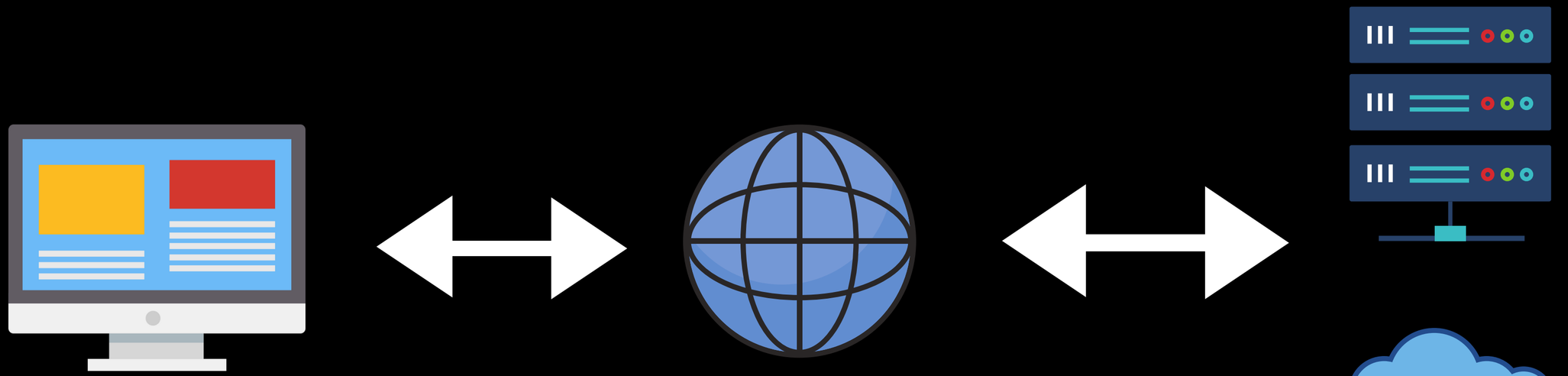
OWASP ERODE

OWASP
SASTRA University

INTRODUCTION TO SQL INJECTION

HOW WEBSITE WORKS?



REQUEST

FOLLOWS
OSI MODEL
STANDARD

RESPONSE

Request:

Data passed from client side to initiate some action on the server

HTTP Request METHODS

Response:

Data passed from client side to initiate some action on the server

HTTP Response STATUS CODES

WHY PROXY?

- Bypass client side mitigations
- Intercept web traffic to analyze
- Best for fuzzing web requests and responses

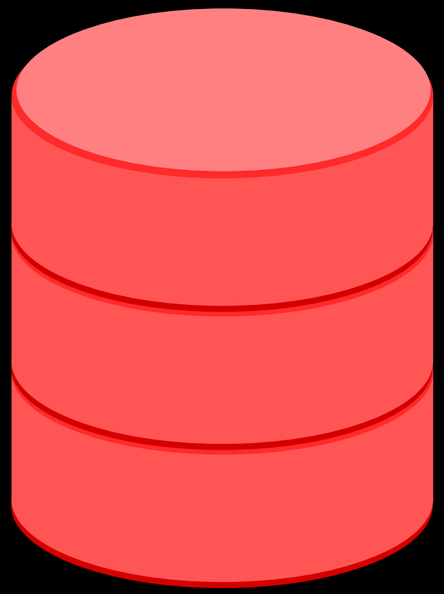
SQL INJECTION

MOST CRUCIAL BUG/VULN

OWASP TOP 10

HIGH IMPACT ON SERVER SIDE

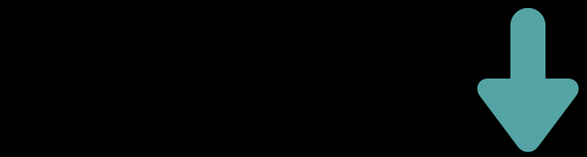
WHY CRUCIAL?



