인터넷에서 사용되는 여러 기술 : HTTPS 1

조인준
KBS 미디어기술연구소 차장

지난 두 편을 통해 REST API가 무엇이며 어떻게 동작하는지에 대해 알아보았습니다. 그 이전에 설명해드린 HTML까지 포함하면 HTTP(Hypertext Transfer Protocol)를 기반으로 우리가 인터넷을 통해 매일 이용하는 것들이 어떤 방식으로 동작하는지의 밑그림이 그려졌을 것 같습니다. 이번 편부터는 HTTP의 보안 강화 버전이라고 볼 수 있는 HTTPS(Hypertext Transfer Protocol Secure)에 관해 알아보겠습니다.

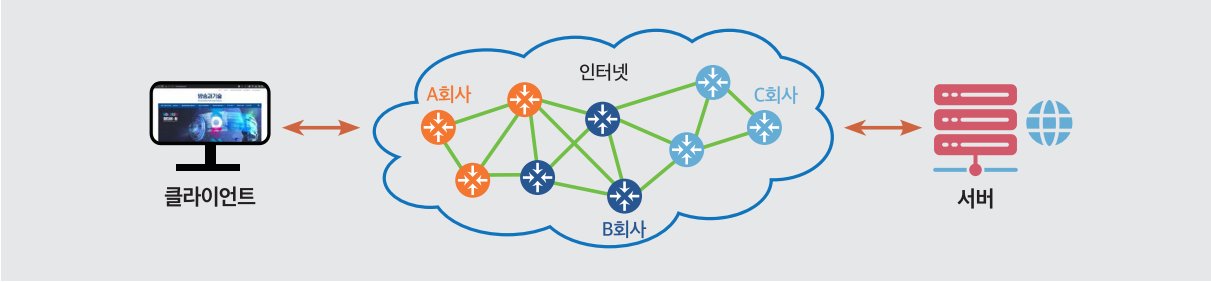

그림 1. 인터넷과 HTTP 기반 정보 전달

우리가 [그림 1]과 같이 웹 사이트에 접속하게 되면 대부분의 경우 HTML이나 JSON으로 된 텍스트 정보를 HTTP로 주고받게 됩니다. 우리가 사용하는 웹브라우저 등의 클라이언트는 HTTP 요청(Request)을 인터넷상의 서버에 보내고, 서버는 클라이언트의 요청을 처리하여 HTTP 응답(Response)을 보내줍니다. 하지만 'HTTP 요청' 및 'HTTP 응답' 모두 인터넷상의 라우터에 보내지고 나면 [그림 1]의 A회사, B회사, C회사로 표시된 ISP(Internet Service Provider)들의 라우터를 거치며 목적지에 도착할 때까지 클라이언트나 서버가 무슨 일이 일어나고 있는지 알거나 이에 대해 할 수 있는 일은 없습니다.

