

학습 및 실습 페이지 제작  
미니 프로젝트 최종 결과 보고서

---

2024.09.13

# 목 차

|                                     |    |
|-------------------------------------|----|
| 1. 개요.....                          | 4  |
| 1.1 목적.....                         | 4  |
| 1.2 목표.....                         | 4  |
| 1.3 인력배치.....                       | 4  |
| 2. 프로젝트 계획.....                     | 5  |
| 2.1 프로젝트 일정.....                    | 5  |
| 2.2 프로젝트 작업 분할 (WBS) .....          | 5  |
| 3. 웹 서비스 구축.....                    | 5  |
| 3.1 개요.....                         | 5  |
| 3.2 시스템 구조.....                     | 6  |
| 4. 학습 페이지 제작.....                   | 7  |
| 4.1 메인페이지.....                      | 7  |
| 4.2 학습페이지.....                      | 7  |
| 4.2.1 Command Line Injection.....   | 8  |
| 4.2.2 SQL Injection (SQLi) .....    | 10 |
| 4.2.3 Cross Site Script (XSS) ..... | 12 |
| 4.2.4 File Upload.....              | 14 |
| 4.2.5 Directory Indexing .....      | 16 |
| 4.2.6 File Download .....           | 18 |
| 5. 실습 페이지 제작.....                   | 19 |
| 5.1 실습 문제 요약.....                   | 19 |
| 5.2 실습 환경 구축.....                   | 20 |
| 5.2.1 파일 구조도 .....                  | 20 |
| 5.2.2 Database 항목.....              | 21 |

|  |    |
|--|----|
| <b>5.3 실습 문제 구축</b> .....                  | 21 |
| <b>5.3.1 Command Line Injection</b> .....  | 21 |
| <b>5.3.2 SQL Injection (SQLi)</b> .....    | 23 |
| <b>5.3.3 Cross Site Script (XSS)</b> ..... | 25 |
| <b>5.3.4 File Upload</b> .....             | 27 |
| <b>5.3.5 Directory Indexing</b> .....      | 28 |

# 1. 개요

## 1.1 목적

PHP 언어를 사용하여 웹 사이트를 구축해보고 웹 공격 기법 학습과 웹 해킹 문제 사이트를 제작하여 웹 해킹 기법에 대한 역량 향상을 목표로 추진

## 1.2 목표

- ✓ 웹 기본 개념 및 구조를 학습
- ✓ 취약점 점검 기준(2021 주요정보통신기반시설 취약점 분석 평가 가이드)에 따라 보안 취약점 식별 및 대응방안 등 학습을 통한 이해
- ✓ 학습 페이지를 구축하고, 공격 기법을 활용할 수 있는 문제 풀이 사이트 제작

## 1.3 인력배치

| 직책 | 성명  | 역할                       |
|----|-----|--------------------------|
| 팀장 | 김명지 | 자료 조사, 문제 코드 작성          |
| 팀원 | 김하늘 | 발표, 문제 코드 작성             |
| 팀원 | 노승찬 | 페이지 디자인, 문제 코드 작성        |
| 팀원 | 박선아 | 발표자료 제작, 최종 보고서 작성       |
| 팀원 | 박수아 | 자료 조사, 페이지 디자인           |
| 팀원 | 유건우 | 실습 페이지 구축, 취약점 설명 페이지 제작 |

[표 1.3.1] 인력 배치

## 2. 프로젝트 계획

### 2.1 프로젝트 일정

| 수행 일정         | 수행 내용                  |
|---------------|------------------------|
| 08.12 ~ 08.15 | 프로젝트 계획 및 웹 구조 학습과 이해  |
| 08.16 ~ 08.24 | 웹 취약점 조사 및 웹 사이트 환경 구성 |
| 08.25 ~ 08.30 | 학습 페이지 개발              |
| 08.31 ~ 09.04 | 문제 페이지 개발              |
| 09.05 ~ 09.08 | 웹 사이트 점검 및 마무리         |
| 09.09 ~ 09.20 | 최종 결과 보고               |

[표 2.1.1] 프로젝트 일정

### 2.2 프로젝트 작업 분할 (WBS)

| 구분             | 8월 |    |    | 9월 |    |    |
|----------------|----|----|----|----|----|----|
|                | 1주 | 2주 | 3주 | 4주 | 5주 | 6주 |
| 계획 및 웹 구조 이해   |    |    |    |    |    |    |
| 조사 및 사이트 환경 구성 |    |    |    |    |    |    |
| 학습 페이지 개발      |    |    |    |    |    |    |
| 문제 페이지 개발      |    |    |    |    |    |    |
| 웹 사이트 점검 및 마무리 |    |    |    |    |    |    |
| 최종 결과 보고       |    |    |    |    |    |    |

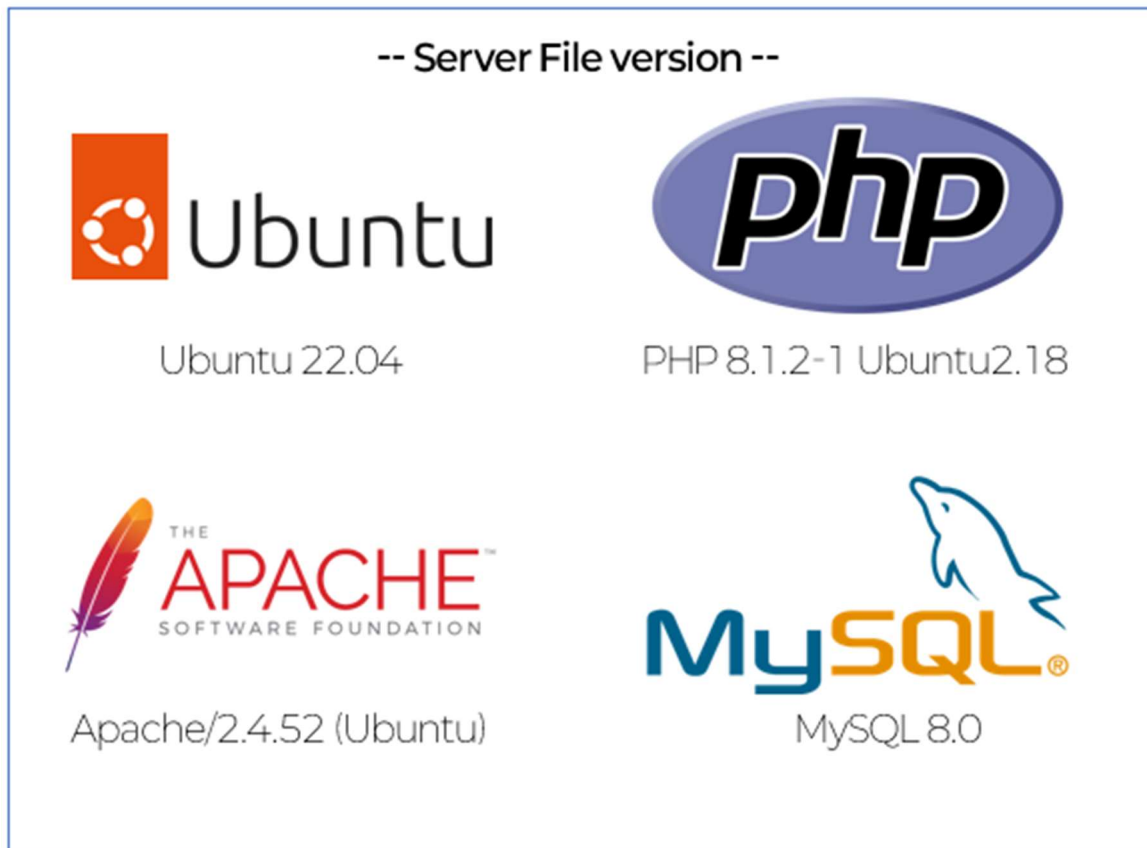
[표 2.1.2] 프로젝트 작업 분할

## 3. 웹 서비스 구축

### 3.1 개요

웹과 웹 취약점에 대한 전반적인 개념을 확인하고, 학습을 기반으로 실제 해당 취약점을 실습할 수 있는 사이트를 생성

### 3.2 시스템 구조



[그림 3.2.1] 시스템 구조

플랫폼: WEB

언어: PHP 8.1.2-1

서버: Apache/2.4.52

Database: MySQL 8.0

호스팅환경: Local Hosting

## 4. 학습 페이지 제작

### 4.1 메인페이지



[그림 4.1.1] 메인 페이지

### 4.2 학습페이지

취약점에 대한 정의, 발생 원인, 동작 방식, 공격 위치, 공격 유형, 사례, 대응방안 설명하고,  
각 취약점에 대한 실습 페이지를 통해 학습에 대한 문제 풀이 가능

## 4.2.1 Command Line Injection

### Web Vulnerability Lab

## Command Injection 취약점

### 1. 정의

웹 애플리케이션에서 시스템 명령을 사용할 때, 세미콜론 혹은 `&`, `&&` 를 사용하여 하나의 Command를 Injection 하여 두 개의 명령어가 실행되게 하는 공격이다.