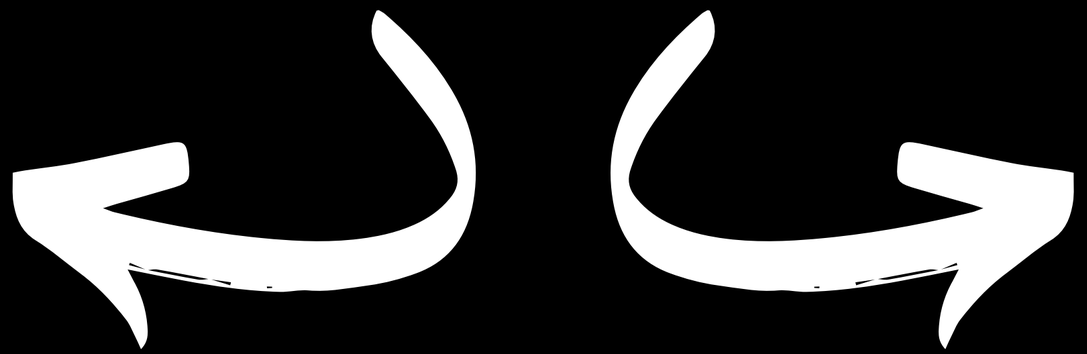
DB

- Data is important for Companies
- Hackers try to steal DB for Money
- Takes more time to recover after impact
- Important data can be leaked (TOP SECRET,NEW PROTOTYPE,PRIVATE DATA)

DATABASE(DB)



BY STRUCTURE



STRUCTURE QUERY LANGUAGE (SQL)

Relational

Not Only SQL (NoSQL)

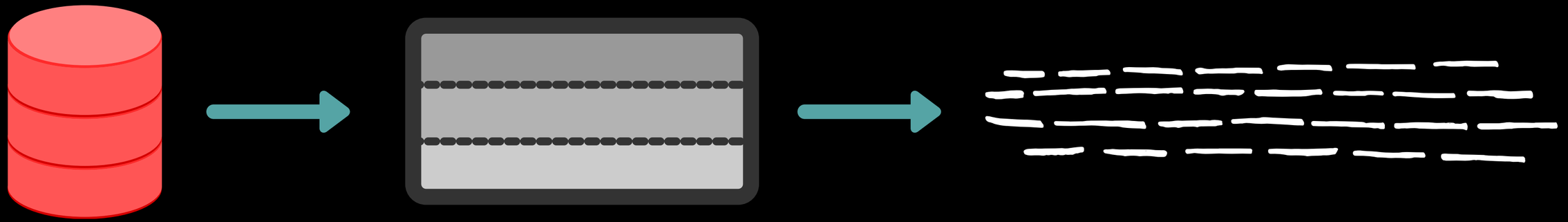
Not Relational

SQL --> Structured Query Language

It is a database with tables and values

Provides data from backend to frontend apps

SQL Struct



DB

Tables

Row & Col

Each column specifies an attribute of the DB

There can be many rows based on the app

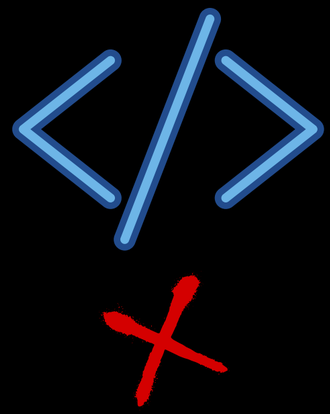
Each row can be referenced by primary key

Tables can be interlinked in SQL using foreign key

Normalization reduces duplicates in the table

HOW SQL DB WORKS?

DATA EMBEDED
IN SOURCE CODE



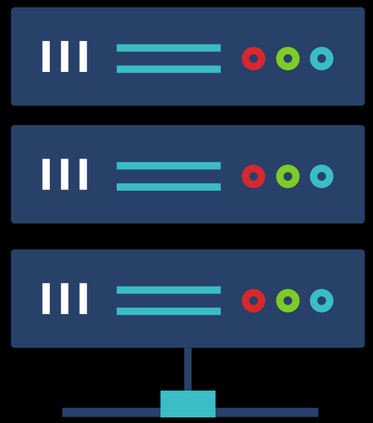
performance,size
security,transparency

DATA CALLED
FROM DB SERVER

finds row with value = moni
retrieves all column attributes

username?=moni

finds "users" table
which contains username



COMMON SQL QUERY

DATA DEFINITION	DATA MANIPULATION	DATA AGGREGATION
CREATE	SELECT	SUM
DROP	UPDATE	AVG
ALTER	INSERT	MAX
TRUNCATE	DELETE	MIN
		SO ON..

username?=moni

TABLE

...

posts

users

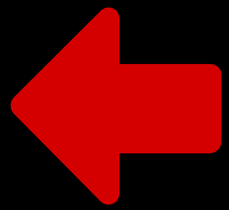
products

...

selects "users" table

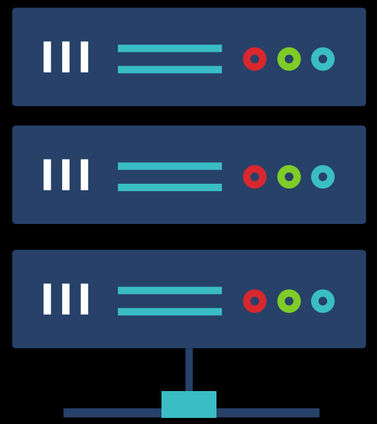
uid	username	email	password	...
1	abc	a@a.com	abcdefgh	...
2	moni	m@a.com	passwd	...
3	nanba	vn@a.com	nasahack	...

