Intro in Web Security & Owasp Top 10

Whoami?

- Yasser Elsnbary
- Student at the Faculty of Computing and Information Sciences, Department of Computer Science, Mansoura University
- Leader Cyber Security Circle @CAT Reloaded
- I participated in CTF competitions (Egypt CTF – Mansoura CTF – Egypt Universities CTF – EGCERT Final)

Agenda :

- Cyber Security Circle
- Intro Web Security
- How To Start In Web Application Security
- Owasp Top 10
- Demos
- References

Cyber Security Circle

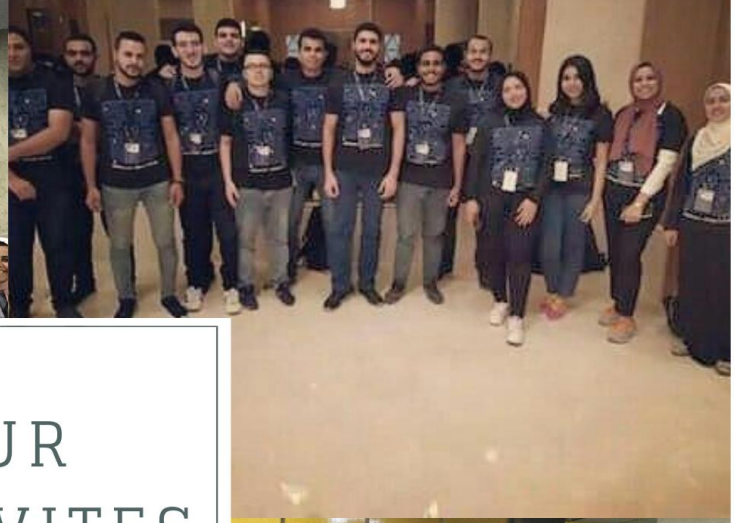
1-Our Re-Founder

Nidal Fikri

Re-founder and the leader of the Cyber Security Circle in Mansoura – 2019

2-We have Three branches:

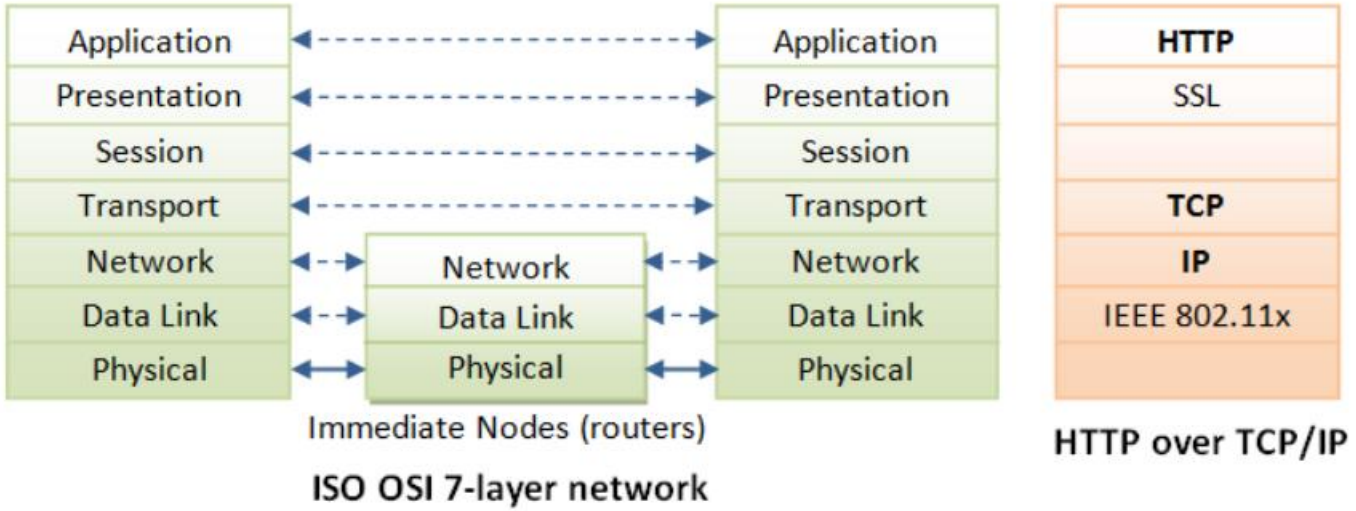
- Web Security
- Reverse Engineering
- Network Security



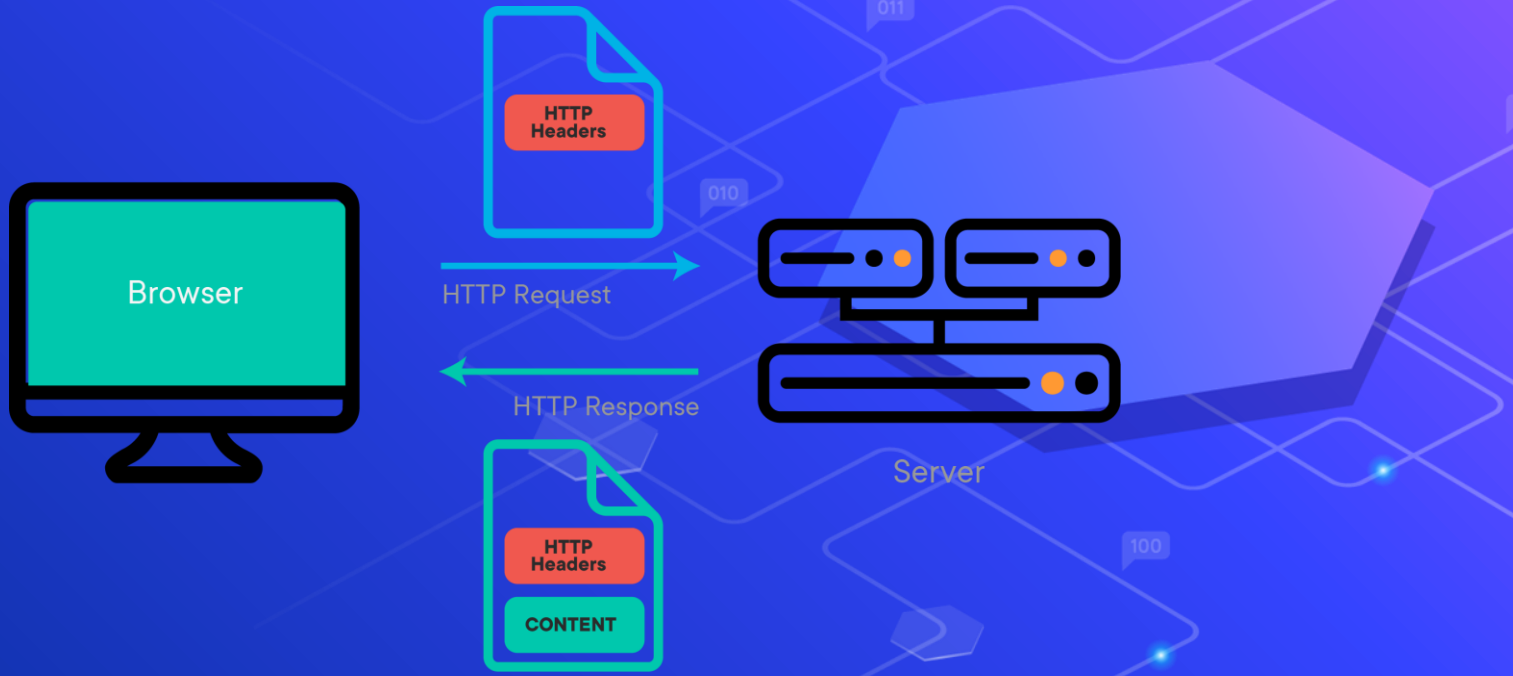
OUR ACTIVITES

HOW Internet and Web work

1-HOW Web works ?



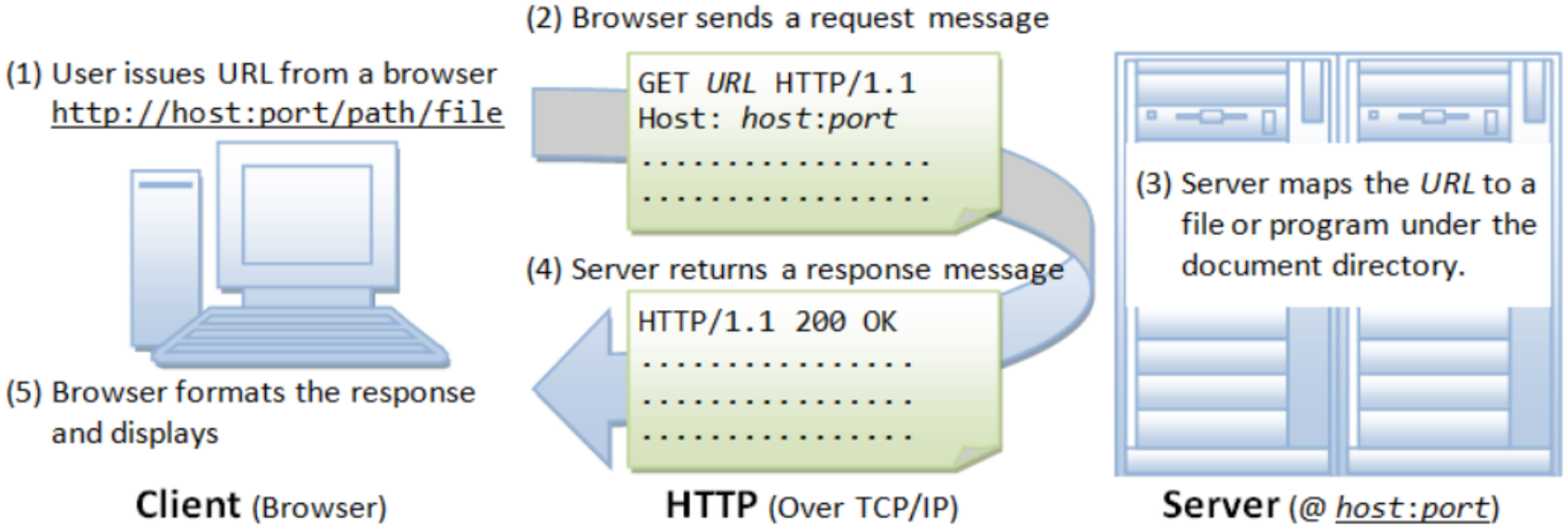
2-HOW Web works ?



HTTP Protocol

1-HTTP split To:

- Request
- Response



2-HTTP Request

HTTP request message

- two types of HTTP messages: *request, response*
- HTTP request message:
  - ASCII (human-readable format)

request line (GET, POST, HEAD commands)

```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
User-agent: Mozilla/4.0
Connection: close
Accept-language:fr
```

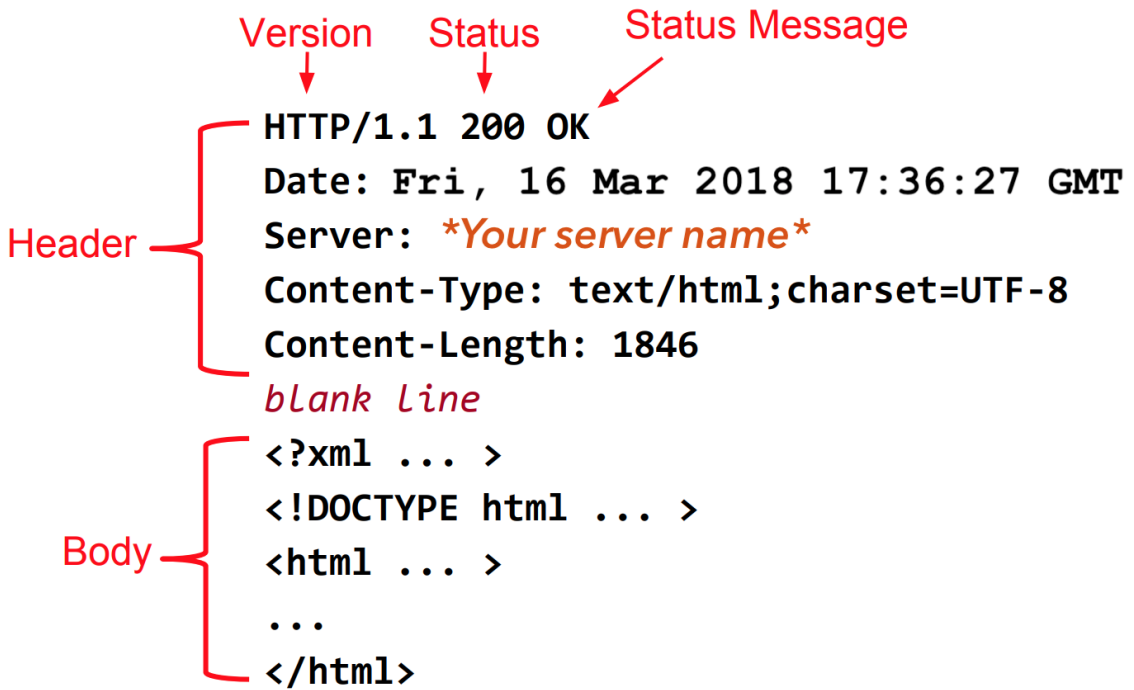
header lines

Carriage return, line feed indicates end of message

(extra carriage return, line feed)

3-HTTP Response

HTTP Response - Read lines from socket



Web Security

Web Security

It basically means protecting a website or web application by detecting, preventing and responding to cyber threats.

