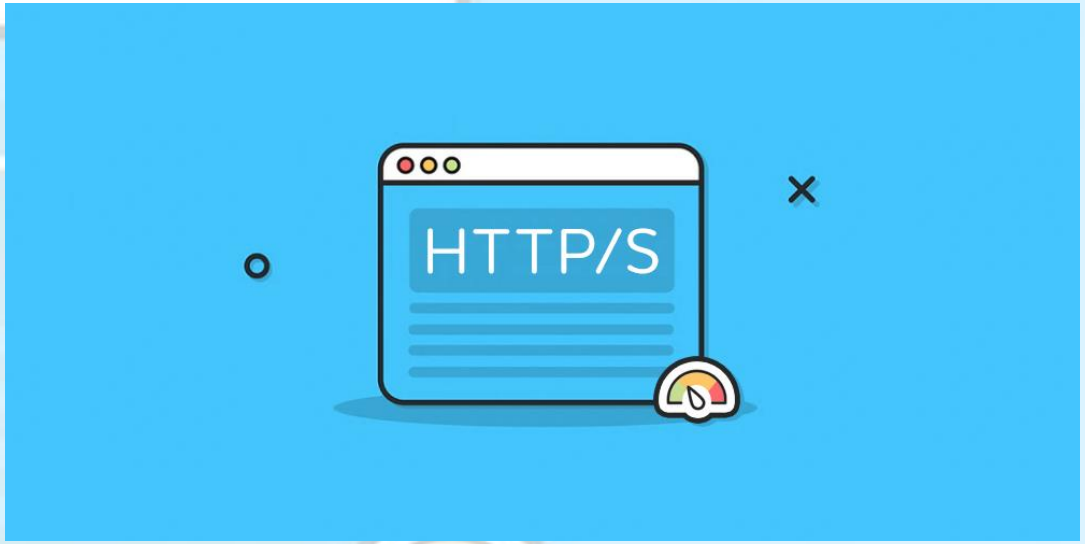
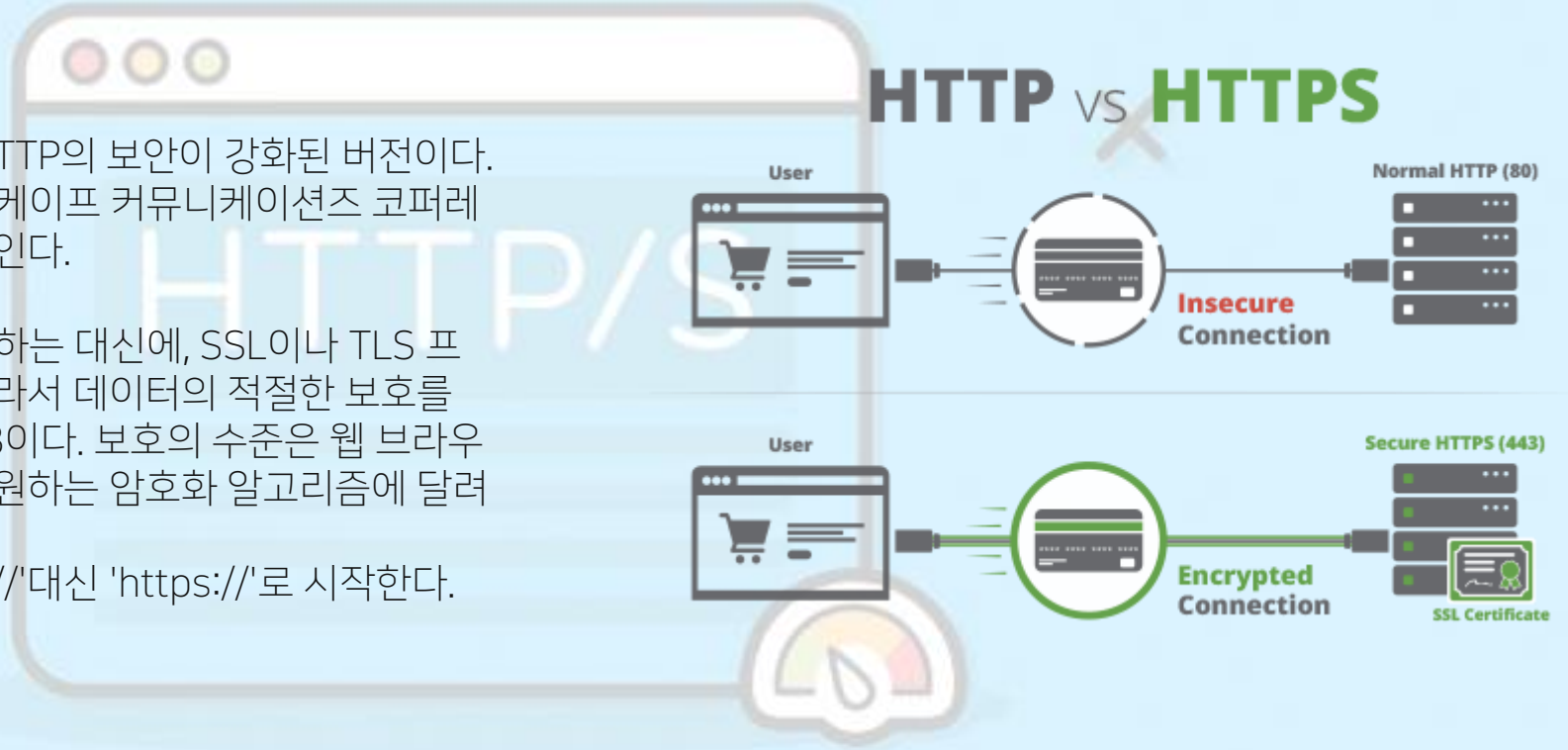
ALL ABOUT HTTPS



HTTPS 개념

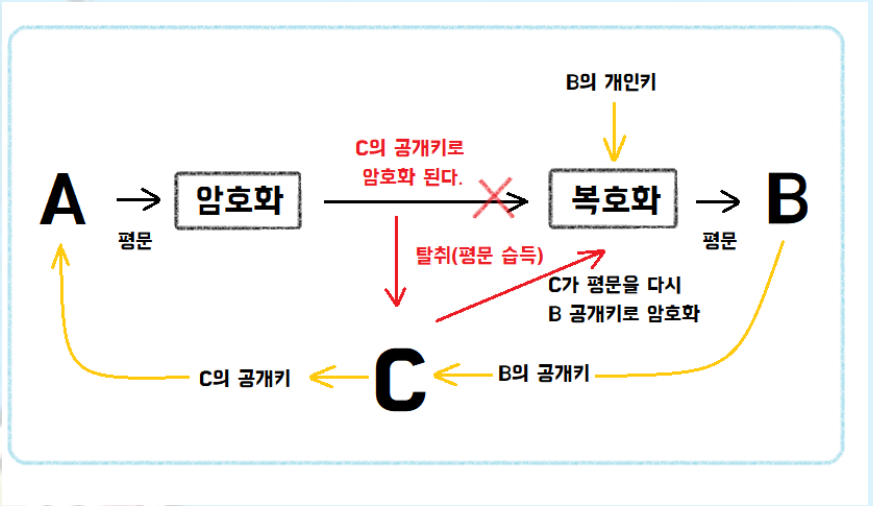
HTTPS는 월드 와이드 웹 통신 프로토콜인 HTTP의 보안이 강화된 버전이다. HTTPS는 통신의 인증과 암호화를 위해 넷스케이프 커뮤니케이션즈 코퍼레이션이 개발했으며, 전자 상거래에서 널리 쓰인다.

HTTPS는 소켓 통신에서 일반 텍스트를 이용하는 대신에, SSL이나 TLS 프로토콜을 통해 세션 데이터를 암호화한다. 따라서 데이터의 적절한 보호를 보장한다. HTTPS의 기본 TCP/IP 포트는 443이다. 보호의 수준은 웹 브라우저에서의 구현 정확도와 서버 소프트웨어, 지원하는 암호화 알고리즘에 달려 있다.
HTTPS를 사용하는 웹페이지의 URI는 'http://'대신 'https://'로 시작한다.

