Blind SQL Injection

CYBER SECURITY &
PRIVACY FOUNDATION

Introduction

Blind SQL Injection is a type of SQL Injection vulnerability in which the web application will be vulnerable to SQL injection but results of injection won't be displayed.

Boolean Based Blind SQL Injection

- By Sending True or False Queries to the server, an attacker is able to compromise the entire Database.

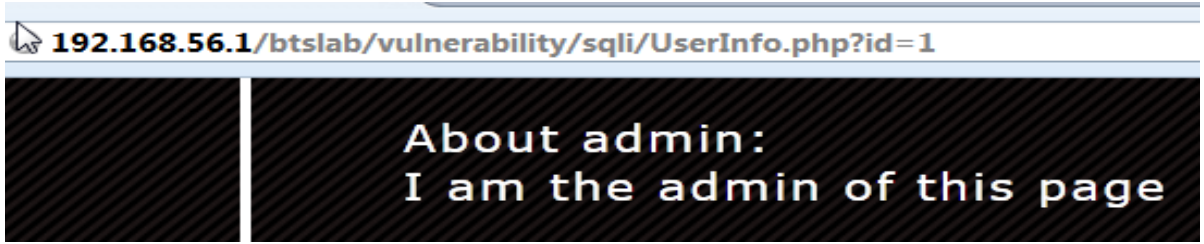
- By comparing the response to the True query with the response to the False query, an attacker can do Blind SQL injection.

Note: Boolean Operator '**and**':

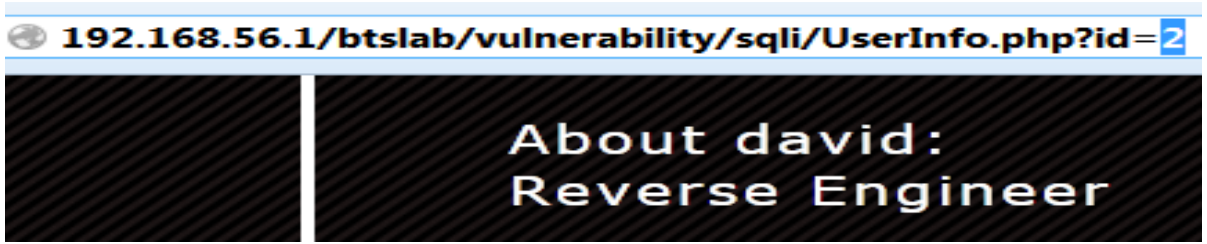
True **and** *1=1* => True
True **and** *1=2* => False

- Here, we have a page that will print out details about a user of a given ID.

When ID is equal to “1”, it prints details of ‘admin’ :

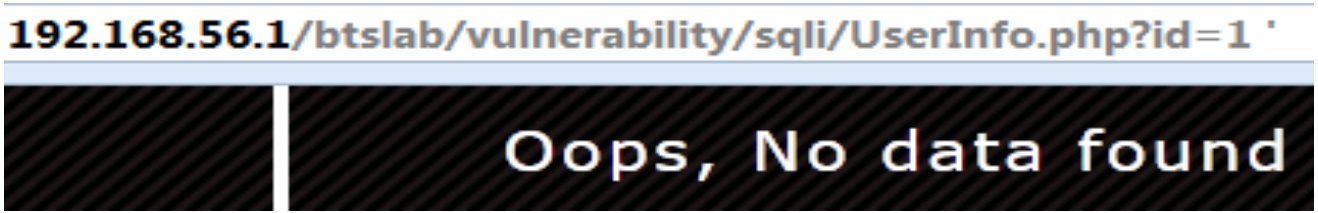


When ID is equal to “2”, it prints details of ‘David’ :



What happens if we enter apostrophe Character in ID value?