### 2. 발생 원인

외부 입력값을 검증하거나 제한하지 않고, 운영체제 명령어 또는 운영체제 명령어의 일부로 사용하는 경우 발생한다.

※ 외부 입력값을 검증하지 않았다.

- 사용자가 입력한 데이터에 대해 유효성 검사를 수행하지 않고 그대로 사용하는 경우를 의미한다.
- 입력값에 `&`, `|`, `:` 등의 메타문자가 포함되거나 악성 명령어가 포함될 수 있다.

※ 외부 입력값을 제한하지 않았다.

- 외부 입력값을 사용할 때, 어떤 값이 허용되는지에 대한 명확한 정의와 제한을 두지 않는 경우를 의미한다.
- 입력값에 대한 검증이나 제한이 부족하면, 공격자가 허용되지 않은 명령어나 데이터를 입력하여 시스템에 영향 줄 수 있다.

### 3. 동작 방식

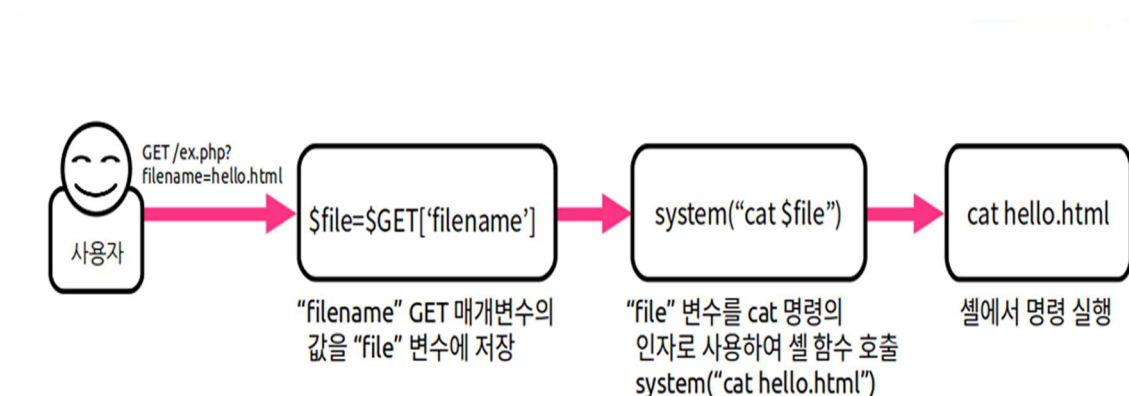
#### 공격을 위한 충족 조건

- 시스템 셸로 전달되는 명령에 사용자 입력값 포함
- 사용자 입력값이 이스케이프 없이 정상적으로 인식
- 웹 애플리케이션을 구동 중인 계정에 시스템 명령 실행 권한 있는 경우

[그림 4.2.1] Command Injection 학습페이지

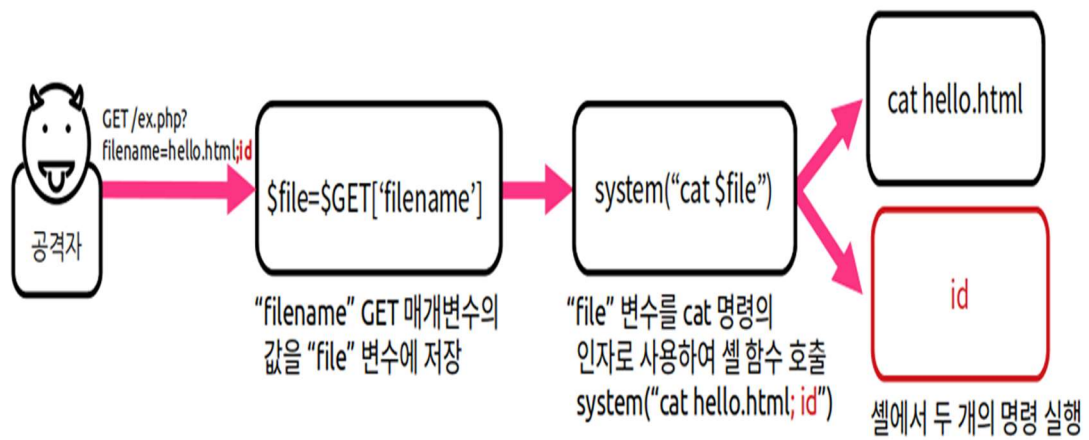
### 학습 페이지 요약 (동작 방식)

일반적 동작 방식과 악의적인 동작 방식 확인을 통해 학습에 대한 이해도를 높임



[그림 4.2.2] Command Injection 일반적 동작





[그림 4.2.3] Command Injection 악의적 동작

**학습 페이지 요약 (대응 방안)**

- ✓ 명령으로 전달되는 입력에 강한 입력 유효성 검사
- ✓ 최소 권한 사용
- ✓ 특수문자 이스케이프 처리 또는 요청 차단
- ✓ 자주 업데이트 및 패치

## 4.2.2 SQL Injection (SQLi)

### Web Vulnerability Lab

## SQL Injection 취약점

### 1. 정의

**SQL Injection**은 공격자가 웹 애플리케이션의 입력 필드를 통해 악의적인 SQL 구문을 삽입하여, 데이터베이스를 조작하거나 민감한 정보를 획득할 수 있는 취약점이다.

### 2. 발생 원인

- 사용자 입력을 신뢰하여 SQL 쿼리를 직접 구성
- SQL 쿼리 내에서 입력값에 대한 유효성 검증 및 필터링 미흡
- 매개변수화된 쿼리나 준비된 문(statement)을 사용하지 않음