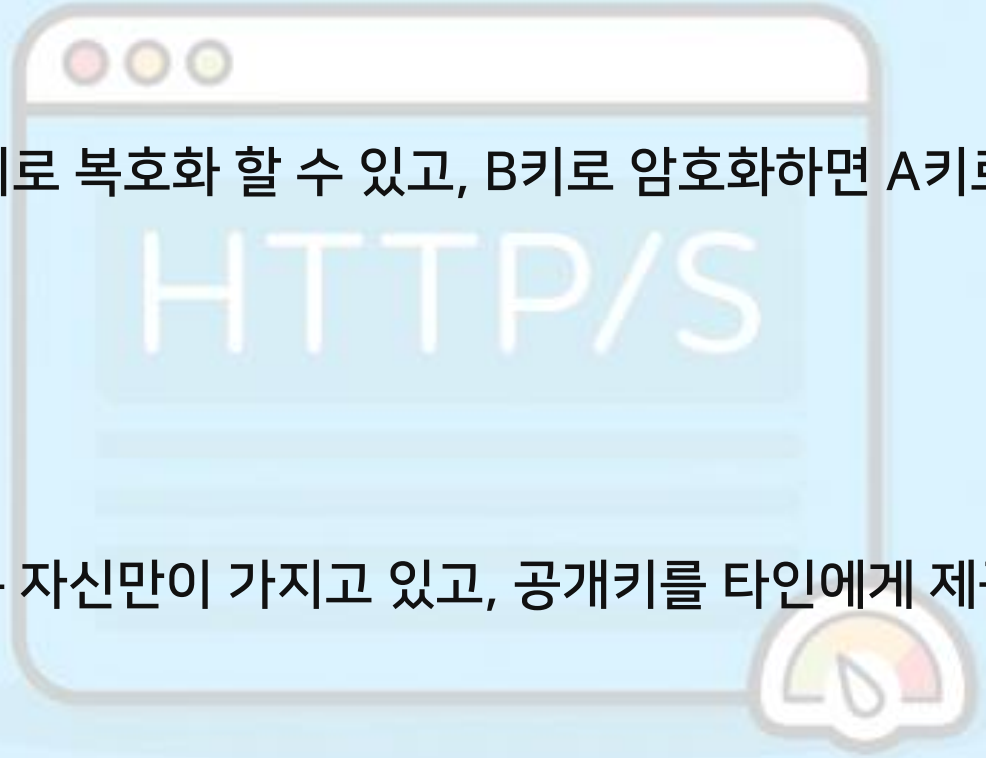


대칭키

암호를 만드는 행위인 암호화를 할 때 사용하는 일종의 비밀번호를 키(key)라고 한다. 이 키에 따라서 암호화된 결과가 달라지기 때문에 키를 모르면 암호를 푸는 행위인 복호화를 할 수 없다. 대칭키는 동일한 키로 암호화와 복호화를 같이 할 수 있는 방식의 암호화 기법을 의미한다. 즉 암호화를 할 때 1234라는 값을 사용했다면 복호화를 할 때 1234라는 값을 입력해야 한다는 것이다. 이해를 돕기 위해서 openssl을 이용해서 대칭키 방식으로 암호화하는 방법을 살펴보자. 아래 명령을 실행하면 plaintext.txt 파일이 생성된다. 그리고 비밀번호를 요구 받을 것이다. 이 때 입력한 비밀번호가 대칭키가 되는 것이다.



공개키 방식은 두개의 키를 갖게 되는데 A키로 암호화를 하면 B키로 복호화 할 수 있고, B키로 암호화하면 A키로 복호화 할 수 있는 방식이다. 이 방식에 착안해서 두개의 키 중 하나를 비공개키(private key, 개인키, 비밀키라고도 부른다)로하고, 나머지를 공개키(public key)로 지정한다. 비공개키는 자신만이 가지고 있고, 공개키를 타인에게 제공한다. 공개키를 제공 받은 타인은 공개키를 이용해서 정보를 암호화한다. 암호화한 정보를 비공개키를 가지고 있는 사람에게 전송한다. 비공개키의 소유자는 이 키를 이용해서 암호화된 정보를 복호화 한다. 이 과정에서 공개키가 유출된다고해도 비공개키를 모르면 정보를 복호화 할 수 없기 때문에 안전하다. 공개키로는 암호화는 할 수 있지만 복호화는 할 수 없기 때문이다.

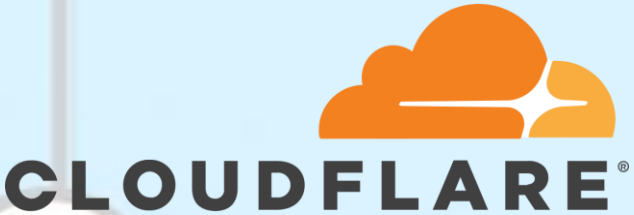


A키로 암호화를 하면 B키로 복호화 할 수 있고, B키로 암호화하면 A키로 복호화 할 수 있는 방식

비공개키는 자신만이 가지고 있고, 공개키를 타인에게 제공한다.

CA

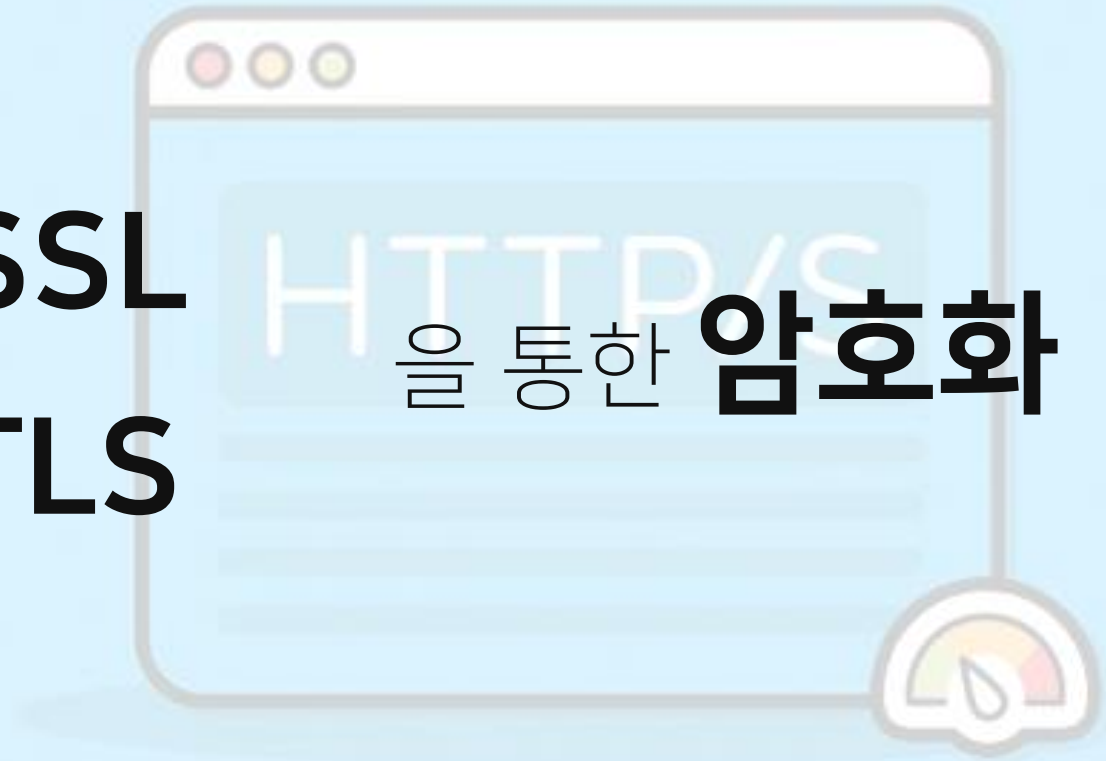
인증서의 역할은 클라이언트가 접속한 서버가 클라이언트가 의도한 서버가 맞는지를 보장하는 역할을 한다. 이 역할을 하는 민간기업들이 있는데 이런 기업들을 CA(Certificate authority) 혹은 Root Certificate 라고 부른다.
CA는 아무 기업이나 할 수 있는 것이 아니고 신뢰성이 엄격하게 공인된 기업들만이 참여할 수 있다.



SSL 인증서의 내용

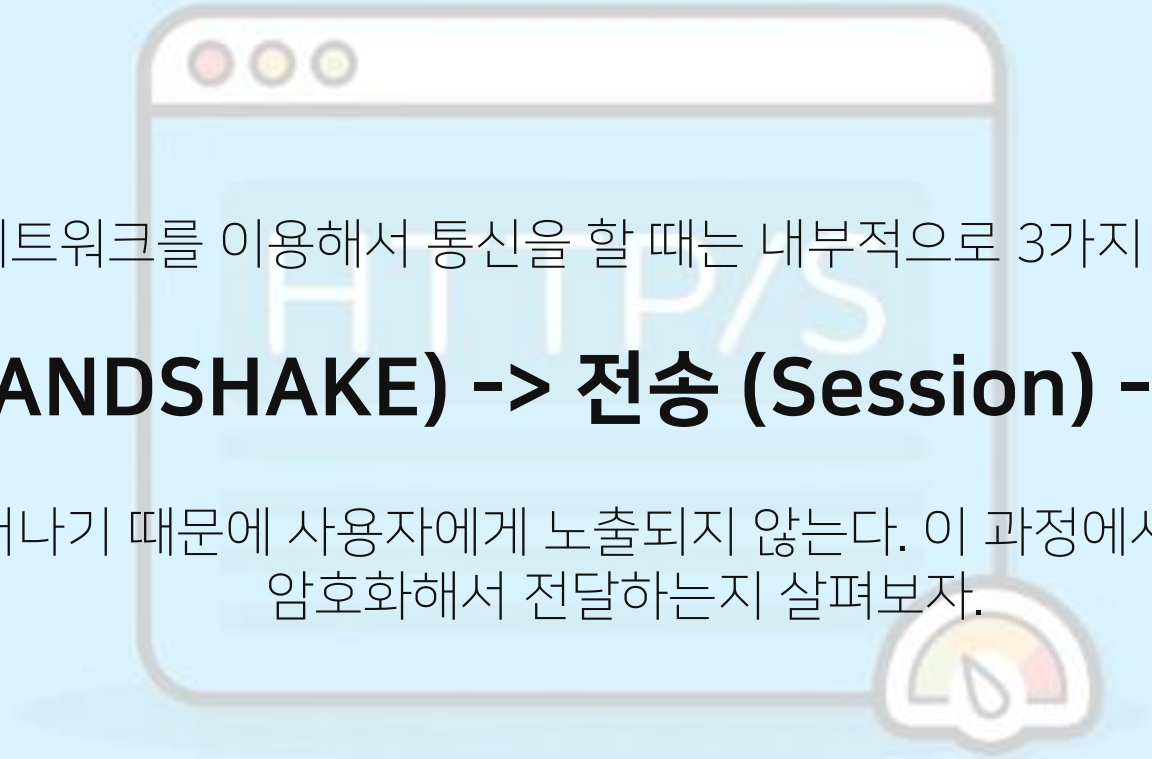
서비스의 정보 (인증서를 발급한 CA, 서비스의 도메인 등등)

서버 측 공개키 (공개키의 내용, 공개키의 암호화 방법)



SSL
TLS

을 통한 **암호화**



컴퓨터와 컴퓨터가 네트워크를 이용해서 통신을 할 때는 내부적으로 3가지 단계가 있다. 아래와 같다.

