Programming

쉽게 알아보는 서버 인증 2편(Access Token + Refresh Token)

이호연 자유로운 오랑우탄 | 2018. 7. 29. 02:23

<26$ JWT 티셔츠 >

안녕하세요! 이전 포스팅에는 크게 세션/쿠키 인증, 토큰 기반 인증(대표적으로 JWT)에 대하여 알아보았습니다. 저희가 앱, 웹 혹은 서버 개발을 하면서 꼭 사용하게 되는 인증(Authorization)은 아주 중요합니다. 만일 설계를 잘못했을 시, 회원들의 정보 유출이 일어날 수도 있으니까요.

이번 포스팅에서는 기본 JWT 방식의 강화버전인 Access Token & Refresh Token 방식에 대해 알아보겠습니다.

Refresh Token?

Access Token(JWT)를 통한 인증 방식의 문제는 만일 제 3자에게 탈취당할 경우 보안에 취약하다는 점입니다.
유효기간이 짧은 Token의 경우 그만큼 사용자는 로그인을 자주 해서 새롭게 Token을 발급받아야 하므로 불편합니다. 그러나 유효기간을 늘리자면, 토큰을 탈취당했을 때 보안에 더 취약해지게 됩니다.
이때 "그러면 유효기간을 짧게 하면서 좋은 방법이 있지는 않을까?"라는 질문의 답이 바로 **"Refresh Token"**입니다.

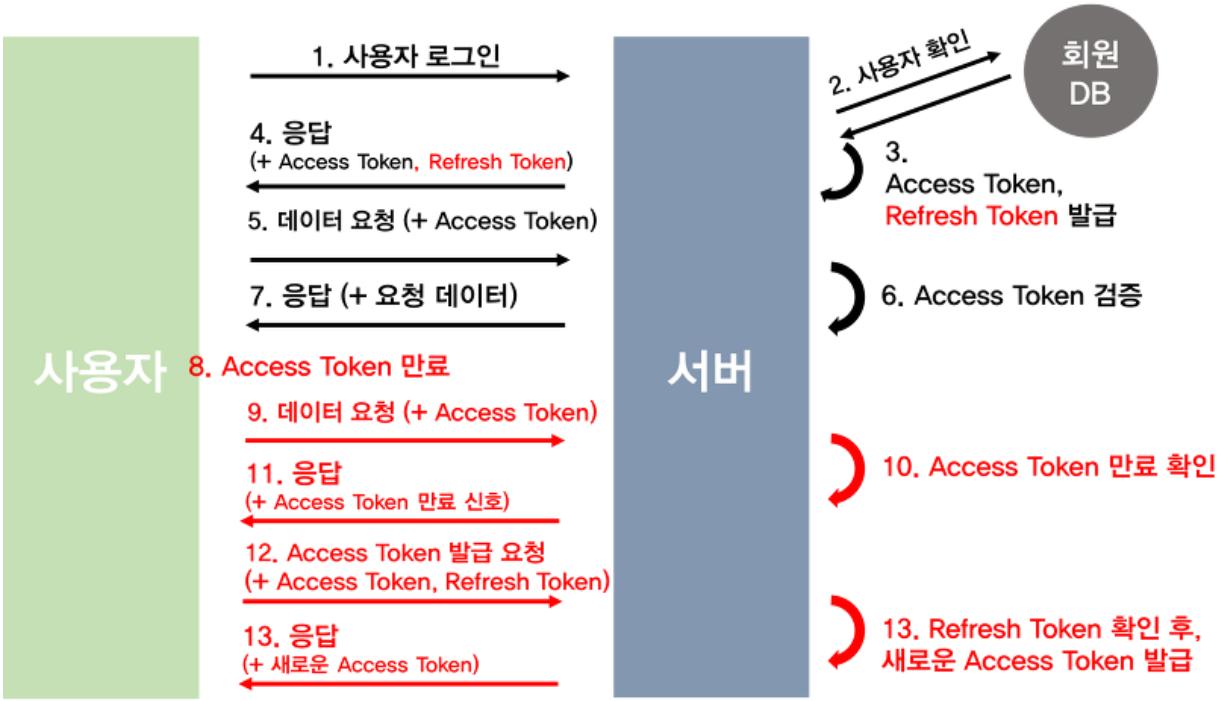
Refresh Token은 Access Token과 똑같은 형태의 JWT입니다. 처음에 로그인을 완료했을 때 Access Token과 동시에 발급되는 Refresh Token은 긴 유효기간을 가지면서, Access Token이 만료됐을 때 새로 발급해주는 열쇠가 됩니다(여기서 만료라는 개념은 그냥 유효기간을 지났다는 의미입니다.)

