

과정명	
05차시	웹 보안운영

### <1> 웹 서버와 웹 보안

#### [1] 웹의 정의

- 웹 서버는 웹 브라우저 클라이언트로부터 HTTP 요청을 받아들이고, HTML 문서와 같은 웹 페이지에서 흔히 찾아볼 수 있는 콘텐츠를 제공하는 서버를 뜻함
- 웹은 전 세계에 흩어져 있는 종업원 및 연구자와 연구 결과나 아이디어를 공유할 수 있는 방법을 모색하는 목적의 프로젝트로 시작하였으며, 처음 프로젝트 계획 당시 웹은 'Hyper Text Project'라고 불렀는데 여기서 하이퍼텍스트는 링크를 통해 한 문서에서 다른 문서로 접근할 수 있는 문서를 뜻함
- HTML은 가장 대표적인 하이퍼텍스트로 웹 페이지를 만들 때 사용되며 텍스트뿐 아니라 링크 정보를 이용해 이미지 및 동영상 등의 다른 자원들을 동시에 나타낼 수 있음
- URL은 웹 상에서 존재하는 각각의 하이퍼텍스트 및 이미지, 동영상 등의 자원에 접근하기 위한 주소로 사용됨

#### [2] 웹 보안

- 웹은 공개된 표준에 의해 모두에게 개방된 환경이기 때문에 접근이 쉬우나 웹 자체의 보안에 대한 고려가 없기 때문에 보안에 다소 취약함
- 웹의 보안 취약성은 악용되어 많은 보안 사고가 발생함

- 웹 보안 위협은 소극적 공격, 공격적 공격 또는 위협의 장소에 따라 분류할 수 있음

##### (1) 소극적 공격과 공격적 공격

###### 1) 소극적 공격

- 브라우저와 서버 사이의 네트워크 트래픽을 도청하여 제한된 웹 사이트 상의 정보에 접근을 허락받는 것

###### 2) 적극적 공격

- 다른 사용자로 위장
- 클라이언트와 서버 사이의 전송 메시지 변경
- 웹 사이트의 정보 수정

##### (2) 보안 위협 장소에 따른 분류

###### 1) 웹 서버 네트워크 트래픽

###### 2) 웹 브라우저 네트워크 트래픽

###### 3) 브라우저와 서버 사이의 네트워크 트래픽

- 웹에 대한 위협 비교

	위협	피해 사항	대응 방법
무결성	<ul style="list-style-type: none"> <li>◆ 사용자 데이터의 변경</li> <li>◆ 트로이 목마 브라우저</li> <li>◆ 메모리의 변경</li> <li>◆ 전송 중 메시지 트래픽의 변경</li> </ul>	<ul style="list-style-type: none"> <li>◆ 정보의 손실</li> <li>◆ 기계에 대한 침해</li> <li>◆ 다른 위협에 대한 취약성</li> </ul>	암호적 검사
기밀성	<ul style="list-style-type: none"> <li>◆ Net 도청하기</li> <li>◆ 서버에서 정보 훔치기</li> <li>◆ 클라이언트에서 데이터 훔치기</li> <li>◆ 네트워크 구성에 대한 정보 알아내기</li> <li>◆ 서버와 통신 중인 클라이언트 알아내기</li> </ul>	<ul style="list-style-type: none"> <li>◆ 기밀성 침해</li> <li>◆ 방해하기</li> <li>◆ 성가시게 하기</li> <li>◆ 사용자의 작업 방해</li> <li>◆ 사용자에게 대한 식별 오류</li> </ul>	암호화, 웹 프록시
서비스	<ul style="list-style-type: none"> <li>◆ 사용자 스레드(threads) 중단시키기</li> </ul>	<ul style="list-style-type: none"> <li>◆ 가짜 정보를 진짜로 오인</li> </ul>	예방하기

거부	<ul style="list-style-type: none"> <li>과잉의 가짜 위협을 기계에 보내기</li> <li>메모리나 디스크 자리 차지하기</li> <li>DNS 공격을 통한 기계 고립화</li> </ul>		어려움
인증	<ul style="list-style-type: none"> <li>합법적 사용자로 위장하기</li> <li>데이터 위조</li> </ul>		암호적 기술

