2023.2.21 스터디 주제 발표

> HTTP 는 Stateless (상태가 없는) 통신 프로토콜이라고 합니다. 따라서, 상태가 없다면 가령 HTTP 를 쓰는 서비스는 매번 로그인을 해 줘야 하거나 사용자 정보를 저장하는 일이 불가능합니다. 그런데 실제론 그렇지 않죠. 어떻게 이런 불편함을 해소했을까요?

HTTP의 특징

- 비연결성
  - 클라이언트가 요청을 하면 서버는 적합한 응답을 하면서 연결을 끊는 성질
- 무상태
  - 비연결적인 특성으로 연결이 해제되면 서버는 클라이언트가 이전에 요청한 결과에 대해 잊어버린다.
  - 클라이언트가 이전과 같은 데이터를 원한다고 해도 다시 서버에 연결해 동일한 요청을 시도해야한다

로그인이나 사용자 정보 어떻게 저장하나?

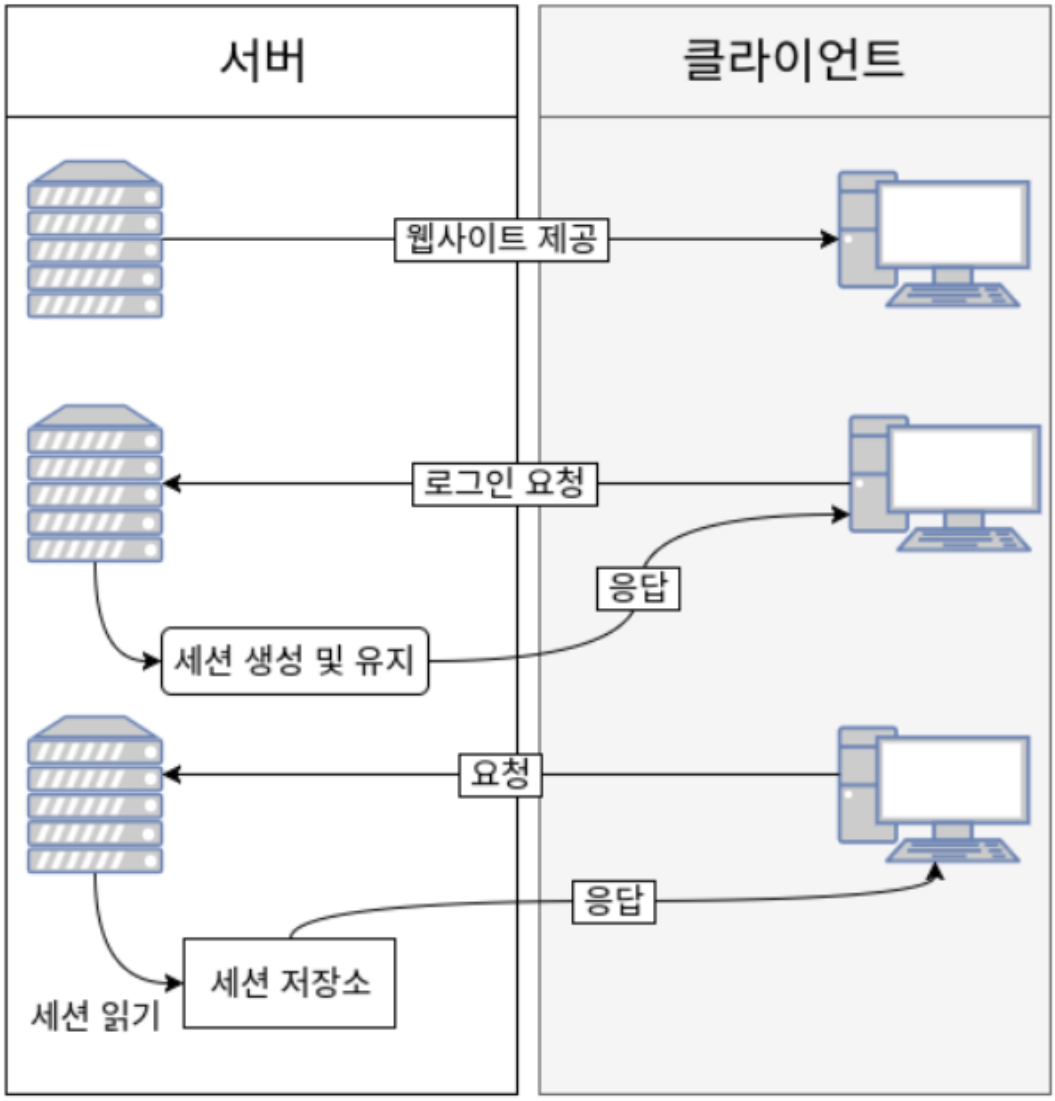
쿠키(Cookie)

Cookie 인증 방식

- 특징
  - 웹 브라우저에 저장(최대 4kb)
  - key - value로 구성
  - 유효시간 설정이 가능하며 특정시간이 지나면 소멸
- 사용 목적
  - 세션관리, 개인화(Facebook 등 광고에서 개인화), 트래킹(GA)
- 단점
  - 네트워크 부하가 있음(쿠키 크기가 클 경우)
  - Header의 Cookie로 전달 되어 중간에 쿠키 탈취가 가능하다.
- 장점
  - 서버에서는 저장공간이 절약된다.