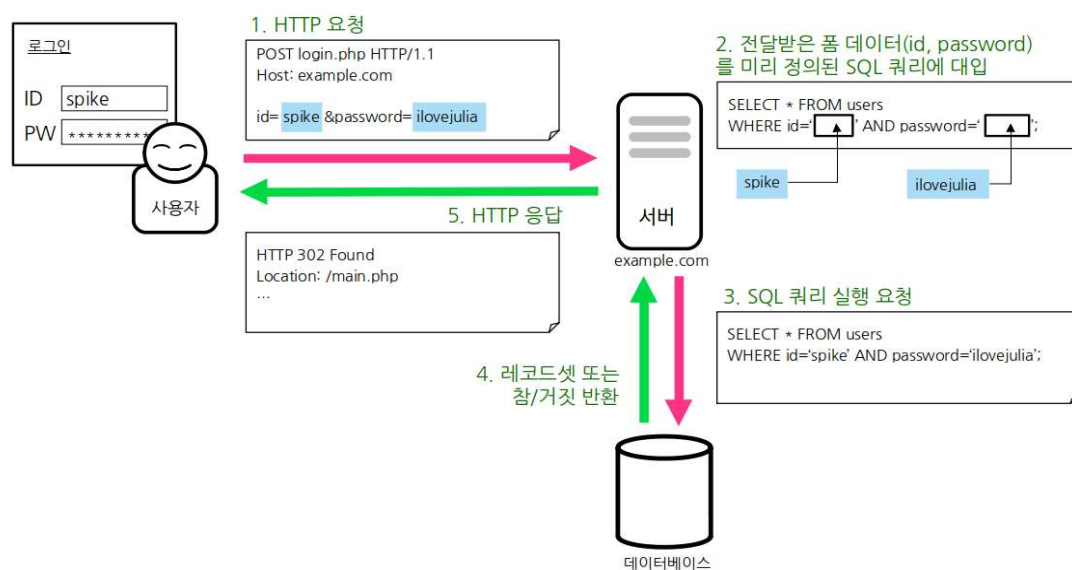
### 3. 동작 방식

#### 3.1 정상적 동작

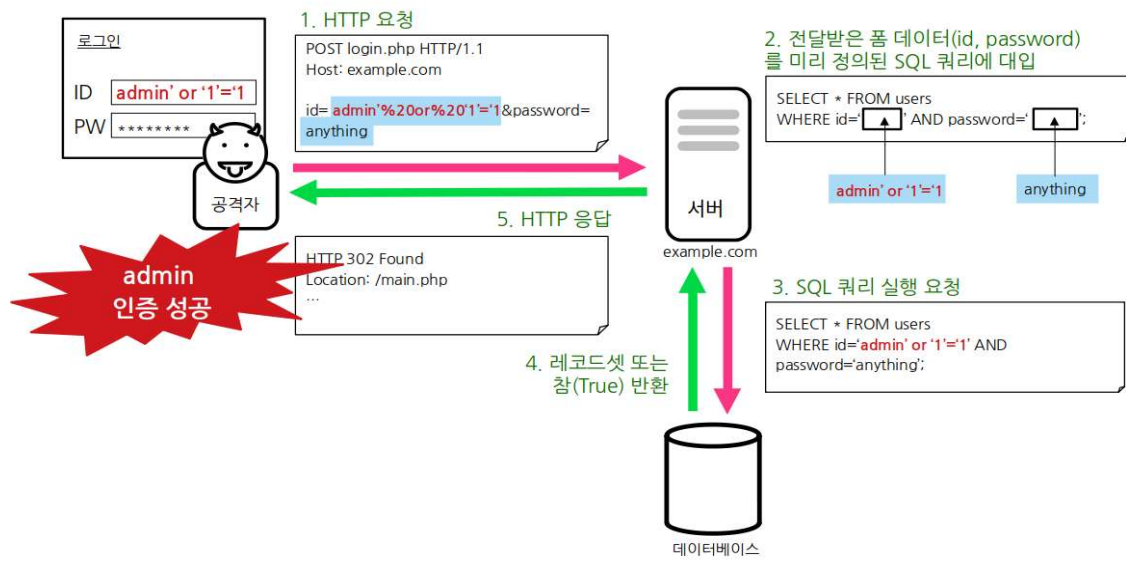
[그림 4.2.4] SQL Injection 학습페이지

### 학습 페이지 요약 (동작 방식)

일반적 동작 방식과 악의적인 동작 방식 확인을 통해 학습에 대한 이해도를 높임



[그림 4.2.5] SQL Injection 일반적 동작



[그림 4.2.6] SQL Injection 악의적 동작

#### 학습 페이지 요약 (대응 방안)

- ✓ 매개 변수화된 쿼리
- ✓ 입력값 검증
- ✓ ORM 사용
- ✓ 최소 권한 원칙 사용

#### 4.2.3 Cross Site Script (XSS)

### XSS 취약점

#### 1. 정의

**XSS** 은 클라이언트 사이드 취약점 중 하나로, 공격자가 웹 리소스에 악성 스크립트를 삽입해 이용자의 웹 브라우저에서 해당 스크립트를 실행하여 공격자가 해당 취약점을 통해 특정 계정의 세션 정보를 탈취하고 해당 계정으로 임의의 기능을 수행할 수 있는 취약점이다.

| 종류            | 설명   |
|---------------|--|
| Stored XSS    | XSS에 사용되는 악성 스크립트가 서버에 저장되고 서버의 응답에 담겨오는 XSS                                 |
| Reflected XSS | XSS에 사용되는 악성 스크립트가 URL에 삽입되고 서버의 응답에 담겨오는 XSS                                |
| DOM-based XSS | XSS에 사용되는 악성 스크립트가 URL Fragment에 삽입되는 XSS<br>- Fragment는 서버 요청/응답에 포함되지 않는다. |
| Universal XSS | 클라이언트의 브라우저 혹은 브라우저의 플러그인에서 발생하는 취약점으로 SOP 정책을 우회하는 XSS                      |

#### 2. 발생 원인

- XSS 공격은 이용자가 삽입한 내용을 출력하는 기능에서 발생한다.
- 클라이언트는 HTTP 형식으로 웹 서버에 리소스를 요청하고 서버로부터 받은 응답, 즉 HTML, CSS, JS 등 웹 리소스를 시각화하여 이용자에게 보여준다.
- 이 때, HTML, CSS, JS와 같은 코드가 포함된 게시물을 조회할 경우 이용자는 변조된 페이지를 보거나 스크립트가 실행될 수 있다.