- 웹 보안 방법은 웹의 응용 범위나 TCP/IP 프로토콜 계층에서 웹이 있는 상대적 위치에 따라 구분됨

#### (1) IP보안(IPSec)

- 종단 사용자와 응용에 투명성을 제공해 주고 범용 해결책 제시
- 필터링 할 수 있는 기능을 가지고 있어 오직 선별된 트래픽만 IP보안을 처리하는 오버헤드가 추가됨

#### (2) 소켓 계층과 전송소켓 계층 보안

- 네스케이프와 마이크로소프트 익스플로어에 소켓 계층이 장착되어 있어 대부분 웹 서버는 이 프로토콜로 구현됨

#### (3) 응용 프로그램별 보안 서비스

- 해당 응용 프로그램에 내장
- 주어진 응용 프로그램의 특정 요구사항에 맞게 변경 가능
- 웹 보안 측면의 중요한 예로 안전한 전자상거래가 있음

## <2> HTTP의 개념

### [1] HTTP 프로토콜의 특성

- HTTP는 웹 상에서 클라이언트와 서버 사이에 정보를 교환하기 위한 프로토콜로서 TCP/IP와 관련된 하나의 응용 프로토콜을 뜻함
- HTTP에 의해 2대의 컴퓨터가 통신하는 경우 어느 한쪽은 반드시 클라이언트가 되고 다른 한쪽은 서버가 되는데 때에 따라서 각 컴퓨터의 역할이 바뀔 수는 있지만 한 번의 통신에서는 클라이언트와 서버의 기능이 정해져 있음
- 클라이언트가 요청 메시지를 먼저 보내야 그 결과로 응답 메시지를 서버가 전달함

### - HTTP 통신 구조

#### (1) 비연결형 기반 통신

- HTTP 요청에 대해 TCP 통신을 설정한 후 요청에 대한 응답이 처리되면 TCP 연결을 끊어버리는 형태
- 서버와 클라이언트가 독립적
- 트랜잭션이 연결된 지속적인 통신에는 부적합하나 인터넷과 같은 다수의 사용자를 대상으로 사용자가 원할 때 필요한 HTML 문서를 서버로부터 불러오는 네트워크 부하가 거의 없는 클라이언트 풀 방식의 서비스에는 적합

#### (2) 무상태 통신

- 기본적인 요청에 대한 응답
- HTTP 트랜잭션이 종료되므로 연속적인 작업에 필요한 트랜잭션 상태 정보를 관리하기 위한 웹 서버의 부하가 필요 없음
- 사용자 상태 정보 관리가 요구되는 작업을 지속적으로 유지하기 위해서는 쿠키 또는 CGI 스크립트 상에서 지원하는 세션 정보 이용

- 정보 교환을 위한 메시지 구조는 요청과 응답의 형태를 가지고 있음

- 클라이언트가 서버와 통신할 때는 TCP 연결→HTTP 메시지 전송→서버 응답→서버가 보낸 응답 분석→TCP 연결 닫기 혹은 다른 요청을 위한 재사용의 과정을 수행함

## [2] HTTP 요청·응답

- HTTP 통신은 클라이언트의 HTTP 요청 메시지에 대해 서버의 HTTP 응답 메시지를 전달받는 형식을 뜻함

- 요청 및 응답 메시지는 HTTP 통신의 핵심이며 각 메시지는 시작라인, 헤더, 메시지 바디로 구성됨

### (1) 시작라인

- 요청 메시지 경우, HTTP 명령어와 URL, HTTP 버전 번호 정보로 구성
- 응답 메시지 경우, 응답 내용의 상태 정보로 구성
- 사용자 목적을 알리는데 사용하는 GET, HEAD, POST 메소드가 포함