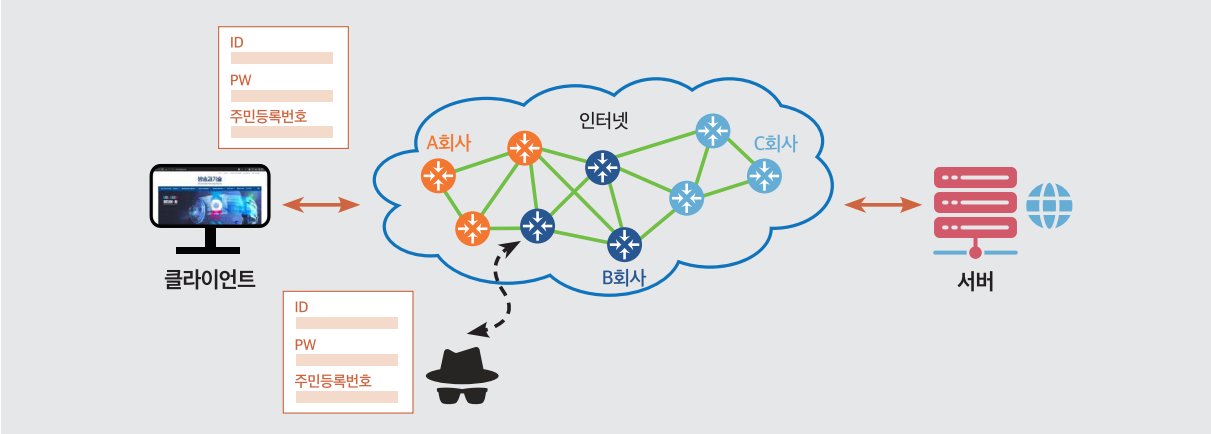

그림 2. 인터넷에서의 정보 감청

[그림 2]와 같이 거래를 위해 특정 웹사이트에 HTML 등의 텍스트 정보로 ID, PW, 주민등록번호 등의 민감한 정보를 보내야 하는 경우 악의를 가진 누군가가 중간에서 데이터를 볼 수 있다면 어떻게 될까요? 일단 보내지고 나면 중간 경로에서 일어나는 일들에 대해 어찌할 방법이 없는 인터넷에서 누군가 내 ID, PW, 주민등록번호 등을 악용하는 일을 막으려면 어떻게 해야 할까요? 이를 위해 만들어진 것이 HTTPS입니다.

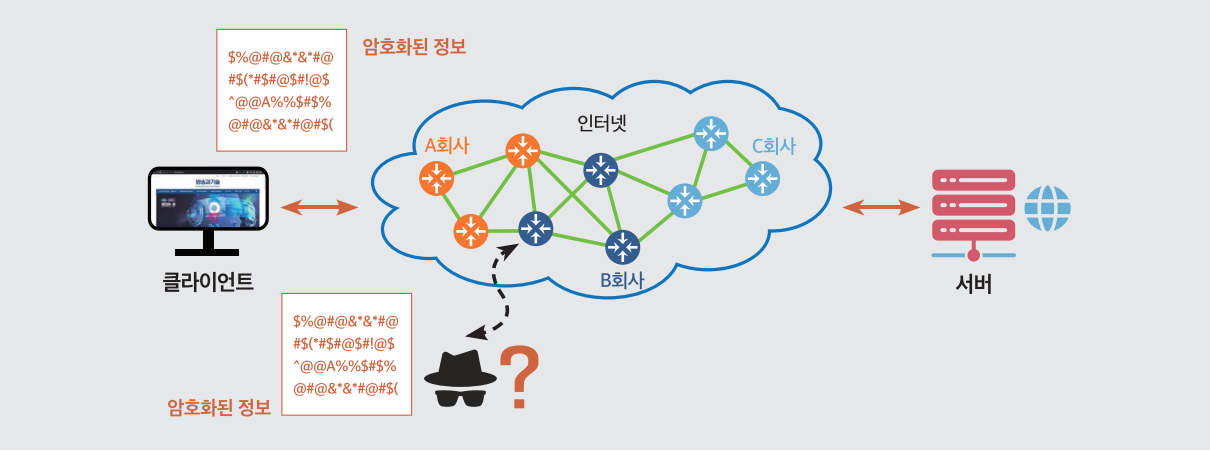

그림 3. 암호화를 통한 인터넷에서의 정보 감청 방지

HTTPS는 [그림 3]의 알 수 없는 문자열과 같이 HTTP에 SSL/TLS라는 기술을 적용하여 전송되는 정보에 암호화를 제공함으로써 보안을 강화한 버전입니다. 암호화를 하면 전송 중 누군가 정보를 빼내더라도 해독이 가능하지 않다면 그 의미를 알 수 없으므로 중요한 정보의 악용을 막을 수 있습니다. 우선 새롭게 소개된 용어인 SSL/TLS에 대한 간략히 짚고 넘어가겠습니다. SSL은 Secure Socket Layer의 약자로 1994년 넷스케이프(Netscape)에 의해 개발된 인터넷 보안 기술입니다. 넷스케이프가 1999년 국제 인터넷 표준화 기구 IETF(Internet Engineering Task Force)로 SSL의 관리를 이전하면서 TLS(Transport Layer Security)로 명칭이 바뀌었으나 예전 명칭인 SSL이 계속 사용되면서 TLS만 사용하거나 SSL/TLS로 병기하기도 합니다.