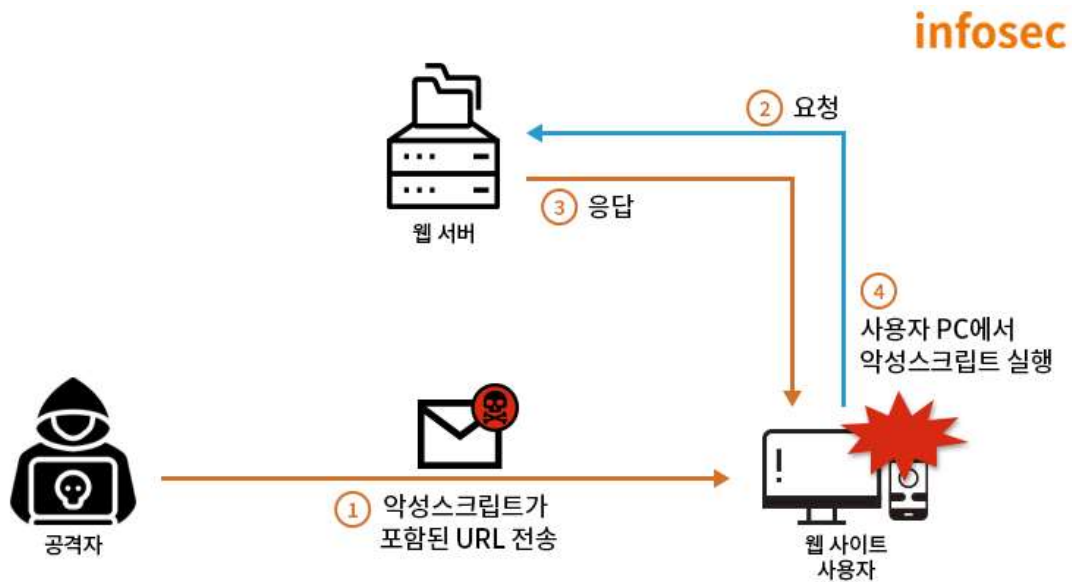
#### 3. 공격 방식

취약한 웹 사이트를 조작해 사용자에게 악성 자바스크립트를 반환하는 방식으로 동작

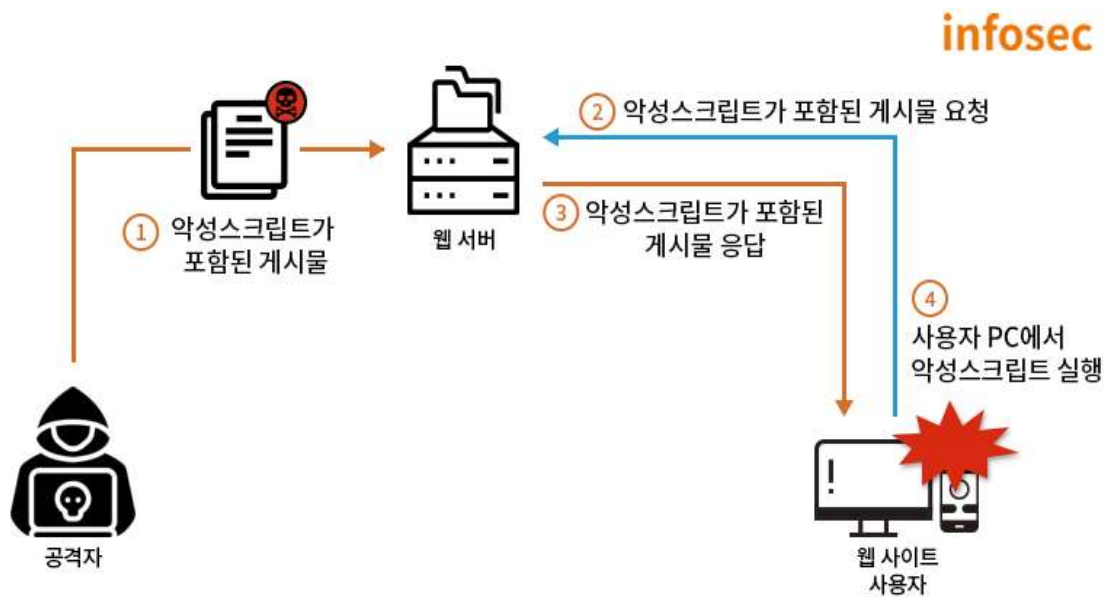
#### [그림 4.2.7] Cross Site Script 학습페이지

#### 학습 페이지 요약 (동작 방식)

XSS 취약점 종류 중 대표적인 Reflected XSS 설명과 Stored XSS 공격 방식에 대한 설명 및 학습



[그림 4.2.8] Reflected XSS 동작



[그림 4.2.9] Stored XSS 동작

#### 학습 페이지 요약 (대응 방안)

- ✓ 사용자 입력값 검증
- ✓ 쿠키에 HTTP Only 플래그 설정
- ✓ 올바른 데이터 출력값 처리
- ✓ CSP(Content Security Policy) 적용

## 4.2.4 File Upload

### File Upload 취약점

#### 1. 정의

파일 업로드 기능이 존재하는 웹 상에서, 서버에서 실행될 수 있는 스크립트 파일(asp, jsp, php 등)이 업로드 되어 실행될 수 있는 취약점이다.

#### 2. 발생 원인

- 파일이 업로드되기 전에 유효성 검사가 구현되지 않음
- 안전하지 않은 방식으로 구현된 유효성 검사
- 파일에 대한 필터링 조치 미흡

#### 3. 동작 방식

##### 3.1 정상적 동작

- 사용자가 자신의 프로필 화면에서 사진 파일(jpeg, png)을 업로드
- 해당 사진 파일은 웹 루트 하위의 profile 디렉토리에 저장됨
- 웹 애플리케이션은 URL을 통해 사진 파일이 저장된 위치에서 파일을 가져와 프로필 조회 화면에 포함시킴

##### 3.2 악의적 동작

- 공격자가 이미지 파일이 아니라 스크립팅 언어로 된 웹셸 파일을 업로드
- 해당 웹셸 파일은 웹 루트 하위의 profile 디렉토리에 저장됨
- 공격자는 서버에 저장된 웹셸 파일을 웹 브라우저에서 열람하거나 임의의 명령 실행
- 시스템 손상 혹은 다른 공격에 필요한 시스템 정보 파악 가능

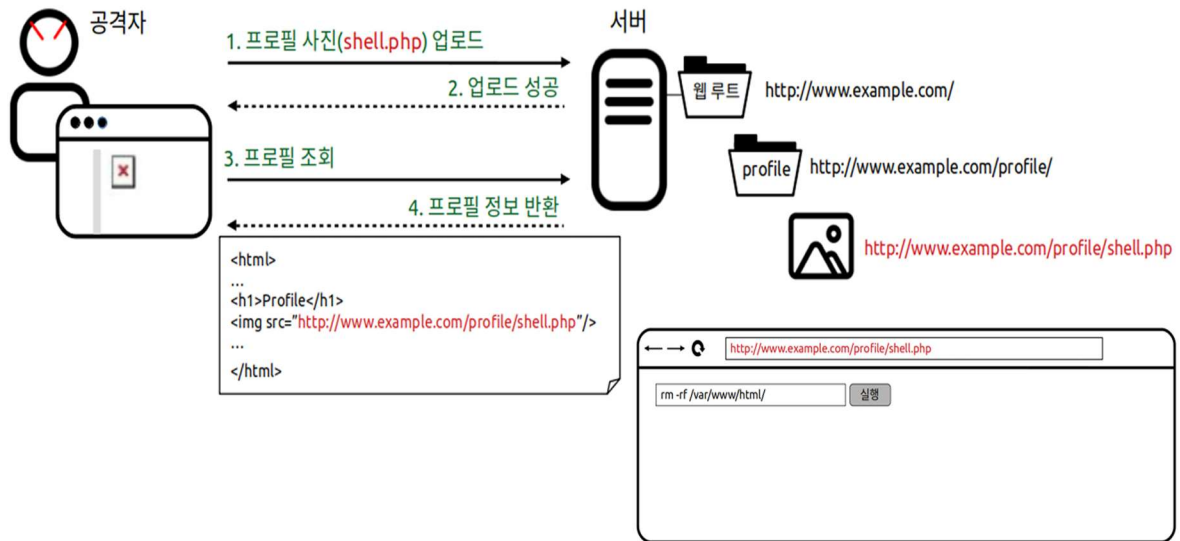
[그림 4.2.10] File Upload 학습 페이지

### 학습 페이지 요약 (동작 방식)

일반적 동작 방식과 악의적인 동작 방식 확인을 통해 학습에 대한 이해도를 높임



[그림 4.2.11] File Upload 일반적 동작



[그림 4.2.12] File Upload 악의적 동작

#### 학습 페이지 요약 (대응 방안)

- ✓ 파일 확장자 검증
- ✓ 파일 유형 검증
- ✓ 파일 Magic Number 검증
- ✓ 파일명 변경 또는 난독화

## 4.2.5 Directory Indexing

### Directory Indexing 취약점

#### 1. 정의

디렉터리 인덱싱(Directory Indexing) 취약점은 웹 서버에서 특정 디렉터리의 콘텐츠가 노출되어, 해당 디렉터리의 파일 리스트가 사용자의 웹 브라우저를 통해 표시되는 보안 취약점을 의미합니다. 이를 통해 민감한 파일이나 서버의 구조가 노출될 수 있습니다.

#### 2. 공격 방식

##### 1. 디렉토리 접근 시도

- 공격자는 웹 서버의 URL 경로를 통해 특정 디렉토리에 직접 접근을 시도합니다.
- 예를 들어, `http://example.com/admin/` 와 같은 경로에 접근할 때, 해당 디렉토리의 인덱스 파일(index.html, index.php 등)이 존재하지 않으면 서버는 디렉토리 내의 파일 목록을 자동으로 표시할 수 있습니다.

##### 2. 파일 목록 열람

- 서버가 디렉토리 인덱싱을 활성화한 경우, 해당 디렉토리의 파일 목록이 웹 브라우저에 표시됩니다.
- 이 목록에는 해당 디렉토리에 존재하는 모든 파일과 서브 디렉토리가 포함되며, 파일 이름, 크기, 수정 날짜 등이 표시될 수 있습니다.

##### 3. 파일 다운로드 및 열람

- 공격자는 파일 목록에서 특정 파일을 선택하여 직접 다운로드하거나 열람할 수 있습니다.
- 특히, 설정 파일이나 데이터베이스 백업 파일 등 민감한 파일이 노출되면 심각한 보안 위협이 발생할 수 있습니다.

##### 4. 추가 공격

- 노출된 파일을 통해 웹 서버의 구성이나 기타 민감한 정보를 파악한 후, 추가적인 공격을 시도할 수 있습니다.
- 예를 들어, 파일 업로드 기능이 있는 디렉토리에 접근하여 악성 코드를 포함한 파일을 업로드하고, 이를 실행하여 서버를 장악하는 등의 2차 공격이 가능해질 수 있습니다.