- Here, the result of injection is not visible.
- But, it displays a custom error message created by developer says “Oops, No data found”



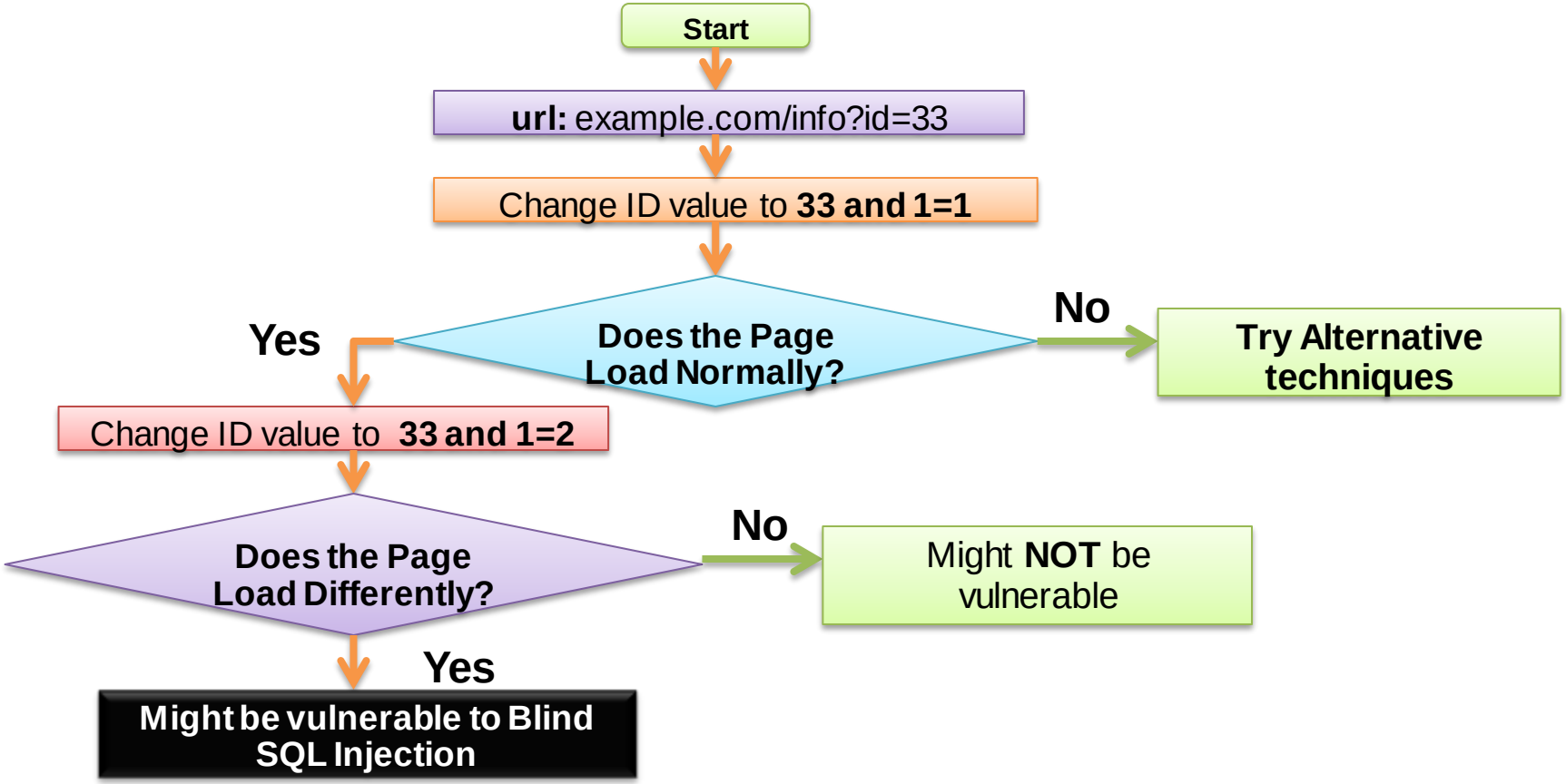
SQL Error Message suppressed in Application Level

