JWT (JSON Web Token)

```
JSON Web Tokens are an open, industry standard [RFC 7519]
method for representing claims securely between two parties.
출처 : https://jwt.io
```

JWT는 웹표준(RFC 7519)으로서 두 개체에서 JSON 객체를 사용하여 가볍고 자가수용적인 방식으로 정보를 안전성 있게 전달해줍니다.

구성요소

JWT는 `.` 을 구분자로 3가지의 문자열로 구성되어 있습니다.

aaaa.bbbbb.ccccc 의 구조로 앞부터 헤더(header), 내용(payload), 서명(signature)로 구성됩니다.

헤더 (Header)

헤더는 typ와 alg 두가지의 정보를 지니고 있습니다. typ는 토큰의 타입을 지정합니다. JWT이기에 "JWT"라는 값이 들어갑니다. alg : 해싱 알고리즘을 지정합니다. 기본적으로 HMAC, SHA256, RSA가 사용되며 토큰을 검증할 때 사용되는 signature부분에서 사용됩니다.

```
{
    "typ" : "JWT",
    "alg" : "HS256"
}
```

정보(payload)

Payload 부분에는 토큰을 담을 정보가 들어있습니다. 정보의 한 조각을 클레임(claim)이라고 부르고, 이는 name / value의 한 쌍으로 이뤄져있습니다. 토큰에는 여러개의 클레임들을 넣을 수 있지만 너무 많아질경우 토큰의 길이가 길어질 수 있습니다.

클레임의 종류는 크게 세분류로 나누어집니다.

1. 등록된(registered) 클레임 등록된 클레임들은 서비스에서 필요한 정보들이 아닌, 토큰에 대한 정보들을 담기위하여 이름이 이미 정해진 클레임들입니다. 등록된 클레임의 사용은 모두 선택적(optional)이며, 이에 포함된 크레임 이름들은 다음과 같습니다.

- `iss` : 토큰 발급자 (issuer)
- `sub` : 토큰 제목 (subject)
- `aud` : 토큰 대상자 (audience)
- `exp` : 토큰의 만료시간(expiration), 시간은 NumericDate 형식으로 되어있어야 하며 언제나 현재 시간보다 이후로 설정되어 있어야 합니다.
- `nbf` : Not before을 의미하며, 토큰의 활성 날짜와 비슷한 개념입니다. 여기에도 NumericDate형식으로 날짜를 지정하며, 이 날짜가 지정하며, 이 날짜가 지나기 전까지는 토큰이 처리되지 않습니다.

- `iat` : 토큰이 발급된 시간(issued at), 이 값을 사용하여 토큰의 age가 얼마나 되었는지 판단 할 수 있습니다.
- `jti` : JWT의 고유 식별자로서, 주로 중복적인 처리를 방지하기 위하여 사용됩니다. 일회용 토큰에 사용하면 유용합니다.

2. 공개(public) 클레임 공개 클레임들은 충돌이 방지된(collision-resistant)이름을 가지고 있어야 합니다. 충돌을 방지하기 위해서는, 클레임 이름을 URI형식으로 짓습니다.

```
{
    "https://chup.tistory.com/jwt_claims/is_admin" : true
}
```

3. 비공개(private) 클레임 등록된 클레임도 아니고, 공개된 클레임들도 아닙니다. 양 측간에(보통 클라이언트 <-> 서버) 합의하에 사용되는 클레임 이름들입니다. 공개 클레임과는 달리 이름이 중복되어 충돌이 될 수 있으니 사용할때에 유의해야합니다.

