Web Server에서 HSM을 사용해야 하는 가 ?

가업이나 기관에서 운영하는 Web Server(웹서버: 일반적으로 기업/기관의 홈페이지 또는 웹사이트라고도 부르고 있음)에 접속하기 위하여 사용자는 인터넷 익스플로러(Internet Explorer)나 크롬(Crome) 같은 Web Browser를 사용합니다. Web Server 와 Web Browser 사이의 통신은 HTTP 나 HTTPS 프로토콜을 사용하여 data를 주고 받습니다. HTTPS 프로토콜은 data를 암호화시킨 후 주고 받습니다. 주고 받는 data속에 개인의 비밀정보나 금융정보나 같은 것이 들어 있으면, 인터넷상에 노출이 되지 않도록 암호화 시켜 통신을 해야 하므로 HTTPS 프로토콜을 사용합니다. 이러한 이유로 HTTPS를 지원하는 Web Server를 보안서버라고 부르기도 합니다.

IETF(국제인터넷기술위원회)가 2015년 5월 차세대 웹 표준 통신으로 HTTP/2를 발표(RFC 7540)함에 따라 HTTP/2를 도입하는 업체가 증가하는 추세에 있습니다. HTTP/2의 2가지 목표(속도 와 안전성)중 하나가 안정성이므로, HTTP/2 표준자체는 암호화된 트래픽만을 요구하지는 않으나, 대부분의 Web Browser들은 암호화된 트래픽만을 지원할 것이라고 선언 하였으므로, HTTP/2 시대를 맞이하여 향후 모든 트래픽은 모두 암호화 트래픽으로 전환될 것으로 여겨지고 있습니다. 대부분의 Web Browser들도 2015년 말까지 HTTP/2를 지원하기로 하였다고 합니다.

HTTPS 프로토콜은 암호화 통신을 위하여 SSL/TLS 라는 암호 프로토콜을 사용합니다. 따라서 HTTPS상의 트래픽을 SSL 기반 암호화 트래픽이라고 부릅니다. SSL/TLS 의 기능을 간단히 설명하면, Web Bowser가 Web Server에 접속하면, 서로 Hello Message를 주고 받습니다. Hello Message 안에는 앞으로 암호통신을 하기 위한 정보가 들어 있으며, 특히 암호키를 만들기 위한 Random Number도 포함되어 있습니다. 두번째 단계는 Web Server가 신뢰하는 기관에서 인정한 신뢰받는 Site 인지를 먼저 확인 시켜 줍니다. 즉 Web Browser는 Web Server로부터 Server Certificate를 받아서 이 Certificate를 검증합니다. 신뢰받은 CA기관에서 발행한 Certificate이면 그 다음 단계로 넘어가지만, 그렇지 않으면 사용자에게 경고 메시지를 보여 줍니다(신뢰할 수 없는 Site 이므로 사용자가 계속 접속을 할 것인지는 스스로 판단하라고...), Server Certificate 안에는 Web Server의 Public-Key(공개키)가 들어 있습니다. 세번째 단계는 사용자의 Web Browser는 서로간의 암호통신을 위하여 2개의 Random Number(자신이 만든 Random Number 와 Web Server에서 받은 Random Number)로 암호키(기능상 session-key라고 부릅니다)를 만들어 Web Server의 Public-Key로 암호화 시켜 Web Server로 보냅니다. Web Server는 자신이 보유하고 있는 Private-Key로 해독하여 암호키(Session-key)를 알게 됩니다. 이로인해 Web Browser 와 Web Server는 통신을 위한 동일한 암호키(Session-Key)를 보유하게 되므로, 이후부터 일어나는 모든 통신은 서로 공유한 암호키(Session-Key)로 암호화시켜 통신을 합니다.

