<웹 기초지식>

- **Web Browser – 웹에 접속하기 위해 사용하는 소프트웨어**

HTTP를 통해 인터넷 상에서 통신을 하며 서버로부터 전달받은 다양한 웹 리소스를 가공해 사용자가 웹과 HTTP의 동작원리를 몰라도 웹을 사용할 수 있도록 하는 소프트웨어

Ex) Chrome, Edge, Safari, Firefox -> 기본 기능은 동일하나, HTML, CSS(Cascading Style Sheet – 표시(디자인) 방식), JS(JavaScript – 웹사이트의 동적 요소) 해석 및 실행을 빠르게 하는 등 차별화를 둠.

아래 사진은 웹 브라우저로 웹사이트에 접속할 때 이루어지는 통신을 간단화한 것.

웹을 사용하기 위한 다양한 방법들:

- Network Program – 직접 HTTP 형태의 데이터 작성해서 전송해야. (ex) nc, telnet
- CLI Program – 서버의 응답을 단순히 출력함(html을 해석함). CSS 및 JS 실행 불가. (ex) curl, wget
- Web Browser – 브라우저가 다해 줌.

- **Web Resource – 웹 상의 모든 콘텐츠**

Ex) input 'http://dreamhack.io/index.html' -> 'dreamhack.io'의 '/index.html' 리소스 요청 수행

- URL(Uniform Resource Locator) – 웹 리소스를 가리키는 주소.

대표적인 웹 리소스들:

- HTML(Hyper Text Markup Language) – 웹 문서의 뼈대
- CSS – HTTP 표시 방법을 정의하는 스타일 시트 언어
- JS – 페이지 내에서의 행위 설정

- **URI(Uniform Resource Identifier)**

리소스를 식별하기 위한 식별자. URL은 URI의 하위개념(웹 사이트의 주소로 웹에 접속하는 것은 URL이자 URI를 이용한 것)

웹 URI 구성 요소:

- Scheme – 이용할 프로토콜 종류
- Host – 웹 서버의 호스트(서버 주소) 정보
- Port – 웹 서버의 포트 정보
- Path – 웹 서버의 경로
- Query – 웹 서버에 전달하는 파라미터(추가 정보). URI에서 '?' 뒤에 붙어
- Fragment – 메인 리소스 내의 서브 리소스 접근에 대한 정보. URI에서 '#' 뒤에 붙어

- **Encoding – 다른 형태나 형식으로 변환하는 처리 및 처리 방식**

-Encoding: 알고리즘이 공개되어 있음. 원래의 정보로 복원 가능

-Decoding: Encoding 된 거 다시 원래 형태로 변경

-Encryption: 양방향 암호. 일치한 알고리즘과 유효 키 있어야 decoding 가능

-URL Encoding(percent encoding) – URI 구조에서 예약어(구분자)로 사용되는 문자들을 전송할 때

-HTML entity Encoding – HTML 문서 내에서 HTML의 태그의 문자로 인식되지 않게 하기 위함.

- **HTTP, HTTPS(HTTP Secure Socket Layer) – 웹에서의 통신을 정의하는 규칙 체계**

  URI의 Scheme(Protocol)에 해당. 컴퓨터 내부 또는 사이에서 데이터가 어떻게 교환되는가?

  TCP(Transmission Control Protocol – 인터넷에서 컴퓨터들이 정보를 주고받는 통신규약) 또는 TLS(암호화된 TCP)를 사용해 통신하고, 기본 포트로 80(HTTP), 443(HTTPS) 포트를 사용함(무조건 이 포트를 이용하는 것은 아님)