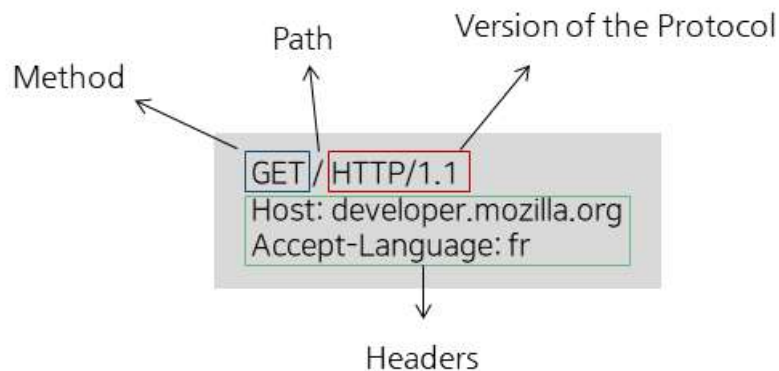
### (2) 헤더

- 추가적인 정보
- 클라이언트 이름과 버전, 자료요청 변경기준일, 리퍼러, 쿠키, 사용자 인증, 캐시 등 정보 포함

### (3) 메시지 바디

- 요청이나 응답에 필요한 내용 포함
- POST 방식의 경우, 메시지의 인코딩 데이터 스트림으로 구성
- 응답 메시지 경우, 응답 문서 내용으로 구성

- 웹 브라우저에서 사용되는 HTTP 요청 메시지



### (1) Method

- 클라이언트가 수행하고자 하는 동작을 정의한 GET, POST, OPTIONS, HEAD를 지칭

메소드	내용
GET	서버가 자원을 전송해 줄 것을 요청
POST	요청 바디를 통해 클라이언트가 서버로 데이터를 전송
HEAD	서버가 자원 내용은 전송하지 않고 헤더만 전송해 줄 것을 요청
OPTIONS	서버가 지원하는 메소드를 정의
TRACE	클라이언트가 보낸 요청을 그대로 반환
PUT	클라이언트가 서버의 지정한 URL로 자원을 전송
DELETE	서버의 자원을 삭제
CONNECT	프록시 터널링을 위해 예약된 메소드

#### 1) GET

- 가장 일반적인 HTTP Request 형태
- 웹 브라우저에 요청 데이터에 대한 인수를 URL을 통해 전송
- 각 이름과 값을 '&'로 결합하여 글자 수를 255자로 제한
- 데이터가 주소 입력란에 표기되기 때문에 최소한의 보안도 매우 취약

#### 2) POST

- URL에 요청 데이터를 기록하지 않고 HTTP 헤더에 데이터를 전송
- 내부의 구분자가 각 파라미터를 구분하여 서버가 각 구분자에 대한 내용을 해석하여 데이터를 처리

- GET에 비해 상대적으로 처리 속도가 늦음
- 인수 값을 URL을 통해 전송하지 않기 때문에 다른 사람이 링크를 통해 해당 페이지를 볼 수 없음