세션(Session)

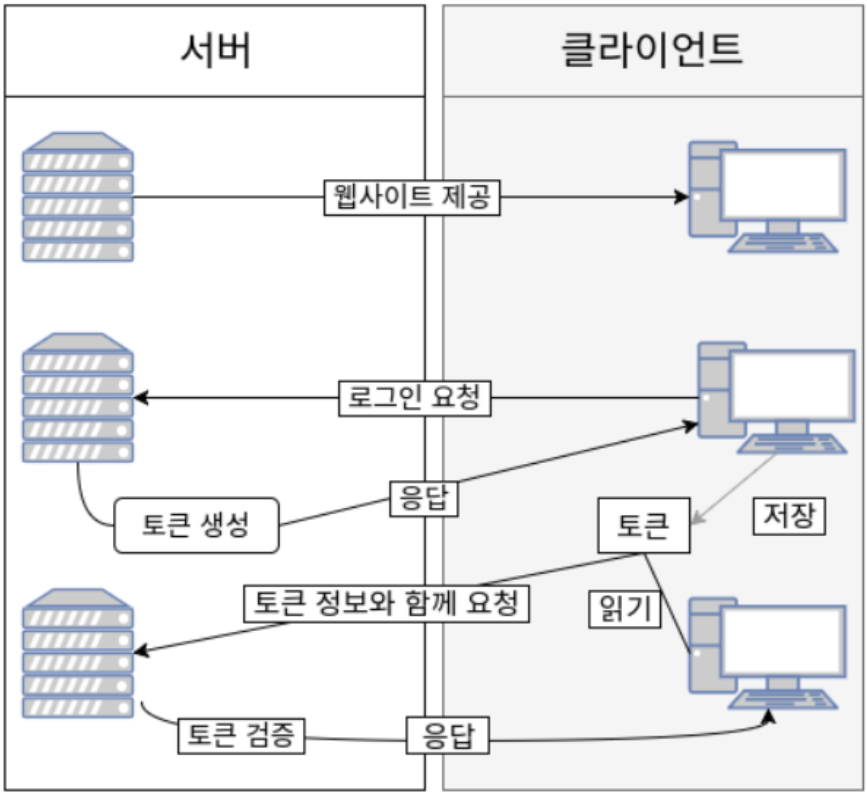
Session 인증 방식



- 특징
  - 서버에 저장되며 고유한 SessionID를 생성한 후 쿠키로 기록
  - 용량 제한 없음
- 장점
  - SessionID만 사용하기에 보안 유지가 된다.(ID 탈취도 가능하지만 IP 특정을 통해 해결가능)
- 단점
  - 서버에서 세션에 대한 데이터를 저장하므로 세션 양이 많을 수록 부하가 커짐(메모리 사용량이 커짐)

JWT(Json Web Tokens)

