오픈 홈 IoT 환경에서 사용자 인증 및 API 접근을 위한 JWT 적용에 관한 연구

홍남수*, 오창현*, 전문석*
*숭실대학교 컴퓨터학과
e-mail:sucream@ssu.ac.kr

A Study on JWT for User Authentication and Access API in Open Home IoT Environment

Namsu Hong*, Changhyun Oh*, Moon-Seog Jun*
*Dept of Computer Science & Engineering, Soongsil University

요 약

최근 IT 기술의 발전으로 다양한 IoT 기기들이 등장하고 있다. 이러한 IoT 기기들에 접근하기 위해 사용자는 인증을 하는 과정을 거쳐야 한다. 하지만 IoT 기기들이 서로 다른 인증시스템을 가지고 있고 각각의 기기들을 통합적으로 관리하기 위해 일반적으로 IoT 게이트웨이를 이용하여 통합된 인증 시스템을 구축하여 사용하고 있다. 하지만 기존 IoT 게이트웨이의 경우 접근 시마다 사용자 계정으로 로그인해야하는 불편함과 세션 연결의 취약점이 존재한다. 따라서 본 논문에서는 IoT 게이트웨이에서 개선된 사용자 인증 및 API 접근을 위한 JWT를 제안한다.

1. 서론

최근 다양한 IT 기술의 발전으로 인간 대 인간 중심의 패러다임이 인간 대 사물, 사물 대 사물 중심으로 바뀌어 가고 있다. 이 때문에 다양한 분야에서 IoT(Internet of Things) 기기들이 등장하고 있으며, IoT 기기들은 일반적으로 사용자에게 정보를 제공하거나 사용자의 요청을 수행한다.

하지만, 사용자에게 서비스를 제공하기 위해 IoT 기기는 사용자를 인증하는 과정을 필요로 하게 된다. 수많은 IoT 기기들은 서로 다른 인증 절차를 가지고 있고, 이는 사용자에게 불편함을 줄 수 있다. 이를 해결하기 위해 많은 논문들에서 통합된 사용자 인증을 위해 IoT 환경에 한 번의 사용자 인증만으로 모든 IoT 기기들에 접근이 가능한 OpenID 기반의 OAuth 시스템 적용에 관한 논문을 제안하였다[1][2]. 하지만 OAuth는 사용자가 API에 접근할 때마다 인가된 사용자임을 증명하는 Access 토큰을 서버에 전달하게 되는데, 서버는 전달받은 Access 토큰의 유효성 확인을 위해 외부에 존재하는 또 다른 Authorization Server를 거친다는 단점이 있다. 따라서 본 논문에서는 외부 Authorization server를 거치지 않고 토큰의 유효성을 자체적으로 검증할 수 있는 IoT 게이트웨이를 이용한 통합된 JWT 사용자 인증 시스템을 제안한다.

2. 관련 연구

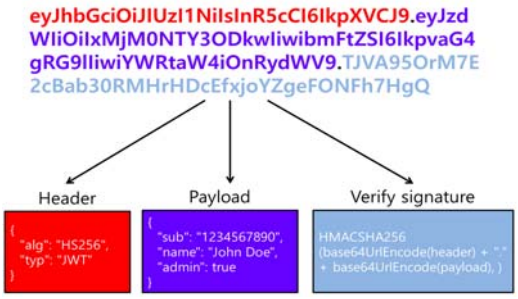
2.1 IoT 게이트웨이

IoT는 사용자의 편리성과 생산성, 안전성 등 다양한 측면에서 이점이 될 것이라 여겨진다. 하지만 아직까지 이러한 IoT 서비스가 우리에게 대단한 편리함을 전해줄 정도로 서비스를 제공하지 못하고 있고, 제공되는 서비스 또한 미미한 수준에 불과하다고 여겨지고 있다. 사용자에게 다양한 서비스를 제공하지 못하는 가장 큰 이유는 사물(Device)들이 만들어내는 서비스의 형태가 서로 다르고, 각각의 사물들을 연결하여도 이를 공통된 로직으로 처리하는 것이 힘들기 때문이다. 이런 문제점을 해결하기 위해 IoT 게이트웨이를 사용하여 호환성 없는 사물들의 통신을 돕도록 하는 개방형 IoT 플랫폼에 관련된 연구들이 등장하고 있다[3][4]. IoT 게이트웨이는 일반적인 IoT 통신기기들이 사용하는 Z-WAVE, ZigBee, RFID등의 통신을 통해 데이터를 수집하거나 사용자로부터 전달받은 API 요청을 IoT 기기에게 전달 후 결과를 사용자에게 전달한다. 또는 수집된 데이터 중 변경된 데이터만을 클라우드 서버에 전달하는 필터링 기능도 제공하고 있다. 사용자는 단지 IoT 게이트웨이에 접속하는 것만으로 IoT 게이트웨이에 연결되어있는 기기들에 접근하는 것이 가능하다.