- **HTTP Request – 사용자가 서버에 요청**

  HTTP 구조에서 각 줄은 CRLF(Carriage Returen Line Feed – 줄 바꿈)로 줄 바꿈이 이루어져야

- (첫째 줄) Method– 수행하려는 동작/ Path– 요청하는 웹 리소스 경로/ Version– HTTP 버전
  - Options – 요청하는 리소스가 허용하는 메소드 목록 반환
  - Head – Response의 Body는 받지 않고 Header만 받음(ex. 서버의 상태확인 etc)
  - Get – 리소스 요청 (ex. 게시글 및 프로필 보기, 이미지 etc)
  - Post – 특정 리소스의 내용을 보낸 값으로 설정 (ex. 생성 및 업데이트 etc)
  - Patch – 특정 리소스의 내용 중 보낸 값의 key만 변경 (ex. 게시글 업데이트 etc)
  - Delete – 특정 리소스를 삭제
  - Trace – 요청 받은 값을 Response의 Body로 다시 되돌려 줌
- (둘째 줄)Heather- '이름:값' 형식. 여러 줄 사용할 수도.
  - Host – 데이터를 보내는 서버의 주소를 의미
  - Cookie – 사용자를 식별하기 위해 사용하는 정보
  - User-Agent – 사용자가 사용하는 프로그램 정보
  - Referer – 페이지 이동시 이전 URI 정보
  - Content-Type – 사용자가 전달하는 데이터의 처리 방식과 형식

- (나머지 줄) Body – 사용자의 데이터 담는 부분

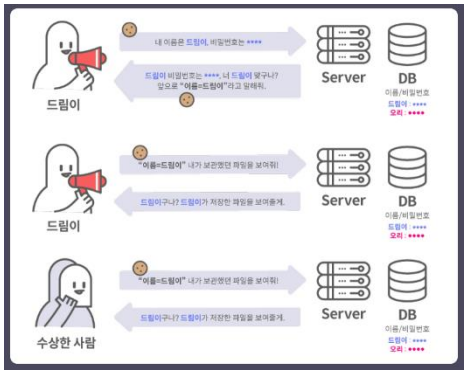
HTTP Response – 사용자의 요청에 대한 서버의 응답

- Version – HTTP의 버전
- Status code – 사용자 요청에 대한 서버 처리 결과
  - 200번대 – 사용자의 요청에 대한 서버의 처리 성공
    - 200 OK
    - 201 Created
  - 300번대 – 사용자가 요청한 리소스가 다른 경로로 변경된 경우. 300번 코드 수신 시 Response Header의 Location 헤더로 리다이렉션.
    - 301 Moved Permanently
    - 302 Found
  - 400번대 – 사용자가 서버에 요청하는 구조 또는 데이터가 잘못됨.
    - 400 Bad Request – 사용자가 전달한 데이터 또는 구조의 잘못된 문법.
    - 403 Forbidden – 사용자가 권한이 없음
    - 404 Not Found – 사용자가 요청한 리소스의 경로에 응답할 데이터가 없음
    - 405 Method Not Allowed – 사용자가 요청한 Method가 서버에서 허용되지 않음
  - 500번대 – 서버의 에러와 관련된 영역
    - 500 Internal Server Error – 서버의 에러발생

◆ 503 Service Unavailable – 서버가 사용자의 요청 처리할 준비가 안됨.

- Header – 사용자와의 상호작용을 위한 데이터 저장 ex) 사용자가 응답 데이터 처리하는 방식 또는 형식에 대한 정보, 서버에서 사용자를 식별하기 위한 쿠키 발급 정보
  - Content-Type – 서버의 응답 데이터를 웹 브라우저에서 처리할 방식과 형식
  - Content-Length – 서버가 사용자에게 응답해주는 데이터의 길이
  - Server – 서버가 사용하는 소프트웨어의 정보
  - Allow – 허용되는 Method 목록을 사용자에게 알려줄 때
  - Location – 300번대 응답코드 시 변경된 리소스 주소
  - Set-Cookie – 사용자에게 쿠키 발급. 해당 헤더를 받은 웹 브라우저가 해당 쿠키를 저장
- Body – 서버의 응답 데이터