Token 인증 방식

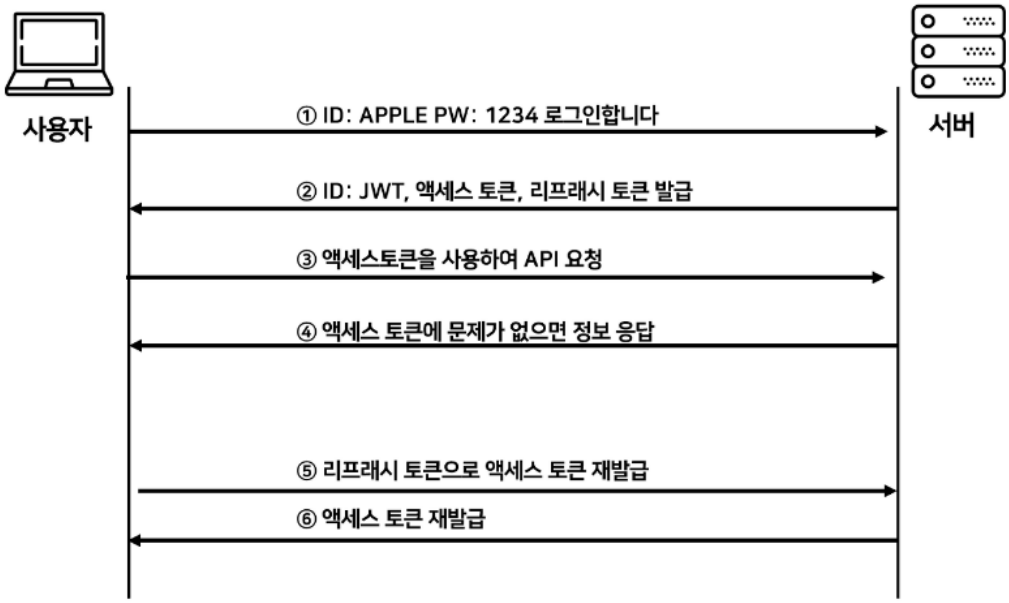


1. 사용자가 아이디와 비밀번호로 로그인을 한다.
2. 서버 측에서 사용자(클라이언트)에게 **유일한 토큰**을 발급한다.
3. 클라이언트는 서버 측에서 전달받은 토큰을 쿠키나 스토리지에 저장해 두고, 서버에 요청을 할 때마다 해당 토큰을 서HTTP 요청 헤더에 포함시켜 전달한다.
4. 서버는 전달받은 토큰을 검증하고 요청에 응답한다.
   토큰에는 요청한 사람의 정보가 담겨있기에 서버는 DB를 조회하지 않고 누가 요청하는지 알 수 있다.

- 특징
  - 정보보호 목적이 아닌 위조 방지의 목적이다.
  - JSON 토큰으로 이루어져 Header, Payload, Signature로 구성되어 있다.

-
- 장점
  - 서버에서 따로 클라이언트를 위한 메모리나 공간을 사용하지 않는다.
  - header와 payload를 가지고 signature를 생성하므로 데이터 위변조를 막는다.
  - 서버 확장성이 우수하다(무상태로 인해)
- 단점
  - 쿠키/세션과 달리 데이터 길이가 길어, 인증요청이 많을 수록 부하가 심해질 수 있음.
  - 토큰을 탈취당하면 대처가 어렵다(사용기간 제한을 통해 극복)

JWT를 이용한 인증 과정



1. 사용자가 ID, PW를 입력하여 서버에 로그인 인증을 요청한다.
2. 서버에서 클라이언트로부터 인증 요청을 받으면, Header, PayLoad, Signature를 정의한다.
   Hedaer, PayLoad, Signature를 각각 Base64로 한 번 더 암호화하여 JWT를 생성하고 이를 쿠키에 담아 클라이언트에게 발급한다.
3. 클라이언트는 서버로부터 받은 JWT를 로컬 스토리지에 저장한다. (쿠키나 다른 곳에 저장할 수도 있음)
   API를 서버에 요청할때 **Authorization header에 Access Token을 담아**서 보낸다.
4. 서버가 할 일은 클라이언트가 Header에 담아서 보낸 JWT가 내 서버에서 발행한 토큰인지 일치 여부를 확인하여 일치한다면 인증을 통과시켜주고 아니라면 통과시키지 않으면 된다.
   인증이 통과되었으므로 페이로드에 들어있는 유저의 정보들을 select해서 클라이언트에 돌려준다.
5. 클라이언트가 서버에 요청을 했는데, 만일 액세스 토큰의 시간이 만료되면 클라이언트는 리프래시 토큰을 이용해서
6. 서버로부터 새로운 엑세스 토큰을 발급 받는다.

참고

https://hyuntaeknote.tistory.com/6

https://hyuntaeknote.tistory.com/3

https://interconnection.tistory.com/74

https://jwt.io/

정리하기

- HTTP는 **비연결성, 무상태**의 특징을 가지고 있다.
- 브라우저는 로그인과 같은 유저 상태를 저장하기위해 3가지 방법을 이용한다.
  - **Cookie, Session, JWT**
- 쿠키와 세션을 비교하고 더 나아간 방식에 대해 말하시오
  - 쿠키는 키-밸류 형식을 가지며 브라우저에 저장됩니다. 브라우저의 정책에 따라 크기가 4kb까지 저장 가능합니다. 서버로 요청 시 헤더정보에 포함되어 노출이 되어 있습니다.
  - 세션은 서버에 필요한 정보들을 저장하며 sessionID를 통해 클라이언트를 찾습니다. 주로 redis나 memcache를 활용합니다. 많은 고객이 유입시 세션 서버의 메모리를 점검해야 합니다.
  - 쿠키와 세션 모두 인가를 위해 처리하는 작업이다보니 서버가 light해질 수 있는 방안을 모색했습니다.

    그 결과 JWT가 등장했습니다. JWT 토큰을 통해 인가를 위한 간단한 정보를 payload로 저장하고 서명을 통해 validation을 확인합니다. 서버는 매 요청에 따라 추가 로직처리 없이 정보를 꺼내 활용할 수 있습니다. 다만 요청 데이터가 크다는 단점이 있습니다.