.....



selects row with "username=moni"

SQLI WORKING



VALUE + QUERY == RESULT

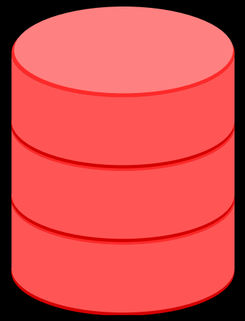
DATA
user input field
form data
url parameters
etc..

FUNCTION
retrieve
update
post
delete & ..

SQL QUERY
works in DB
based on
"**values**" from "**DATA**"
"**queries**" from "**FUNCTION**"

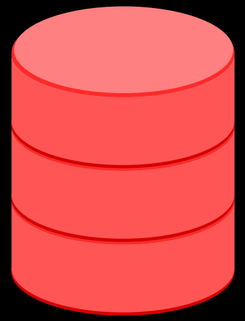
USER INPUTS
or
USER RELATED DATA
IS THE KEY FOR
SUCCESSFUL SQLI

correct
input data



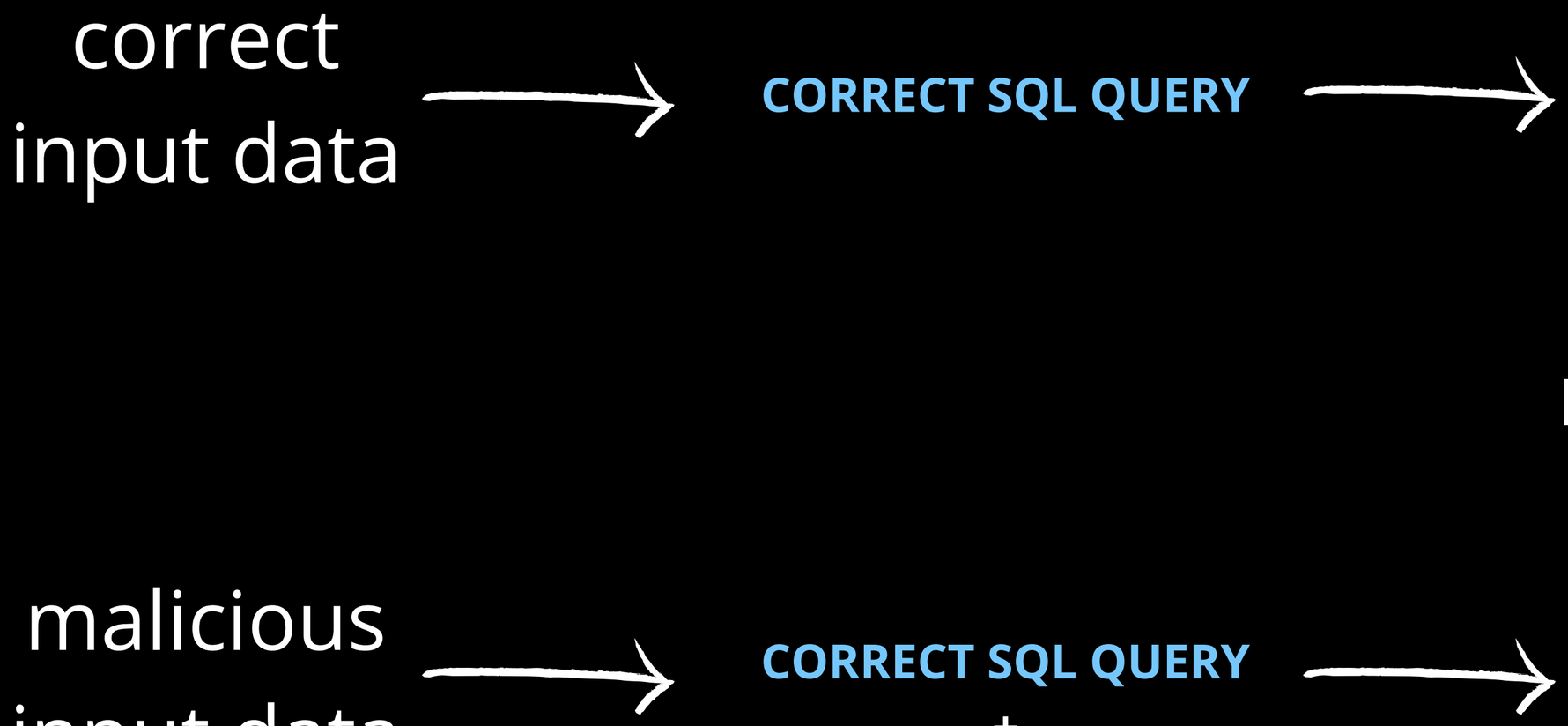
SQL Query
produces
"**Expected
Correct
Result To User**"