악수 (HANDSHAKE) -> 전송 (Session) -> 세션종료

이것은 은밀하게 일어나기 때문에 사용자에게 노출되지 않는다. 이 과정에서 SSL가 어떻게 데이터를 암호화해서 전달하는지 살펴보자.

악수 (HANDSHAKE)

1. 클라이언트가 서버에 접속하며 랜덤 데이터를 전송. (Client hello)

Client

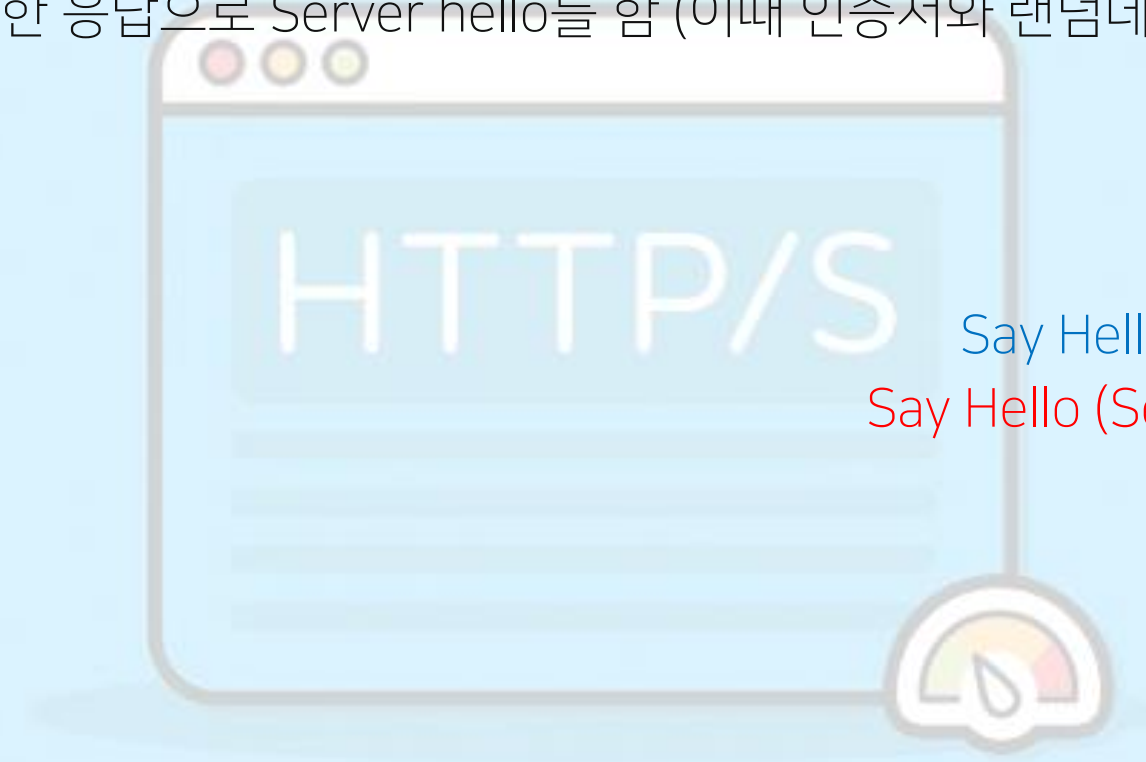
Say Hello (Cli 암호화)

Server

악수 (HANDSHAKE)

2. 서버가 Client hello에 대한 응답으로 Server hello를 함 (이때 인증서와 랜덤데이터 제공)

Client



Server

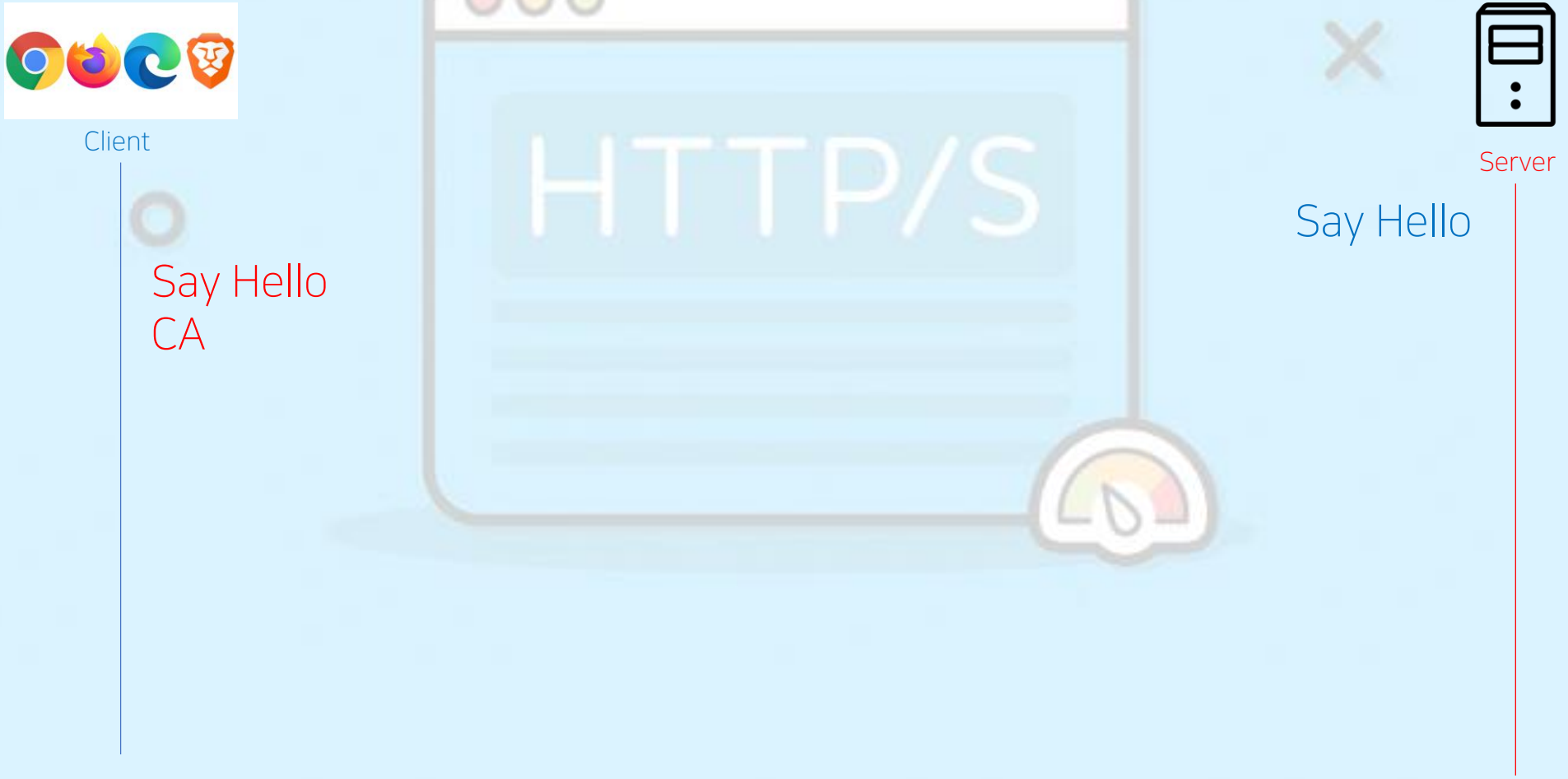
Say Hello (Cli 암호화)

Say Hello (Server 암호화)

CA (인증서)

악수 (HANDSHAKE)

2. 서버가 Client hello에 대한 응답으로 Server hello를 함 (이때 인증서와 랜덤데이터 제공)



악수 (HANDSHAKE)

3. 클라이언트의 브라우저에서 서버가 건네준 인증서가 CA에서 발급된 건지를 확인하고 공개키로 인증서를 복호화 한다. (인증서가 믿을 수 있다고 판단함)

Client

Server

Is CA in the List?

CA

악수 (HANDSHAKE)

3. 클라이언트의 브라우저에서 서버가 건네준 인증서가 CA에서 발급된 건지를 확인하고 공개키로 인증서를 복호화 한다. (인증서가 믿을 수 있다고 판단함)

Client

Server

Is CA in the List?

NO WARNING!!!

