HTTP & HTTPS

HTTP란 ?

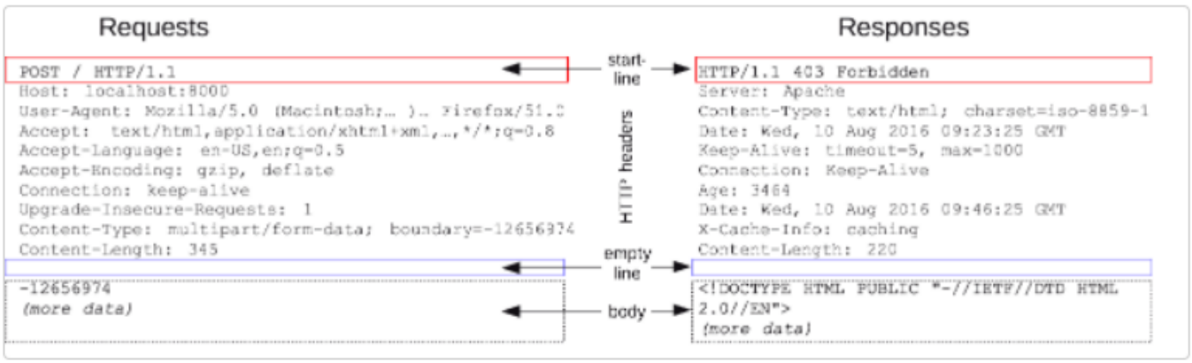
Hyper Text Transfer Protocol의 약자로, 서버/클라이언트 모델을 따라 데이터를 주고 받기 위한 프로토콜, 80번 포트 사용

[특성]

- 비연결지향
- 상태정보 유지안함

(→ 이 특성을 보완하기위해 쿠키와 세션을 사용)

[구조]



✨ 요청

- start-line
  - 요청 method
  - URI
  - HTTP Version
- header
  - Host URL
  - User-Agent : 클라이언트 정보
  - Accept
  - Authorization

✨ 응답

- start-line
  - HTTP Version
  - 상태 코드
  - 상태 텍스트
- header
  - Date
  - Content-Type
  - Cache-Control
  - 기타 등등

- 기타 등등

> ✨ 공통 부분

- empty line : 헤더와 본문 구분용 빈줄
- body : 요청과 관련된 데이터나 응답과 관련된 데이터 또는 문서를 포함

하지만 HTTP는 암호화가 되지 않은 평문 데이터 전송 프로토콜이므로 ,누군가 네트워크에서 신호를 가로채는 경우 내용이 노출되는 보안이슈 발생

→ 이 문제를 해결해주는 프로토콜이 **HTTPS**

HTTPS란?

> HTTP + Secure

HTTP에 데이터 암호화가 추가된 프로토콜. 443번 포트 사용, 데이터 암호화를 지원한다.

[암호화 방식] - 간단히 설명

▼ 대칭키 암호화

암호화/복호화 사용하는 키가 동일

장점 : 수행 시간이 짧음

단점 : 키 교환중 탈취가능성 多, 사람이 증가할 수록 키 관리가 어려워짐

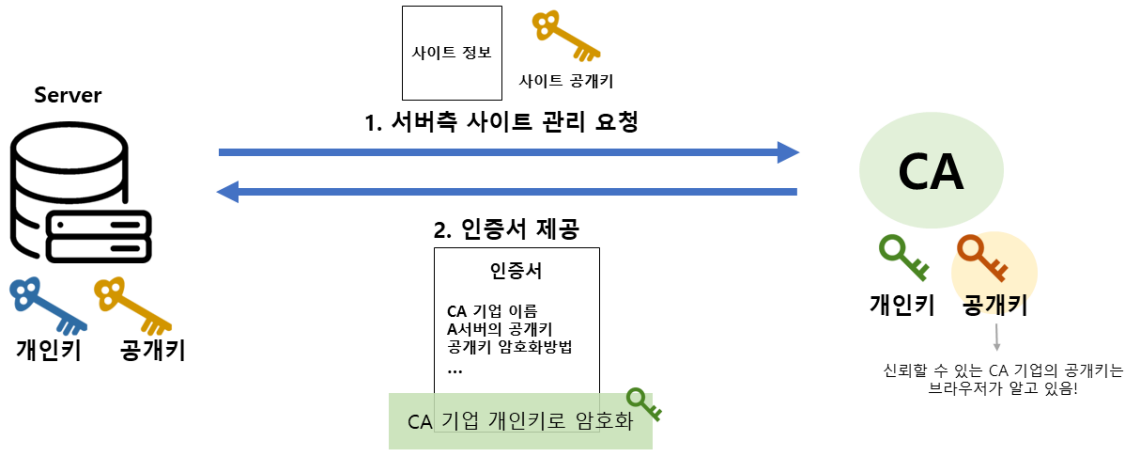
▼ 비대칭키 암호화

암호화/복호화 사용하는 키가 다름. (공개키/개인키)