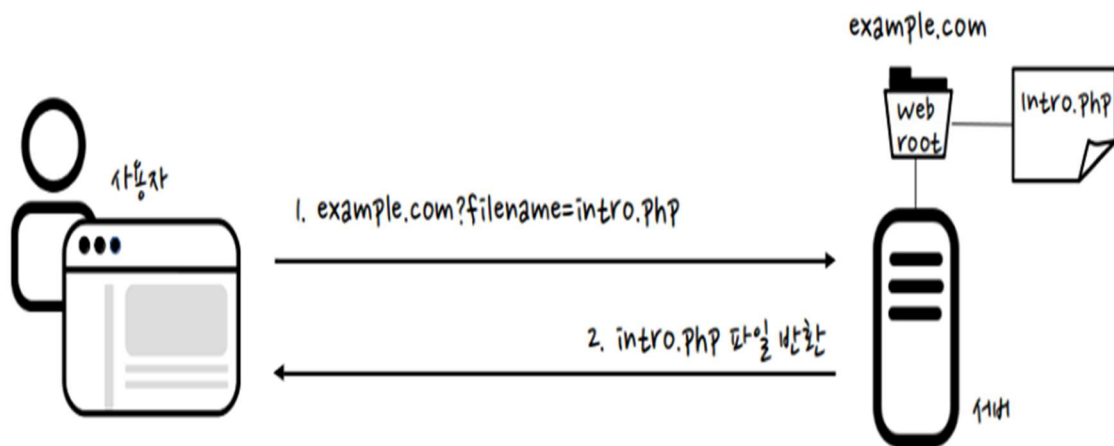
#### 3. 보안 위협

[그림 4.2.13] Directory Indexing 학습 페이지

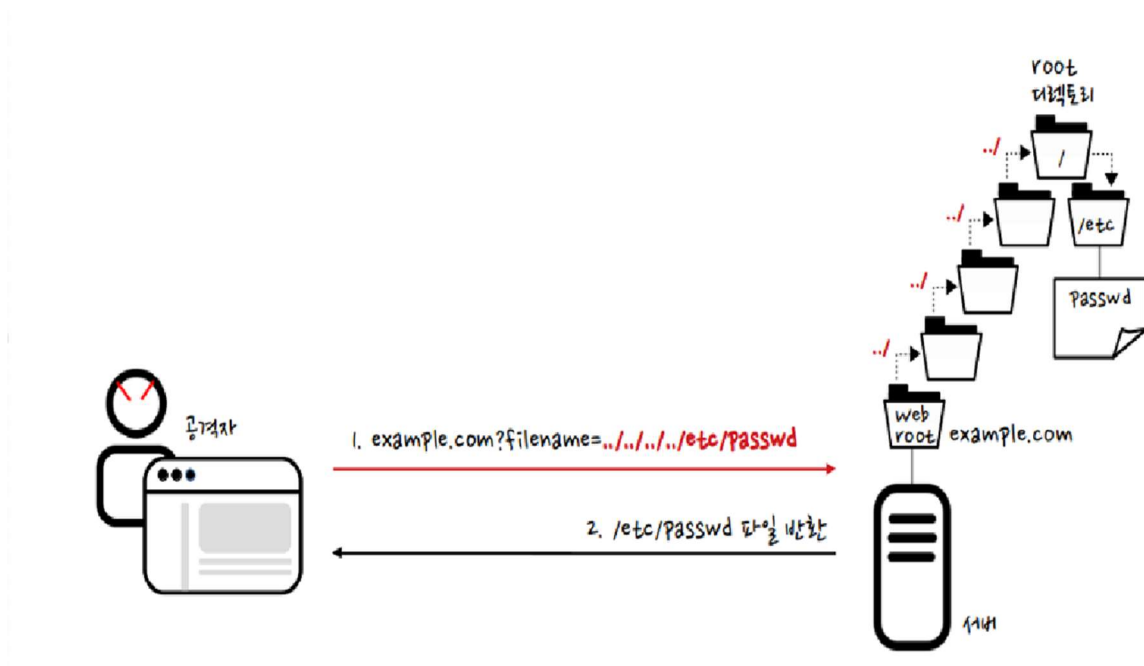
### 학습 페이지 요약 (동작 방식)

일반적 동작 방식과 악의적인 동작 방식 확인을 통해 학습에 대한 이해도를 높임





[그림 4.2.14] Directory Indexing 일반적 동작



[그림 4.2.15] Directory Indexing 악의적 동작

#### 학습 페이지 요약 (대응 방안)

- ✓ 상위 디렉터리 접근 방지
- ✓ 디렉터리 인덱싱 비활성화
- ✓ 보안 헤더 설정 강화
- ✓ .htaccess 파일 사용(Apache 서버)

## 4.2.6 File Download

# Web Vulnerability Lab

## File Download 취약점

### 1. 정의

파일 다운로드 기능 사용 시 임의의 문자나 주요 파일의 입력을 통해 웹 서버의 홈 디렉터리를 벗어나 임의의 위치에 있는 파일을 다운 가능한 취약점(passwd, 중요파일/백업, 데이터베이스, 소스코드 정보 등)

### 2. 발생 원인

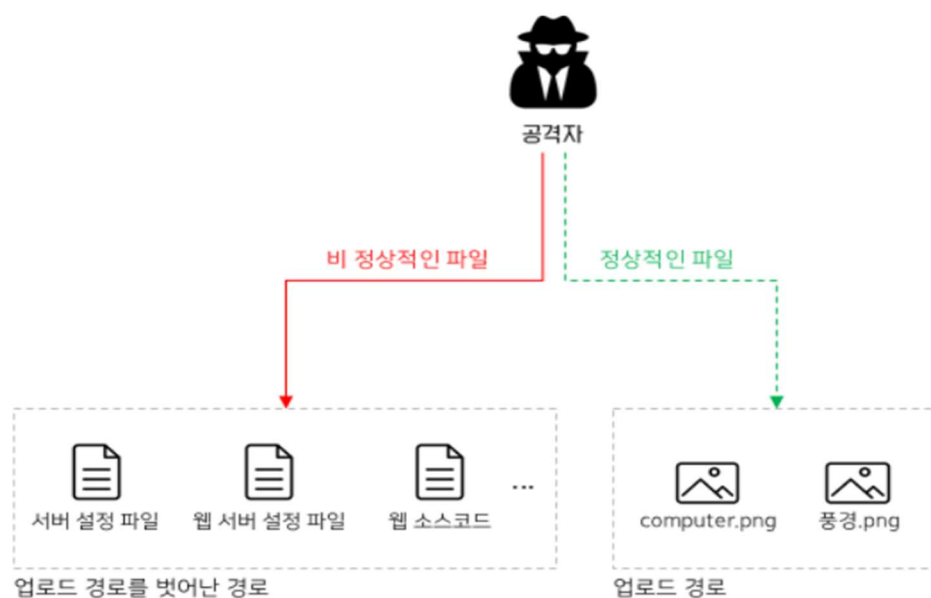
- 파일 다운로드 시 파일의 절대경로 또는 상대 경로가 노출되는 경우
- 다운로드 모듈이 파일의 경로나 이름을 파라미터로 사용하는 경우
- 파일 경로와 파일명 필터링 미흡 (ex. ../ 를 사용해 상위 디렉토리 접근)
- 다운로드 경로가 노출되지 않더라도 구조가 단순하여 파라미터 변조를 통해 접근이 허용되지 않는 파일에 접근 가능할 경우

### 3. 동작 방식

[그림 4.2.16] File Download 학습 페이지

### 학습 페이지 요약 (동작 방식)

악의적인 동작 방식 확인을 통해 학습에 대한 이해도를 높임





[그림 4.2.17] File Download 악의적 동작

#### 학습 페이지 요약 (대응 방안)

- ✓ 파일 이름과 경로명을 데이터베이스에서 관리 및 비교
- ✓ 특정 디렉토리에서만 다운로드가 가능하도록 설정
- ✓ 다운로드 경로를 사용자가 확인할 수 없게 제한
- ✓ ./w%와 같은 특수문자를 필터링 처리

## 5. 실습 페이지 제작

### 5.1 실습 문제 요약

다음 취약점에 대해 총 11개 문제를 제작하였고, 문제 난이도 기준을 다음과 같이 선정

| 문제 난이도 | 기준             |
|--------|----------------|
| 1단계    | 개념 설명만으로 풀이 가능 |
| 2단계    | 간단한 필터링 우회 필요  |
| 3단계    | 복잡한 필터링 우회 필요  |