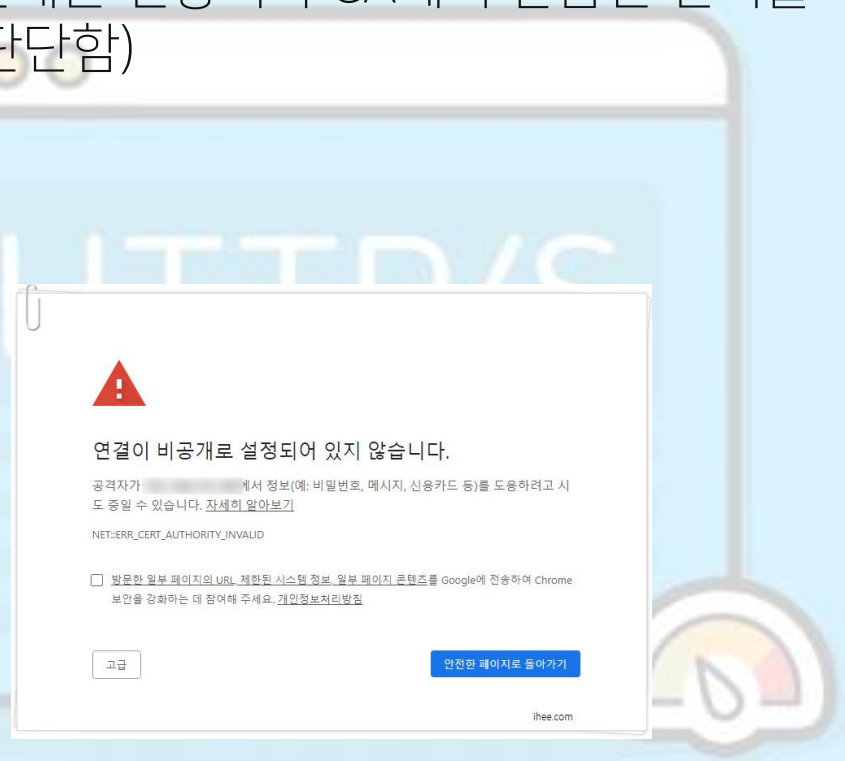
악수 (HANDSHAKE)

3. 클라이언트의 브라우저에서 서버가 건네준 인증서가 CA에서 발급된 건지를 확인하고 공개키로 인증서를 복호화 한다. (인증서가 믿을 수 있다고 판단함)

Client

Is CA in the List?

NO WARNING!!!



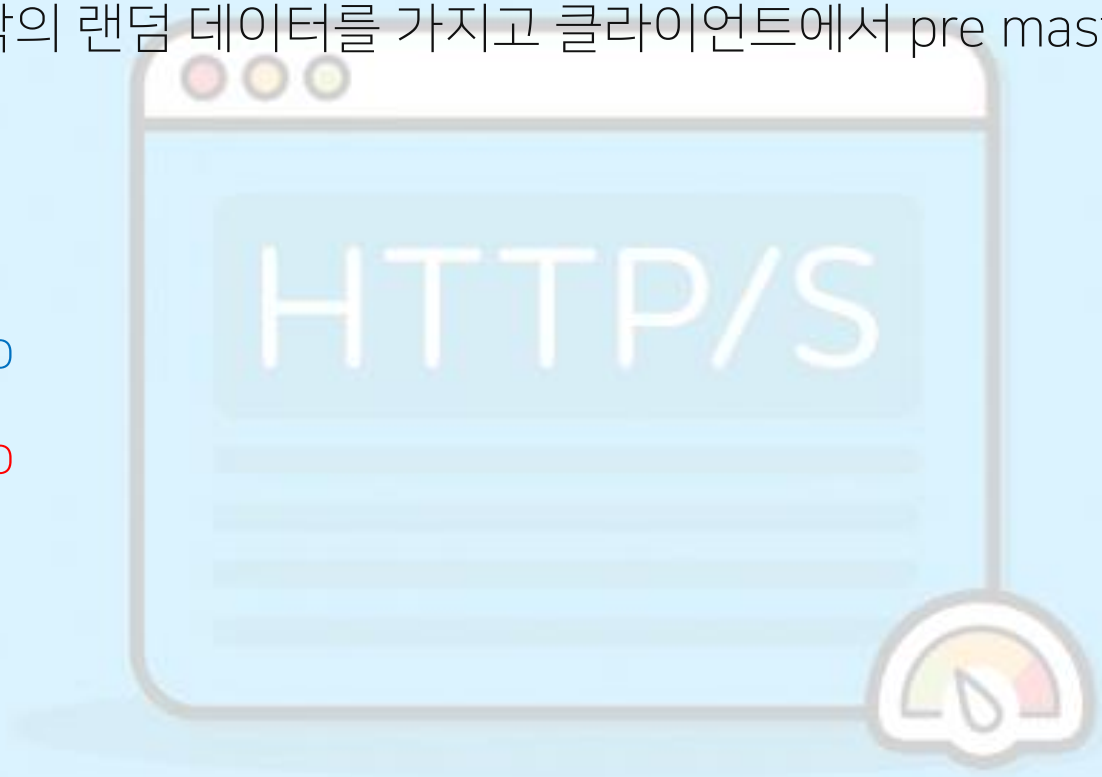
Server

악수 (HANDSHAKE)

4. 클라이언트와 서버의 각각의 랜덤 데이터를 가지고 클라이언트에서 pre master secret값을 만들어냄 (대칭키)

Client

Say Hello
+
Say Hello

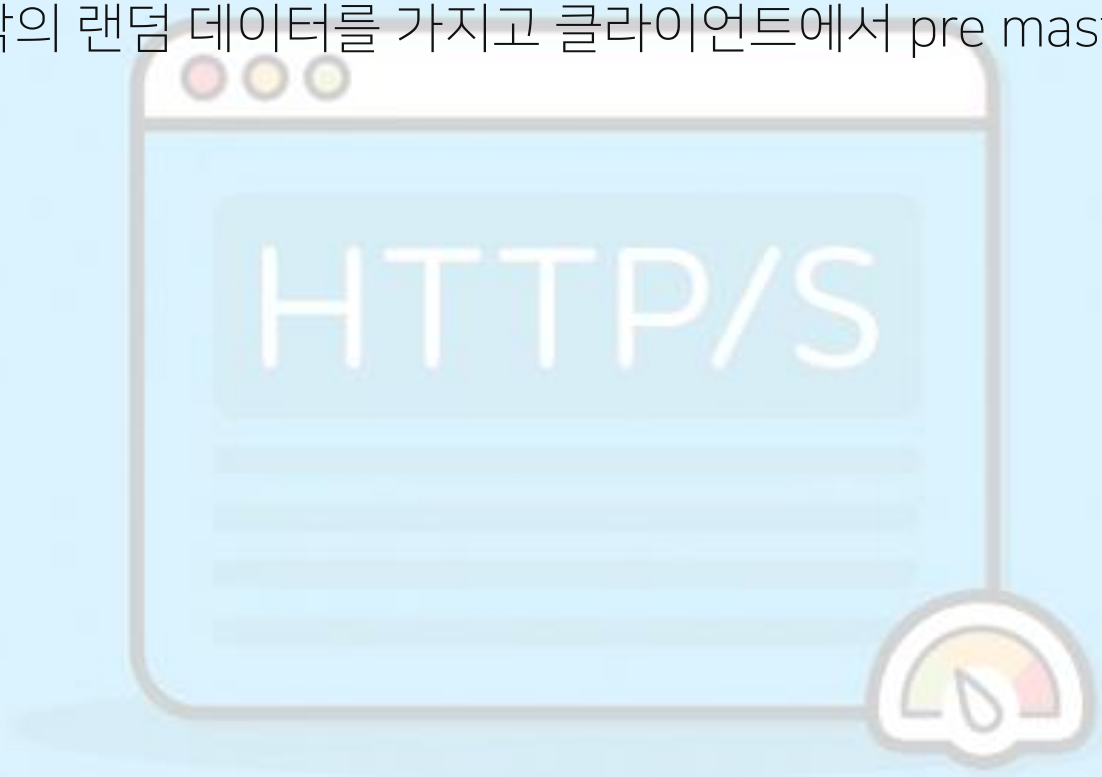


Server

악수 (HANDSHAKE)

4. 클라이언트와 서버의 각각의 랜덤 데이터를 가지고 클라이언트에서 pre master secret값을 만들어냄 (대칭키)