This work was supported by the ICT R&D program of MSIP/IITP. [R0112-14-1061, The analysis technology of a vulnerability on an open-source software, and the development of Platform]

2.2 JWT(JSON Web Token)

일반적인 웹 통신에서는 HTTP Request, HTTP Response를 통해 서버와 클라이언트가 통신을 한다. 하지만 이러한 웹 통신은 일시적이기 때문에 서버는 클라이언트의 이전 상태를 알 수 없는데, 이를 Stateless라고 한다. 따라서 클라이언트가 서버에 접속하여 로그인에 성공하였더라도 서버는 클라이언트의 로그 정보를 유지할 수가 없다. 서버와 클라이언트 사이의 지속된 연결을 유지하기 위해 세션(Session)과 쿠키(Cookie)를 사용하여 Stateless의 문제를 해결하였다. 하지만 단순한 세션과 쿠키의 사용은 CSRF(Cross-Site Request Forgery), XSS(Cross-Site Scripting)등의 문제가 발생할 수 있다[5]. 세션과 쿠키의 보안을 위해 SSL/TLS 통신에서만 사용이 가능한 쿠키 값을 사용하거나 최근에는 세션이 아닌 토큰 방식의 사용자 인증 시스템을 사용하고 있다. 대표적인 토큰 방식으로 OAuth가 있는데, 사용자는 API 서버에 접근할 때마다 Access 토큰과 Refresh 토큰을 이용하여 사용자의 인증과 권한을 확인하는 방식이다[6]. 하지만 OAuth는 Access토큰의 유효성을 검사하기 위해 외부의 Authorization Server를 통해 복잡한 과정을 거쳐 확인한다는 단점이 있다. 또한 많은 통신과정이 있기 때문에 오버헤드가 발생할 수 있으며, OAuth는 signature를 사용하지 않고 SSL/TLS 기능에만 의존하기 때문에 공격자가 Access 토큰을 탈취할 경우, 심각한 문제가 발생할 수 있다. 최근에는 JWT(JSON Web Token)이라는 새로운 토큰 인증 시스템이 등장하였다[7]. JWT는 OAuth와 마찬가지로 사용자에게 토큰을 발급해준다. 하지만 JWT가 OAuth와 다른 점은 사용자에게 발급해준 토큰이 단순한 토큰이 아니라 토큰 내부에 특정 정보를 담고 있다는 것이다. JWT는 Header, Payload, Signature 필드로 이루어져있다.



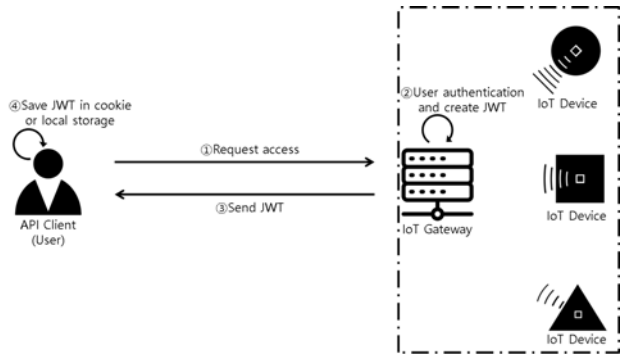
(그림 1) JWT의 구성요소

JWT는 각각의 필드를 Base64로 인코딩하여 ".(dot)"으로 구분한다. 먼저 Header필드에는 JWT에서 사용할 암호화 방식을 기술하고 있다. Payload필드에서는 사용자ID, 권한, 토큰 발급기관, 토큰의 유효기간 등 원하는 정보를 넣을 수 있다. Signature필드는 Header필드와 Payload필드를 Secret key를 통해 Hash한 값이다. 따라서 Signature필드를 통해 토큰에 대한 무결성을 보장받을 수 있다. 또한 이 Secret Key는 서버만이 알고 있기 때문에 악의적인 사용자가 토큰을 변조하여도 Secret Key가 없기 때문에 서버에서 토큰 검증 시 유효하지 않은 토큰으로 분류되어 올바른 API 접근이 불가능하다. JWT의 또 다른 장점은 서버가 토큰의 유효성을 검증하기 위해 외부의 Authorization server로 토큰을 전송할 필요가 없기 때문에 서버와 클라이언트 사이의 보안에만 신경 쓸 수 있고 비교적 오버헤드가 적다는 장점이 있다.