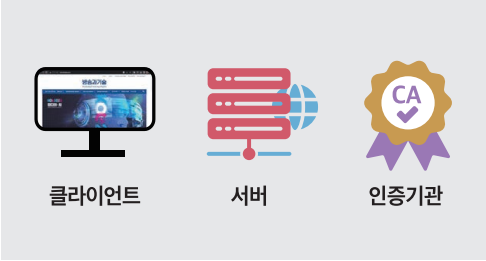

그림 4. SSL/TLS를 이용한 인터넷 보안통신의 주요 주체

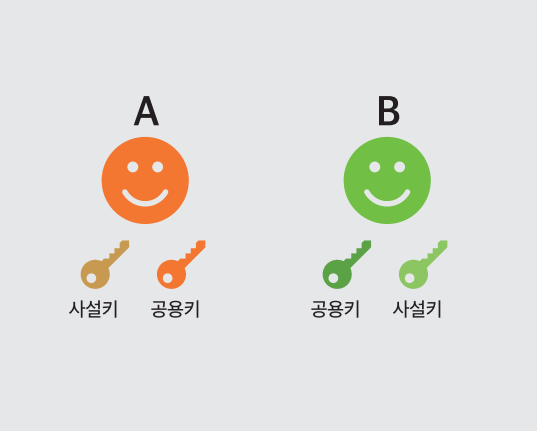

그림 5. 비대칭키 쌍(Asymmetric Key Pair)

SSL/TLS의 이해를 위해서는 클라이언트, 서버, 인증기관(CA : Certificate Authority)의 상호작용에 대한 이해가 필요하고 이를 위해서는 공용키(Public Key)와 사설키(Private Key)를 이용한 비대칭키 쌍(Asymmetric Key Pair) 암호화(Encryption)에 대해 알아야 합니다. [그림 5]와 같이 A와 B가 각각 공용키와 사설키 쌍을 하나씩 가지고 있다고 가정합니다. 공용키는 말 그대로 누군가 필요할 때 공개해 줄 수 있는 키이고 사설키는 타인에게 공개 안 하고 자신만이 알고 있는 키입니다. A와 B가 서로 다른 키 쌍을 가지고 있을 때 이를 비대칭키 쌍이라고 합니다. 공용키와 사설키 쌍은 둘 중 하나로 암호를 걸은 경우 다른 하나로 그 암호를 풀 수 있는 세상에서 유일한 쌍입니다.

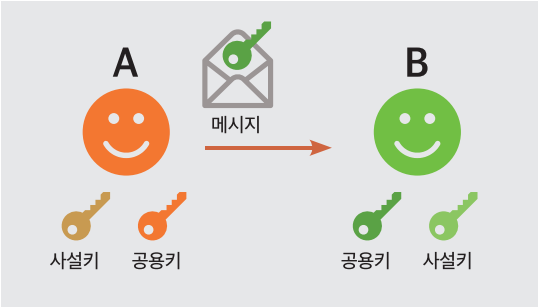

그림 6. 비대칭키 쌍을 이용한 메시지 암호화

• 암호화(Encryption)

A가 B만 알아볼 수 있게 어떤 메시지를 보내고 싶을 때 A는 B의 공용키를 이용해 자신의 메시지를 암호화할 수 있습니다. 이렇게 암호화된 A의 메시지는 B의 사설키로만 풀어 볼 수 있습니다. 이는 다른 말로 이 메시지를 해독해서 그 내용을 확인할 수 있는 사람은 세상에서 B밖에 없다는 것이 됩니다. 이를 인터넷 보안에서는 기밀성(Confidentiality)이라 합니다.