(2) Path

- 가져오려는 리소스의 경로로 프로토콜, 도메인, TCP 포트인 요소들을 제거한 리소스의 URL

(3) Version of the Protocol

- HTTP 프로토콜의 버전

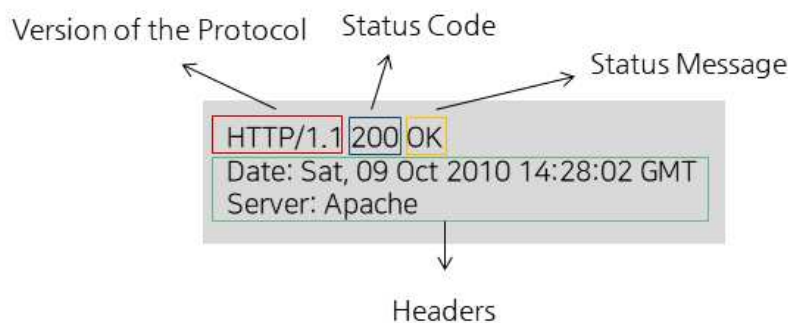
(4) Headers

- 서버에 대한 추가 정보를 전달하는 선택적인 헤더들

(5) Etc

- POST와 같은 몇 가지 메소드를 위해 전송된 리소스를 포함하는 응답의 본문과 유사한 본문

- 웹 브라우저에서 사용되는 HTTP 응답 메시지



(1) Version of the Protocol

- HTTP 프로토콜의 버전

(2) Status Code

- 요청의 성공 여부와 그 이유를 나타내는 상태 코드
- 대표적인 상태 코드

값의 범위	구분	내용
100~199	정보 전송(information)	◆ 요청을 받아 처리가 계속되고 있음
200~299	성공(success)	◆ 요청이 정상적으로 처리됨
300~399	리다이렉션(redirection)	◆ 요청을 처리하기 위해 클라이언트가 다른 URL로 이동되어야 함
400~499	클라이언트 오류(client error)	◆ 클라이언트의 요청에 문제가 있어 처리를 할 수 없음
500~599	서버 오류(server error)	◆ 서버 측에서 오류가 발생하여 요청을 처리할 수 없음

(3) Status Message

- 아무런 영향력이 없는 상태 코드의 짧은 설명을 나타내는 상태 메시지

(4) Headers

- 요청 헤더와 비슷한 HTTP 헤더들

(5) Etc

- 선택 사항으로 가져온 리소스가 포함된 본문

### <3> 웹 서버의 공격 유형과 서버 관리

[1] 웹 서버 공격

- 웹 서버 공격은 웹 사용자 클라이언트의 취약점을 이용한 사용자 컴퓨터 공격과 웹 서버의 취약점을 이용한 웹 서버 공격으로 분류할 수 있으며, 웹 서버에 대한 공격은 해당 네트워크의 방화벽을 우회하여 내부 네트워크를 공격하는 시작점이 되므로 특히 주의해야 함
- 웹 서버 관련 공격은 최신 해킹 경향에 따라 끊임없이 새로운 공격 기법이 발견되므로,

항상 최신 보안 경향에 관심을 두고 매년 발표되는 OWASP TOP10에 대해 이해하고 있어야 함

#### - 웹 서버 공격 유형

##### (1) 부적절한 파라미터 조작

- HTML 요청을 변조하여 보안 매커니즘을 우회하는 기법
- 파라미터를 조작하여 시스템 명령 실행
- Tainted 인자의 사용 여부 확인, 소스 코드 상세 분석, WebScarab과 같은 도구를 사용한 Tainted 인자 검사로 취약성 판단
- 데이터 유형 검증, 허용된 문자셋 검증, 최대/최소 길이 검증, Null 값의 허용 여부 검증, 중복 허용 여부 검증, 숫자의 범위 검증 등으로 대응

##### (2) 원격지 파일의 명령 실행

- 게시판 소스 코드 중 Include문을 이용하여 Passthru나 System과 같이 원격지에서 명령 실행이 가능한 함수를 추가하여 원격지에 명령을 실행
- 게시판 소스 코드에 Include문이 있는지와 Include문에 의해 원격지의 파일을 포함할 수 있는지 확인하는 것으로 취약성 판단 가능
- PHP의 경우 php.ini 파일에서 allow\_url\_fopen 옵션을 off값으로 설정하여 대응

##### (3) SQL Injection

- 공격자가 입력값을 조작하여 원하는 SQL 구문을 실행하는 기법
- 검색어 및 로그인 필드에 큰 따옴표, 작은 따옴표, 세미클론을 입력하여 DB 오류 발생 여부를 확인 하는 것으로 취약성 판단 가능
- 사용자의 입력에 특수 문자가 포함되어 있는지 검증하거나 SQL 서버의 오류 메시지 미표시, 일반 사용자 권한으로 시스템 저장 프로시저 접근 불허 등으로 대응

##### (4) 파일 업로드

- 해당 게시판의 웹 애플리케이션과 동일한 언어의 스크립트 파일을 업로드 한 후 다시 이를 SSI특성을 이용하여 실행시킴으로써 웹 서버의 내부 명령어를 실행시켜 공격
- 게시판에 글쓰기 권한과 파일 첨부 기능이 있는지와 확장자가 jsp, php, asp, cgi 등의 파일들이 업로드가 가능한지 확인하는 것으로 취약성 판단
- 업로드되는 파일의 확장자 검증과 업로드 파일을 위한 디렉토리의 실행 권한 제거로 대응