3. JWT를 이용한 사용자 인증 및 API 접근 제안

사용자(API Client)는 IoT 디바이스들이 연결된 IoT 게이트웨이에 접근하고자 할 때, SSL/TLS 통신 환경에서 자신의 계정으로 로그인한다. 이후 서버는 사용자의 IMEI(International Mobile Equipment Identity)값과 SSL/TLS 핸드쉐이크시 사용한 premaster key값을 인증하고 사용자 고유의 Secret Key를 생성한다. 그리고 Secret key를 이용하여 JWT를 생성한 뒤 사용자에게 발급하고 사용자는 발급받은 JWT를 통해 API에 접근하는 시스템을 제안한다. 이 때, IoT 게이트웨이와 IoT 디바이스 사이에는 신뢰되는 대칭키 통신을 사용하고 API Client 디바이스의 고유한 IMEI번호를 IoT 게이트웨이에 사전에 등록했다고 가정한다. 또한 IoT 게이트웨이는 SSL인증서를 발급받아 안전하게 HTTPS 통신을 한다고 가정한다.

3.1 사용자 인증 및 토큰 발급



(그림 2) JWT 토큰 발급 순서도

제안하는 JWT 토큰 발급 순서는 다음과 같다.

Step.1) API Client와 IoT 게이트웨이 사이의 SSL/TLS 연결 확립

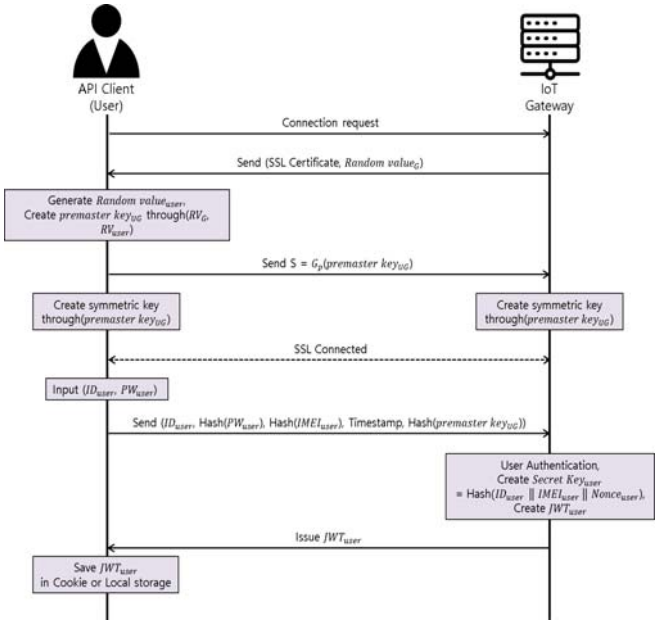
Step.2) API Client의 ID와 패스워드 입력

Step.3) $Hash(PW_{user1})$, $Hash(IMEI_{user1})$, Timestamp, $Hash(premasterkey_{UG})$를 IoT 게이트웨이에 전송

Step.4) IoT 게이트웨이는 API Client의 정보와 IMEI, premaster key를 확인하여 인가된 사용자임을 확인

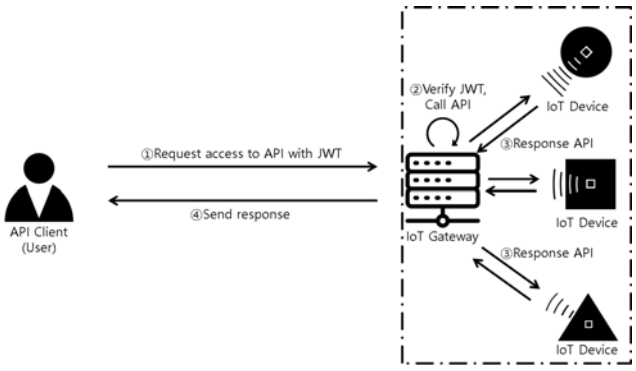
Step.5) $Hash(ID_{user} \| IMEI_{user} \| Nonce_{user})$값을 API Client의 secret key로 사용하여 API Client의 JWT를 생성 후 발급

Step.6) API Client는 발급받은 JWT를 Cookie 또는 Local storage에 안전하게 저장



(그림 3) JWT 토큰 발급 프로토콜

3.2 JWT를 이용한 사용자 API 접근



(그림 4) 사용자 API 접근 순서도

제안하는 사용자 API 접근 순서는 다음과 같다.