[표 5.1.1] 문제 난이도

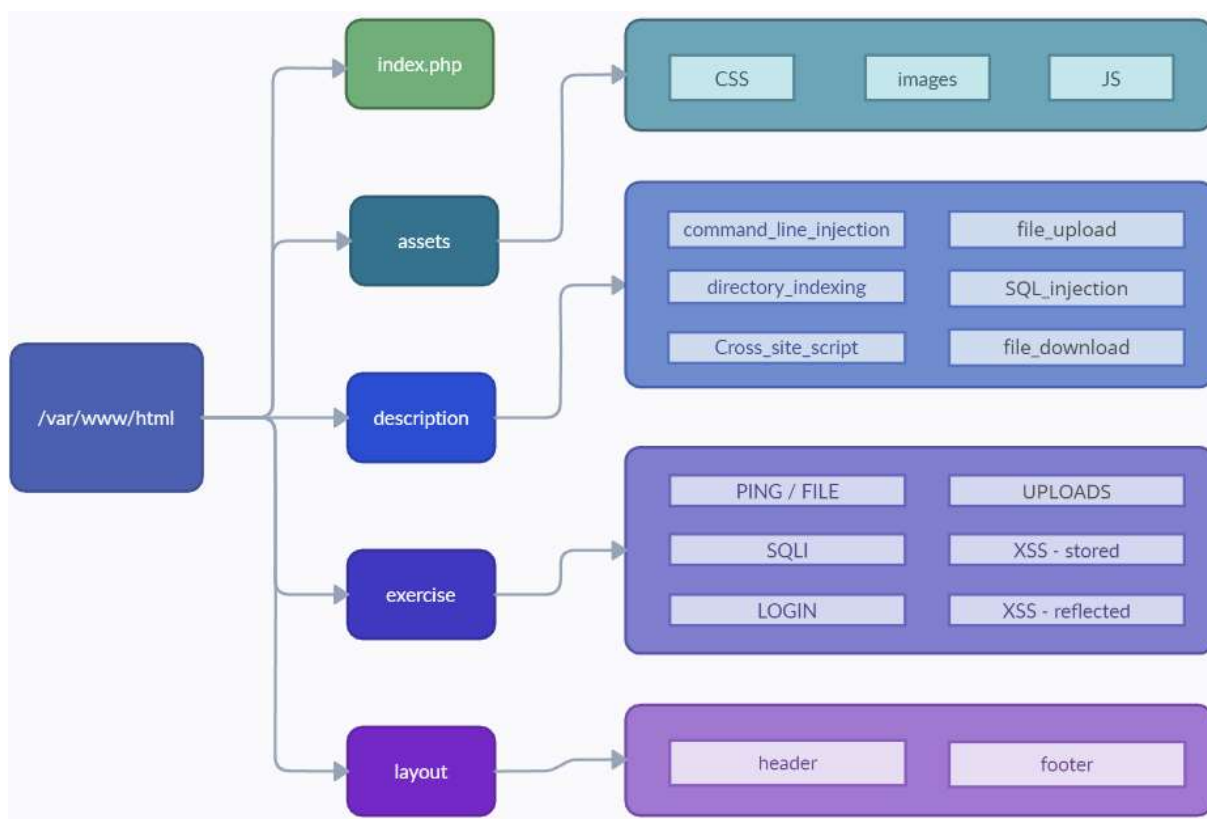
| 취약점 종류             | 문제명            | 문제 난이도 |
|--------------------|----------------|--------|
| SQL Injection      | Find User 1&2  | 1 단계   |
|                    | Login          | 2 단계   |
| XSS                | XSS            | 1 단계   |
|                    | XSS_Stored 1&2 | 2 단계   |
| Command Injection  | Ping 1&2       | 2 단계   |
|                    | File           | 1 단계   |
| Directory Indexing | 페이지 자체 취약점     | 1 단계   |
| File Upload        | Upload         | 1 단계   |
|                    | Upload-Base64  | 2 단계   |

[표 5.1.2] 실습 페이지 구성

## 5.2 실습 환경 구축

### 5.2.1 파일 구조도

CSS, 레이아웃, 설명 페이지, 실습 페이지 파일의 위치를 구조도를 통해 다음과 같이 나타내었음



[그림 5.2.1] 파일 구조도

## 5.2.2 Database 항목

현재 페이지 내의 구성된 데이터베이스는 사용자 정보 users와 게시판 정보 memo가 있으며, 이 정보는 XSS\_Stored 문제를 위해 관리되고 있음

```
mysql> desc users;
```

| Field    | Type        | Null | Key | Default | Extra          |
|----------|-------------|------|-----|---------|----------------|
| id       | int         | NO   | PRI | NULL    | auto_increment |
| name     | varchar(30) | NO   |     | NULL    |                |
| password | varchar(30) | NO   |     | NULL    |                |
| email    | varchar(40) | NO   |     | NULL    |                |

4 rows in set (0.00 sec)

```
mysql> desc memo;
```

| Field      | Type         | Null | Key | Default           | Extra             |
|------------|--------------|------|-----|-------------------|-------------------|
| id         | int          | NO   | PRI | NULL              | auto_increment    |
| title      | varchar(255) | NO   |     | NULL              |                   |
| content    | text         | NO   |     | NULL              |                   |
| author     | varchar(50)  | NO   |     | NULL              |                   |
| created_at | timestamp    | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED |

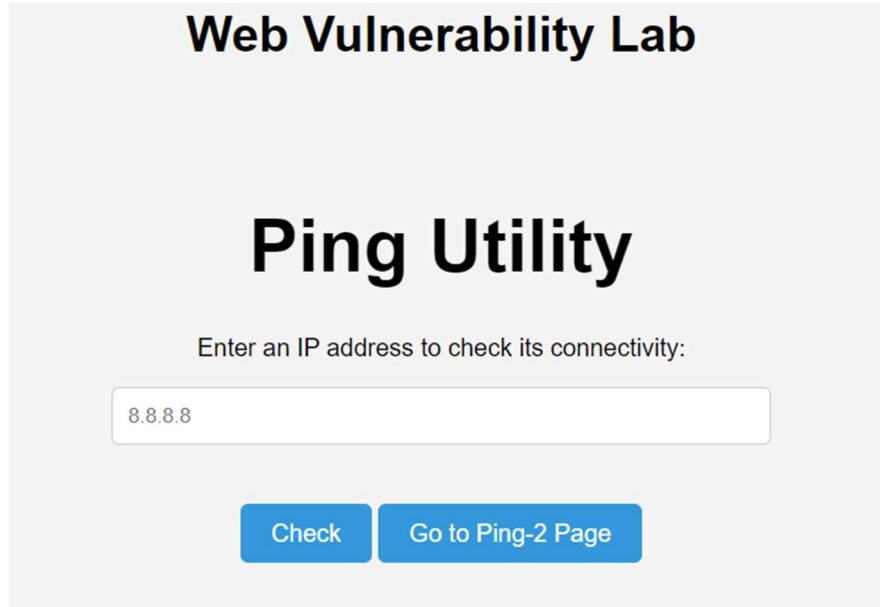
5 rows in set (0.00 sec)

