**JWT 토큰 보안 Local storage와 cookie 비교**

**1. JWT(JSON Web token)**

Json web token의 약자로 모바일이나 웹의 사용자 인증을 위해 사용하는 암호화된 토큰을 의미한다. 현재 프로젝트에서는 (약칭: Accestoken ) JWT 정보를 담아 사용자의 인증을 확인하여 게시판의 수정, 삭제, 쓰기 등을 수행하고 있다.

**2. XSS(Cross Site Scripting)**

XSS라고 불리는 이유는 CSS가 이미 약자가 있기 때문이고, code injection attack 이라고도 한다.   
XSS란 웹 해킹 공격 중 하나로서, 게시판이나 웹 메일 등에 자바 스크립트와 같은 코드를 삽입 해 개발자가 고려하지 않은 기능이 작동하게 하는 치명적인 공격이다. XSS 공격은 크게 Relected XSS와 Stored XSS 그리고 DOM based XSS 세가지로 분류할 수 있다.

**3. CSRF(Cross site Request Forgery)**

정상적인 requests를 가로채 백엔드 서버에 변조된 request를 보내서 악의적인 동작을 수행하는 공격을 의미한다. (피해자 정보 수정, 정보 열람) 예를 들어, 본인이 쓰지 않는 광고성 글이 페이스북에 올라가는 식의 공격.

대략적인 공격 과정은 다음과 같다.  
1. 공격자는 유저가 img를 열람하도록 하거나, link를 클릭하도록 유도한다.   
2. 이 action은 사용자 의도와는 관계없이 http request를 보낸다  
3. 유저가 로그인이 되어 있는 상태라면(필수), 이 request 는 정상적으로 서버에 동작을 수행한다.

정리하자면,   
XSS는 공격 대상이 Client 이고,   
CSRF는 공격 대상이 Server로 보면 된다.   
쉽게 말하면 XSS는 스크립트를 이용한 공격  
CSRF는 내용 자체 변조를 통한 공격이다.

위와 같은 공격을 당하지 않고 보안을 강화하기 위해 JWT같은 개인 정보를 저장할 수 있는 방법에는 2가지 정도가 존재한다. -> **Local storage, cookie**

**1.** **LocalStorage에 저장**

**장점: CSRF 공격에는 안전하다.**   
그 이유로는 자동으로 request에 담기는 쿠키와 다르게 js 코드에 의해 헤더에 담기므로 XSS를 뚫지 않는 이상 공격자가 사용자인 척 request를 보내기가 어렵다.

**단점: XSS 취약하다.**공격자가 LocalStorage 에 접근하는 JS 코드 한 줄만 주입하면 localStorage를 공격자가 내 집처럼 드나들 수 있다

**2. cookie에 저장**

**장점: XSS 공격으로부터 localStorage에 비해 안전하다.**1. 쿠키의 httpOnly 옵션을 사용하면 Js에서 쿠키에 접근 자체가 불가능하다. 그래서 XSS 공격으로 쿠키 정보를 탈취할 수 없다. (httpOnly 옵션은 서버에서 설정할 수 있음) HTTP Only는 자바스크립트의 document.cookie 이용해서 쿠키에 접속하는 것을 막는 옵션이다.

2. Secure로 웹 브라우저와 웹 서버가 https로 통신하는 경우에만 웹 브라우저가 쿠키를 서버로 전송하는 옵션도 사용할 수 있다.

\*하지만 XSS 공격으로부터 완전히 안전한 것은 아니다. httpOnly 옵션으로 쿠키의 내용을 볼 수 없다고 해도, js로 request를 보낼 수 있으므로 자동으로 request에 실리는 쿠키의 특성 상 사용자의 컴퓨터에서 요청을 위조만 할 수 있기 때문이다. 즉, XSS가 뚫린다면 httpOnly cookie도 안전하지 않다.

**단점: CSRF 공격에 취약하다**자동으로 http request 에 담아서 보내기 때문에 공격자가 request url만 안다면 사용자가 관련 link를 클릭하도록 유도하여 request를 위조하기 쉽다.

**★결론 best Option**현재 진행하고 있는 프로젝트(Oauth Api gateway)는 refresh token 설정을 해놨기 때문에, refresh token을 httpOnly 쿠키로 설정하고 url이 새로고침 될때마다 refresh token을 request에 담아서 새로운 accessToken을 발급 받는 것이다. 발급 받은 accsessToken을 js private vaiable에 저장한다.

현재 이 부분은 ajax 공통 모듈을 같이 설계하면서 진행할 부분이다. (어디에 어떻게 넣을지)

이런 방식을 사용하는 경우 refresh token이 csrf에 의해 사용된다 하더라도 공격자는 acessToken을 알 수 없다. 요청을 위조하여 피해자가 의도하지 않은 서버 동작을 일으키는 공격 방법이기 때문에 refresh token을 통해 받아온 response(accessToken)은 공격자가 확인할 수 없다.

따라서 쿠키를 사용(httpOnly)하여 XSS를 막고 refresh token 방식을 이용하여 CSRF를 막을 수 있다.[[1]](#footnote-1) 여기서 httpOnly 부분은 서버에서 설정해야 한다. 프론트 쿠키 설정로는 불가능.

**SSL 이란 무엇인가?**

SSL은 Secure Sockets layer의 약자로 웹 표준 암호화 통신으로서 웹 브라우저와 서버 사이에 정보를 암호화해주는 방식이다. SSL과 TLS(Transport Layer Security)는 동일한 용어로 넷스케이프에 의해서 발명된 SSL 이 폭넓게 사용되다가 표준화 기구인 IETF의 관리로 변경 되면서 TLS라는 이름으로 바뀌었다.

따라서 HTTPs는 HTTP Over Secure Socket Layer의 약자로 아래 그림처럼 HTTP 통신이 SSL Layer 위에서 이루어지는 것을 HTTPS라고 한다.

SSL 디지털 인증서클라이언트와 서버가 통신할 때, 클라이언트는 서버로부터 데이터를 받기 전에 해당 서버가 의도한 서버가 맞는지, 즉 신뢰할 수 있는 서버인지 확인해야한다.

이러한 역할을 수행하는 것이 바로 디지털 인증서다. 디지털 인증서란 클라이언트와 서버간의 통신을 제3자가 보증해주는 전자화된 문서이다.

**Http와 Https의 차이->**

HTTP는 하이퍼 텍스트 전송 프로토콜의(Hypertext Transfer Protocol)의 약자.  
HTTPS는 하이퍼 텍스트 전송 프로토콜 보안(Hypertext Transfer Protocol Secure)의 약자

일반 HTTP 프로토콜의 문제점은 서버에서부터 브라우저로 전송되는 정보가 암호화되지 않았으나, HTTPS 프로토콜은 SSL(보안 소켓 계층)을 사용함으로써 이 문제를 해결했다.   
(현재 진행하고 있는 프로젝트는 http에서 https로 변경 되어있다.)

또한 HTTPS 프로토콜은 SSL(보안 소켓 계층)을 사용하여 이를 구현했는데, SSL은 서버와 브라우저 사이에 안전하게 암호화된 연결을 만들 수 있게 도와주고, 서버 브라우저가 민감한 정보를 주주 받을 때 이것이 도난 당하는 것을 막아준다.

1. 2021년 8월 4일(팀장님) 회의를 통한 결론으로는 현재 프로젝트에서 저장하는 정보가 session id AccessToken만 존재하고, https도 사용하고 있기 때문에 local storage 와 cookie의 보안 차이가 엄청 많이 나지 않을 것으로 판단 된다고 하심. [↑](#footnote-ref-1)