• 서명(Signature)

A가 B나 다른 사람에게 메시지를 보내고 이것이 자신이 보낸 것임을 보장하기 위한 목적으로 자신의 사설키로 암호화한 메시지를 보낼 수 있습니다. A가 사설키로 암호화한 메시지는 A의 공용키로밖에 해독이 불가하므로 A의 공용키로 해독되어 정상적인 메시지가 나온다면 이는 A가 보낸 것이 맞습니다. 이를 인터넷 보안에서는 신원인증(Authentication)이라고 합니다. 또한, A의 공개키로 해독된 메시지가 정상적이라는 것은 중간에 누군가 메시지를 가로채서 내용을 수정하지 않았다는 증명도 됩니다. 만약에 A가 어떻게 암호화했는지 모르는 누군가가 암호화된 상태로 메시지를 수정한다면 A의 공용키로 해독한 내용은 알아볼 수 없는 메시지가 될 것이기 때문입니다. 이를 인터넷 보안에서는 무결성(Integrity)이라고 합니다.

그림 7. 비대칭키 쌍을 이용한 서명

그림 8. 대칭키(Symmetric Key)

위와 같이 비대칭키 쌍을 이용하면 암호화와 서명을 통해 기밀성, 신원인증 및 무결성 세 가지 보안요소를 달성할 수 있습니다. 하지만 현실에서 이를 구현하는 것에는 문제가 있습니다. 비대칭키 쌍을 이용한 암호화 및 이의 해독은 CPU 자원을 많이 소모합니다. 이렇다 보니 대량의 데이터에 적용할 경우 시스템의 연산 자원 및 속도에 문제가 발생합니다. 그렇다면 암호화를 이용한 인터넷 보안은 그림의 떡인가요? 그렇지는 않습니다. 궁하면 통한다고 우리에겐 [그림 8]과 같이 비교적 간단한 연산으로 대량의 데이터를 암호화하고 해독할 수 있는 대칭키(Symmetric Key) 방식이 있습니다. 대칭키라는 명칭은 A와 B가 같은 키를 소유하기 때문에 붙여진 이름입니다. 그런데 대칭키 방식에서 A와 B가 같은 키를 가지기 위해서는 둘 중의 한 명이 임의로 키를 만들어서 상대방에게 줘야 합니다. 만약 전달 과정에서 이 키가 누군가에게 유출이 된다면 이 키로 암호화한 정보들은 더 이상 비밀이 보장되지 않습니다.

보기 편하게 정리하면 [표 1]과 같이 비대칭키 방식은 보안이 우수하지만 연산 속도에 문제가 있고, 대칭키 방식은 연산 속도에 장점이 있지만 유출이 되면 무용지물이 될 수 있습니다.

구분	비대칭키	대칭키
단점	연산속도 느림 암호문이 원문보다 사이즈가 큼	보안성 약함
장점	보안성 강함	연산속도 빠름 암호문과 원문의 사이즈가 같음

표 1. 비대칭키 vs 대칭키 비교

비대칭키와 대칭키를 놓고 어떻게 하면 실용적 보안 기술을 마련할 수 있을까하는 고민 끝에 [그림 9, 10]과 같은 기술이 만들어졌습니다. [그림 9]에서 A는 대칭키를 임의로 생성한 후 이를 B의 공용키로 암호화하여 B에게 전달합니다. A가 생성한 대칭키의 데이터양은 크지 않으므로 비대칭키로 암호화해도 크게 문제 될 것이 없습니다. B의 공용키로 대칭키를 암호화했으므로 이 대칭키를 복원할 수 있는 키는 B의 사설키가 세상에서 유일합니다. 누군가 중간에 대칭키를 가로채도 복원할 방법이 없으므로 B에게 안심하고 보낼 수 있습니다.