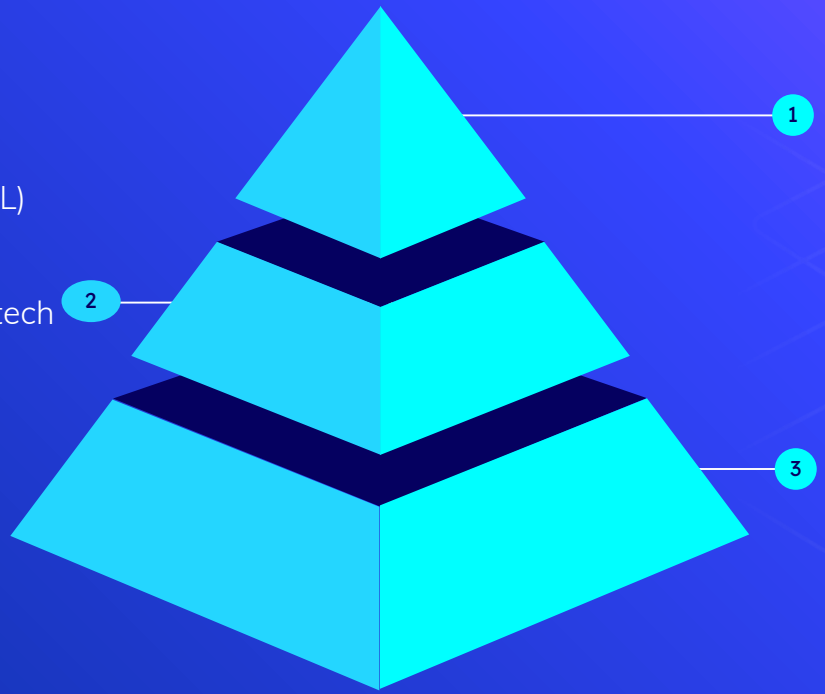
1- Likelihood of Threat

SQL injection
Password breach
Cross-site scripting
Data breach
Code injection
Xss
IDOR
DOS Attack

2- OUR PATH in Circel

Intermediate lvl

1. Web Dev (html-js-php-mySQL)
2. HTTP Protocol
3. Web Servers
4. Components of modern web tech & Terms
5. Study Vulnerabilities bugCrowd VRT



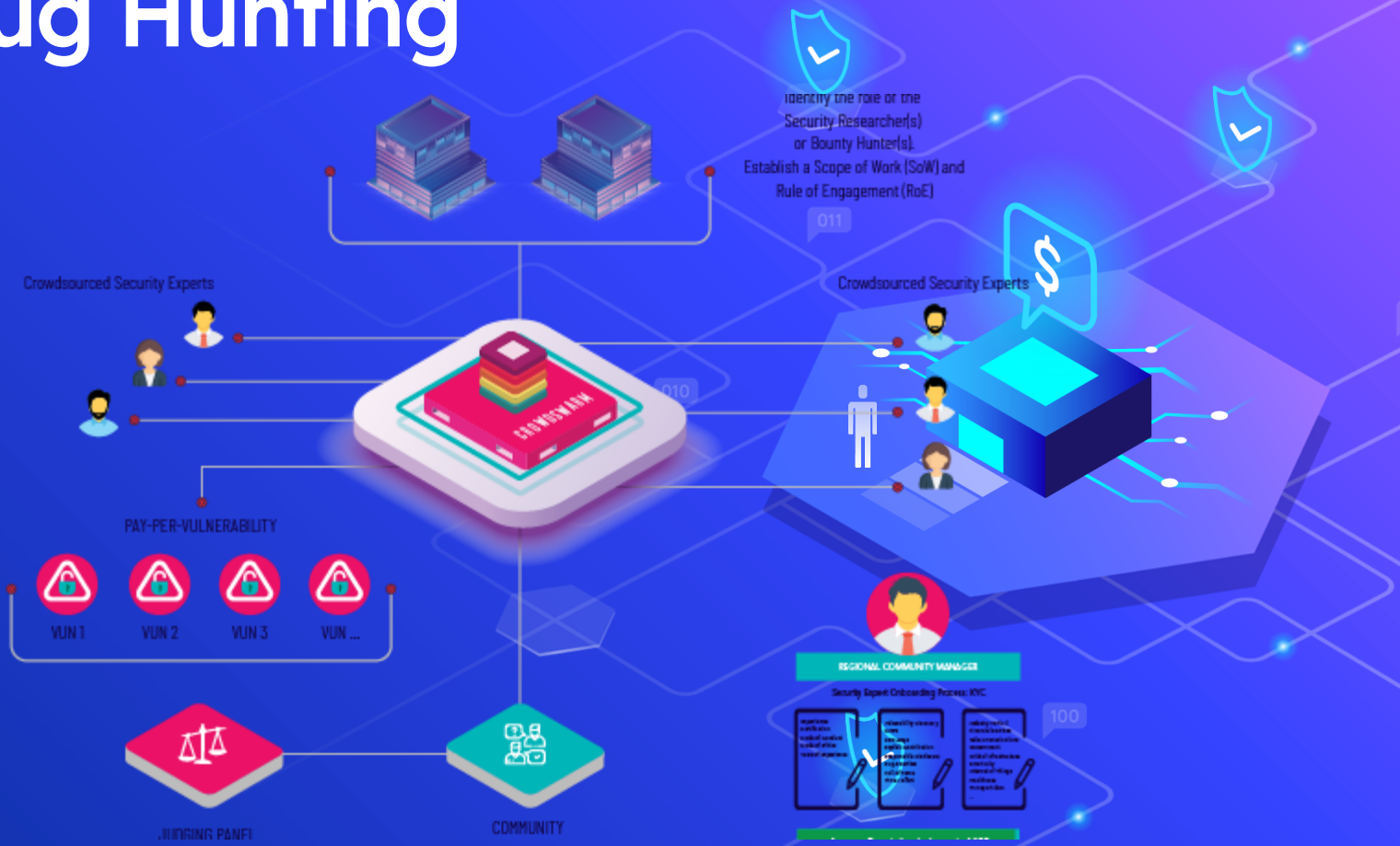
Advanced lvl

1. Practice your Knowledge Download owasp broken Web application Project
2. Play CTFs
3. Start Bug Hunting
4. build your mind set
5. Learn Recon

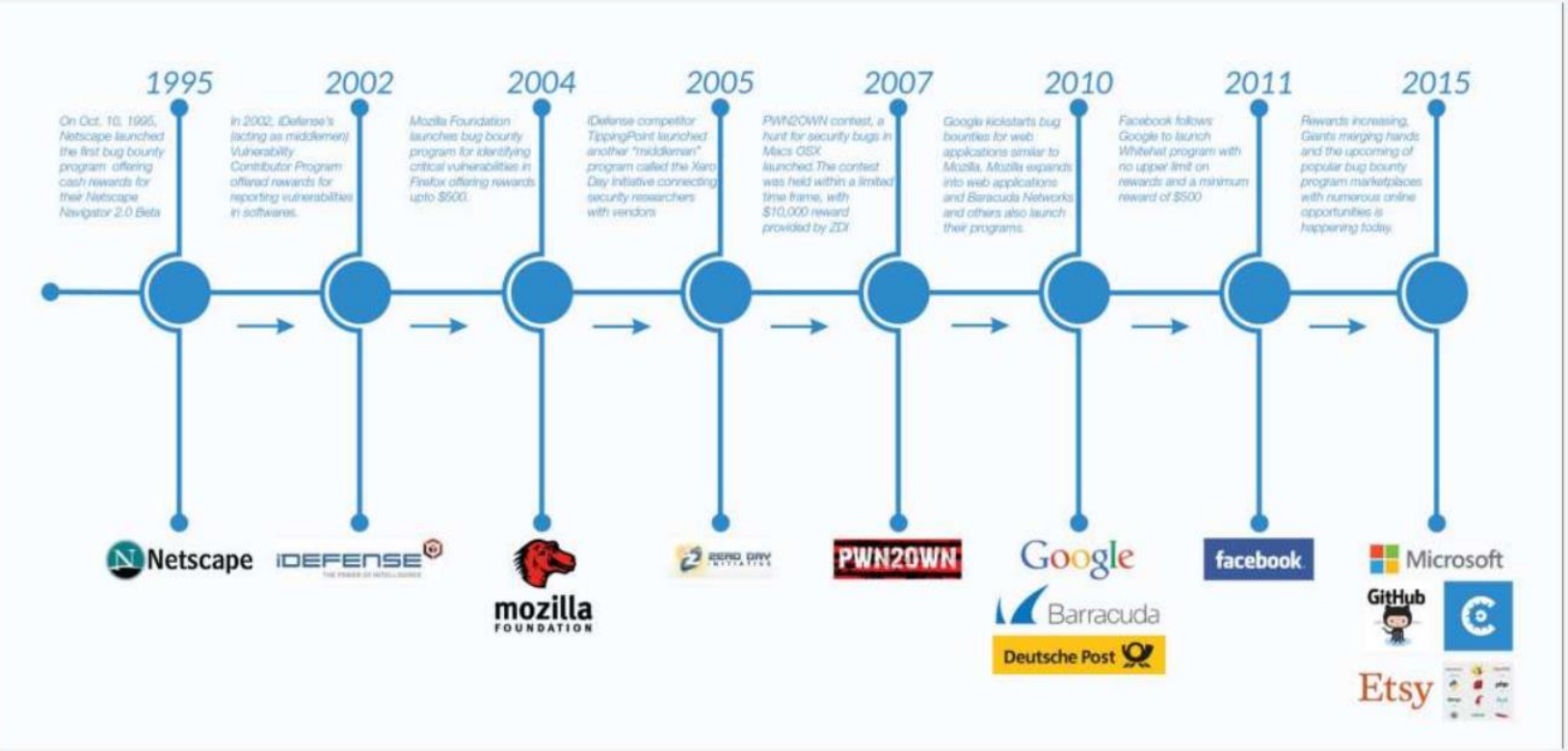
Entry lvl

1.Linux
2.Programming Skills
3.Basics of Networks
4.Git

3- Bug Hunting



HISTORY


1995
On Oct. 10, 1995, Netscape launched the first bug bounty program offering cash rewards for their Netscape Navigator 2.0 Beta
2002
In 2002, iDefense's (acting as middlemen) Vulnerability Contributor Program offered rewards for reporting vulnerabilities in softwares.
2004
Mozilla Foundation launches bug bounty program for identifying critical vulnerabilities in Firefox offering rewards upto $500.
2005
iDefense competitor TippingPoint launched another "middleman" program called the Xero Day Initiative connecting security researchers with vendors
2007
PWN2OWN contest, a hunt for security bugs in Macs OSX launched. The contest was held within a limited time frame, with $10,000 reward provided by ZDI
2010
Google kickstarts bug bounties for web applications similar to Mozilla. Mozilla expands into web applications and Barracuda Networks and others also launch their programs.
2011
Facebook follows Google to launch Whitehat program with no upper limit on rewards and a minimum reward of $500
2015
Rewards increasing, Giants merging hands and the upcoming of popular bug bounty program marketplaces with numerous online opportunities is happening today.
Netscape
iDEFENSE
mozilla FOUNDATION
PWN2OWN
Google
Barracuda
Deutsche Post
facebook
Microsoft
GitHub
Etsy

RUNNING A BUG BOUNTY PROGRAM

Self-Hosted Bug Bounty Program



Using a Bug Bounty Management Provider


GOOGLE VULNERABILITY REWARD PROGRAM

2017 Year in Review

1,230
INDIVIDUAL REWARDS

274
PAID RESEARCHERS

113
COUNTRIES REPRESENTED IN BUG REPORTS

60
COUNTRIES REPRESENTED IN BUG REWARDS

$112,500
BIGGEST SINGLE REWARD

$160,000+
DONATED TO CHARITY

Ahmed Aboul-Ela
@aboul3la

Replying to @michielprins

@michielprins @Hacker0x01 @jobertabma What's the rank of Egypt ? :)

11:25 PM · Mar 29, 2016 · Twitter Web Client

2 Likes

Jobert Abma @jobertabma · Mar 30, 2016
Replying to @aboul3la
@aboul3la @michielprins @Hacker0x01 10th place with 43 hackers!