- **Cookie – 사용자 인증상태를 유지함. 사용자의 브라우저에 저장됨.**



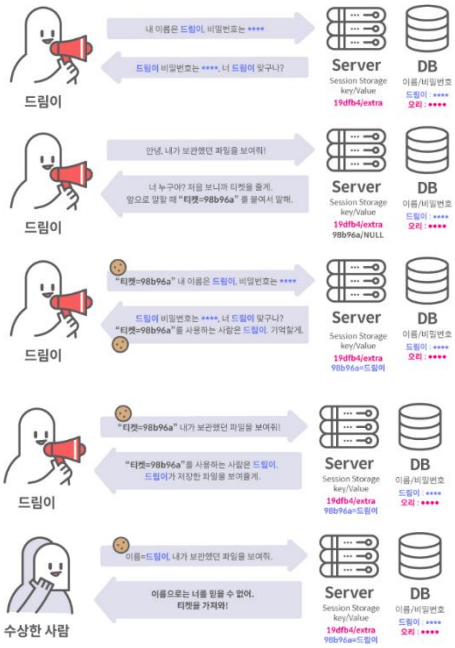
웹 브라우저는 HTTP response의 set-cookie header나 JS의 document.cookie를 통해 저장됨.

인증된 이후 HTTP request를 보낼 때, 쿠키를 브라우저가 자동으로 헤더에 추가함.

- **Session – 서버에 저장되는 데이터.**

쿠키의 경우 사용자가 임의 사용자로 인증된 것처럼 요청을 조작할 가능성.

서버가 유추할 수 없는 랜덤 문자열 키를 만들고, 브라우저는 키를 쿠키에 저장하여 request에 요청을 보내면 서버에서 키의 인증 상태를 확인.



- HTTP의 경우 평문으로 통신을 하기 때문에 MITM 공격에 취약함. HTTPS는 암호화.

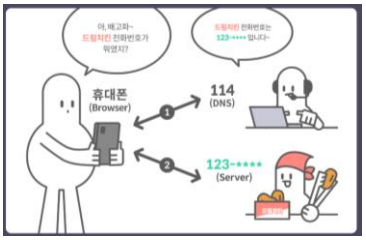
- **Domain Name / Host Name**

URI 구성요소 중 'Host'는 웹 브라우저가 어디에 연결할지 정함. Domain Name이나 IP(Internet Protocol) Address가 Host에 사용됨.

Ex) http://example.com/path1?search=1#fragment => Host: example.com

- IP Address – 네트워크 상에서 장치들을 식별하기 위한 불규칙한 숫자로 이루어진 주소. 의미를 부여해 사용하기 위해 Domain Name을 사용.
- Domain Name으로 Host 조회할 때는 Domain Name 과 IP Address를 매핑해 저장하는 DNS(Domain Name Server)에서 조회해 등록된 IP Address를 가져와 사용함. 'nslookup'으로 Domain Name 정보 확인 가능.

  Ex) 웹 브라우저에서 http://example.com/에 접속하면 DNS에서 wxample.com(Domain Name)의 아이피를 가져와 이와 통신함.



- **Web Server – 사용자의 HTTP 요청을 해석하여 처리한 후 응답하는 역할**

  Ex) nginx, Apache, Tomcat, IIS

  웹 서버는 사용자의 request를 웹서버 자체로 처리할지, 알맞은 내부 서비스로 연결할지 결정.

  Ex) 클라이언트가 접근한 URI가 .html을 가진 리소스에 요청 시 해당경로의 html 반환, .php 확장자를 가진 리소스에 요청 시 php 엔진을 통해 요청 처리, /payment/ 경로로 시작하는 요청은 payment 처리를 위한 어플리케이션에게 요청을 연결.

- **Web Application – 사용자의 요청을 동적으로 처리할 수 있도록 만들어진 어플리케이션**

  웹 어플리케이션을 작성할 때사용자의 request를 동적으로 처리하기 위해 Web Application Language – PHP, NodeJS, Python, Java가 사용됨.

- **DataBase Management System(DBMS) – DB 내의 데이터 조회, 수정, 삽입을 용이하게 사용할 수 있도록 도와줌.**

  Ex) MySQL, MS-SQL

<웹 해킹 공격 벡터>

- **Client-side Attack**

  서비스 사용자에 대한 공격

  사용자가 웹서버로부터 제공받는 데이터가 변조됨. -> 이를 웹 브라우저에서 render되는 과정에서 취약점 발생.

- **Server-side Attack**

  서비스를 운용하는 서버에 대한 공격

  공격 성공 시, 어플리케이션 코드 및 타 사용자의 정보 유출, 서버 탈취로 이어질 수

  서비스 형태마다 특별한 공격 방법이 존재할 수