Client

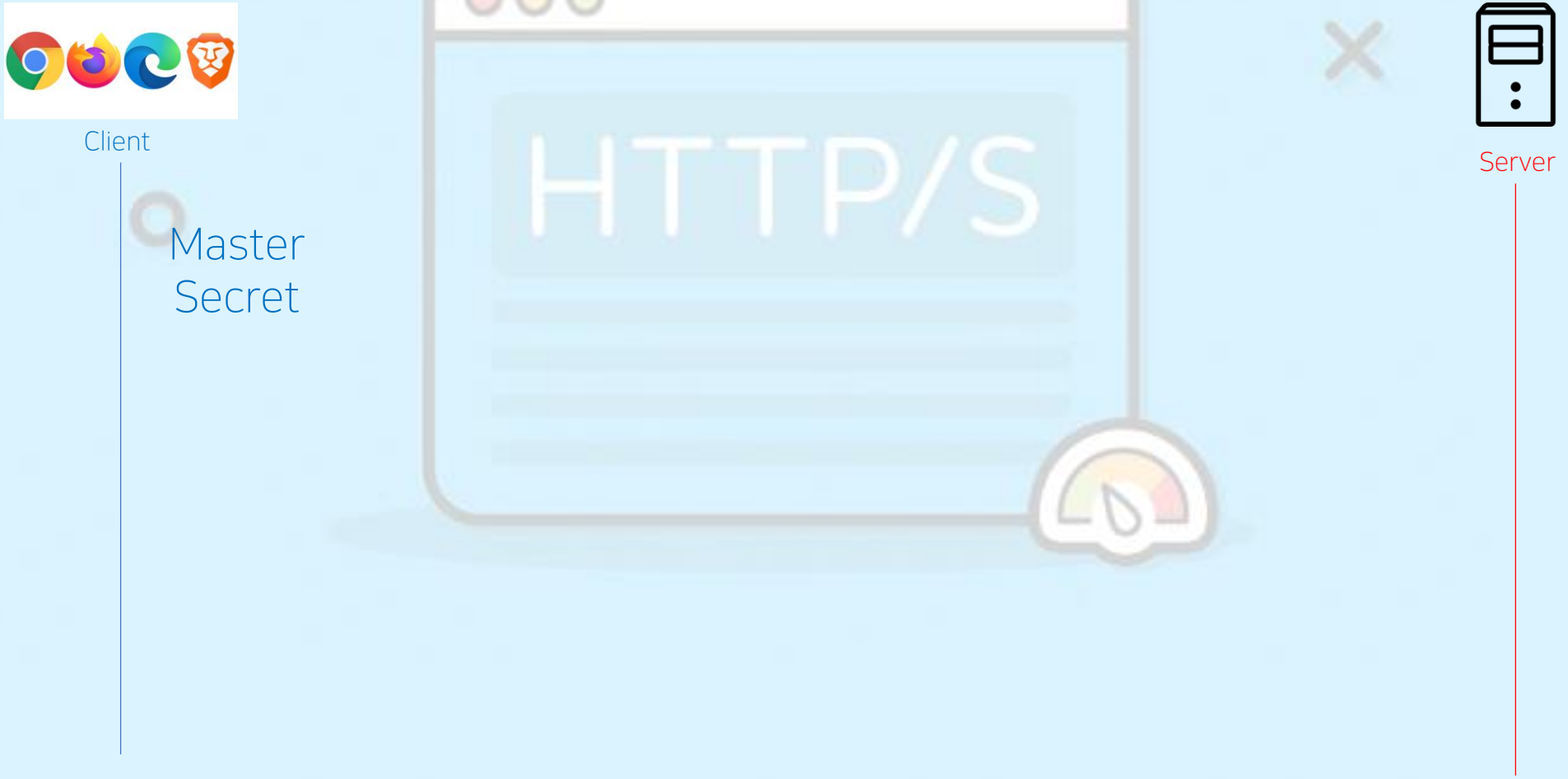


Server

" 합 – 체 "

악수 (HANDSHAKE)

4. 클라이언트와 서버의 각각의 랜덤 데이터를 가지고 클라이언트에서 pre master secret값을 만들어냄 (대칭키)



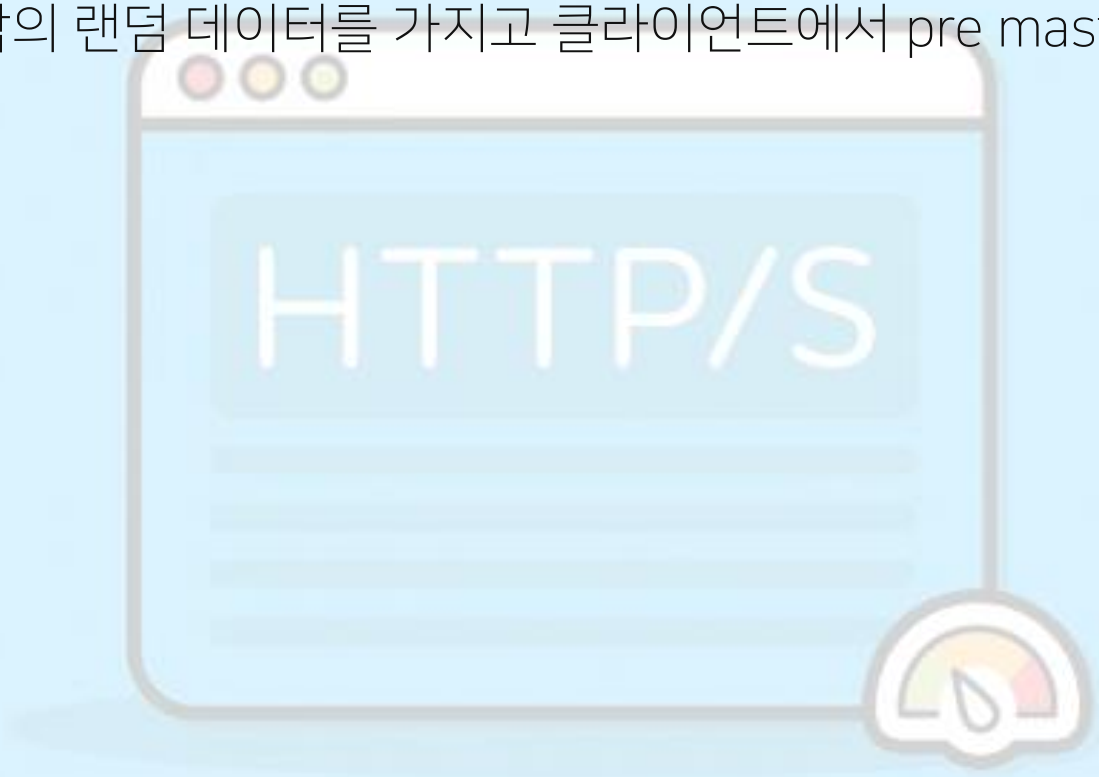
악수 (HANDSHAKE)

4. 클라이언트와 서버의 각각의 랜덤 데이터를 가지고 클라이언트에서 pre master secret값을 만들어냄 (대칭키)

Client

CA 공개키

Master Secret



Server

악수 (HANDSHAKE)

4. 클라이언트와 서버의 각각의 랜덤 데이터를 가지고 클라이언트에서 pre master secret값을 만들어냄 (대칭키)

Client

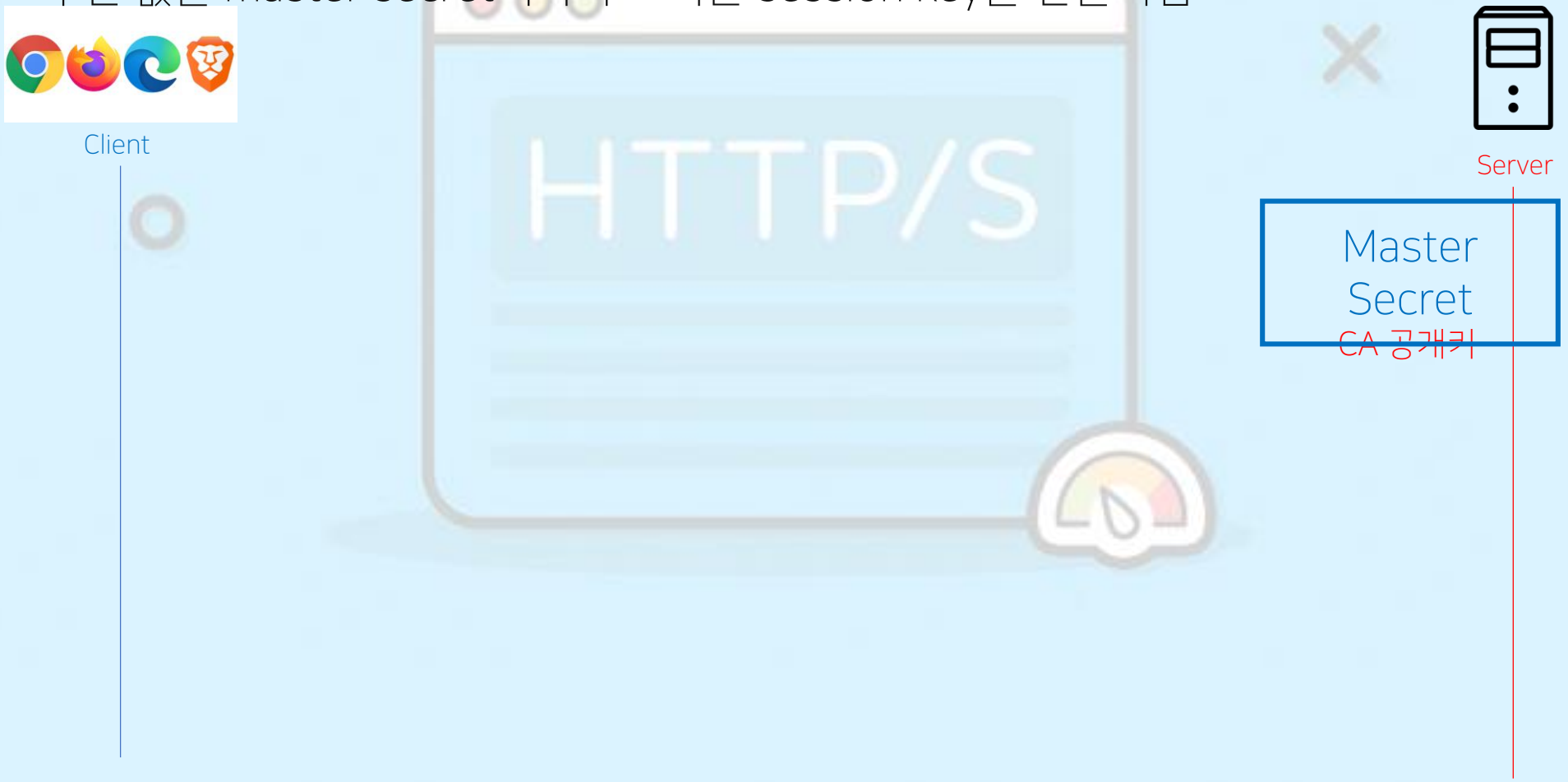
Master Secret
CA 공개키

= Pre Master Key

Server

악수 (HANDSHAKE)