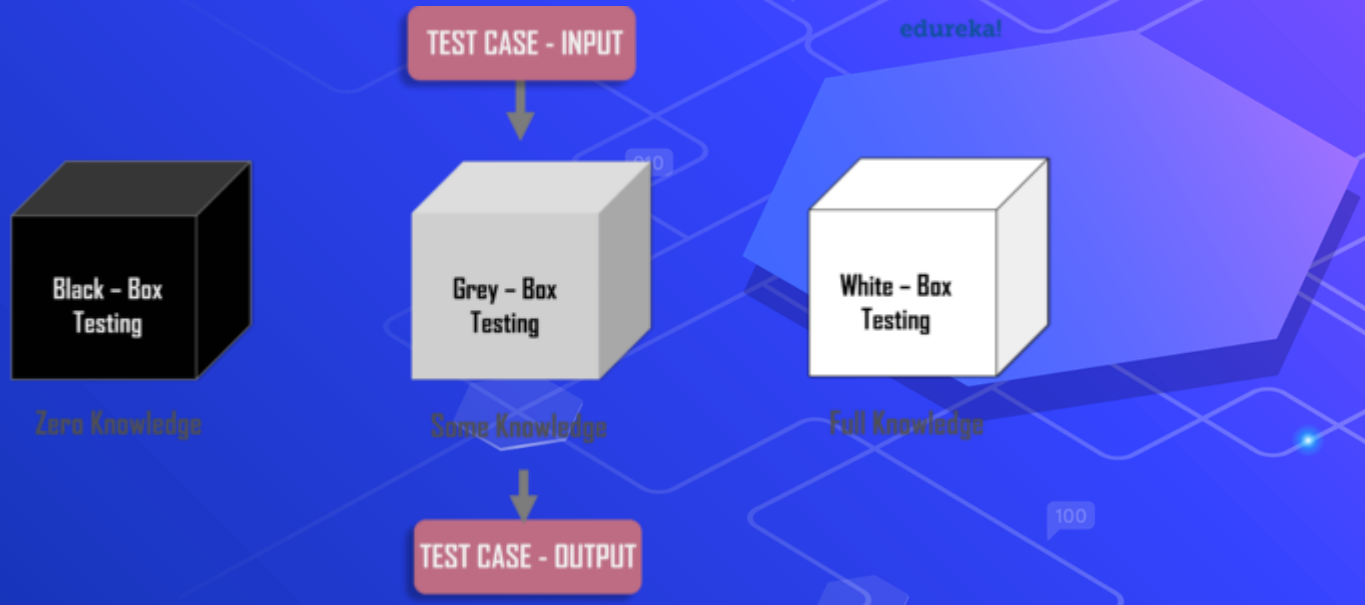
2 4

Hussain Adnan @hussain_0x3c · Apr 5, 2016
@jobertabma @aboul3la @michielprins @Hacker0x01 @Hat_Mast3r I hope you raise the rank of Iraqi security researcher on HackerOne :)

1

4- Penetration Testing

Types of Penetration Testing:



Owasp

What is Owasp ?

OWASP stands for the Open Web Application Security Project

What is Owasp Top 10 ?

OWASP Top 10 - 2013		OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	U	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	U	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	☒	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	☒	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

A1:2017- Injection

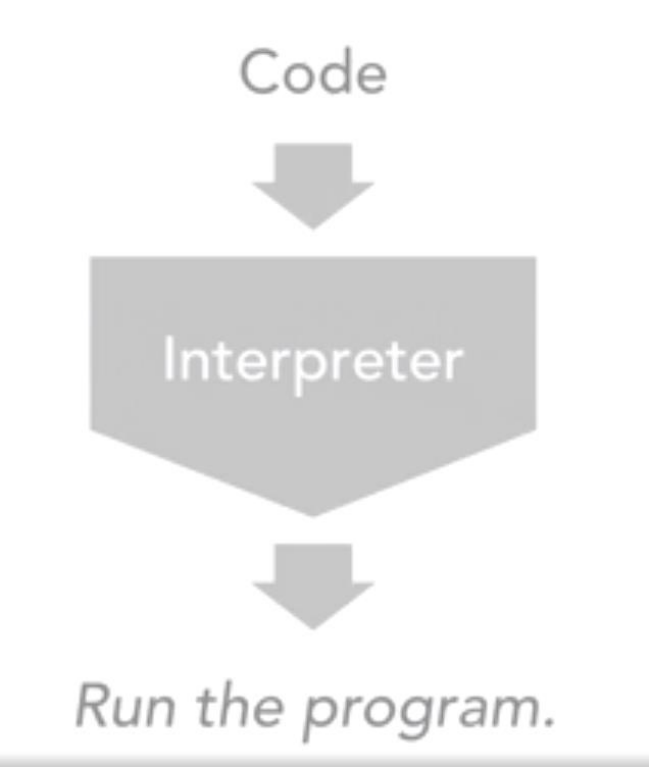
A1:2017- Injection

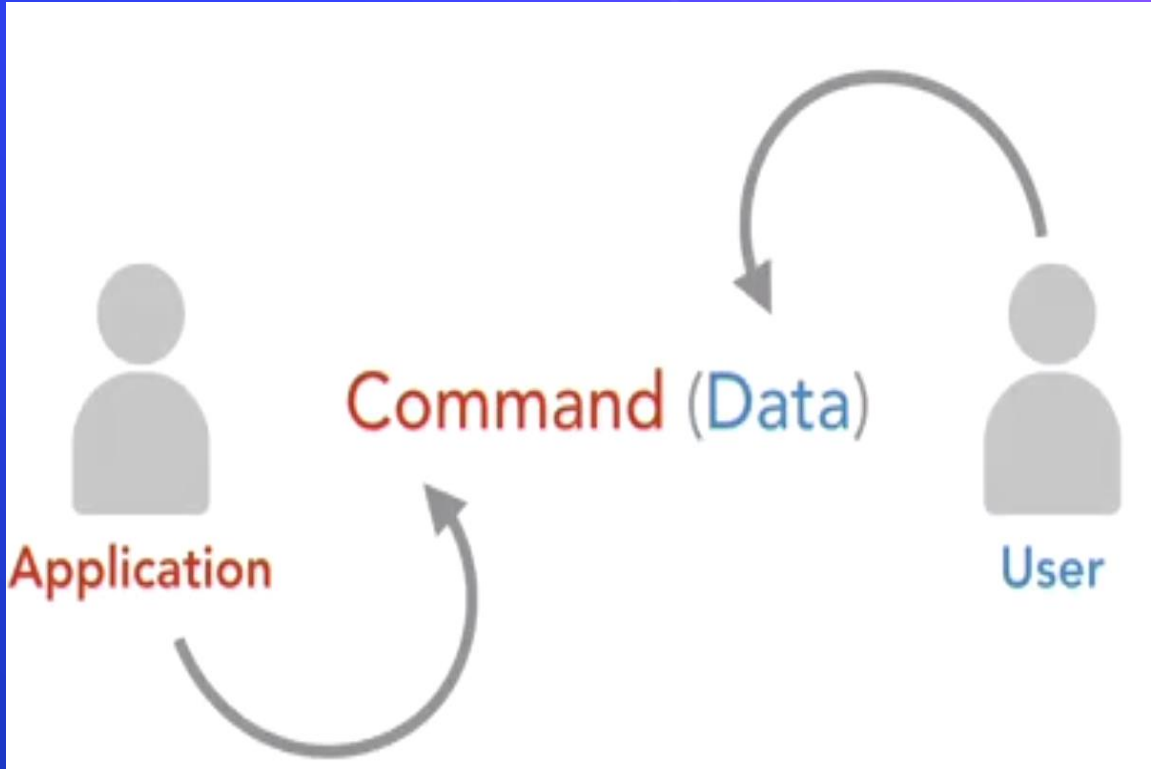
injection flaws, such as SQL, OS, XXE, and LDAP injection occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.

Code = Data = Commands

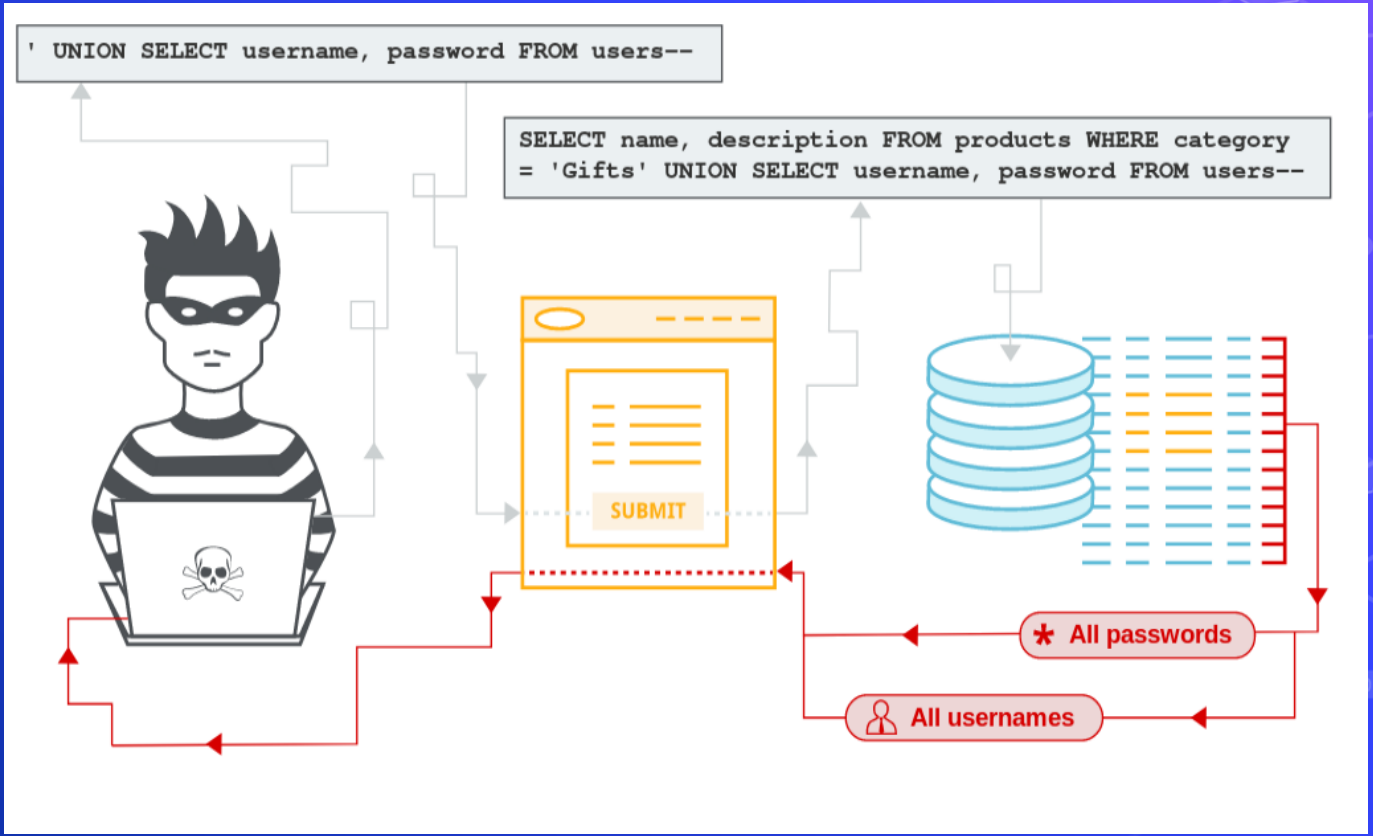
A1:2017- Injection

Injection – How Does it Work ?