Step.1) API Client는 IoT 게이트웨이에 접근

Step.2) API Client가 API 요청시, Cookie나 local storage에 저장된 JWT와 $Hash(IMEI_{user1})$, Timestamp, $Hash(premasterkey_{UG})$를 같이 전달

Step.3) IoT 게이트웨이는 API Client 정보와 IMEI, premaster key를 확인하여 인가된 API Client임을 확인

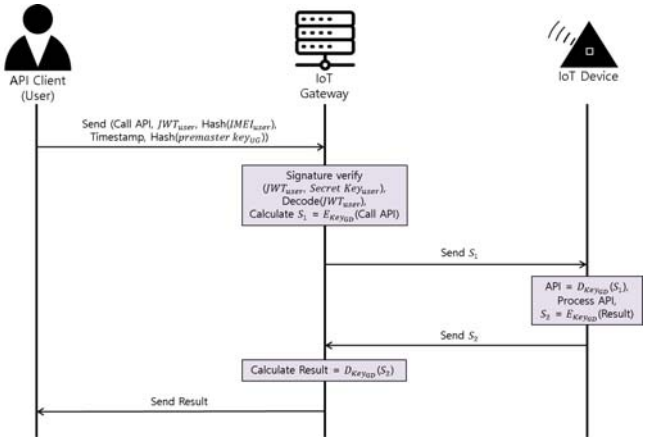
Step.4) JWT의 secret key를 이용하여 토큰의 무결성 확인

Step.5) 인가된 API Client인 경우에 토큰 디코딩

Step.6) IoT 게이트웨이는 IoT 디바이스에게 대칭키 Key_{GD}를 이용하여 API 요청을 전달

Step.7) IoT 디바이스는 API요청을 처리후 결과를 Key_{GD}를 이용하여 IoT 게이트웨이에 전달

Step.8) IoT 게이트웨이는 결과를 API Client에게 전달



(그림 5) 사용자 API 접근 프로토콜

4. 결론

사용자는 다양한 IoT 디바이스에 접근하기 위해 서로 다른 인증을 거치지 않고 IoT 게이트웨이를 통해 한 번의 인증으로 원하는 IoT 디바이스에 접근이 가능하게 된다.

하지만 이러한 서비스를 일반적으로 제공하는 OAuth는 단순히 SSL/TLS에만 의존하며 토큰의 Signature가 없기 때문에 토큰을 탈취당할 경우, 심각한 문제를 초래할 수 있다. 이를 해결하기 위해 본 논문에서는 JWT를 이용한 사용자 인증 및 API 접근 기법을 제안하였다. 이로 인해 사용자마다 서로 다른 권한을 가지는 토큰을 생성할 수 있고 인가된 사용자만이 IoT 게이트웨이에 접근하여 원하는 API를 사용할 수 있게 된다. 또한 세션 통신을 하지 않고 CORS의 사용을 제한하여 JSON 데이터의 이동만을 허용하며, JWT는 HTTP Only cookie 또는 Secure cookie등 안전한 cookie에 저장하거나 local storage에 저장하기 때문에 CSRF, XSS등의 공격으로 인한 취약점 문제와 사용자가 지속적으로 API에 접근할 때 발생하는 추가적인 오버헤드 문제도 해결될 것으로 보인다.

참고문헌

[1] “An Authentication mechanism for IoT Network based on OAuth Protocol”,Young Kyu Choi, Seon Jeong Kim, Kang Seok Kim, Ki-Hyoung Kim, 2015.6.

[2] “OAuth 2.0 For Event-based IoT Device Data Communication” Kun Woong Yuk, Lim Taebeom, 2015.7.

[3] “Development of IoT Gateway based on Open Source H/W”, Dae-Hyun Ryu, 2015.9.

[4] “Study on Lightweight IoT Sensor Gateway Using Open Source Hardware”, Seung-Hyeok Shi, 2015.10

[5] “Cross-Site Request Forgeries: Exploitation and Prevention”, William Zeller, Edward W. Felten, 2015,5.

[6] “The OAuth 2.0 Authorization Framework”, https://tools.ietf.org/html/rfc6749, 2012.10.

[7] “JWT intoduction”, https://jwt.io/introduction/