[그림 5.2.2] Database

## 5.3 실습 문제 구축

### 5.3.1 Command Line Injection

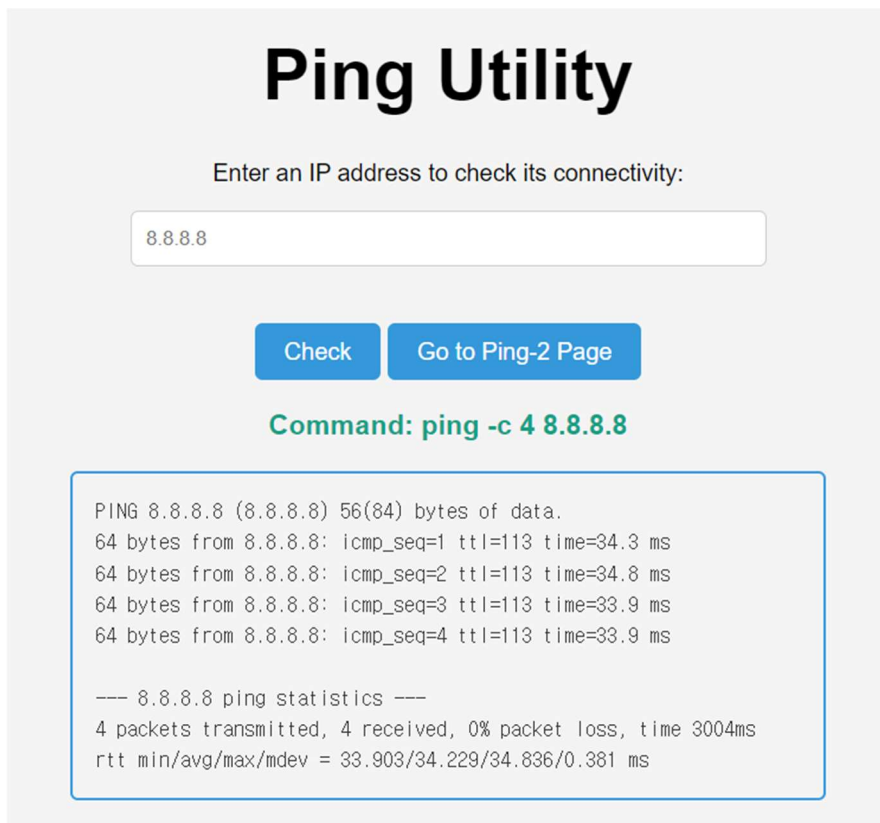
Command Line Injection 기본 페이지는 Ping을 통한 학습이 가능하도록 되어있으며, Ping 명령어를 입력하면 해당 Ping에 대한 응답이 출력됨



[그림 5.3.1] Command Line Injection Ping 문제 화면

#### 실습 페이지 정상 작동

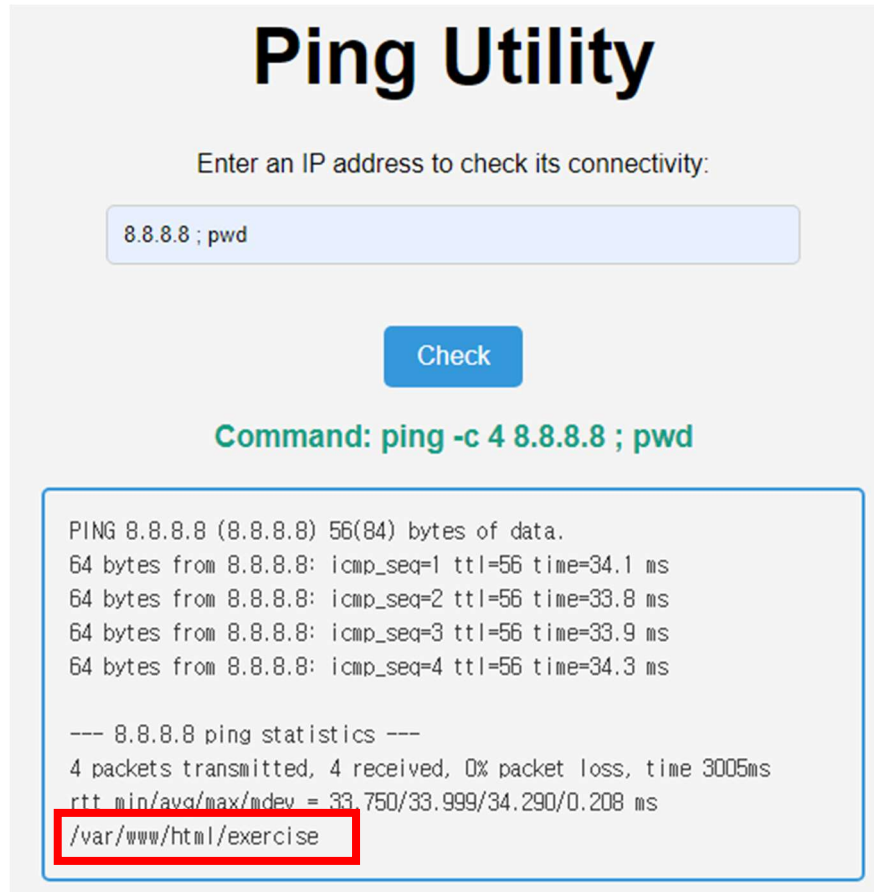
기본적으로 8.8.8.8 Ping을 날렸을 경우 정상적으로 Ping이 도달하는 것을 확인할 수 있음



[그림 5.3.2] Command Line Injection 정상적인 작동

### 실습 페이지 악의적 작동

악의적인 명령어를 통해 명령어에 해당하는 민감한 정보들을 얻을 수 있으며, 해당 명령어가 시스템에서 그대로 실행되는 것을 확인



**Ping Utility**

Enter an IP address to check its connectivity:

8.8.8.8 ; pwd

Check

**Command: ping -c 4 8.8.8.8 ; pwd**

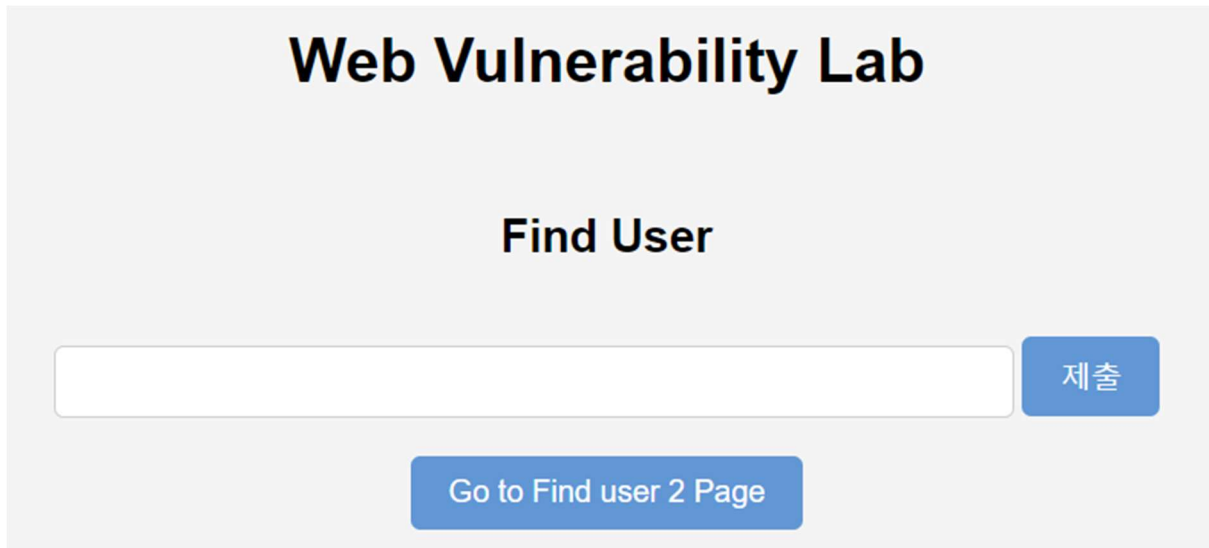
```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:
64 bytes from 8.8.8.8: icmp_seq=1 ttl=56 time=34.1 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=56 time=33.8 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=56 time=33.9 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=56 time=34.3 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt_min/avg/max/mdev = 33.750/33.999/34.290/0.208 ms
/var/www/html/exercise
```

[그림 5.3.3] Command Injection 악의적 작동

### 5.3.2 SQL Injection (SQLi)

SQL Injection 기본 페이지는 User 이름을 입력하면 해당 ID, 이름, 이메일에 대한 정보를 얻을 수 있으며, 다음 단계인 Find user2로 이동할 수 있음



The image shows a web interface titled "Web Vulnerability Lab" with a sub-header "Find User". Below the header is a text input field and a blue button labeled "제출" (Submit). At the bottom, there is a blue button labeled "Go to Find user 2 Page".

[그림 5.3.4] SQL Injection Find User 문제 화면

#### 실습 페이지 정상 작동

admin 또는 User, guest를 입력하면 해당 정보의 ID, 이름, 이메일을 알 수 있음



The image shows the same "Web Vulnerability Lab" interface as Figure 5.3.4, but with the input field containing "admin" and the "제출" button highlighted. Below the input field, the results are displayed: "ID: 1", "이름: admin", and "이메일: admin@example.com".