- In the Database server perspective, nothing changes.
- However, the Error message produced from the DB server will be suppressed in the Application Level(In PHP code)

```php
$id=$_GET['id'];
$result=mysql_query("select * from users where id=".$id);//removed die() function to suppress sql error
$data=@mysql_fetch_array($result); // '@' symbol to suppress error message
if($data)
{
echo "<br>About ".$data['username'].":<br>".$data['about'];
}
else
{
echo "<br/>Oops, No data found";
}
```

Testing

Methodology



Sending True Query

- Loads normal Page:

- When we send the above request, the SQL query will becomes :

select * from users where id=2 and 1=1;

```
mysql> select * from users where id=2 and 1=1;
+----+----------+--------+----------+--------------------+-----------+-------------+
| ID | username | email  | password | about              | privilege | avatar      |
+----+----------+--------+----------+--------------------+-----------+-------------+
|  2 | david    | david  | pass     | Reverse Engineer   | user      | default.jpg |
+----+----------+--------+----------+--------------------+-----------+-------------+
1 row in set (0.00 sec)
```

Sending False Query

- Loads Different Page:

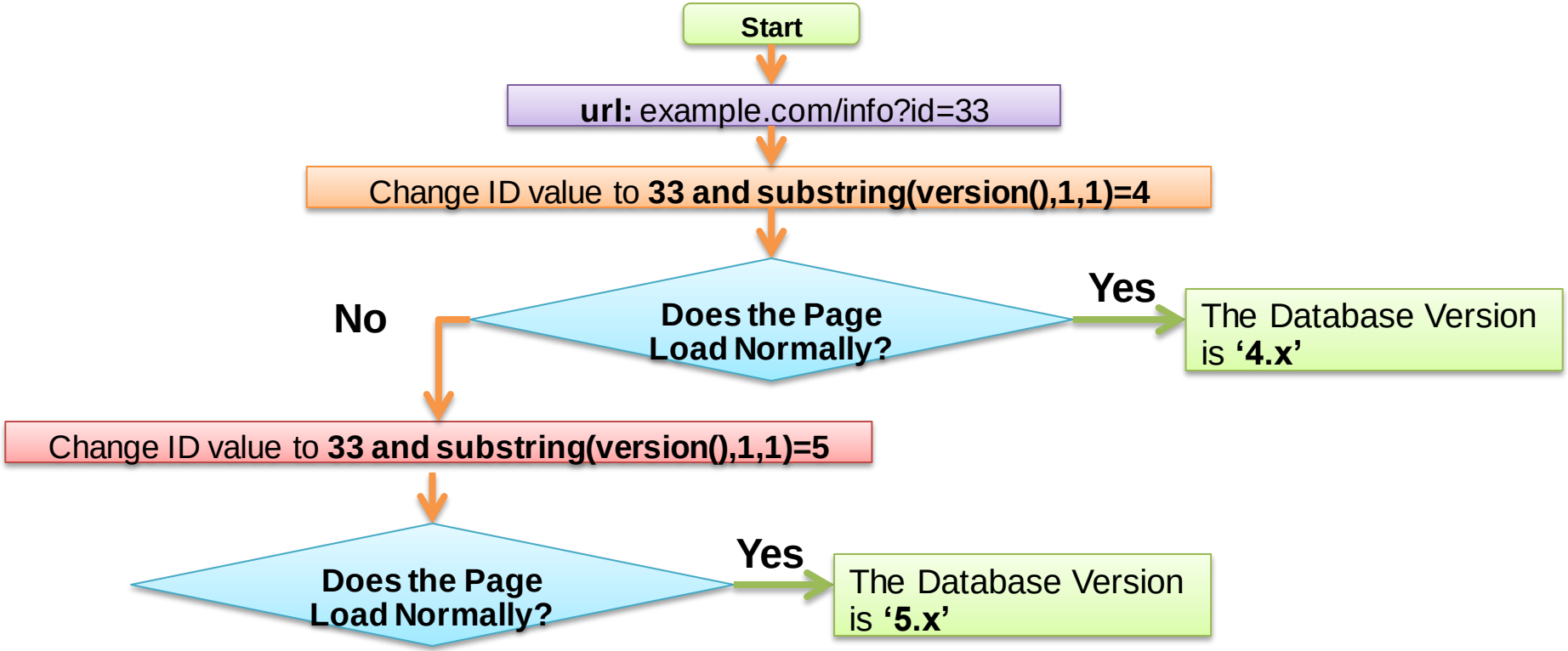
- When we send the above request, the SQL query will becomes :

