2017년 1학기 네트워크 기초 과제

**SSL / TLS 프로토콜**

게임공학과 2015182014 성예솔(목요일 10~12교시)

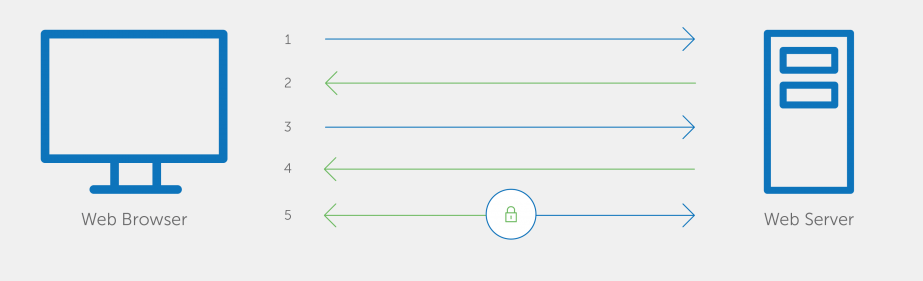
SSL(Secure Socket Layer) 은 인터넷을 이용한 정보 보안의 안전한 거래를 위해서 Netscape 회사에서 만든 인터넷 통신 규약 프로토콜이다. 이에 기초로 해서 IETF에서 SSL보다 안전하고 정확한 TSL(Transport Layer Security)을 만들었다. 전송계층에서 서버와 클라이언트를 상대로 인증과 암호화를 하는 보안용 프로토콜들이다. 응용계층과 전송계층 사이에서 안전한 보안채널을 형성해주는 역할을 한다. 인터넷으로의 의사소통에서 악의적인 코드 삽입을 방지 할 수 있다.

SSL (Secure Sockets Layer)은 서버 및 클라이언트 간에 암호화 된 링크를 설정하기위한 표준 보안 기술이다.

SSL은 신용 카드 번호, 사회 보장 번호와 같은 민감한 정보를 안전하게 전송하기 위해 접근 가능한지를 판단하고 전송한다.

SSL 프로토콜은 전송되는 링크와 데이터 모두에 대한 암호화의 변수를 결정한다.

SSL은 특히 온라인 거래 중 또는 기밀 정보를 전송할 때, 매일 인터넷에서 사람들의 수백만 데이터를 보호한다. SSL 보안 웹 사이트는 http가 아닌 https로 시작한다.



SSL(Secure Socket Layer)의 작동 과정

1. [웹브라우저] 브라우저는 웹사이트에 연결된다. SSL로 암호화된 페이지에 서버가 자신을 식별하도록 요청한다.
2. [웹서버] Public Key를 인증서 사본과 함께 전송한다.
3. [웹브라우저] 인증서가 믿을 만하다고 판단한 CA(Certificate Authority)로부터 서명된 것인지 확인하고 만료나 손상되지 않았으면 브라우저가 서버의 공개키를 사용하여 랜덤 대칭 암호화키를 생성, 암호화 하여 다시 보낸다.
4. [웹서버] Private Key를 이용해서 랜덤 대칭 암호화키와 URL, http 데이터를 해독하고 웹 브라우저로부터 받은 랜덤 대칭 암호화키를 이용하여 다시 웹 브라우저로 전송한다.
5. [웹브라우저,웹서버] 전송 된 모든 데이터를 세션 키로 암호화 한다.

IETF는 몇 차례 반복을 통해서 프로토콜을 정의하였고, 대부분의 브라우저와 HTTPS 에서 웹서비스가 지원된다. TLS 1.2 버전은 만들어 진지 9년이 지났다. 9년동안 이 프로토콜의 결함이 드러나는 몇 가지 공격을 받은 뒤로 IETF는 TLS 1.3 프로토콜을 개발하였다. TLS 1.3은 최근에 인터넷 사용자를 위해서 속도와 보안이 향상된 새로운 암호화 프로토콜이다. 2016년 말에 완성되었고 1.2와 다른 점은 보안이 강화되었고 속도도 향상되었다. TLS1.3은 이전 버전의 프로토콜을 공격한 방법으로는 전혀 영향을 끼치지 못한다. 그리고 1.3 버전은 간소화되어 서버 운영자가 훨씬 간단하게 구성 할 수 있게 되었다. 그리고 훨씬 속도가 빠르고 안전한 웹 브라우징 환경을 제공한다.



TLS(Transport Layer Security)의 작동 과정

암호화 된 사이트로 메시지를 보내려면 먼저 공유 암호 키를 설정해야 한다. 이 프로세스를 암호화 핸드 셰이크라고 한다. 브라우저와 웹 서버 간에 특별한 메시지를 주고받는다. TLS 핸드 셰이크는 브라우저로 암호화 된 사이트에 연결할 때마다 뒤에서 발생한다. TLS 1.2를 사용하면 요청을 보내기 전에 핸드 셰이크를 완료하려면 브라우저와 웹서버간 왕복을 두 번이나 해야 했다. 이에 반해, TLS 1.3은 초기 핸드 셰이크가 절반으로 줄어들어 단 한 번만 왕복을 하면 된다.

1. 클라이언트에서 서버로 ClientHello 메시지를 보낸다. 여기에는 클라이언트에서 가능한 TLS 버전, 세션 식별자, 암호 설정 등의 정보가 포함된다.

2. 클라이언트의 메시지를 받은 서버는 ServerHello 메시지를 클라이언트에게 보낸다. 여기에는 ClientHello 메시지의 정보 중 서버에서 사용하기로 선택한 TLS 버전, 세션 식별자, 암호 설정 등의 정보가 포함된다.

3. 서버가 클라이언트에 Certificate 메시지를 보낸다. 여기에는 서버의 인증서가 들어간다. 이 인증서는 별도의 인증 기관에서 발급받은 것이며, 서버가 신뢰할 수 있는 자임을 인증한다. 전송이 끝나면 ServerHelloDone 메시지를 보내 끝났음을 알린다.

4. 클라이언트는 서버에서 받은 인증서를 검증한다. 인증서의 유효 기간이 만료되지 않았는지, 그 인증서가 해당 서버에게 발급된 인증서가 맞는지 등을 확인한다. 인증서를 신뢰할 수 있다고 판단하였다면 다음 단계로 넘어간다.

5. 클라이언트는 임의의 pre-master secret을 생성한 뒤, 서버가 보낸 인증서에 포함된 공개 키를 사용해 암호화한다. 이렇게 암호화된 pre-master secret을 ClientKeyExchange 메시지에 포함시켜 서버에 전송한다.

6. 서버는 전송받은 정보를 복호화하여 pre-master secret을 알아낸 뒤, 이 정보를 사용해 master secret을 생성한다. 그 뒤 master secret에서 세션 키를 생성해내며, 이 세션 키는 앞으로 서버와 클라이언트 간의 통신을 암호화하는데 사용할 것이다. 물론 클라이언트 역시 자신이 만들어낸 pre-master secret을 알고 있으므로, 같은 과정을 거쳐 세션 키를 스스로 만들 수 있다.

7. 서버와 클라이언트는 각자 동일한 세션 키를 가지고 있으며, 이를 사용해 대칭 키 암호를 사용하는 통신을 할 수 있다. 따라서 우선 서로에게 ChangeCipherSpec 메시지를 보내 앞으로의 모든 통신 내용은 세션 키를 사용해 암호화해 보낼 것을 알려준 뒤, Finished 메시지를 보내 각자의 핸드셰이킹 과정이 끝났음을 알린다.