[그림 5.3.5] SQL Injection 정상적인 검색 출력

#### 실습 페이지 악의적 작동

SQL Injection 공격 구문을 통해 데이터베이스에 저장되어 있는 모든 사용자의 정보가 출력됨



## Find User

'or 1=1 #

제출

|        |            |                        |
|--------|------------|------------------------|
| ID: 1  | 이름: admin  | 이메일: admin@example.com |
| ID: 2  | 이름: user1  | 이메일: user1@example.com |
| ID: 3  | 이름: guest  | 이메일: guest@test.com    |
| ID: 4  | 이름: guest1 | 이메일: guest1@test.com   |
| ID: 5  | 이름: guest2 | 이메일: guest2@test.com   |
| ID: 6  | 이름: guest3 | 이메일: guest3@test.com   |
| ID: 7  | 이름: guest4 | 이메일: guest4@test.com   |
| ID: 8  | 이름: guest5 | 이메일: guest5@test.com   |
| ID: 9  | 이름: guest6 | 이메일: guest6@test.com   |
| ID: 10 | 이름: guest7 | 이메일: guest7@test.com   |

[그림 5.3.6] SQL Injection 악의적 동작

### 5.3.3 Cross Site Script (XSS)

XSS 기본 페이지는 게시판 정보를 확인할 수 있으며, Title 검색 탭에 구문 삽입을 통해 실습해 볼 수 있도록 구성되어 있음

## Web Vulnerability Lab

### Memo Board - Search Memos

Title:

| Title                      | Content   | Author   | Date                |
|----------------------------|---|----------|---------------------|
| SQL Injection Example      | This memo details SQL injection techniques.               | Bob      | 2024-09-07 23:28:16 |
| Server Security            | A memo about securing web servers from attacks.           | Charlie  | 2024-09-07 23:28:16 |
| Cross-Site Scripting       | An explanation on how XSS can be exploited.               | David    | 2024-09-07 23:30:00 |
| Password Best Practices    | Best practices for securing user passwords.               | Emma     | 2024-09-07 23:31:00 |
| Database Encryption        | The importance of encrypting sensitive data in databases. | Frank    | 2024-09-07 23:32:00 |
| Phishing Attack Prevention | Tips on how to prevent phishing attacks.                  | Grace    | 2024-09-07 23:33:00 |
| Web Application Firewalls  | How to protect your website with WAFs.                    | Henry    | 2024-09-07 23:34:00 |
| SSL/TLS Best Practices     | Guidelines for implementing SSL/TLS on websites.          | Isabella | 2024-09-07 23:35:00 |
| DNS Security               | Ensuring that your DNS configurations are secure.         | Jack     | 2024-09-07 23:36:00 |

[그림 5.3.7] XSS Stored\_XSS 문제 화면

## 실습 페이지 정상 작동

사용자가 검색을 통해 Title 정보를 입력하면 해당 내용이 출력

### Web Vulnerability Lab

#### Memo Board - Search Memos

Title:

Search results for: 'Security'

| Title           | Content   | Author  | Date                |
|-----------------|---|---------|---------------------|
| Server Security | A memo about securing web servers from attacks.   | Charlie | 2024-09-07 23:28:16 |
| DNS Security    | Ensuring that your DNS configurations are secure. | Jack    | 2024-09-07 23:36:00 |

[그림 5.3.8] XSS 정상적인 작동

## 실습페이지 악의적 작동

검색창에 스크립트 구문(<script>alert()</script>)을 입력하여 쿠키 또는 세션 값을 획득할 수 있음

### Memo Board - Search Memos

Title:

59.15.158.20:7676 내용:

Cookie=Security\_Academy

[그림 5.3.9] XSS 악의적 동작

### 5.3.4 File Upload

File Upload 기본 페이지는 기존 파일을 업로드 할 수 있도록 구성되어 있으며, 업로드한 파일의 리스트는 출력되지 않도록 함



[그림 5.3.10] File Upload base64 문제 화면

### 실습 페이지 정상 작동

파일이 올라가면 'File is successfully uploaded.' 문구가 출력되며 파일명이 base64로 인코딩되어 저장됨



[그림 5.3.11] File upload 정상적인 동작

### 실습 페이지 악의적 작동

파일 형식의 검사 없이 파일이 업로드 된다는 점을 이용하여 악의적인 사용자가 악성 스크립트가 저장된 파일을 올릴 수 있으며, 웹셀을 통한 여러 공격이 가능하게 됨

|   |                                  |                  |     |
|---|----------------------------------|------------------|-----|
|  | <a href="#">ZmxhZw==.txt</a>     | 2024-08-29 10:50 | 35  |
|  | <a href="#">d2Vic2h1bGw=.php</a> | 2024-08-29 16:57 | 40  |
|  | <a href="#">dGVzdA==.php</a>     | 2024-08-28 23:18 | 376 |

[그림 5.3.12] base64 인코딩 된 파일 저장 목록

### 5.3.5 Directory Indexing

Directory Indexing 기본 페이지는 페이지 자체적으로 취약하게 설정되어 있으며, 해당 페이지에서 디렉터리 경로 조작을 통해 확인이 가능

## 7. 실습

- 해당 페이지에 **Directory Indexing** 취약점이 있습니다.
- 어떤 식으로 접근을 하면 해당 취약점을 이용할 수 있는지 확인해보세요.

[그림 5.3.13] Directory Indexing 문제 화면

### 실습 페이지 악의적 작동

해당 URL 접근 시, Index of 페이지에 접근이 가능하며 해당 페이지가 아닌 다른 페이지의 저장된 파일을 확인 및 저장이 가능하게 됨



## Index of /exercise

| Name                          | Last modified    | Size | Description |
|-------------------------------|------------------|------|-------------|
| Parent Directory              | -                | -    | -           |
| flag.txt                      | 2024-08-28 16:18 | 37   |             |
| login.tar                     | 2024-09-06 10:28 | 13K  |             |
| login/                        | 2024-09-06 02:54 | -    |             |
| ping-2.php                    | 2024-08-29 17:37 | 941  |             |
| ping.php                      | 2024-08-28 16:10 | 907  |             |
| sql_i_find_user.php           | 2024-09-05 13:59 | 2.2K |             |
| sql_i_find_user_2.php         | 2024-09-07 22:58 | 1.8K |             |
| sql_i_login.php               | 2024-09-05 14:19 | 2.2K |             |
| trash/                        | 2024-08-29 17:44 | -    |             |
| upload-base64.php             | 2024-08-28 23:45 | 2.5K |             |
| upload.php                    | 2024-08-29 17:36 | 1.8K |             |
| uploads-base64/               | 2024-08-29 16:57 | -    |             |
| uploads/                      | 2024-09-07 22:59 | -    |             |
| xss - 복사본.php:Zone.Identifier | 2024-09-07 23:28 | 0    |             |
| xss.php                       | 2024-09-07 23:56 | 1.6K |             |
| xss_stored.php                | 2024-09-07 23:30 | 1.0K |             |

Apache/2.4.52 (Ubuntu) Server at 59.15.158.20 Port 7676

[그림 5.3.14] 디렉터리 인덱싱 /exercise 경로 확인



## Index of /description

| Name                       | Last modified    | Size | Description |
|----------------------------|------------------|------|-------------|
| Parent Directory           | -                | -    | -           |
| command_line_injection.php | 2024-08-29 17:39 | 7.4K |             |
| directory_indexing.php     | 2024-09-02 22:27 | 6.2K |             |
| file_upload.php            | 2024-08-29 17:34 | 5.9K |             |
| sample.php                 | 2024-08-29 09:47 | 3.2K |             |
| sql_i.php                  | 2024-09-06 11:27 | 7.6K |             |
| trash/                     | 2024-08-29 17:22 | -    |             |
| xss.php                    | 2024-09-07 23:30 | 4.8K |             |

Apache/2.4.52 (Ubuntu) Server at 59.15.158.20 Port 7676

[그림 5.3.15] 디렉터리 인덱싱 /description 경로 확인