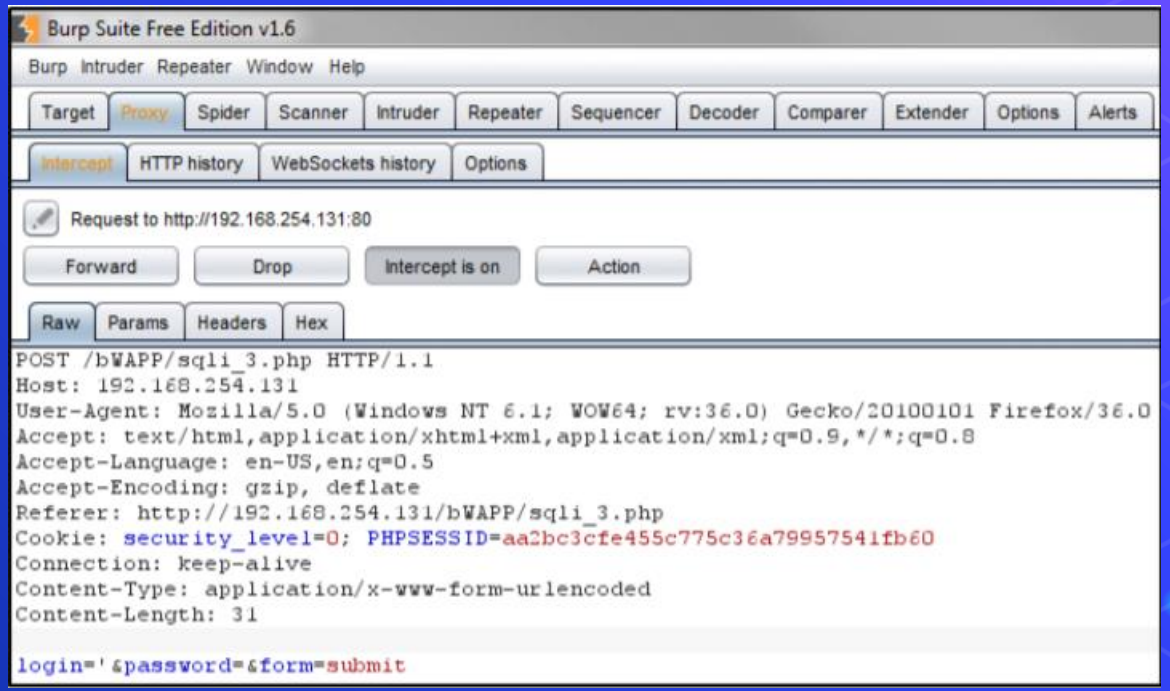


Demo SQL-injection



Demo SQL-injection

A1:2017- Injection -Demo



A1:2017- Injection -Demo

‘ or 1=1#

Username	Password	Match
Charles	zxcvbnm	false
admin	password	true

```
$sql = "SELECT * FROM users WHERE name = '" . $_POST["user"] . "' AND pass = '" .
$result = mysqli_query($db,$sql);
$count = mysqli_num_rows($result);

if ($count >= 1){
        login();
}
```

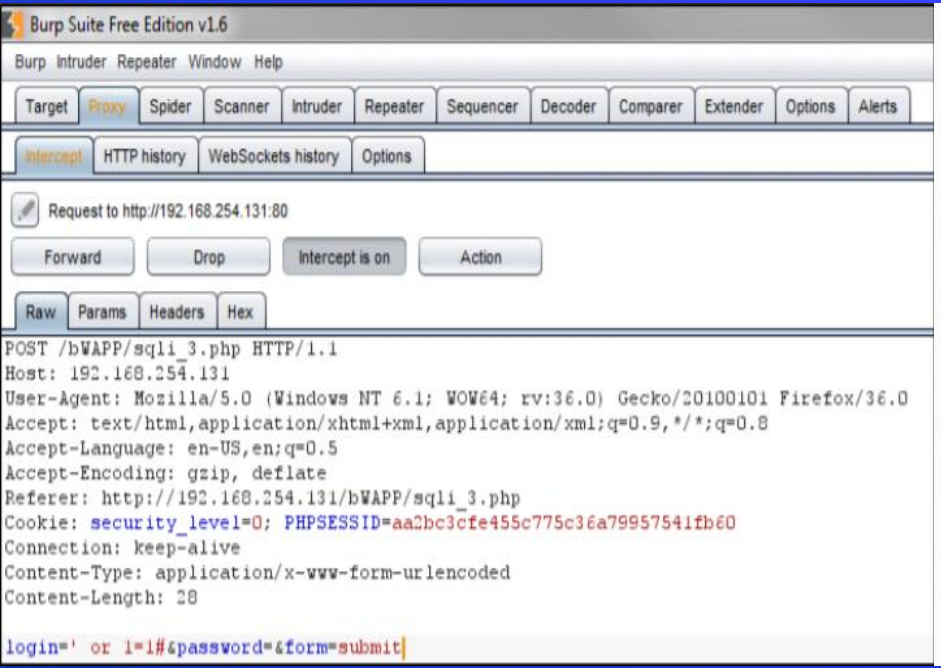
A1:2017- Injection -Demo

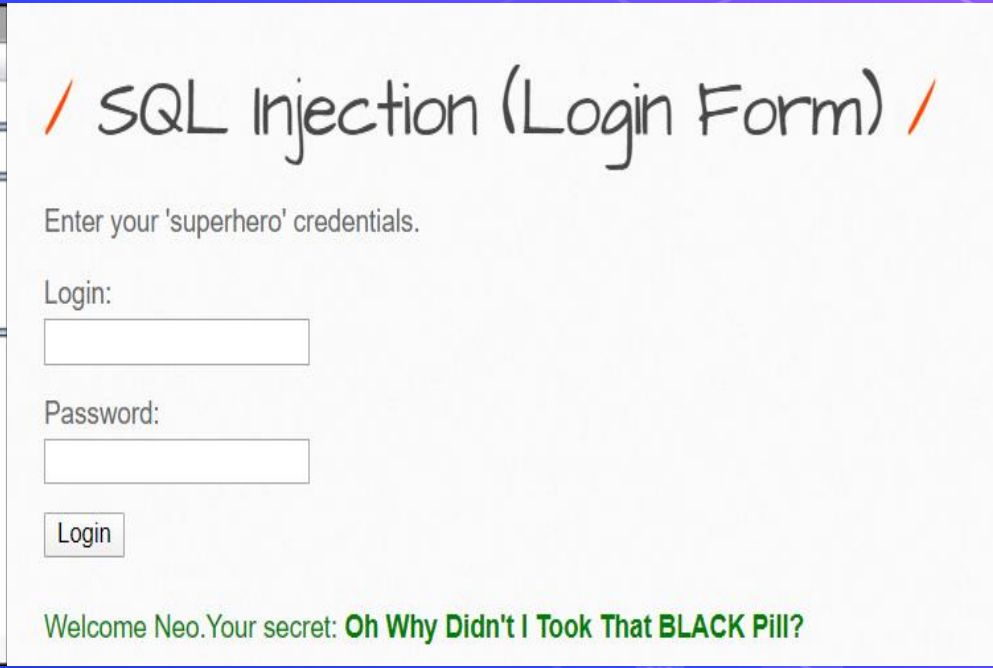
```
$sql = "SELECT * FROM users WHERE name = '" . $_POST["user"] . "' AND pass = '" .
$result = mysqli_query($db,$sql);
$count = mysqli_num_rows($result);

if ($count >= 1){
        login();
}
```

Username	Password	Match
Charles	zxcvbnm	true
admin	password	true
admin2	summer2017	true

A1:2017- Injection -Demo





A2 Broken Authentication

Broken Authentication : Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently

A2:2017-Broken Authentication

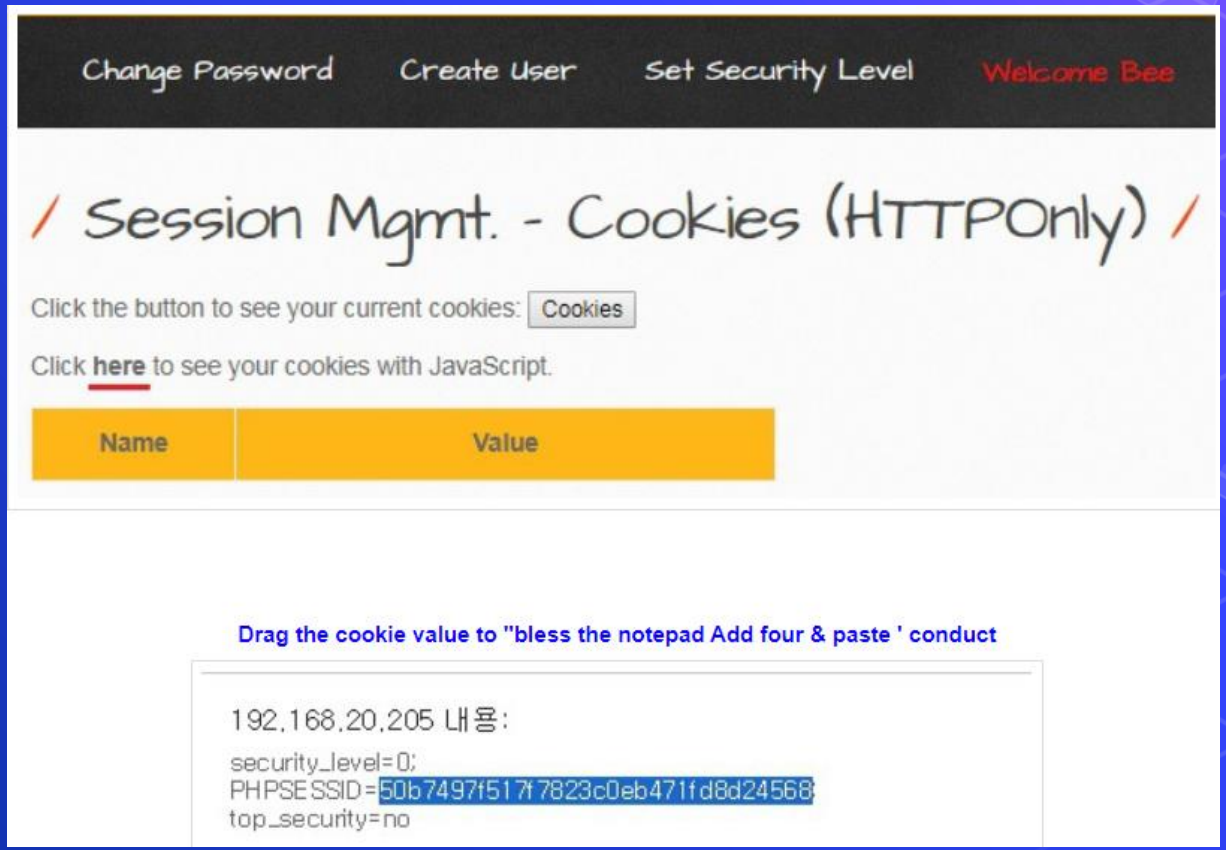
Authentication

- Who are you?
- What is yours?

Session

- Active account
- In use by account owner

Demo


Change Password
Create User
Set Security Level
Welcome Bee
/ Session Mgmt. - Cookies (HTTPOnly) /
Click the button to see your current cookies:
Cookies
Click here to see your cookies with JavaScript.
Name
Value
Drag the cookie value to "bless the notepad Add four & paste ' conduct
192.168.20.205 내용:
security_level=0;
PHPSESSID=50b7497f517f7823c0eb471fd8d24568
top_security=no

Secure VS HttpOnly

`Secure` and `HttpOnly` cookies

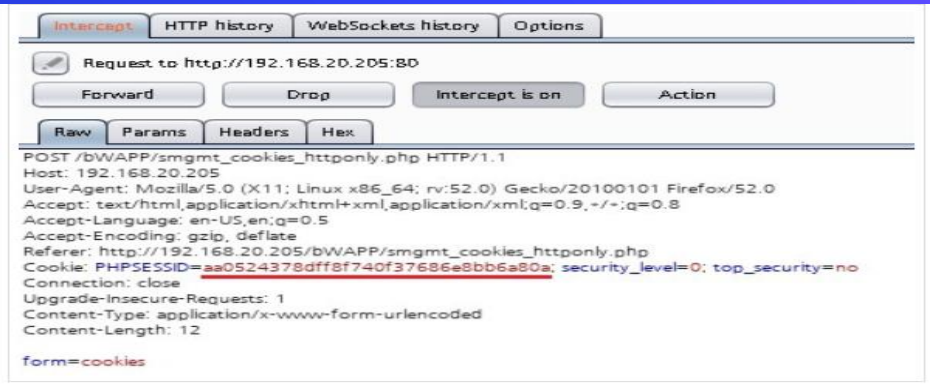
A secure cookie is only sent to the server with an encrypted request over the HTTPS protocol. Even with `Secure`, sensitive information should *never* be stored in cookies, as they are inherently insecure and this flag can't offer real protection. Starting with Chrome 52 and Firefox 52, insecure sites (`http:`) can't set cookies with the `Secure` directive.

To help mitigate cross-site scripting (XSS) attacks, `HttpOnly` cookies are inaccessible to JavaScript's `Document.cookie` API; they are only sent to the server. For example, cookies that persist server-side sessions don't need to be available to JavaScript, and the `HttpOnly` flag should be set.

