시스템은 다음과 같은 동작 순서를 갖는다. 먼저 Verifier가 Prover로 검증 값 요청을 전달한다. 요청을 받은 Prover는 AP의 Application 영역을 해쉬하고, 해당 값을 SE로 전달한다. SE는 제조과정에서 생성된 Private Key를 이용하여 전달받은 Application 해쉬값을 서명하고 서명된 값을 Verifier로 전달한다. 마지막으로 Verifier는 전달받은 서명 값을 검증하고 해당 결과를 Prover로 전달한다.

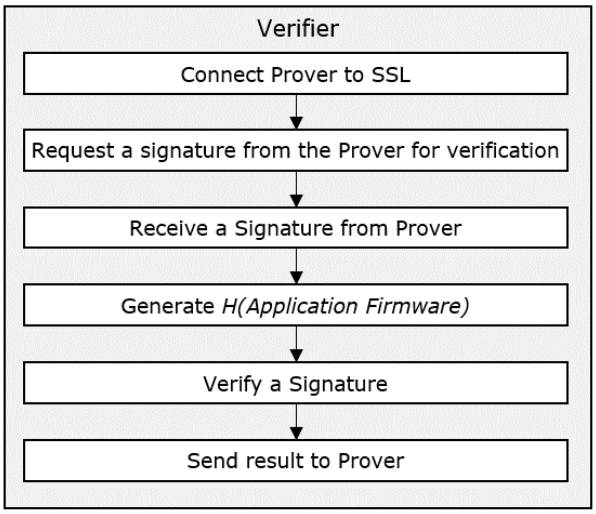
3.1 Prover

Prover는 AP와 SE로 구성된다. 저성능 IoT 디바이스인 AP는 Verifier와 안전한 데이터 교환을 위해 SSL 라이브러리, 검증 값 생성을 위해 Hash Function, AP를 증명하기 위한 AP 고유의 값인 Device ID가 해쉬된 값을 가지고 있다. 이러한 AP는 [그림 1]과 같은 순서로 동작한다. 먼저, Verifier와 안전한 데이터 교환을 위해 SSL을 이용하여 세션을 연결한다. 세션 연결이 완료되면 Verifier는 Prover에게 검증 값 전달 요청 메시지를 전달한다. 해당 메시지를 전달받은 AP는 Hash Function을 통해 Application 영역을 해쉬 한다. Application 해쉬 값은 해쉬된 Device ID 값과 함께 SE로 전달된다. SE는 Private Key를 가지고 있으며 해당 Key를 통해 전달된 해쉬 값들을 서명한다. 서명된 값은 AP로 전달되며, AP는 다시 Verifier로 전달한다. 마지막으로 Verifier로부터 검증 결과를 받게 되며 해당 결과 값을 확인하여 Application을 실행 또는 Block하게 된다.

3.2 Verifier



[그림 1] Prover 실행 순서



[그림 2] Verifier 실행 순서

Verifier는 Prover가 전달하는 검증 값을 검증하는 역할을 하며 Prover와의 안전한 데이터 교환을 위해 사용하는 SSL 라이브러리, 자체적으로 검증 값을 생성하기 위해 필요한 AP의 Application Firmware 및 Hash Function, 해쉬된 AP의 Device ID, 마지막으로 전달되는 서명된 검증 값을 복호화하기 위해 사용되는 SE의 Public Key를 가지고 있다. 이러한 Verifier는 [그림 2]와 같은 순서로 동작한다. 먼저, Prover와 안전한 데이터 교환을 위해 SSL을 이용하여 세션을 연결한다. 세션 연결이 완료되면 Verifier는 Prover에게 검증 값 전달 요청 메시지를 전달한다. 이후 Prover는 검증 값을 생성하고 생성된 값을 Verifier로 전달한다. 검증 값을 전달받은 Verifier는 가지고 있는 AP의 Application Firmware를 해쉬 한다. 이후 전달 받은 서명된 검증 값을 SE의 Public Key를 이용하여 복호화한다. 복호화 된 검증 값과 가지고 있는 해쉬된 Application Firmware 값과 해쉬된 AP의 Device ID를 이용하여 검증한다. 마지막으로 검증 결과 값을 Prover로 전달한다.

3.3 시나리오

제안하는 시스템을 통한 Remote Attestation을 통해 기존 Remote Attestation의 취약점을 개선할 수 있다. 먼저, 저성능 IoT 디바이스의 검증 값인 HMAC 값을 생성하는데 사용되는 Protected Key는 IoT 디바이스 내부에 저장되어 있어 공격자로부터의 위협에 노출되는 취약점이 있다. 이러한 취약점은 제안하는 시스템의 SE에 Protected Key를 저장함으로써 안전한 Key 저장을 할 수 있다. 또한 HMAC을 이용한 검증 값 대신 Private Key를 통한 서명 값 생성 시 AP의 Device ID를 같이 서명함으로써 Prover에서 값이 생성되었다는 것을 증명할 수 있어 검증 값의 신뢰성을 높일 수 있다.

4. 결론 및 향후 계획

본 논문에서는 저성능 IoT 디바이스에서의 Remote Attestation 구현에 대해 제시하였다. 제시하는 방안은 저성능 IoT 디바이스에 하드웨어 보안 칩인 SE를 장착하여 Remote Attestation에서 사용되는 검증 값을 생성하는데 사용되는 Key를 안전하게 보관함으로써 보안성을 강화하고, 저성능 IoT 디바이스를 외부 서버인 Verifier로부터 검증받음으로써 저성능 IoT 디바이스의 무결성을 검증할 수 있다. 향후 계획으로는 시나리오에서 서술한바와 같이 Protected Key를 이용하여 생성된 HMAC 값 대신 서명을 이용하여 저성능 IoT 디바이스의 무결성을 검증한다.

참 고 문 헌

[1] 신수민, 김학범, “사물 인터넷 환경에서의 센서 네트워킹 보안 기술 분석,” 정보보호학회지, 24(4), 56-65, 2014.08.

[2] George Coker, Joshua Guttman, Peter Loscocco, Amy Herzog, Jonathan Millen, Brian O’ Hanlon, John Ramsdell, Ariel Segall, Justin Sheehy, Brain Sniffen, “Principles of remote attestation,” ICICS, 10(2), 63-81, 2011.06.

[3] Eldefrawy Karim, Francillon Aurelien, Perito Daniele, Tsudik Gene, “SMART: Secure and Minimal Architecture for (Establishing a Dynamic) Root of Trust,” NDSS, 2012.02.

[4] 정인성, 신용태, ”OpenSSL을 이용한 보안 통신 API의 설계 및 구현,” 인터넷정보학회논문지, 4(5), 87-96, 2003.10.

[5] 박상호, 권태경, “안드로이드 앱을 위한 플랫폼 보안 분석,” 인터넷정보학회논문지, 14(2), 79-80, 2013.11.