위의 방식의 문제점은 누군가 인터넷상의 모든 Web Traffic을 Capture한 다음에(스노든은 미국 NSA는 Network Backbone을 해킹 하였다고 폭로 하였습니다), 특정 Web Server의 Private-Key를 획득하면, 통신시 사용한 암호키(Session-Key)를 해독할 수 있으므로 특정 Web Server와 통신한

모든 내용을 볼 수 있게 됩니다. 만약 금융기관이라고 한다면, 그 금융기관에 접속한 모든 사용자의 비밀정보 및 금융정보가 해커에 손에 들어가게 된다는 것입니다. 이러한 문제점을 피하기 위하여 나온 해결책으로 암호키(Session-Key)를 다른 방식으로 공유하는 방법인데,. Diffie-Hellman 방식이라고 부릅니다. Web Server의 Public-Key로 암호화 하는 대신에 각각 DH Parameter를 서로 보내어 암호키(Session-Key)를 공유하는 방법입니다(Diffie-Hellman 방식은 Man-In-The-Middle 공격에 취약하다고 합니다). 또 다른 방법은 PFS(Perfect Forward Secrecy) 방식인데 현재의 Private-Key가 유출 되더라도, 지난 Session의 암호키(session-Key)는 해독할 수 없게 만드는 구조입니다. 이론적으로 Public-Key를 Session당 Random하게 만드는 방식입니다. 실제로 PFS를 지원하는 알고리즘은 ECDHE-RSA 와 DHE-RSA가 있으며, 단점은 CPU 성능을 더 필요로 한다고 합니다. 아직까지 PFS 방식을 사용하는 Web Server는 많지 않다고 합니다.

암호키(Session-Key) 공유 방식을 Web Server의 Public-Key를 사용할 경우, HSM 장비 사용을 고려하곤 합니다. 주로 3가지 목적으로 사용합니다.

첫째, Web Server의 Private-Key를 안전하게 보존하기 위해서 입니다. HSM 장비를 사용하지 않을 경우는 Private-Key는 Web Server상의 어딘가에 보존되어 있을 것이며 또한 메모리상에 존재하게 됩니다. 2014년도에 발생한 Heartbleed(원격지에서 특정 서버의 메모리 내용을 볼 수 있기 때문에 암호키 같은 민감한 정보가 유출될 수 있는 길을 열어주는 Bug)같은 Web Server 시스템의 Security Bug가 발생하더라도, HSM 장비로 Private-Key를 보관하고 있으면 피해를 보지 않게 됩니다. HSM 장비에 Private-Key를 보존할 경우는 Private-Key는 HSM 장비를 절대로 떠나지 않으며, Private-Key를 사용한 해독 연산 자체도 HSM 장비 내에서만 이루어지기 때문입니다. 이러한 목적으로 외국의 은행들은 자사의 Web Server의 Private-Key보존을 위해 HSM 장비를 사용하고 있다고 합니다. 사용하는 암호 알고리즘도 RSA에서 ECC로 전환되는 과정에 있다고 합니다(컴퓨팅 성능의 발전으로 기존 2048 비트 RSA 키는 해독이 될 가능성이 있다고 하며, .4096 비트를 사용하면 너무 속도가 늦어지기 때문이라고 합니다). 요즈음 출시되는 HSM 장비는 ECC 알고리즘을 지원하고 있습니다. HSM 장비를 사용하여 Private-Key를 보관하면, Private-Key가 누출될 가능성이 없으므로, 굳이 PFS 방식을 사용하지 않더라도 암호키(Session-Key)의 해독을 방지 할 수 있습니다. 즉 PFS 방식은 HSM 장비를 사용하지 않을 경우에 필요한 솔루션이라고 생각합니다..

둘째, Web Server에서 동시에 접속하는 수많은 사용자를 지원하는 데, 문제가 없도록, Private-Key로 암호키(Session-Key)를 해독하는 기능을 전용 HSM 장비에 맡기는 것입니다, Private-Key로 연산하는 데는 CPU 자원을 많이 사용하기 때문입니다. 또한 Web Server의 CPU 부하를 줄일 수 있으므로 더 많은 동시 사용자를 지원할 수 있게 됩니다. 이것을 SSL Acceleration 기능이라고 부릅니다.

셋째, 암호키(Session-Key)를 만들기 위한 Random Number를 만들 때 엔트로피가 높은 Random Number를 생성하기 위해서 입니다. 엔트로피가 높은 Random Number란 유추하기가 더 어렵다라는 의미입니다. 즉 만들어진 암호키(Session-Key)가 유추하기가 어렵게 된다는 의미입니다. S/W 암호모듈을 만들 때 가장 어려운 부분이 난수발생기(Random Number Generator)의 구현이라고

합니다. 좋은 성능은 가진 HSM은 H/W Seed(예측 불가능한 잡음 Source)를 기반으로 truly Random Number를 생성 할 수 있다고 합니다.