##### (5) 파일 다운로드

- 다운로드를 위한 게시판의 파일을 이용하여 임의의 문자나 주요 파일명의 입력을 통해 웹 서버의 홈 디렉토리를 벗어나 시스템 내부의 다른 파일로 접근하여 다운로드 하는 공격

##### (6) 적절한 인증 없이 주요 기능 허용

- 인증과 관련하여 가장 대표적인 보안 위협

##### (7) XSS

- 웹 브라우저에서 사용자가 입력하는 입력 값에 대한 유효성을 검증하지 않고 그대로 웹 서버로 전송했을 때 발생할 수 있는 보안 취약점
- 일반적으로 자바 스크립트에서 주로 발생하나 VB스크립트, ActiveX 등 클라이언트에서 실행되는 동적 데이터를 생성하는 모든 언어에서 발생 가능

## [2] 웹 서버 관리

### (1) 웹 서버의 종류

#### 1) 단일 스레드 웹 서버

- 한 번에 하나씩 요청을 처리
- 트랜잭션이 완료되면 다음 커넥션이 처리
- 처리 도중에는 다른 모든 커넥션이 무시되므로, 심각한 성능 문제를 만들

## 2) 멀티 프로세스와 멀티 스레드 웹 서버

- 여러 요청의 동시 처리를 위해 여러 개의 프로세스 혹은 고효율 스레드 할당
- 스레드/프로세스는 필요 시 만들어질 수 있고 미리 만들어 질 수도 있음
- 서버가 무수히 많은 동시 커넥션 처리 시, 수 많은 프로세스/스레드가 많은 메모리나 시스템 리소스를 소비해 최대 개수에는 제한을 걸어야 함

## 3) 다중 I/O 서버

- 모든 커넥션이 동시에 활동을 감시 당함
- 커넥션의 상태가 바뀌면 커넥션에 대해 작은 양의 처리를 수행
- 처리가 완료되면 커넥션은 다음 번 상태 변경을 알리기 위해 열린 커넥션 목록으로 복귀
- 스레드와 프로세스의 리소스 낭비가 적음

## 4) 다중 멀티 스레드 웹 서버

- 여러 개의 스레드가 각각 열려 있는 커넥션을 감시하고, 각 커넥션에 대해 조금씩 작업을 수행

## (2) 웹 서버 관리

### 1) 계정 관리

- 웹 서버의 계정 관리와 관련하여 가장 중요한 점은 웹 서버를 실행시키는 운영체제 상의 사용자는 최소 권한을 가져야 한다는 것인데 이는 웹 서버가 아직까지 알려지지 않은 보안 취약점으로 인해 보안 공격자에게 탈취 당할 가능성이 항상 존재하기 때문임
- 웹 서버를 동작시키는 사용자가 최소 권한을 가졌다면 만약 보안공격자가 웹 서버를 탈취해도 권한이 없기 때문에 시스템 설정을 변경할 수 없어 악의적인 행동의 범위가 감소됨

### 2) 파일 관리

- 디렉터리 검색 기능과 불필요하게 설치된 파일 제거 필요
- 디렉터리 검색 기능의 활성화는 외부의 보안 공격자가 내부 디렉터리 내용을 확인할 수 있게 되어 서버의 내부 구조 및 백업 파일 혹은 소스 파일들이 공개됨
- 웹 서버 설치 시 기본적으로 제공되는 매뉴얼 파일 및 예제 스크립트 파일은 보안 공격자에게 웹 서버의 정보 및 운영체제에 대한 정보를 알려주기 때문에 더 큰 보안사고를 만들 수 있어 제거해야 함

### 3) 서비스 관리

- 파일 업로드 혹은 다운로드의 최대 크기를 설정하지 않으면 부주의한 사용자나 악의적 사용자에게 의해 간혹 서비스가 중지되는 사고가 발생할 수 있음
- 소수의 대용량 사용자들이 있다면 전체 시스템이 늦어져 다른 사용자들이 피해를 입기도 함
- 대용량 파일의 허용은 웹 취약점 등으로 인한 보안 공격자에게 의해 내부의 중요 자료가 대량 유출되는 경우도 발생시킬 수 있으므로 불필요한 불법 업로드, 다운로드를 허용하지 않도록 웹 서버의 환경을 설정해야 함