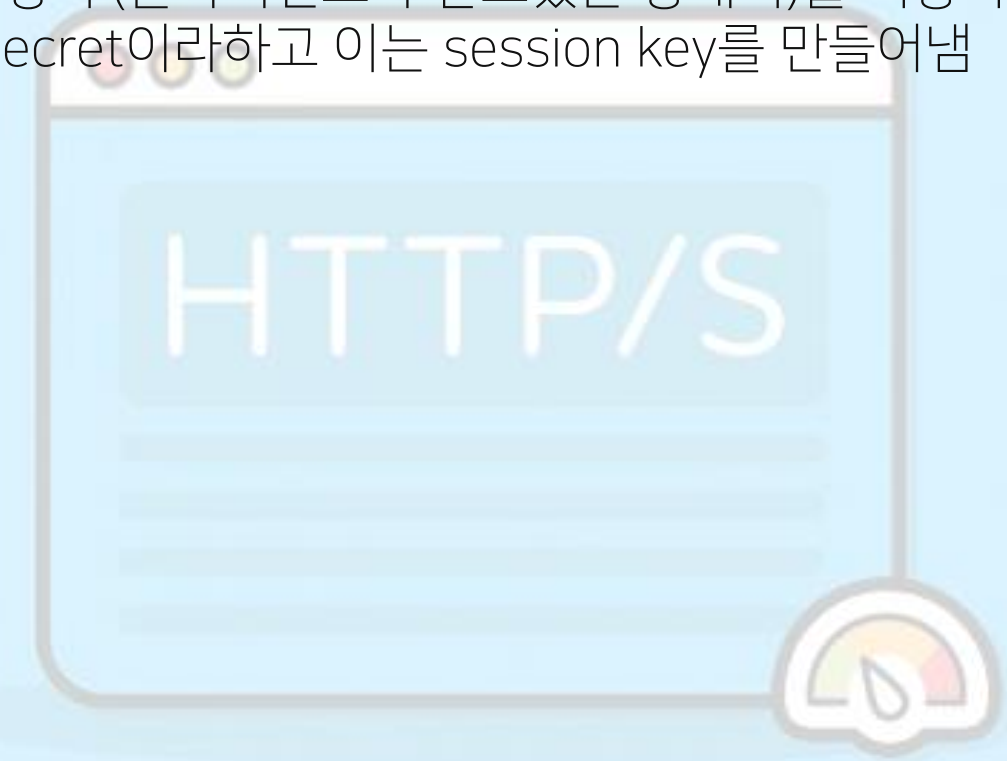
5. pre master secret 값을 비대칭키 (클라이언트가 알고있는 공개키)를 이용하여 암호화하고 이를 서버에 보냄
이렇게 암호화 된 값을 master secret이라하고 이는 session key를 만들어냄



악수 (HANDSHAKE)

5. pre master secret 값을 비대칭키 (클라이언트가 알고있는 공개키)를 이용하여 암호화하고 이를 서버에 보냄
이렇게 암호화 된 값을 master secret이라하고 이는 session key를 만들어냄

Client



Server

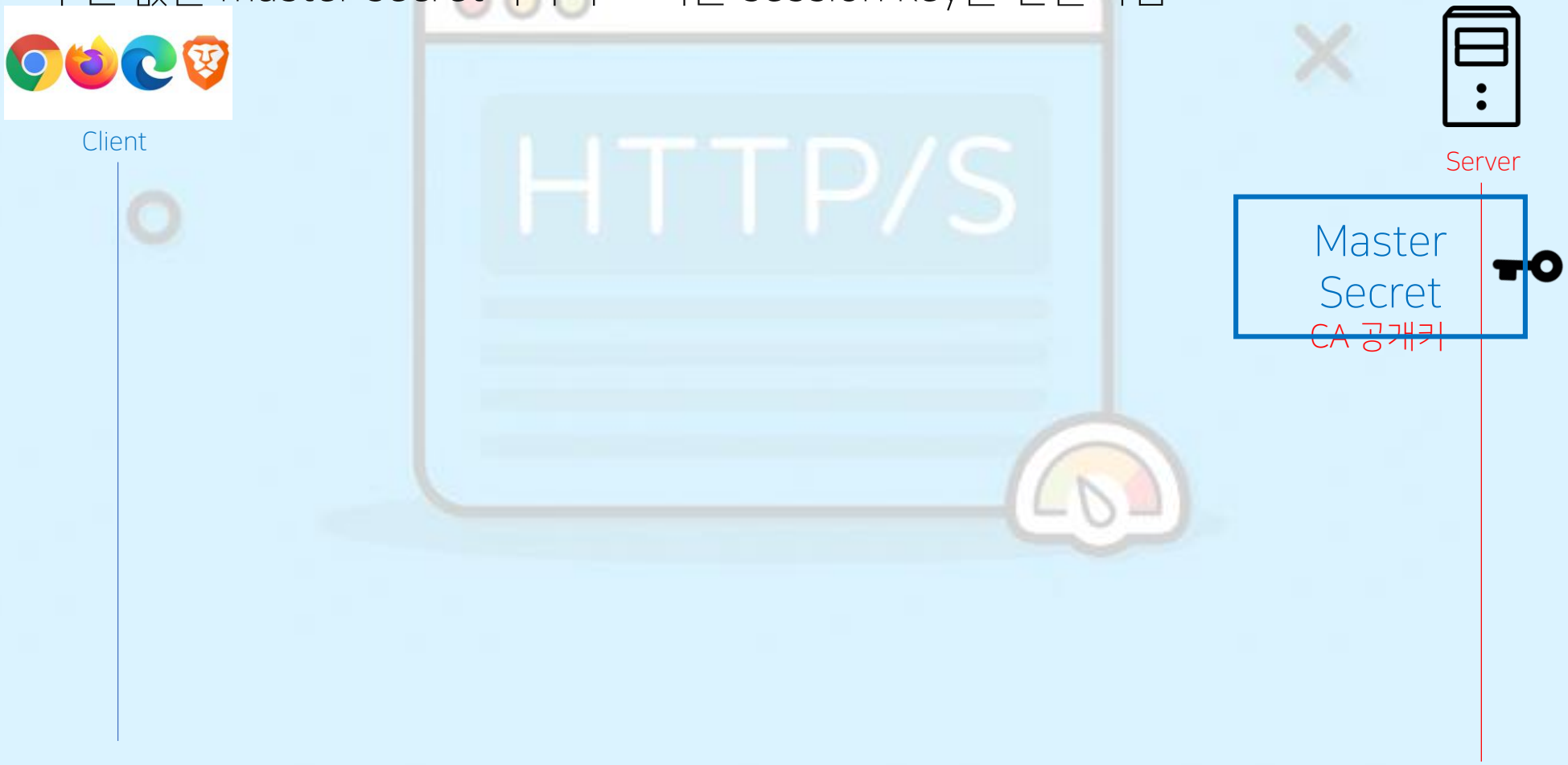
Master Secret

CA 공개키

CA 비공개키

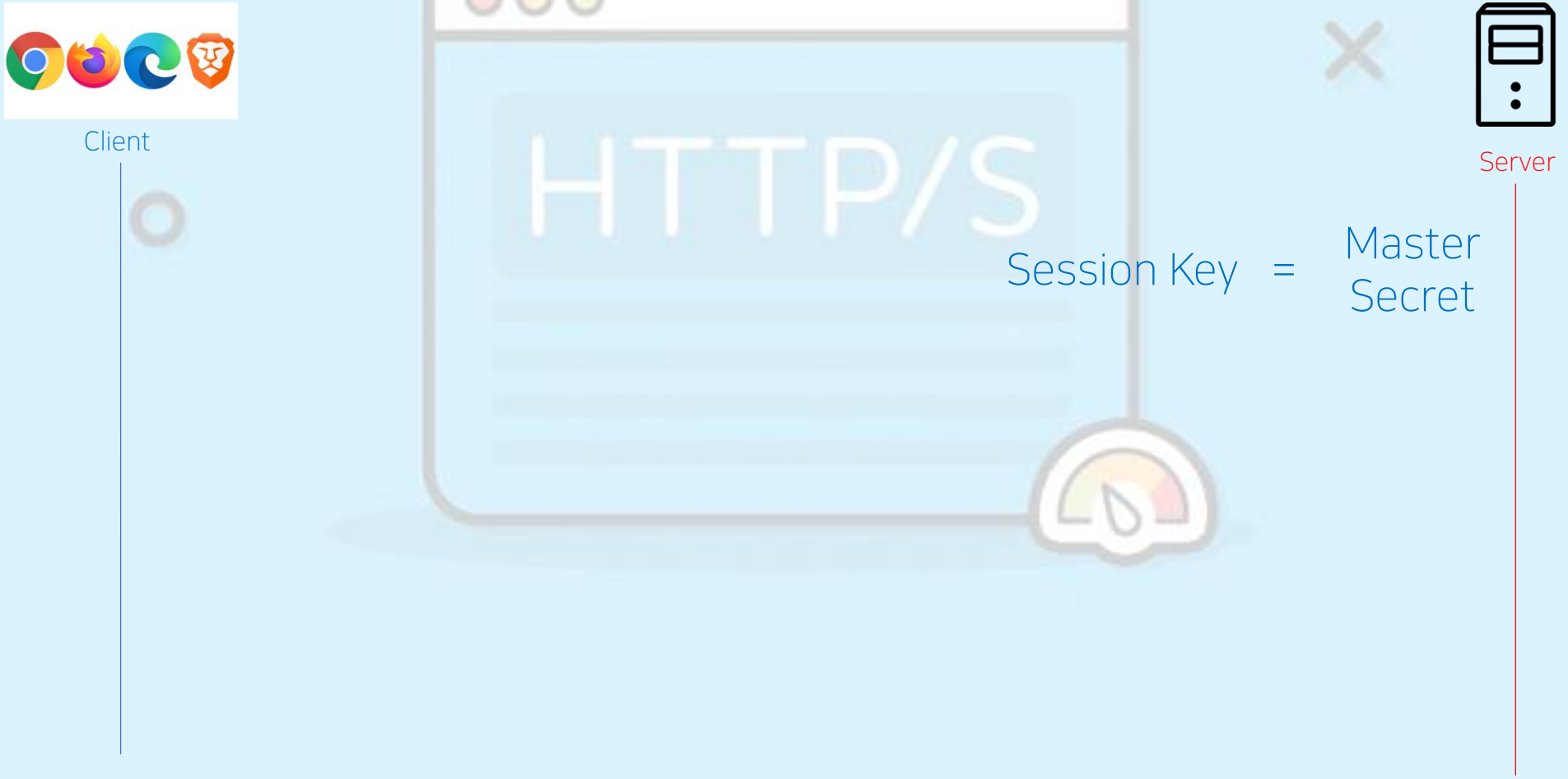
악수 (HANDSHAKE)