Demo


Change Password
Create User
Set Security Level
Welcome Test
/ Session Mgmt. - Cookies (HTTPOnly) /
Click the button to see your current cookies:
Cookies
Click here to see your cookies with JavaScript.
Name
Value

Demo



Change cookie information of test account to cookie information of bee account

form = cookies
Host: 192.168.20.205
User-Agent: Mozilla / 5.0 (X11; Linux x86_64; rv: 52.0) Gecko / 20100101 Firefox / 52.0
Accept: text / html, application / xhtml + xml, application / xml; q = 0.9, * / *; q = 0.8
Accept-Language: en-US, en; q = 0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.20.205/bWAPP/smgmt_cookies_httponly.php
Cookie: PHPSESSID = aa0524378dff8f740f37686e8bb6a80a; security_level = 0; top_security = no
50b7497f517f7823c0eb471fd8d24568 < -change to cookie information in bee account
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application / x-www-form-urlencoded
Content-Length: 12

form = cookies

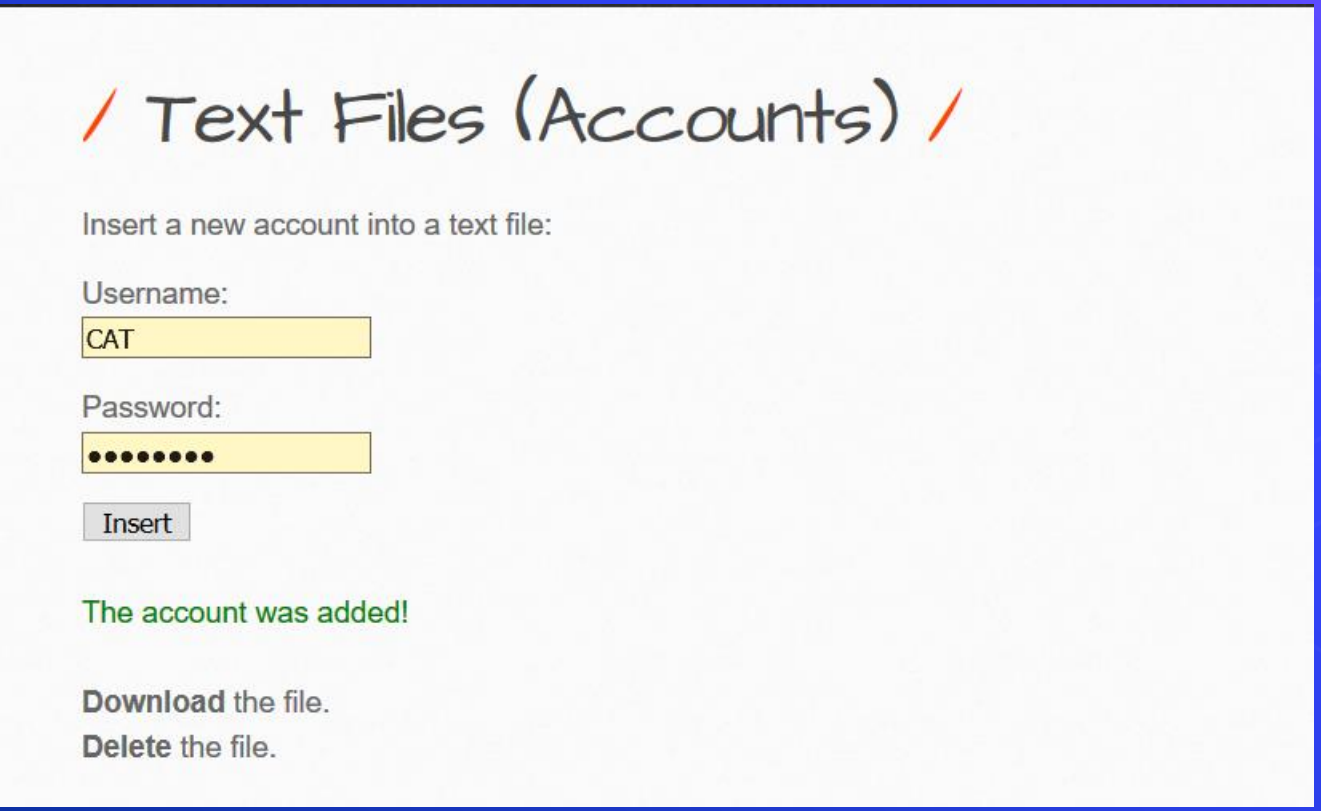
Demo


Change Password
Create User
Set Security Level
Welcome Bee
/ Session Mgmt. - Cookies (HTTPOnly) /
Click the button to see your current cookies:
Cookies
Click here to see your cookies with JavaScript.

A3 Sensitive Data Exposure

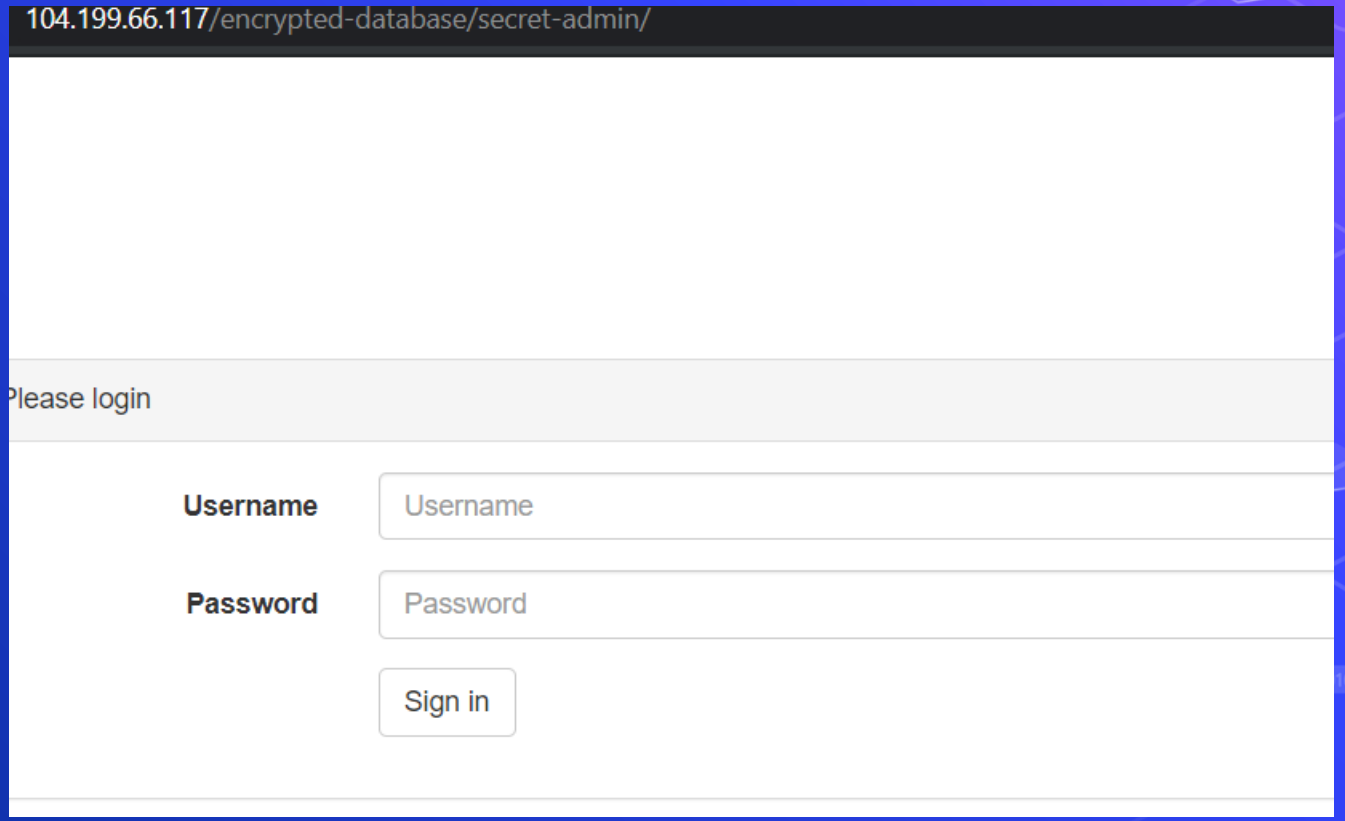
“Sensitive data exposure : Sensitive data exposure vulnerabilities can occur when an application does not adequately protect sensitive information from being disclosed to attackers. For many applications this may be limited to information such as passwords, but it can also include information such as credit card data, session tokens, or other authentication credentials.”

A3:2017-Sensitive Data Exposure Demo



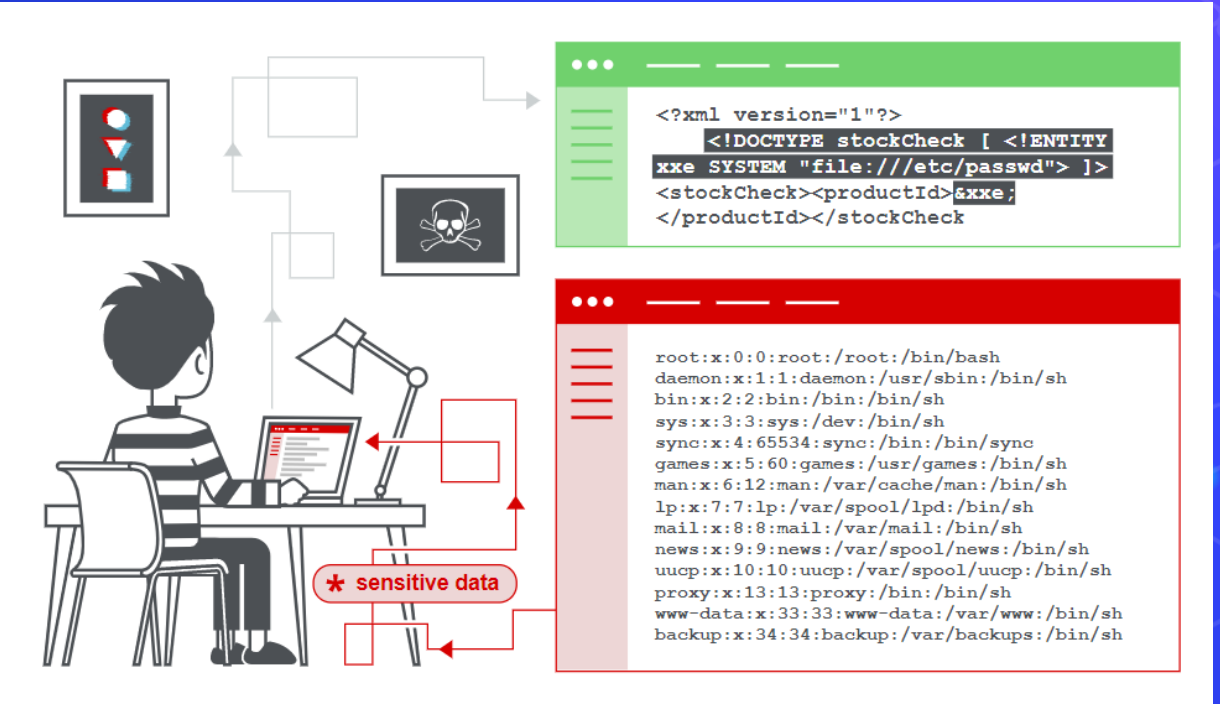

```
'bee', 'bug'
'yasser', 'mohamed'
'CAT ', 'Reloaded'
```

Demo
CTF- CyberTalents



A4-XML External Entities (XXE)

Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.



XXE-How Does It Work ?

Supply

Order

Demo

```
POST /product/stock HTTP/1.1
Host: ace61f5a1e08bd8a80ac221300520089.web-security-academy.net
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://ace61f5a1e08bd8a80ac221300520089.web-security-academy.net/product?productId=1
Content-Type: application/xml
Origin: https://ace61f5a1e08bd8a80ac221300520089.web-security-academy.net
Content-Length: 182
Connection: close
Cookie: session=CNf23VmWXYKrOOupvaVvOJTFnlkFVBn8
// yasser
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE test [ <!ENTITY xxe SYSTEM "file:///etc/passwd"> ]>
<stockCheck>
<productId>&xxe;</productId>
<storeId>1</storeId>
</stockCheck>
```

Demo

```
HTTP/1.1 400 Bad Request
Content-Type: application/json
Connection: close
Content-Length: 1144

"Invalid product ID: root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
peter:x:2001:2001::/home/peter:/bin/bash
user:x:2000:2000::/home/user:/bin/bash
dnsmasq:x:101:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
messagebus:x:102:101::/nonexistent:/usr/sbin/nologin
"
```

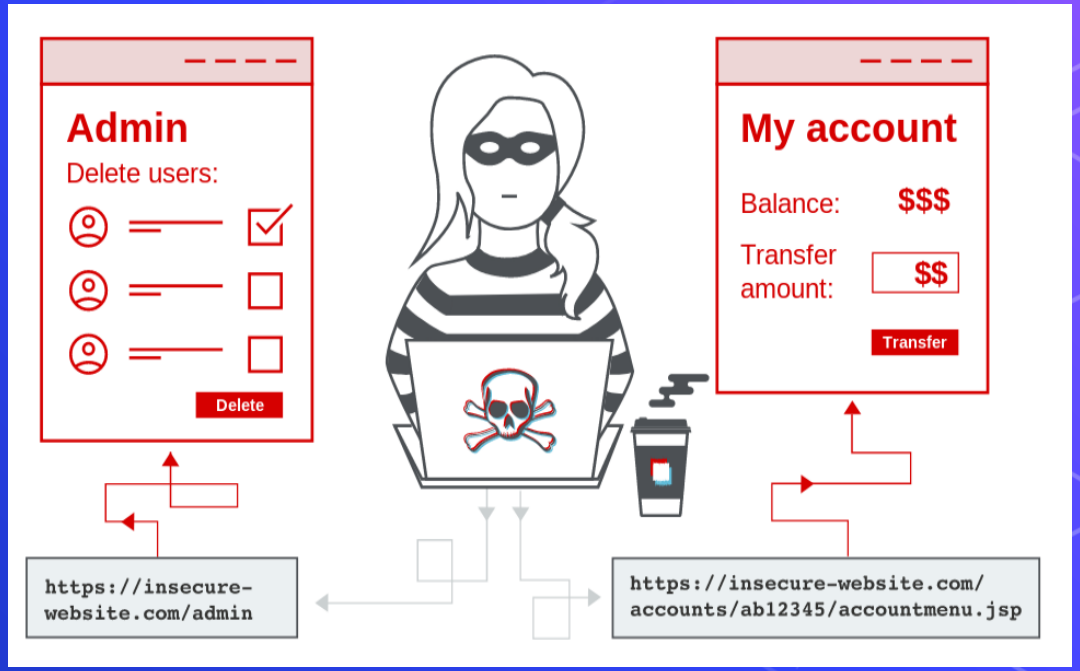
A5 : Broken Access Control

Access control : determines whether the user is allowed to carry out the action that they are attempting to perform.