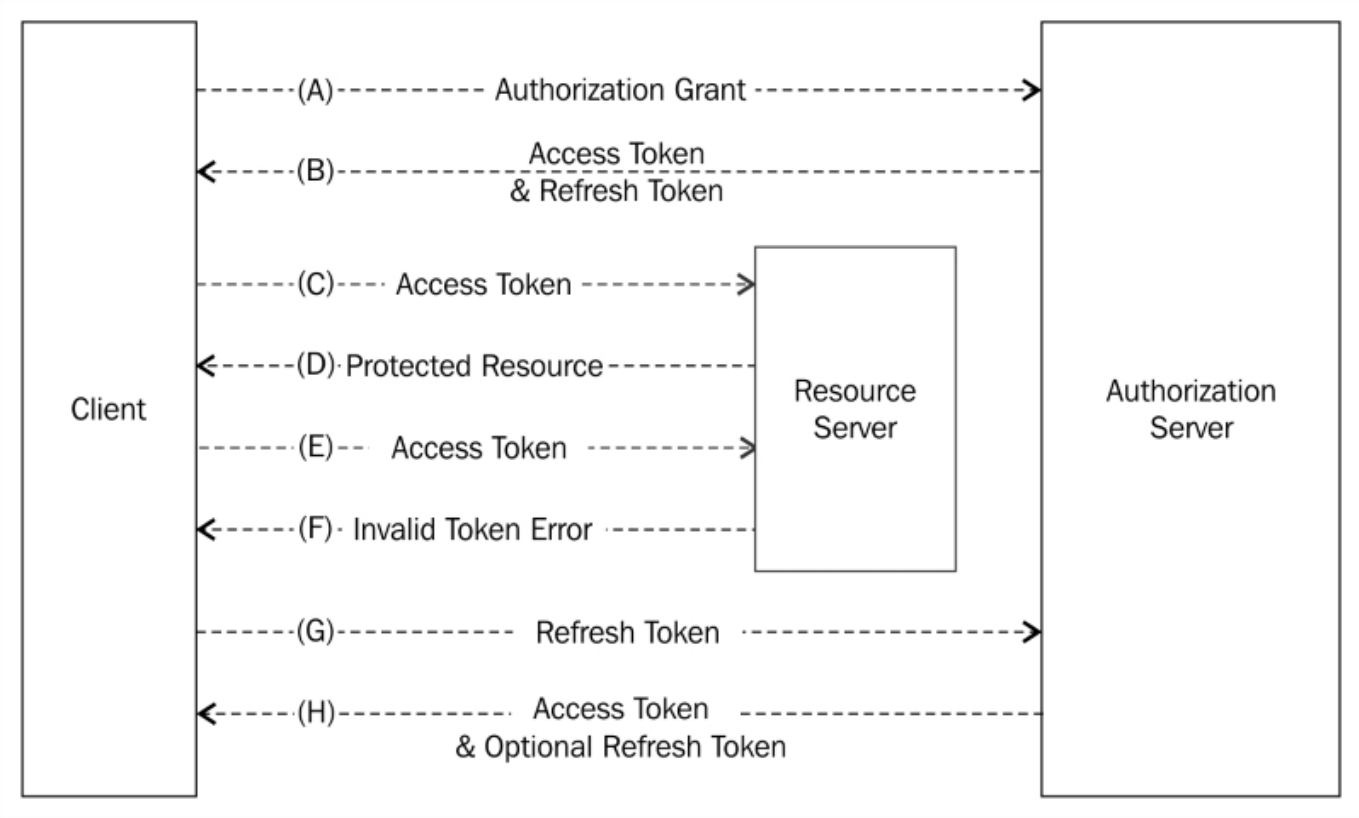
서명(signature)

서명은 헤더의 인코딩값과 정보의 인코딩값을 합친후 주어진 비밀키로 해쉬를 하여 생성합니다. 이렇게 만든 해쉬를 `base64`형태로 나타내게 됩니다.

로그인 인증시 JWT 사용

만약 유효기간이 짧은 Token을 발급하게되면 사용자 입장에서 자주 로그인을 해야하기 때문에 번거롭고 반대로 유효기간이 긴 Token을 발급하게되면 제 3자에게 토큰을 탈취당할 경우 보안에 취약하다는 약점이 있습니다. 그 점들을 보완하기 위해 **Refresh Token** 을 사용하게 되었습니다. Refresh Token은 Access Token과 똑같은 JWT입니다. Access Token의 유효기간이 만료되었을 때, Refresh Token이 새로 발급해주는 열쇠가 됩니다. 예를 들어, Refresh Token의 유효기간은 1주, Access Token의 유효기간은 1시간이라고 한다면, 사용자는 Access Token으로 1시간동안 API요청을 하다가 시간이 만료되면 Refresh Token을 이용하여 새롭게 발급해줍니다. 이 방법또한 Access Token이 탈취당한다해도 정보가 유출이 되는걸 막을 수 없지만, 더 짧은 유효기간때문에 탈취되는 가능성이 적다는 점을 이용한 것입니다. Refresh Token또한 유효기간이 만료됐다면, 사용자는 새로 로그인해야 합니다. Refresh Token도 탈취 될 가능성이 있기 때문에 적절한 유효기간 설정이 필요합니다.

Access Token + Refresh Token 인증 과정


Client
(A) Authorization Grant
(B) Access Token & Refresh Token
(C) Access Token
(D) Protected Resource
(E) Access Token
(F) Invalid Token Error
Resource Server
Authorization Server
(G) Refresh Token
(H) Access Token & Optional Refresh Token

[참고 자료]

- 링크