malicious
input data

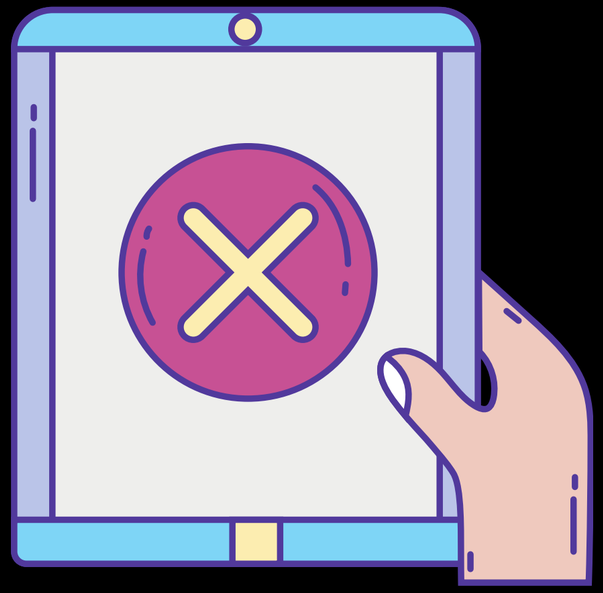


SQL Query
produces
"**Unexpected
Different
Result To User**"

FRONTEND SIDE


correct
input data
CORRECT SQL QUERY
SQL Query
produces
"Expected
Correct
Result To User"
malicious
input data
CORRECT SQL QUERY
+
INJECTED MALICIOUS
QUERY
SQL Query
produces
"Unexpected
Different
Result To User"
BACKEND SIDE

HOW TO IDENTIFY SQLi?



- Different result than expected(important) in frontend
- Unusual errors (like server error etc..)
- Its important to identify the difference between correct page and sqli error page (comparers)

SQLi TYPES

- DIRECT INPUT Based Attacks
- UNION Based Attacks
- BOOLEAN Based Attacks
- ERROR Based Attacks
- TIME DELAY Based Attacks

RARE SQLi TYPES:

- **BLIND SQLi:**
  - **No difference in error pages**
  - **Should be manually tested**
  - **Can be checked by ERROR,BOOLEAN & DELAYS conditions**
  - **Rare to find**

RARE SQLi TYPES:

- Second Order SQLi:
  - Gets stored in DB
  - Should be manually tested
  - Critical Flaw (~ High severity)
  - Persistent attack if saved in DB
  - Rare to find

HOW TO APPROACH?

- Test URL Parameters
- Test User Input Fields
- Test Search Bars
- Test Request Headers related to Users
- Search for BlindSQLi
- Try for Second Order SQLi

AUTOMATION FOR SQLi?

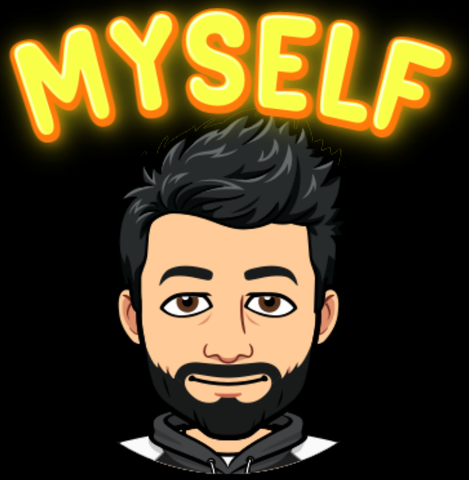
- SQLMap (Best for UNION Attacks)
- BBQSQL (Good for Blind SQLi)
- Leviathan
- jSQL
- SQLNinja

DO MANUAL TESTING, AUTOMATION ALWAYS DOESN'T GIVE RESULTS

PREVENTION?

- Validate user inputs
- Sanitize data with limited special characters
- Prefer whitelist over blacklist
- Limit privileges and read-access
- Do not link to other valuable tables
- Scanning and updating DB
- Usage of WAF

THANK YOU



@aidenpearce369