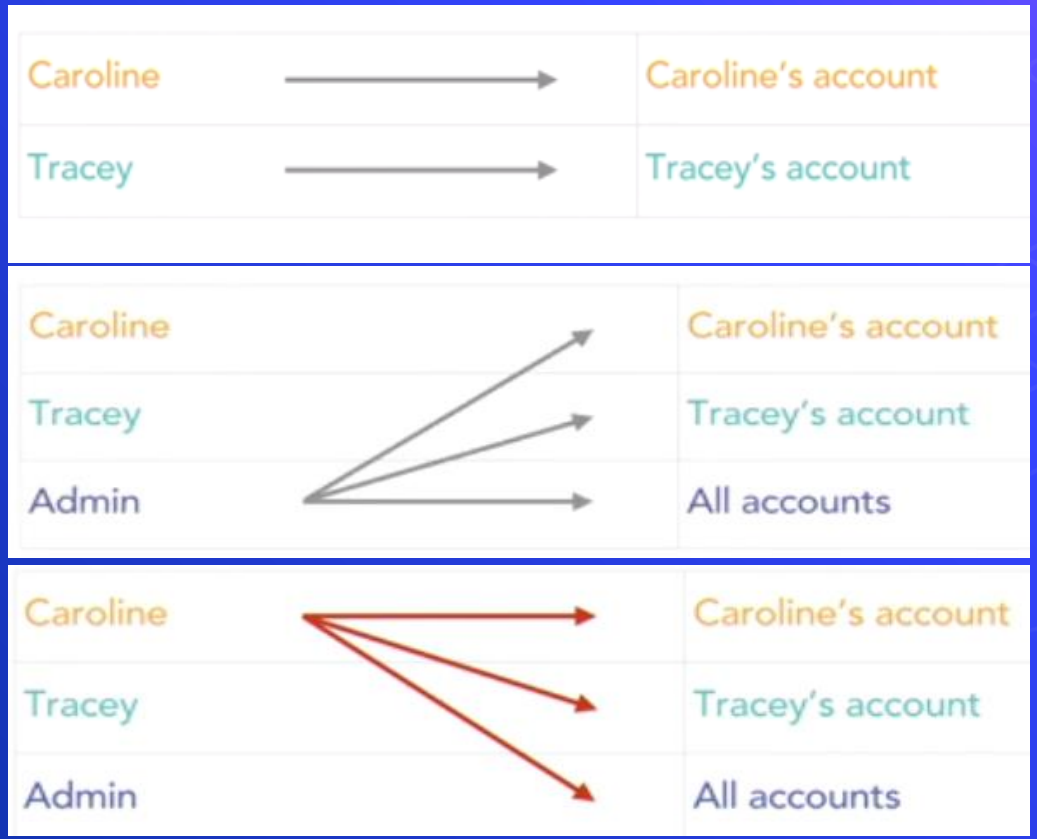
authorization : the idea that, depending on your role, you should have access to specific things and be able to perform specific actions

be divided into the following categories:

- **Vertical access controls**
- If a user can gain access to functionality that they are not permitted to access then this is vertical privilege escalation
- **Horizontal access controls**
- arises when a user is able to gain access to resources belonging to another user



A5 : Broken Access Control



Normal users

Admin

Broken Access control

subcategory of access control vulnerabilities. IDOR arises when an application uses user-supplied input to access objects directly and an attacker can modify the input to obtain unauthorized access

Demo -Insecure direct object references (IDOR)

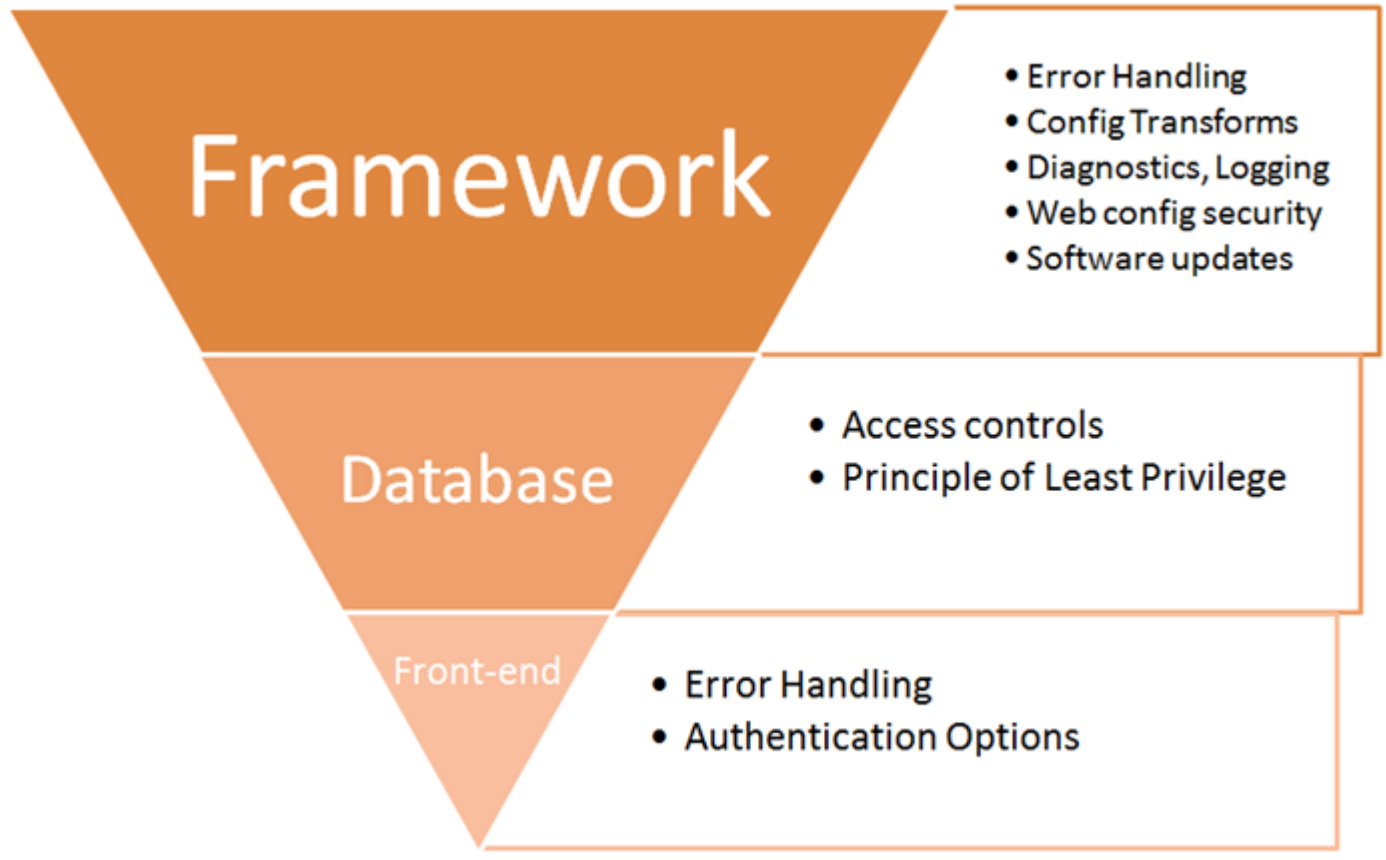
A6 Security Misconfiguration

Software development life cycle :

Security Misconfiguration occurs when a system is configured in a way that makes it vulnerable

Ex:

- Default system credentials: user accounts, passwords in factory default or unchanged status
- Directory and file listings: not disabled and easily available through search engines
- User traces: pages returned to users with error messages that have too much information in them
- Unnecessary pages: sample apps, old privileges, and user accounts, for example
- Software: not up-to-date, legacy systems, patches not utilized

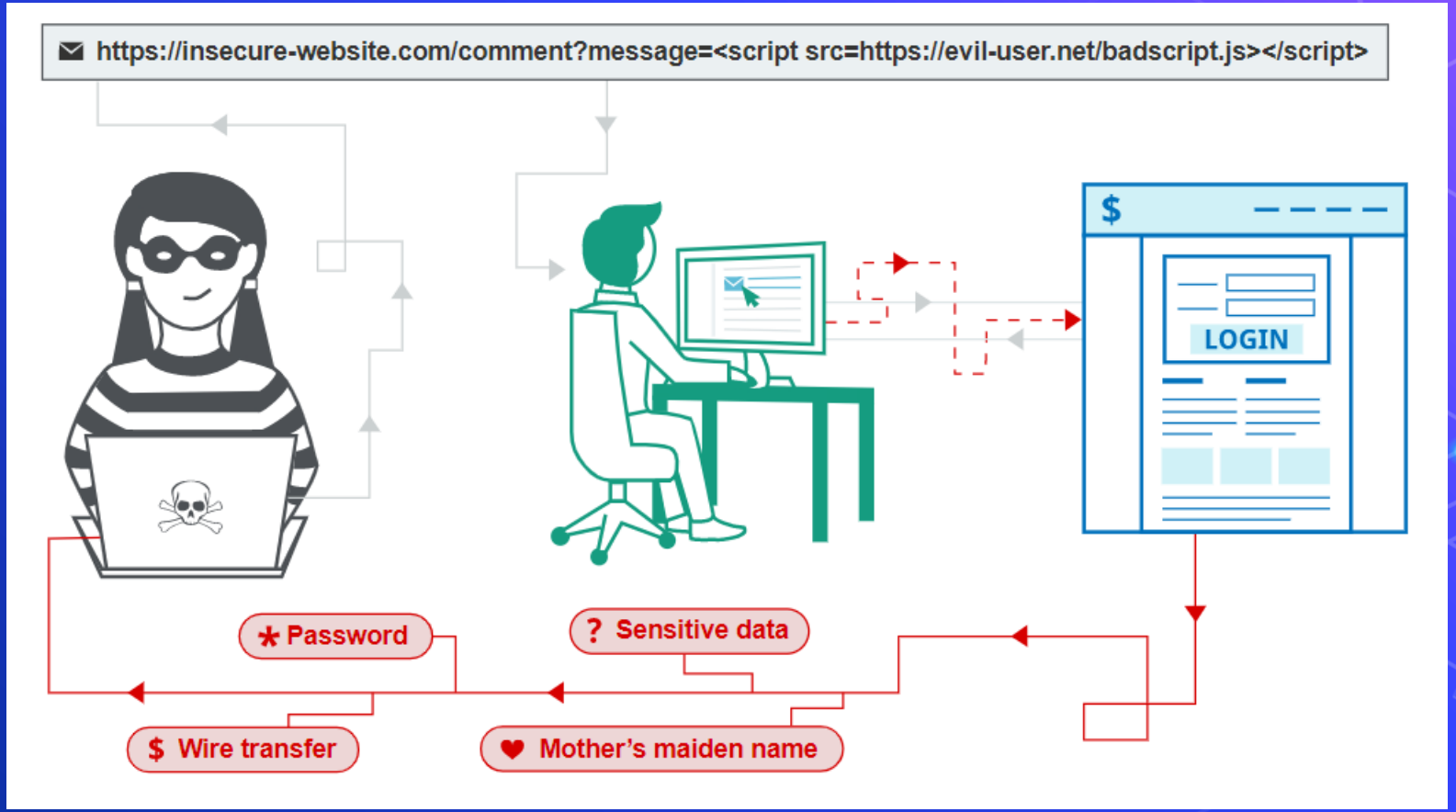

Framework
• Error Handling
• Config Transforms
• Diagnostics, Logging
• Web config security
• Software updates
Database
• Access controls
• Principle of Least Privilege
Front-end
• Error Handling
• Authentication Options

A7:2017-Cross-Site Scripting (XSS)

