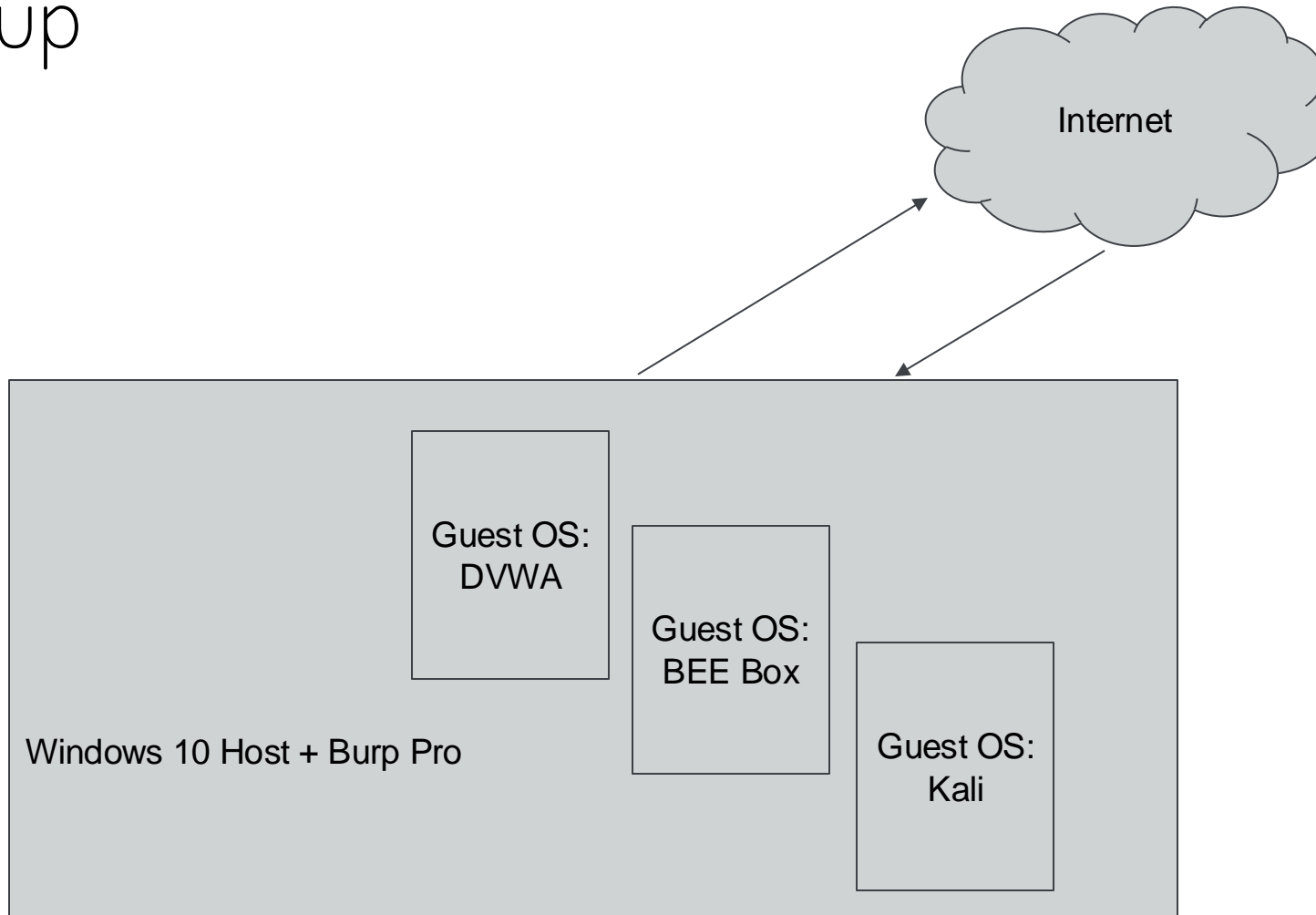




Web Hacking 101

Sheikh Rizan

Lab Setup



1. Kali: 8GB RAM + 4 CPU + bridge
2. DVWA (default) + Bridge
3. BEE Box (Default) + Bridge

Course Outline

9 AM – 11:00AM

Introduction

Tooling & Methodology

Security Testing Cycles

Common Vulnerabilities

11:00AM – 11:20 AM

Break

11:20 AM – 12:30 Noon

Code Review

Secure Coding Guide

Intro to Burpsuite

Lab Practice I

12 :30– 1:30 PM

Lunch Break

1:30 PM – 5:00 PM

Lab Practice II (3hours)