select * from users where id=2 and 1=2;

```
mysql> select * from users where id=2 and 1=2;
Empty set (0.00 sec)
```

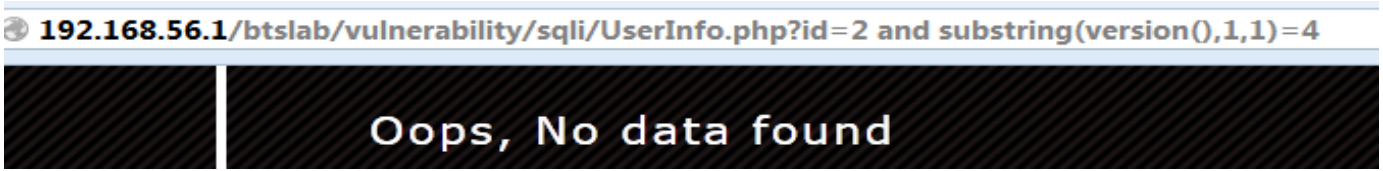
Exploitation

Determining Database Server Version



Determining Database Server Version

- Checking whether the db server version is 4.x

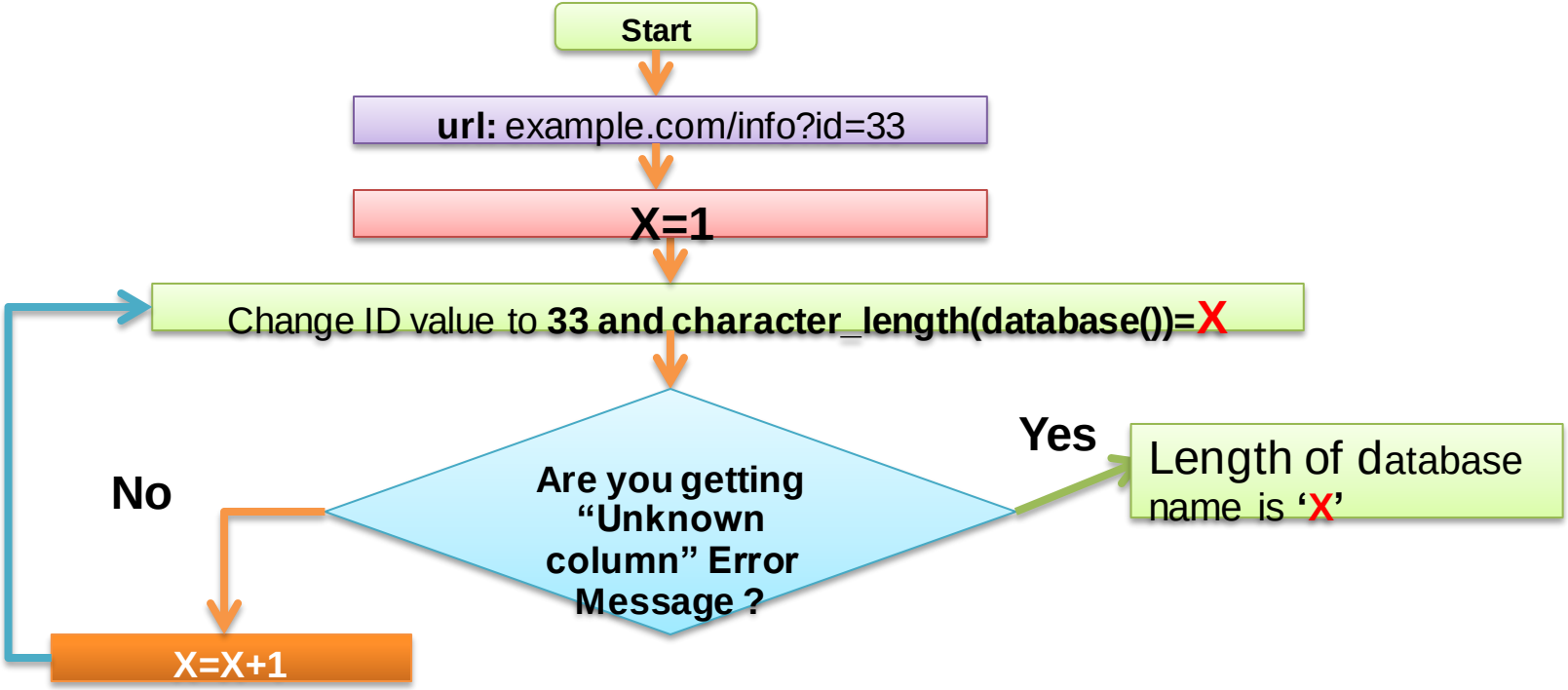


Result: False

- Checking whether the db server version is 5.x

Result: True

Determining length of database name

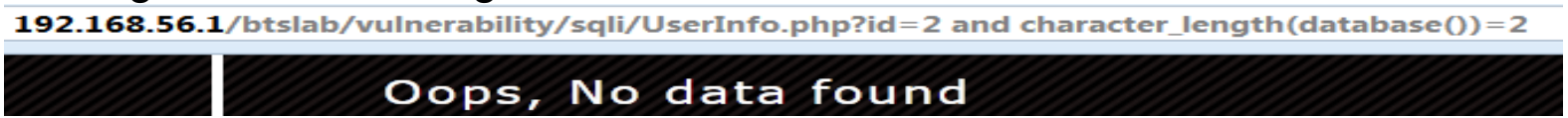


Determining length of database name

- Checking whether the length is '1'

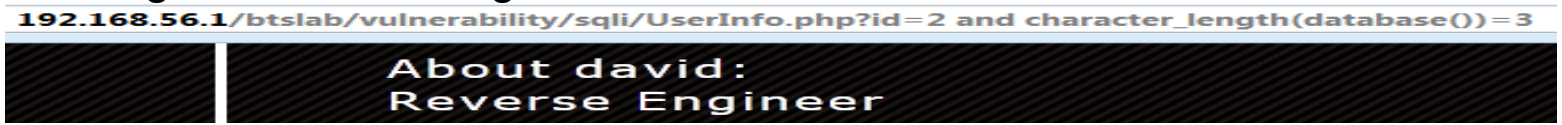
Result : False

- Checking whether the length is '2'



Result : False

- Checking whether the length is '3'



Result : TRUE

- So, the Database Name length is **'3'**

Determining Database Name

Determining the First Character of Database name

- Checking whether the first character is ‘a’

Result: True

The First Character of Database name is : ‘a'

Determining the Second Character

- Checking whether the 2nd character is ‘a’

Result: False

- Checking whether the 2nd character is ‘b’

Result: True

The Second Character of Database name is : ‘b'

Determining the Third Character

- Checking whether the 3rd character is ‘a’

Result: False

- Checking whether the 3rd character is ‘b’

Result: False

- Checking whether the 3rd character is ‘c’

Result: True

The 3rd Character of Database name is : ‘c'

From previous injection attacks, we determined the following:

- ~~The length of Database Name is : 3~~
- The First Character of DB Name is : **a**
- The 2nd Character of DB Name is : **b**
- The 3rd Character of DB Name is : **c**

- Hence the Database Name is : **“abc”**

- In this way, we can gather information from the Database Server.

- A Successful exploitation allows an attacker to dump entire database.