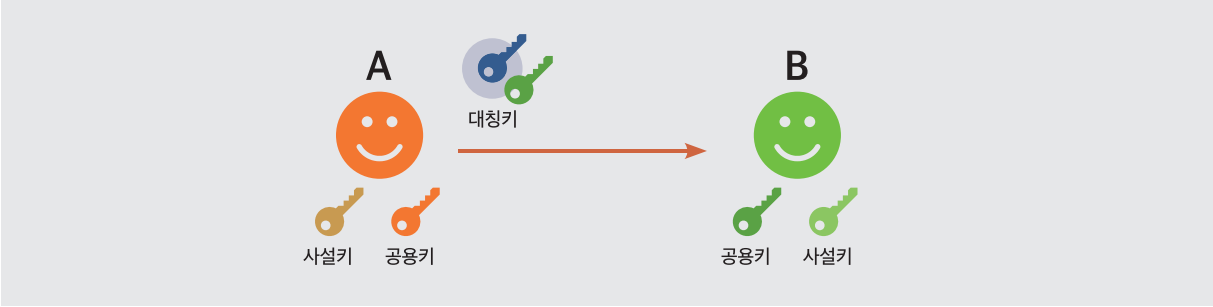

그림 9. 비대칭키를 이용한 안전한 대칭키 전달

A가 생성하여 B의 공용키로 암호화한 대칭키를 받은 B가 자신의 사설키로 대칭키를 복원하면 [그림 10]과 같이 양쪽 모두 안전하게 대칭키를 소유하게 됩니다. 이 대칭키로 암호화한 것들은 이 대칭키로 밖에 해독이 불가하므로 이제부터 A와 B 사이에 오가는 메시지는 누군가 가로채도 해독할 수 없습니다.

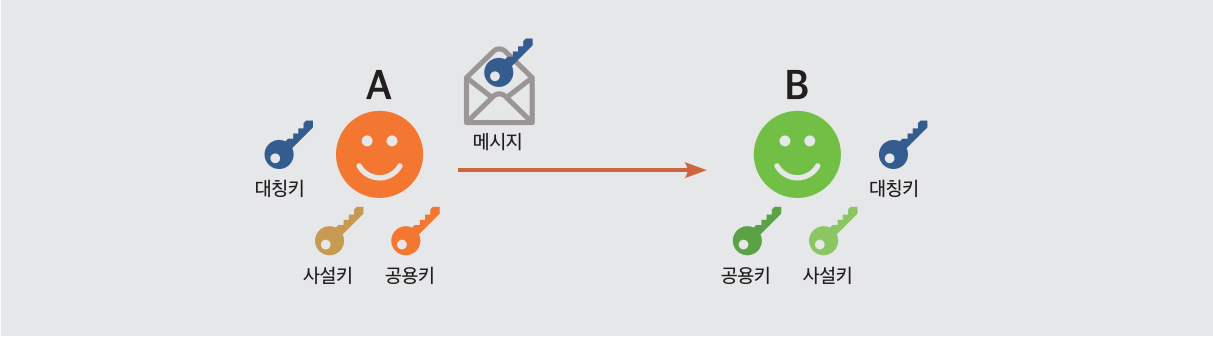

그림 10. 안전하게 공유된 대칭키를 통한 메시지 암호화

이렇듯 대칭키를 전달하는데 비대칭키 방식을 사용하여 높은 수준의 기밀을 유지하고 이렇게 공유된 대칭키로 대량의 데이터를 암호화하여 연산 속도 문제를 해결한 방식이 SSL/TSL 기술이 사용하는 방식입니다. 이후의 연재를 통해 위 개념이 어떤 방식으로 SSL/TSL에서 동작하고 이를 기반으로 HTTPS가 구현되는지에 관해 차례차례 전해드리겠습니다.

P.S.

C군이 여러분께 전하는 내용 중 전문적 성격이 짙은 것은 엄밀한 언어를 사용하여 설명하기에는 한계가 있습니다.
본 내용은 설명하는 대상에 대한 전체적 맥락의 이해에만 이용하시고, 그 이상은 권위 있는 전문자료를 참고하시기 바랍니다.