사용 예를 간단히 들어보겠습니다. Refresh Token의 유효기간은 2주, Access Token의 유효기간은 1시간이라 하겠습니다. 사용자는 API 요청을 신나게 하다가 1시간이 지나게 되면, 가지고 있는 Access Token은 만료됩니다. 그러면 Refresh Token의 유효기간 전까지는 Access Token을 새롭게 발급받을 수 있습니다.

** Access Token은 탈취당하면 정보가 유출되는건 동일합니다. 다만 짧은 유효기간 안에만 사용이 가능하기에 더 안전하다는 의미입니다.

** Refresh Token의 유효기간이 만료됐다면, 사용자는 새로 로그인해야 합니다. Refresh Token도 탈취될 가능성이 있기 때문에 적절한 유효기간 설정이 필요해보입니다(보통 2주로 많이 잡더군요)

Access Token + Refresh Token 인증 과정



이번에는 과정이 좀 복잡합니다. 차근차근 하나씩 알아봅시다. 빨간색은 1편의 JWT 인증방식에서 추가로 들어간 과정입니다.

1. 사용자가 ID , PW를 통해 로그인합니다.

2. 서버에서는 회원 DB에서 값을 비교합니다(보통 PW는 일반적으로 암호화해서 들어갑니다)

3~4. 로그인이 완료되면 Access Token, Refresh Token을 발급합니다. 이때 일반적으로 회원DB에 Refresh Token을 저장해둡니다.

5. 사용자는 Refresh Token은 **안전한 저장소**에 저장 후, Access Token을 헤더에 실어 요청을 보냅니다.

6~7. Access Token을 검증하여 이에 맞는 데이터를 보냅니다.

8. 시간이 지나 Access Token이 만료됐다고 보겠습니다.

9. 사용자는 이전과 동일하게 Access Token을 헤더에 실어 요청을 보냅니다.

10~11. 서버는 Access Token이 만료됨을 확인하고 권한없음을 신호로 보냅니다.

** Access Token 만료가 될 때마다 계속 과정 9~11을 거칠 필요는 없습니다.
사용자(프론트엔드)에서 Access Token의 Payload를 통해 유효기간을 알 수 있습니다. 따라서 프론트엔드 단에서 API 요청 전에 토큰이 만료됐다면 바로 재발급 요청을 할 수도 있습니다.

12. 사용자는 Refresh Token과 Access Token을 함께 서버로 보냅니다.

13. 서버는 받은 Access Token이 조작되지 않았는지 확인한후, Refresh Token과 사용자의 DB에 저장되어 있던 Refresh Token을 비교합니다. Token이 동일하고 유효기간도 지나지 않았다면 새로운 Access Token을 발급해줍니다.

14. 서버는 새로운 Access Token을 헤더에 실어 다시 API 요청을 진행합니다.

Refresh Token이 들어가면서 과정이 좀 복잡해졌습니다. 하지만 Access Token의 약점을 보완해주기 때문에 보안이 중요한 프로젝트에서는 사용하기를 권장합니다.

(큰 장점)

기존의 Access Token만 있을 때보다 안전합니다.

(단점)

1. 구현이 복잡합니다. 검증 프로세스가 길기 때문에 자연스레 구현하기 힘들어졌습니다(프론트엔드, 서버 모두)

2. Access Token이 만료될 때마다 새롭게 발급하는 과정에서 생기는 HTTP 요청 횟수가 많습니다. 이는 서버의 자원 낭비로 귀결됩니다.

저도 현재 프로젝트에 Access Token + Refresh Token 방식을 도입하였습니다. 처음에 구현하기가 좀 힘들 수 있으나 충분히 가치가 있다고 봅니다.

다음 포스팅에서는 페이스북, 네이버 로그인에 쓰이는 Oauth에 대해 알아보도록 하겠습니다. Oauth도 Access Token + Refresh Token 방식으로 진행되기 때문에 어렵지 않게 이해하실 수 있을 겁니다. 짜이찌엔!

[참고]

현재 그랩이라는 닉네임으로 크리에이터 활동을 하고 있습니다. 많은 관심 부탁드립니다 :)