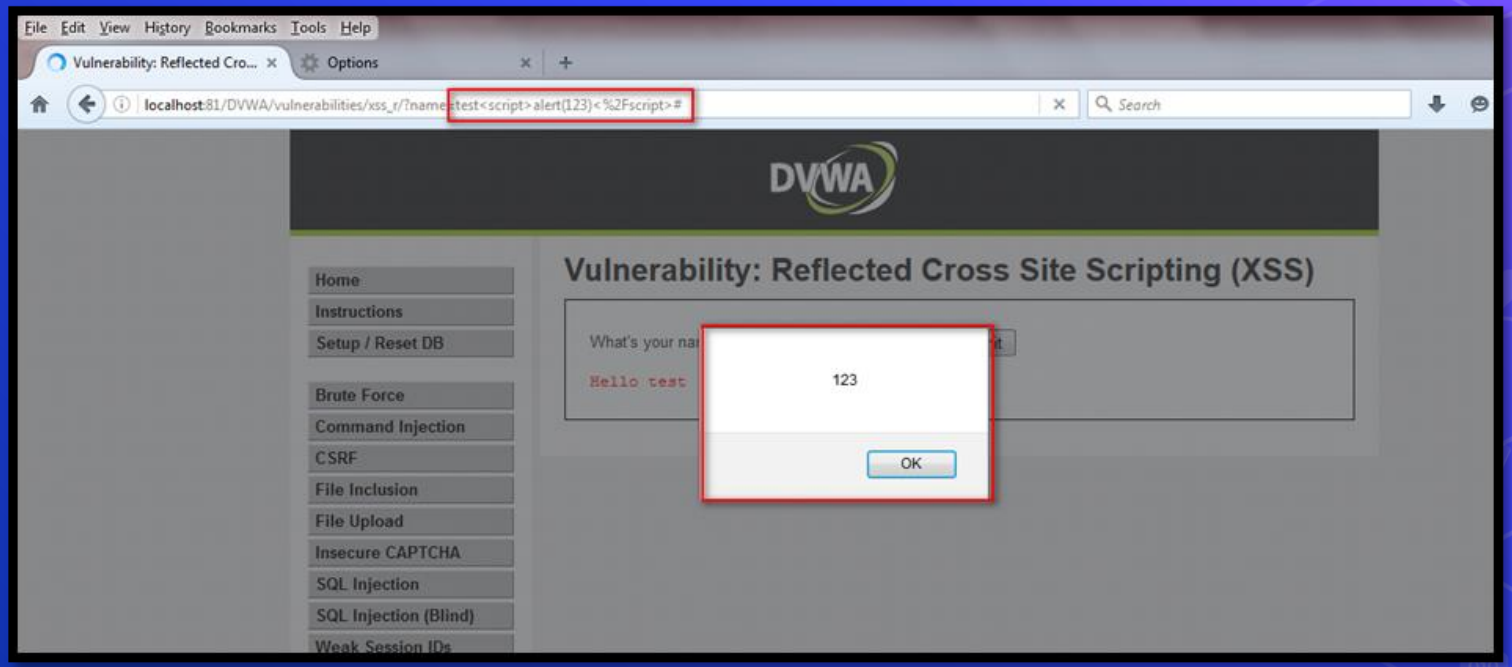
“XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user supplied data using a browser API that can create JavaScript. XSS allows attackers to execute scripts in the victim’s browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites. ”

```
<script> do cool things </script>
<script> do malicious things </script>
```

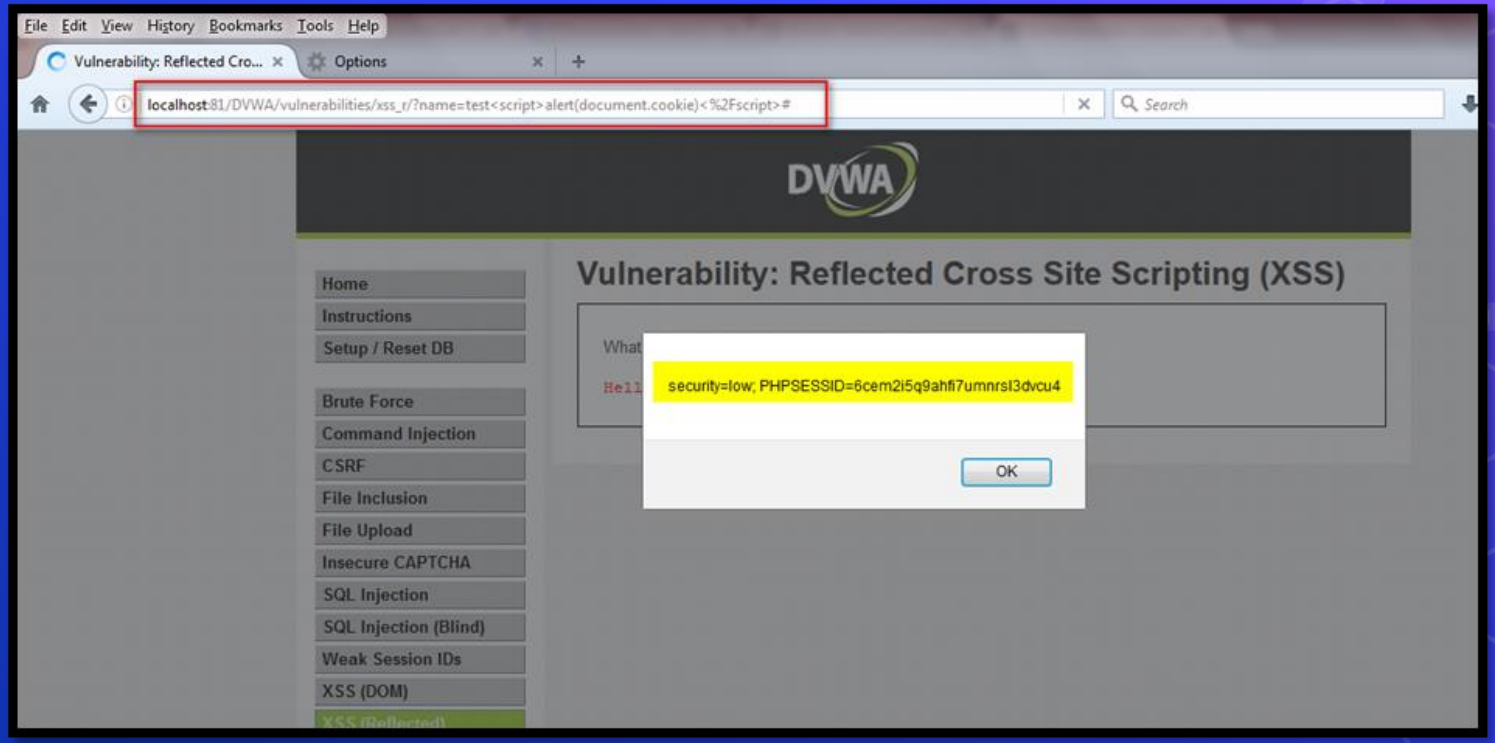
A7 Cross-Site Scripting (XSS)



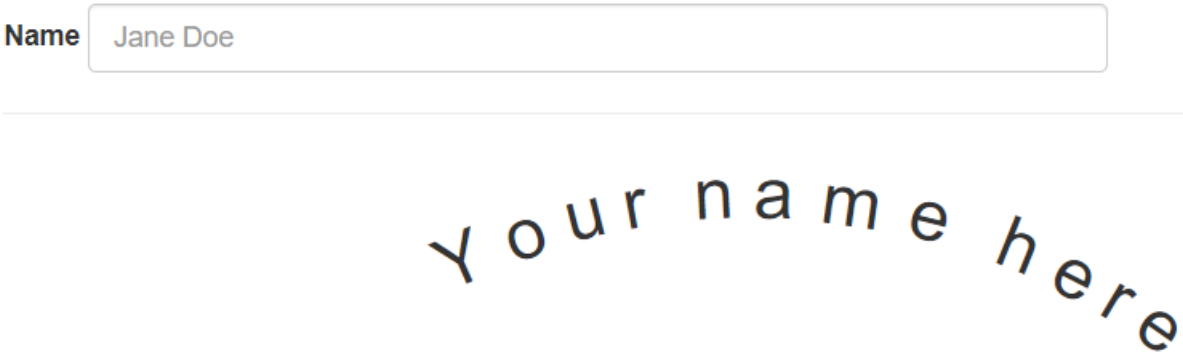
A7 Cross-Site Scripting (XSS)



A7 Cross-Site Scripting (XSS)



A7:2017-Cross-Site Scripting (XSS)-DEMO CTF Cybertalents



Demo2 Google XSS Game

- **<script>alert(document.cookie)</script> reflected xss**
- **<img src=x onerror="alert("CAT Reloaded")"> stored xss**
- **x' onerror="alert('CAT Reloaded')“ OR 3'><script>alert("CAT Reloade")</script>**

A8 Insecure Deserialization

allows attackers to transfer a payload using serialized objects. This happens when integrity checks are not in place and deserialized data is not sanitized or validated

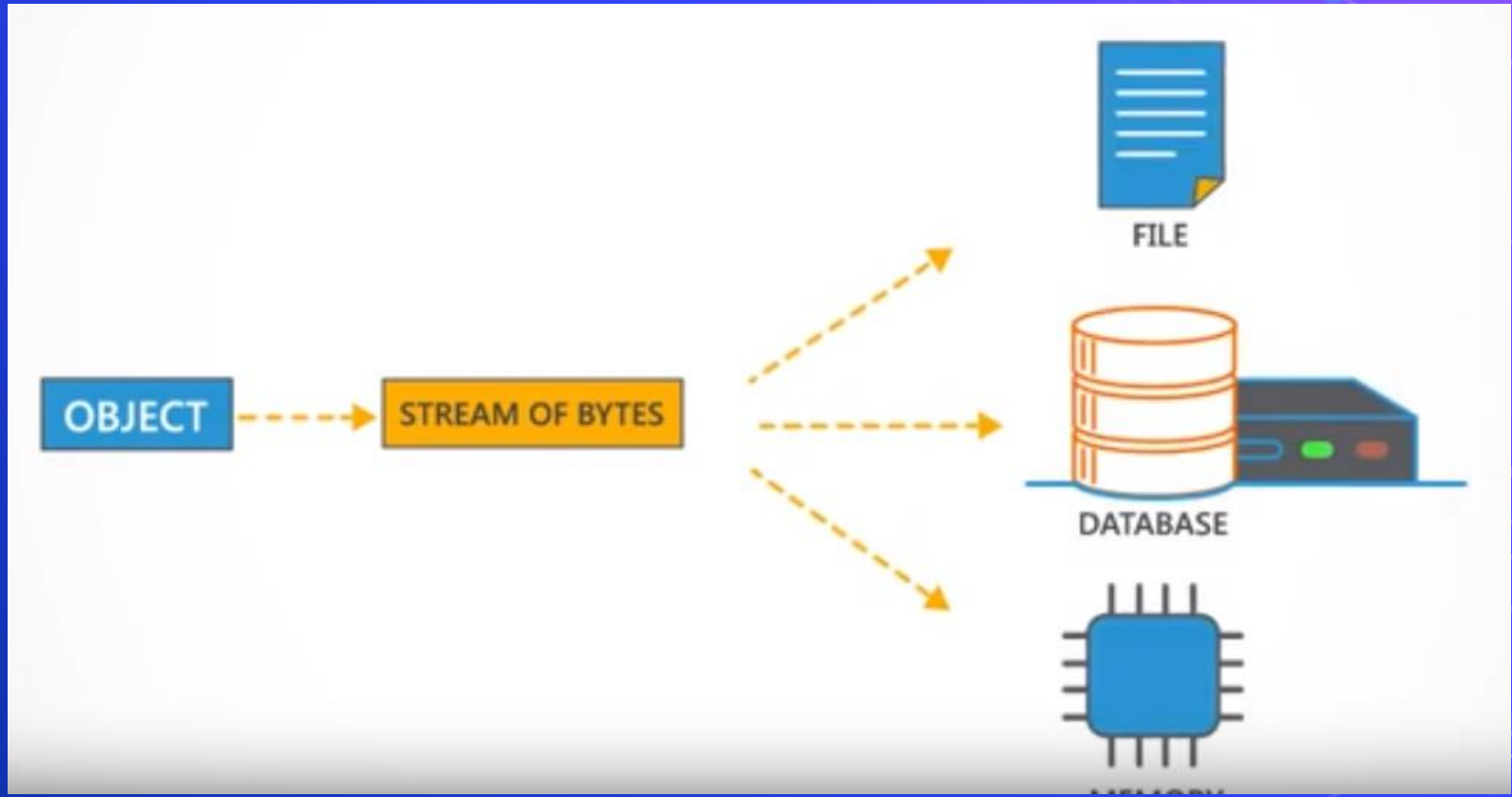
This conversion is called serialization. Serialization is the process of taking an object and translating it into plaintext. This plaintext can then be encrypted or signed, as well as simply used the way it is. The reverse process is called deserialization, i.e. when the plaintext is converted back to an object.

a:4:{i:0;i:132;i:1;s:7:"Mallory";i:2;s:4:"user"; i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}