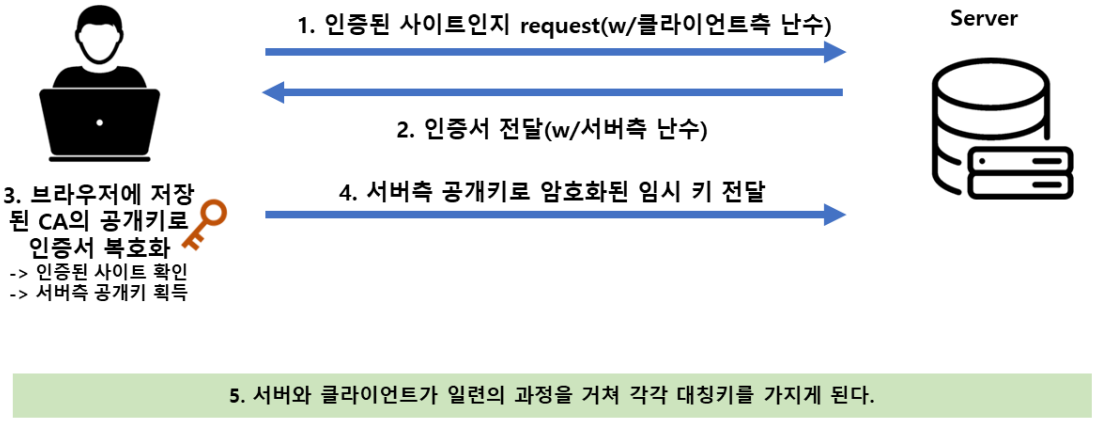
장점 : 키분배 필요X, 기밀성/인증/부인방지 기능 제공

단점 : 대칭키 암호화 방식에 비해 속도가 느림

[통신 흐름]

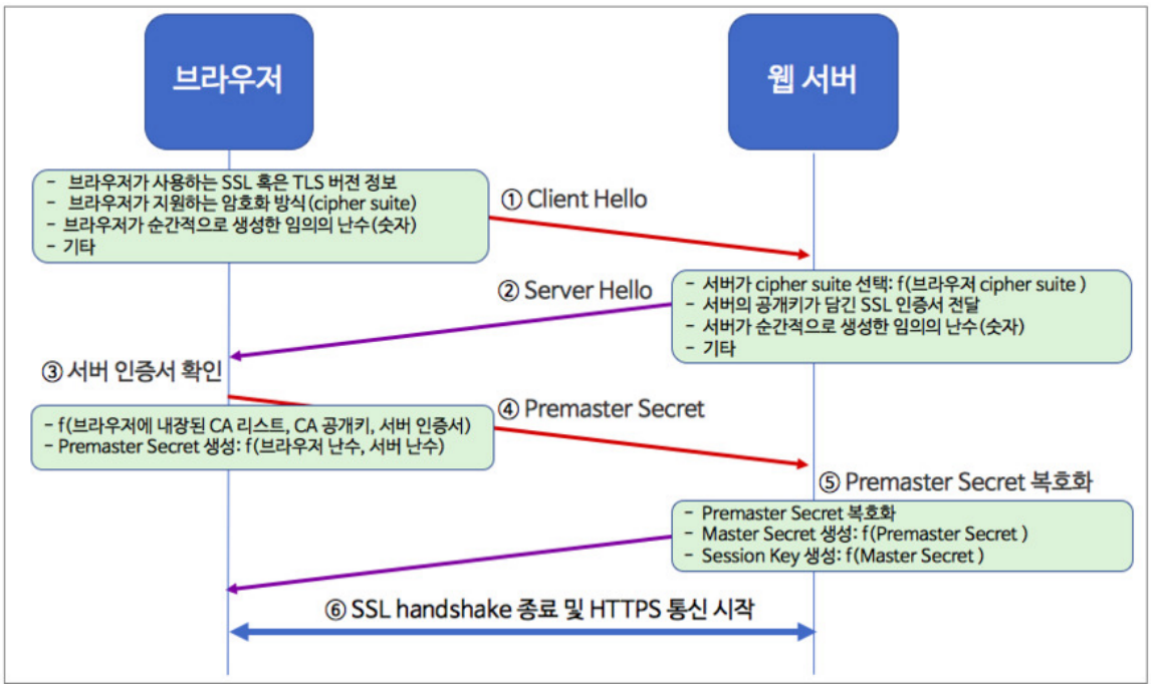


- CA : Certificate Authority(디지털 서명을 해주는 인증 기관)



3번에서 서버측 개인키를 이용해서 데이터들을 암호화해서 사용하면 되겠다고 생각 할 수 있다. 하지만 비대칭키로 암호화된 데이터를 복호화하는 과정은 속도가 느리므로 클라이언트와 서버가 공유하는 대칭키를 만든다.

▼ 조금 더 자세히 말하자면



[handShake 과정] - 비대칭키

[handShake 이후의 통신] - 대칭키

HTTPS도 무조건 안전한 것은 아니다. (신뢰성 있는 CA기업이 아닌 자체 인증서를 발급한 경우 등)

HTTP와 HTTPS

개인 정보와 같은 민감한 데이터를 주고 받아야 한다면 HTTPS를 이용해야 하지만, 노출이 되어도 괜찮은 단순한 정보 조회 등 만을 처리하고 있다면 HTTP를 이용하면 된다.

reference

- https://brunch.co.kr/@sangjinkang/38
- https://www.youtube.com/watch?v=H6lpFRpyl14
- https://jeong-pro.tistory.com/89
- https://www.uname.in/129
- https://mangkyu.tistory.com/98