5. pre master secret 값을 비대칭키 (클라이언트가 알고있는 공개키)를 이용하여 암호화하고 이를 서버에 보냄
이렇게 암호화 된 값을 master secret이라하고 이는 session key를 만들어냄



악수 (HANDSHAKE)

6. 이 session key를 이용하여 서버와 클라이언트는 대칭키 방식으로 암호화 통신을 함



TLS(SSL) – HANDSHAKE

1. 클라이언트가 서버에 접속하며 랜덤 데이터를 전송. (Client hello)

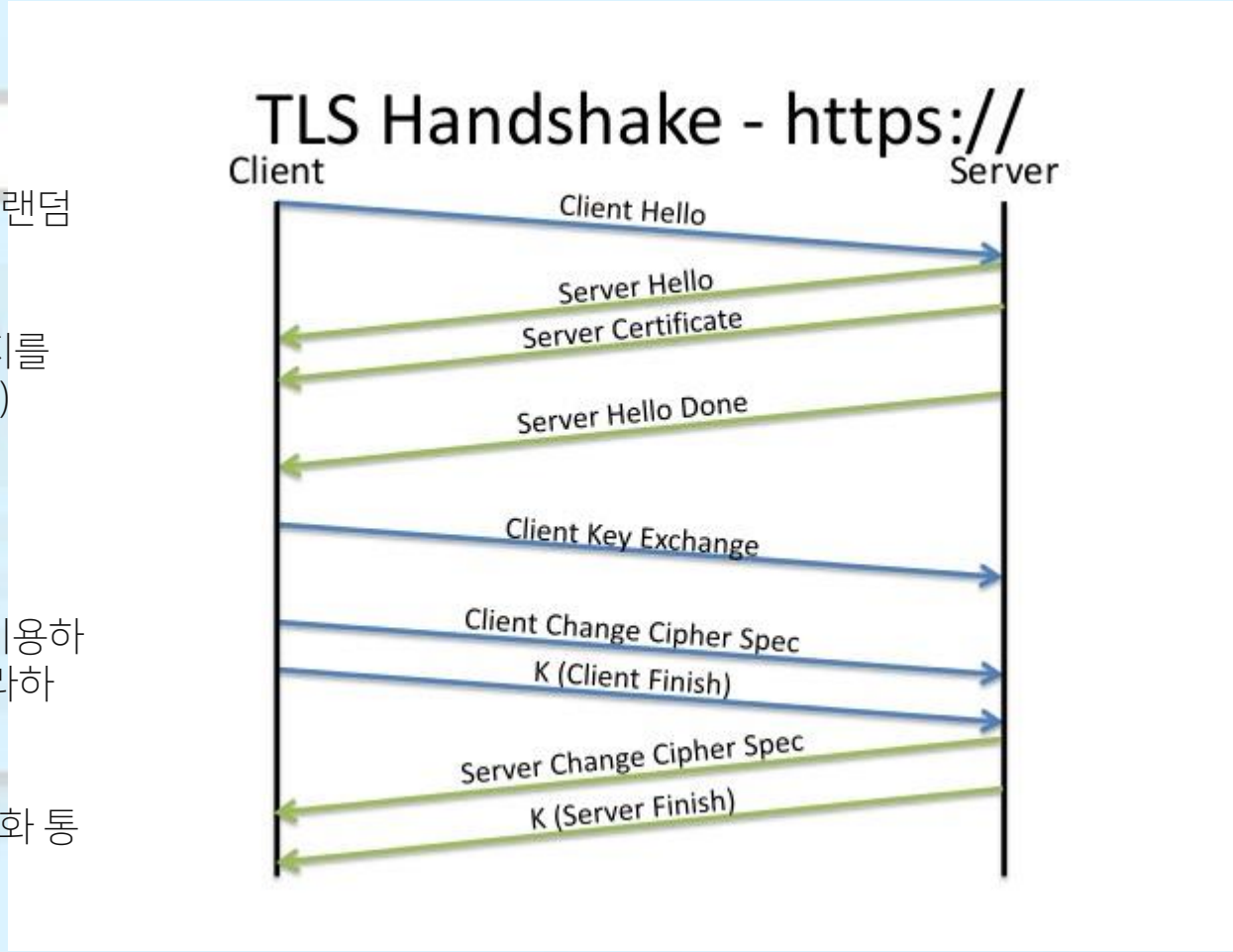
2. 서버가 Client hello에 대한 응답으로 Server hello를 함 (이때 인증서와 랜덤 데이터 제공)

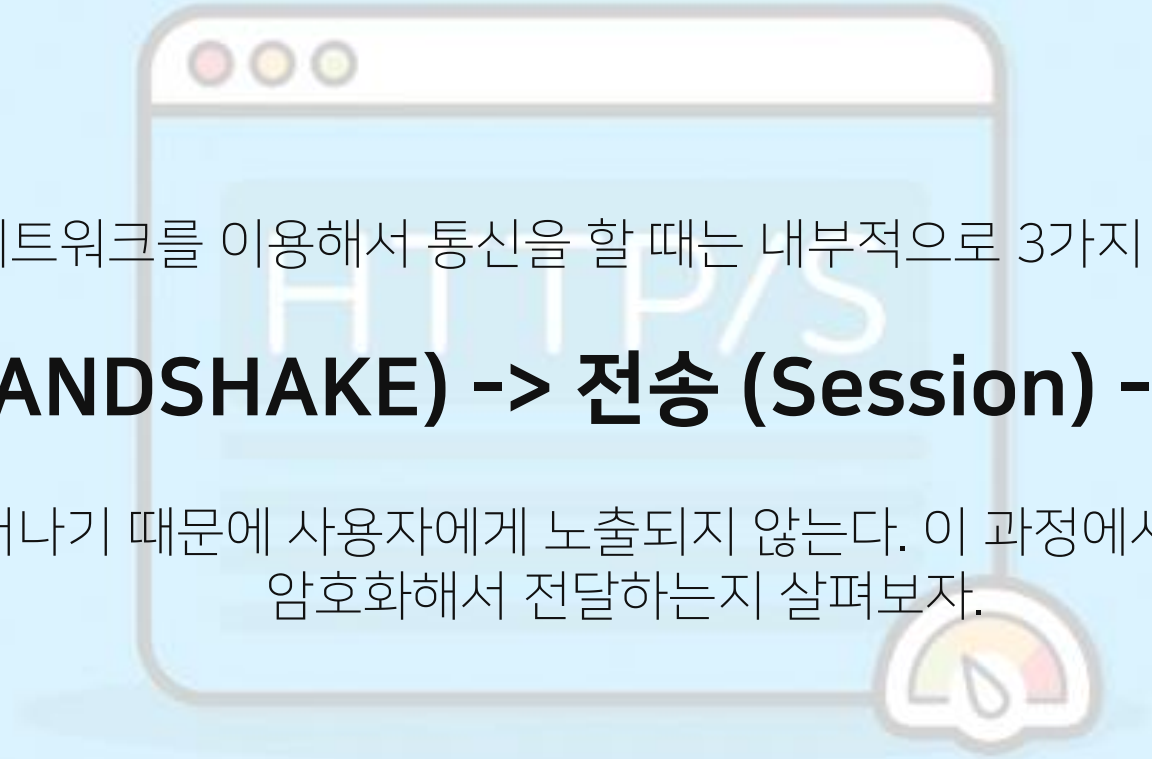
3. 클라이언트의 브라우저에서 서버가 건네준 인증서가 CA에서 발급된건지를 확인하고 공개키로 인증서를 복호화한다. (인증서가 믿을 수 있다고 판단함)

4. 클라이언트와 서버의 각각의 랜덤 데이터를 가지고 클라이언트에서 pre master secret값을 만들어냄 (대칭키)