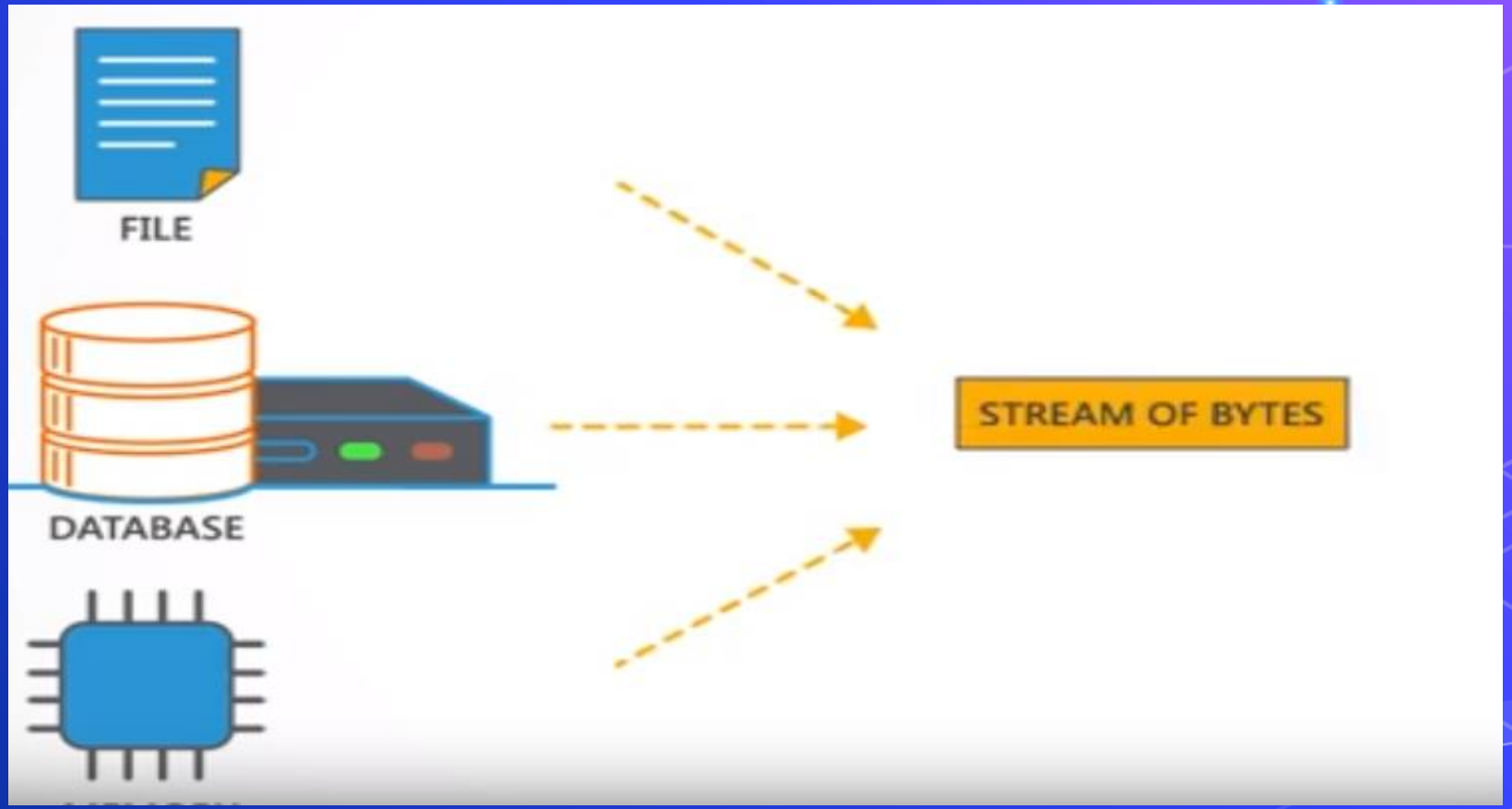
An attacker changes the serialized object to give themselves admin privileges:

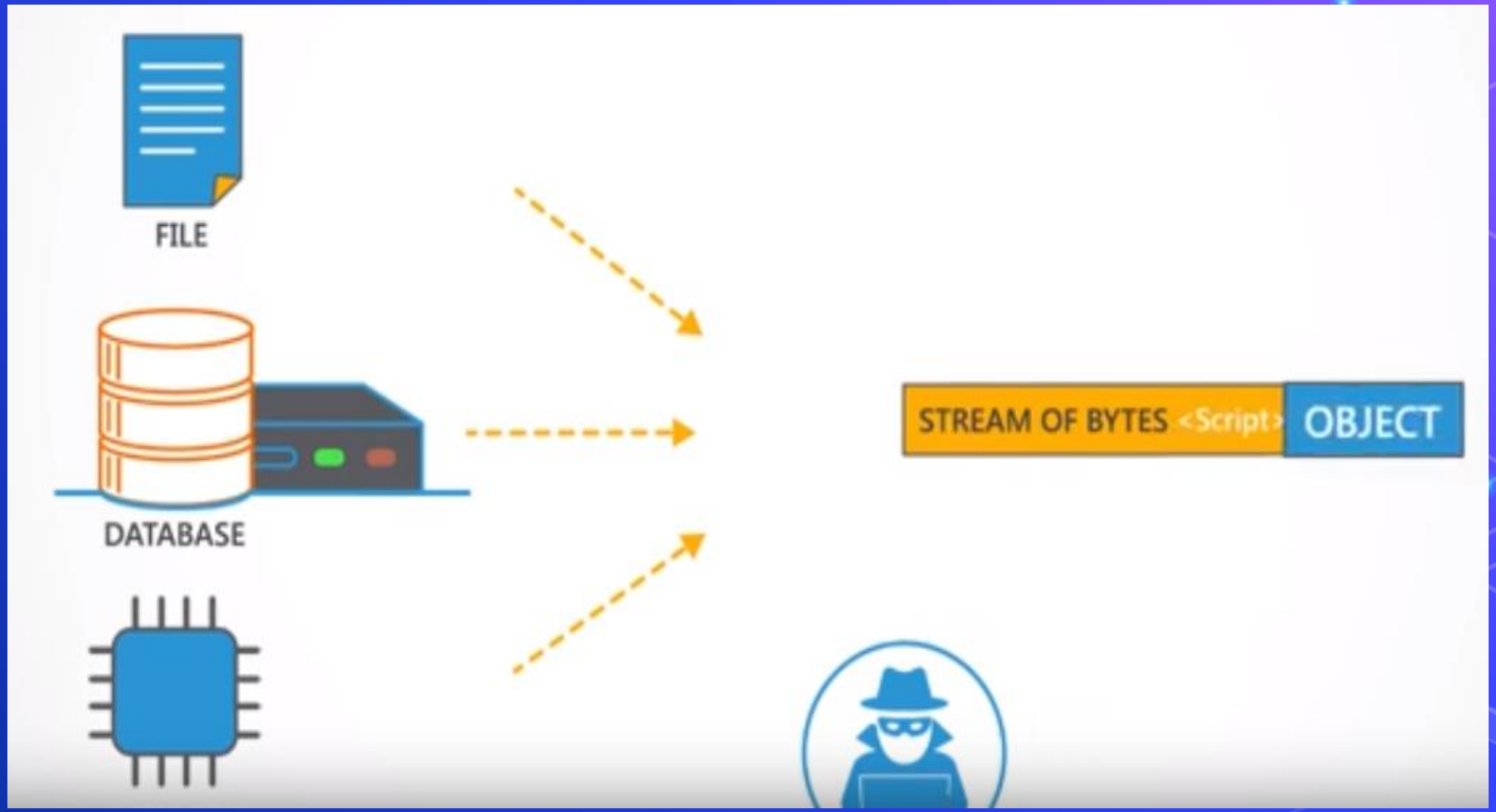
a:4:{i:0;i:1;i:1;s:5:"Alice";i:2;s:5:"admin"; i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}

serialization

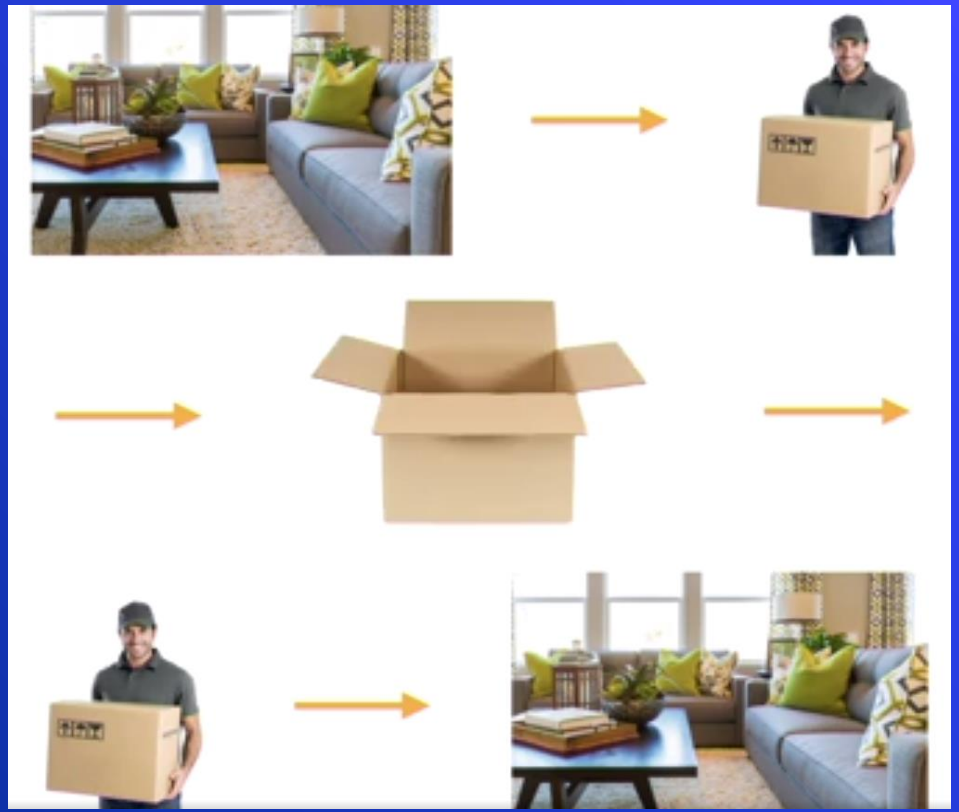


Deserialization




FILE
DATABASE
STREAM OF BYTES <Script> OBJECT

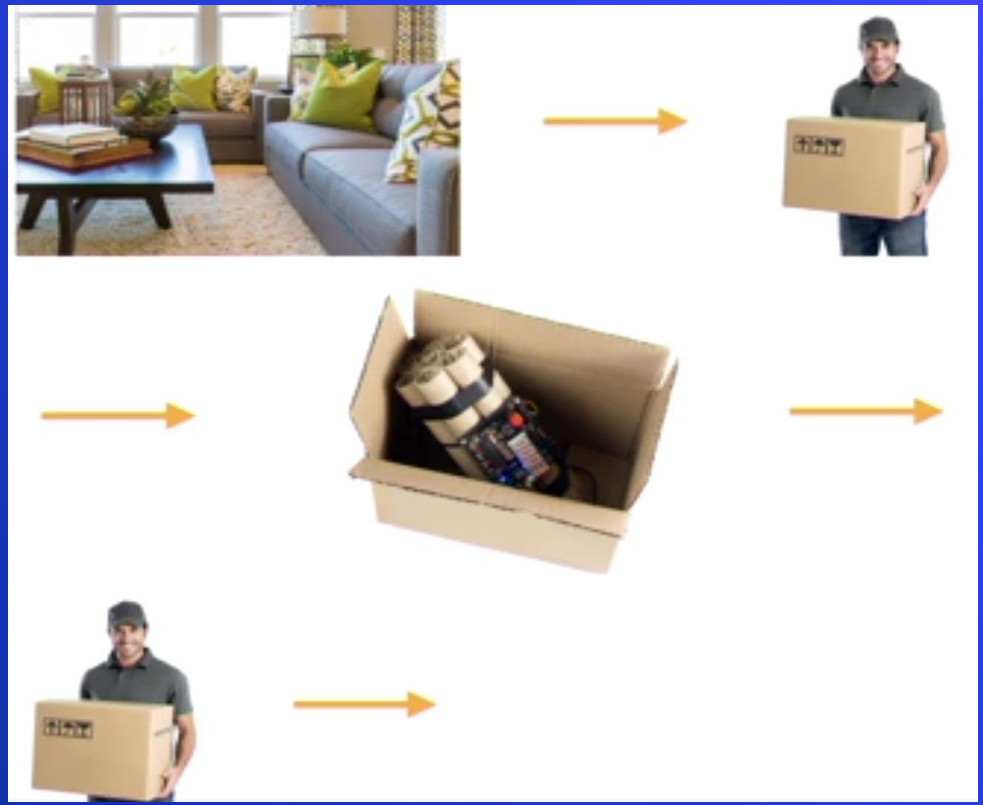
A8 secure Deserialization



serialization

serialization

A8 Insecure Deserialization



serialization

serialization

Demo

A8 Insecure Deserialization

Demo

A9 Using Components with Known Vulnerabilities

Demo

A10 :2017 Insufficient Logging & Monitoring

Any questions?

Thanks!

Any questions?

You can find me at:

Twitter : @YasserElsnbary2

Ask : https://ask.fm/ElsnbaryYasser