

# 01

## SQL Injection



## TABLE OF CONTENTS

01

### What Is SQL Injection

What is SQLi? And what is it's impact?

02

### SQL Injection detection

How to detect?

03

### SQL Injection examples & hands on

Payloads example

04

### SQL Injection Prevention

How Can we Protect our Website From SQL Injection?

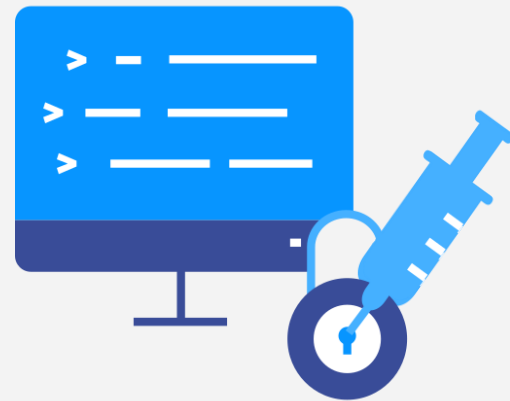
05

### Recap and Q/A

Fast Recap and Answering Questions

## What is SQL Injection

- SQL injection (SQLi) is a type of web application security vulnerability that allows an attacker to interfere with the queries that an application makes to its database, which an attacker can use inject malicious SQL code into a weakly protected user input fields or other input fields. The injected SQL code is then executed by the application's database server.



### What is the impact of a successful SQL Injection attack ?

- SQL injection attack can result in unauthorized access to sensitive data, such as passwords, credit card details, or personal user information.
- SQL injection attack can result in Manipulation and Deletion of Data.
- SQL injection attack in some cases can cause an attacker to take Complete Control over Database Server.



## TABLE OF CONTENTS

01

### What Is SQL Injection

What is SQLi? And what is it's impact?

02

### SQL Injection detection

How to detect?

03

### SQL Injection examples & hands on

Payloads example

04

### SQL Injection Prevention

How Can we Protect our Website From SQL Injection?

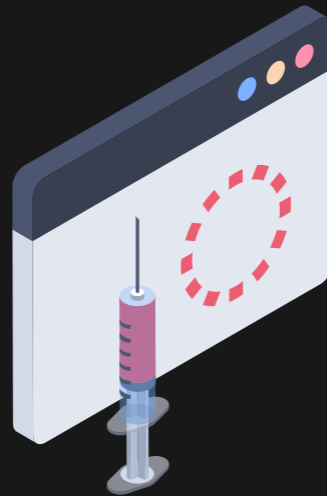
05

### Recap and Q/A

Fast Recap and Answering Questions

# 02

## SQL Injection detection



## How as an attacker you can find & detect SQL Injection vulnerabilities ?

**SQL injection can be detected manually via systematic set of tests against every entry point in the application :**

1. We can start with submitting a single quote character [ ' ] and look for errors.
2. Submitting some Boolean conditions such as [ OR 1=1 ] and look for differences in the application response.
3. We can try and Delay the request via a Time-Based injection.



## TABLE OF CONTENTS

01

### What Is SQL Injection

What is SQLi? And what is it's impact?

02

### SQL Injection detection

How to detect?

03

### SQL Injection examples & hands on

Payloads example

04

### SQL Injection Prevention

How Can we Protect our Website From SQL Injection?

05

### Recap and Q/A

Fast Recap and Answering Questions




# 03

## SQL Injection examples



## Examples on SQL INJECTION :

- Let's first have a look behind the scenes understand how things operate in the back-end at the data-base so a vulnerable query would look like the image where it takes the input directly into the query.



```
SELECT ? FROM ? WHERE ?='%value%'
```

## Examples on SQL INJECTION :

- Now let's try and exploit our application

We can start by entering a single quote character [ ' ] in our login form and use any random password , as you can see we get an Error showing a part of the original query.