5. pre master secret 값을 비대칭키 (클라이언트가 알고있는 공개키)를 이용하여 암호화하고 이를 서버에 보냄 이렇게 암호화 된 값을 master secret이라하고 이는 session key를 만들어냄

6. 이 session key를 이용하여 서버와 클라이언트는 대칭키 방식으로 암호화 통신을 함





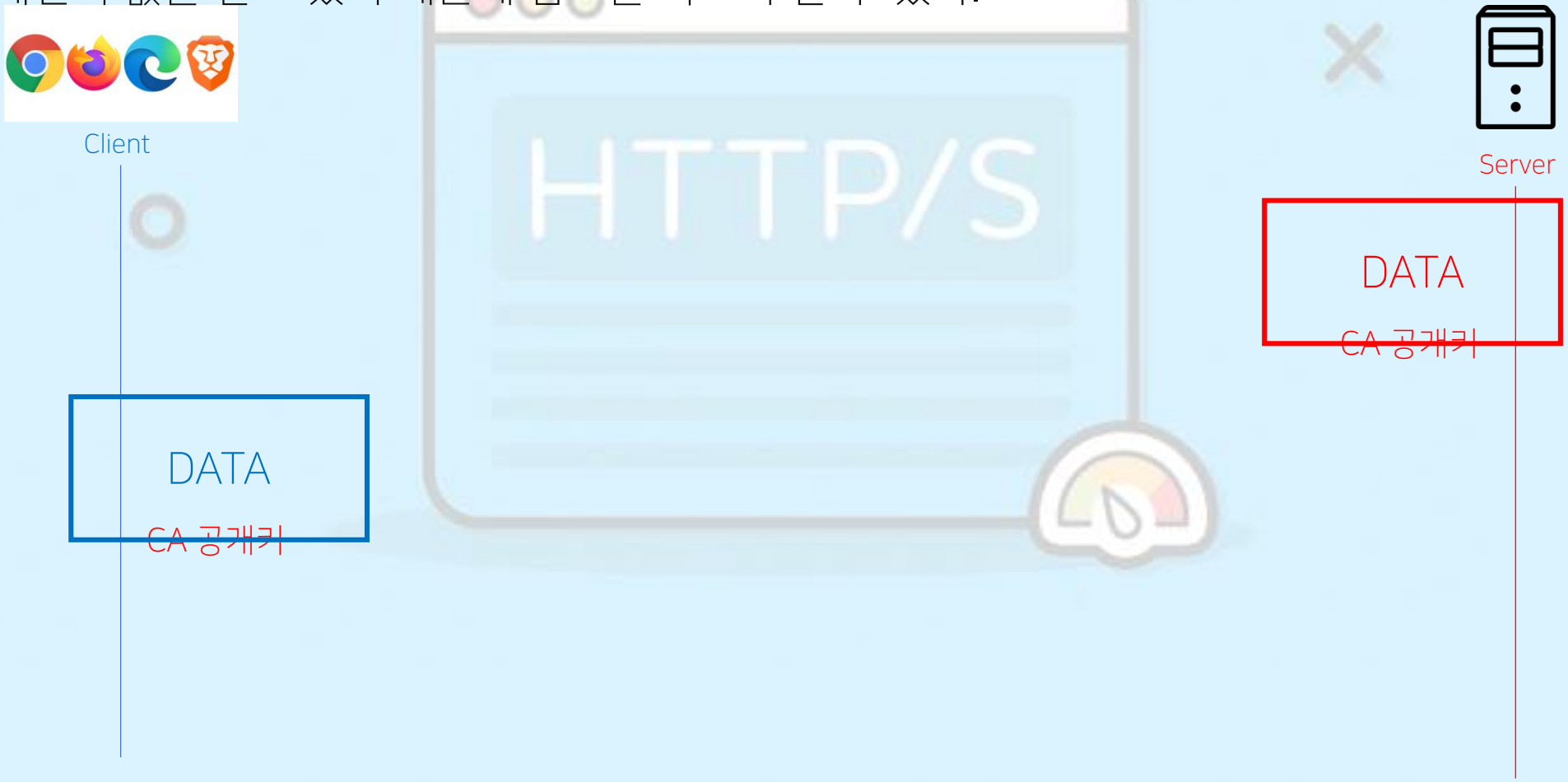
컴퓨터와 컴퓨터가 네트워크를 이용해서 통신을 할 때는 내부적으로 3가지 단계가 있다. 아래와 같다.

악수 (HANDSHAKE) -> 전송 (Session) -> 세션종료

이것은 은밀하게 일어나기 때문에 사용자에게 노출되지 않는다. 이 과정에서 SSL가 어떻게 데이터를 암호화해서 전달하는지 살펴보자.

전송 (Session)

session key 값을 이용해서 대칭키 방식으로 암호화 한다는 점이다. 암호화된 정보는 상대방에게 전송될 것이고, 상대방도 세션키 값을 알고 있기 때문에 암호를 복호화 할 수 있다.

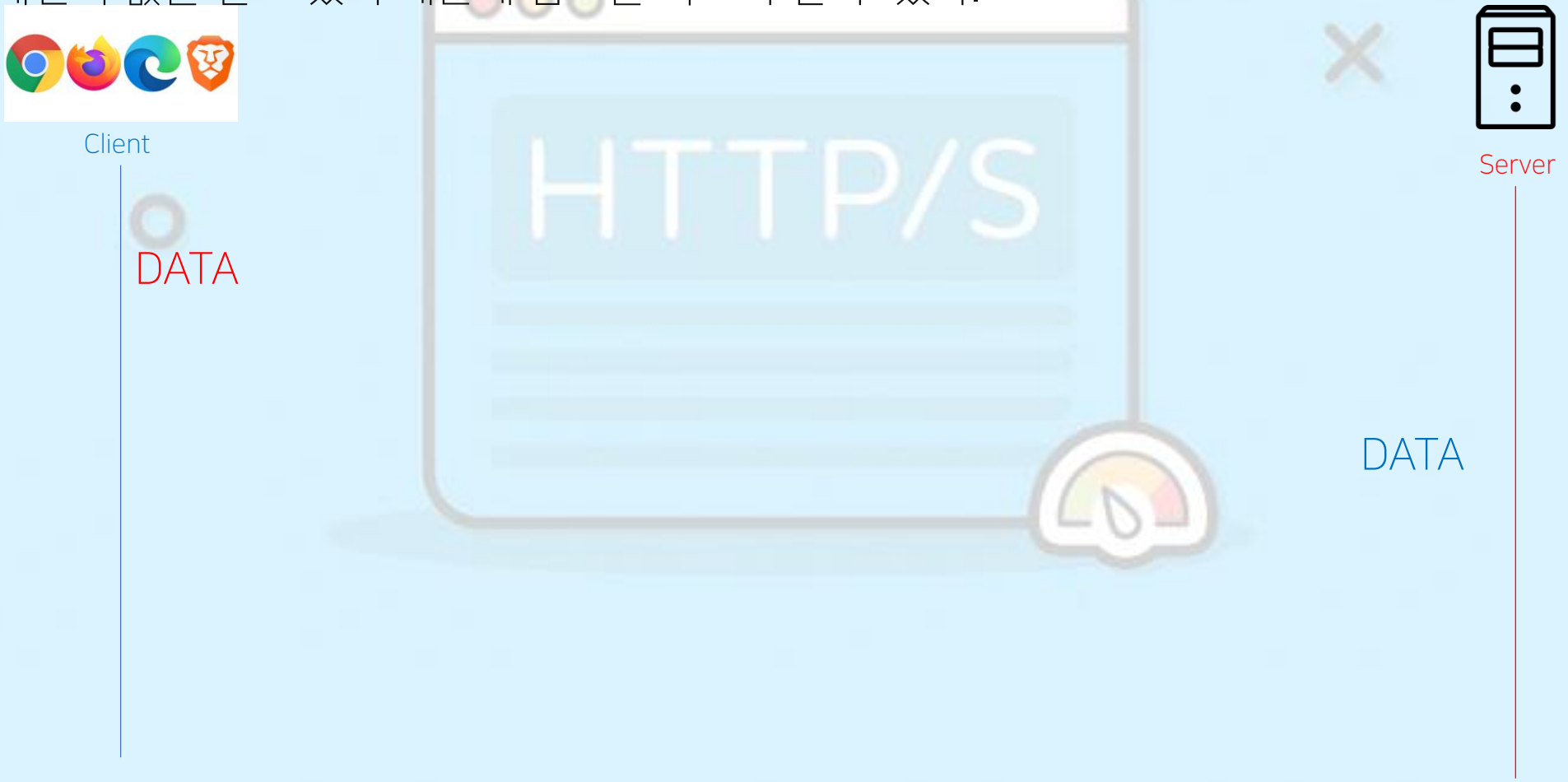


전송 (Session)

session key 값을 이용해서 대칭키 방식으로 암호화 한다는 점이다. 암호화된 정보는 상대방에게 전송될 것이고, 상대방도 세션키 값을 알고 있기 때문에 암호를 복호화 할 수 있다.

전송 (Session)

session key 값을 이용해서 대칭키 방식으로 암호화 한다는 점이다. 암호화된 정보는 상대방에게 전송될 것이고, 상대방도 세션키 값을 알고 있기 때문에 암호를 복호화 할 수 있다.



인증서 구입 -> 웹서버 셋팅 (Apache 서버 이용)

opentutorials.org StarSSL 설치 -> 사용 링크