### OperationalError

```
sqlite3.OperationalError: near "' AND password = '": syntax error
```

## Examples on SQL INJECTION :

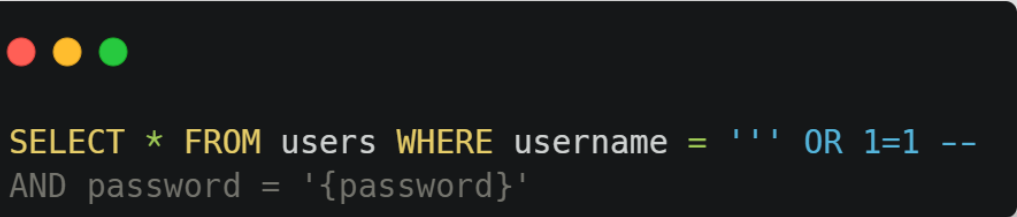
- Next let's do something serious by entering some true statement  
Like [ ' OR 1=1 -- ] now that will return a TRUE so the WHERE clause  
always evaluate to true which is supposed to return all the users  
from the database, Now let's Break it down.

```
SELECT * FROM users WHERE username = '' OR 1=1 --  
AND password = '{password}'
```

## Examples on SQL INJECTION :

- **[ ' ]**: The single quote is used to close the initial string value.
- **[ OR 1=1 ]**: This is a condition that always evaluates to true by using the logical OR operator and the condition 1=1.
- **[ -- ]**: The double hyphen signifies the start of a comment

Which will make the database ignore  
Everything after it & allow the attacker  
To enter any password.



```
SELECT * FROM users WHERE username = '' OR 1=1 --  
AND password = '{password}'
```

## Examples on SQL INJECTION :

- Now our application only return one user at a time so to bypass this we will use the [ LIMIT & OFFSET ] clauses .
- The LIMIT clause limits the rows of the output.

employee_id	first_name	last_name
121	Adam	Fripp
103	Alexander	Hunold
115	Alexander	Khoo
193	Britney	Everett
104	Bruce	Ernst
179	Charles	Johnson
109	Daniel	Faviet
105	David	Austin
114	Den	Raphaely

LIMIT 5



employee_id	first_name	last_name
121	Adam	Fripp
115	Alexander	Khoo
103	Alexander	Hunold
193	Britney	Everett
104	Bruce	Ernst

## Examples on SQL INJECTION :

- The offset clause will shift the rows of the outcome.
- using them together will shift & limit The outcome rows of our query allowing us To login as other users.

employee_id	first_name	last_name
121	Adam	Fripp
103	Alexander	Hunold
115	Alexander	Khoo
193	Britney	Everett
104	Bruce	Ernst
179	Charles	Johnson
109	Daniel	Faviet
105	David	Austin
114	Den	Raphaely



OFFSET 3




LIMIT 5

employee_id	first_name	last_name
193	Britney	Everett
104	Bruce	Ernst
179	Charles	Johnson
109	Daniel	Faviet
105	David	Austin

## Examples on SQL INJECTION :

- Now if we try this payload using [ LIMIT & OFFSET ] every time we change the number of the OFFSET we can login as different users .



```
SELECT * FROM users WHERE username = '' OR 1=1  
LIMIT 1 OFFSET 2 -- AND password = '{password}'
```



## TABLE OF CONTENTS

01

### What Is SQL Injection

What is SQLi? And what is it's impact?

02

### SQL Injection detection

How to detect?

03

### SQL Injection examples & hands on

Payloads example

04

### SQL Injection Prevention

How Can we Protect our Website From SQL Injection?

05

### Recap and Q/A

Fast Recap and Answering Questions

# 04

## SQL Injection Prevention



## How to prevent your web application from SQL INJECTION :

- Use Parameterized Queries (Prepared Statements).
- Input Validation and Sanitization.
- Least Privilege Principle.
- Escaping User Input.
- Disable Error Messages.
- Web Application Firewall (WAF).



## RESOURCES

<https://portswigger.net/web-security/sql-injection#what-is-sql-injection>  
<https://www.sqltutorial.org/sql-limit/>





# Thank You

Presented by USIF ARABY