Trainer

BAE SYSTEMS

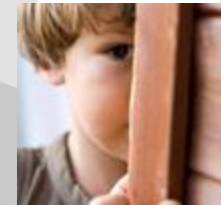
- More than 20 years in IT Security, 7 years offensive security.
- Worked for Telecoms, Oil & Gas, Military & Law Enforcement surveillance solutions.
- CISSP, CISA, OSWE, OSCE, OSCP, CREST CRT & Burp Certified Practitioner.
- Bug bounties findings for US DoD, Spotify, Amazon, Sony, Alibaba, Dell, etc
- Published custom exploits at github, packetstorm & exploit-db.
- Presented @ B-Sides Singapore, Rootcon Philippines & RawSec Malaysia.
- Vulnerability Research Publications: 9 CVEs & 3 private 0-days.

Introduction

BAE SYSTEMS

- 40 % theory + 60% lab practice.
- Designed for beginners with no or little knowledge in Web Penetration Testing.
- CTF (Capture the Flag) VM will be used as target practice.
- Burpsuite pro is highly recommended.

- **Don't be shy** to ask any questions; no such thing as a silly question.
- The more you practice, the better you'll get. Pen-testing is all about practice.

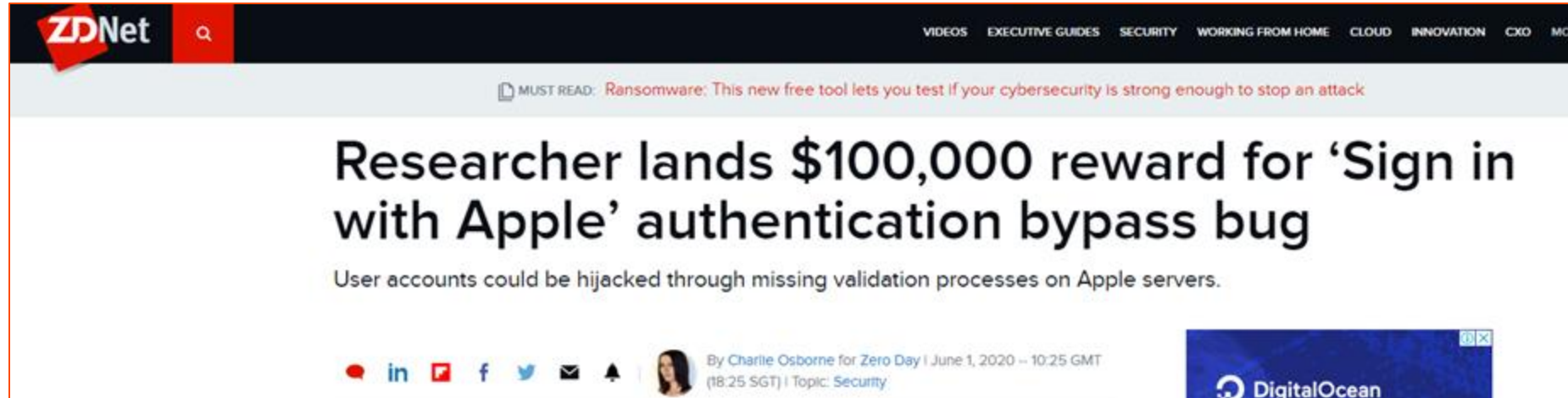


Disclaimer

BAE SYSTEMS

DO NOT ATTEMPT TO HACK OR GAIN UNAUTHORIZED ACCESS TO ANY PRIVATELY OWNED WEBSITES, SERVERS, HOSTS ON THE INTERNET. ANY UNLAWFUL ACTIVITY IS STRICTLY NOT CONDONED AND WILL GET YOU IN TROUBLE WITH THE LAW. BAE SYSTEMS WILL NOT BE HELD LIABLE FOR ANY UNLAWFUL ACTIVITIES THAT YOU CONDUCT AS A RESULT OF THIS COURSE. THE CONTENTS OF THIS COURSEWARE IS STRICTLY FOR EDUCATIONAL PURPOSES ONLY.

Vulnerability Disclosure Programs & Bug Bounties



Facebook awards \$55k bug bounty for third-party vulnerabilities that could compromise its internal network

Adam Bannister 23 March 2021 at 11:56 UTC
Updated: 02 July 2021 at 12:01 UTC

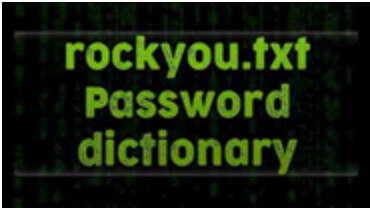
Tooling & Methodologies

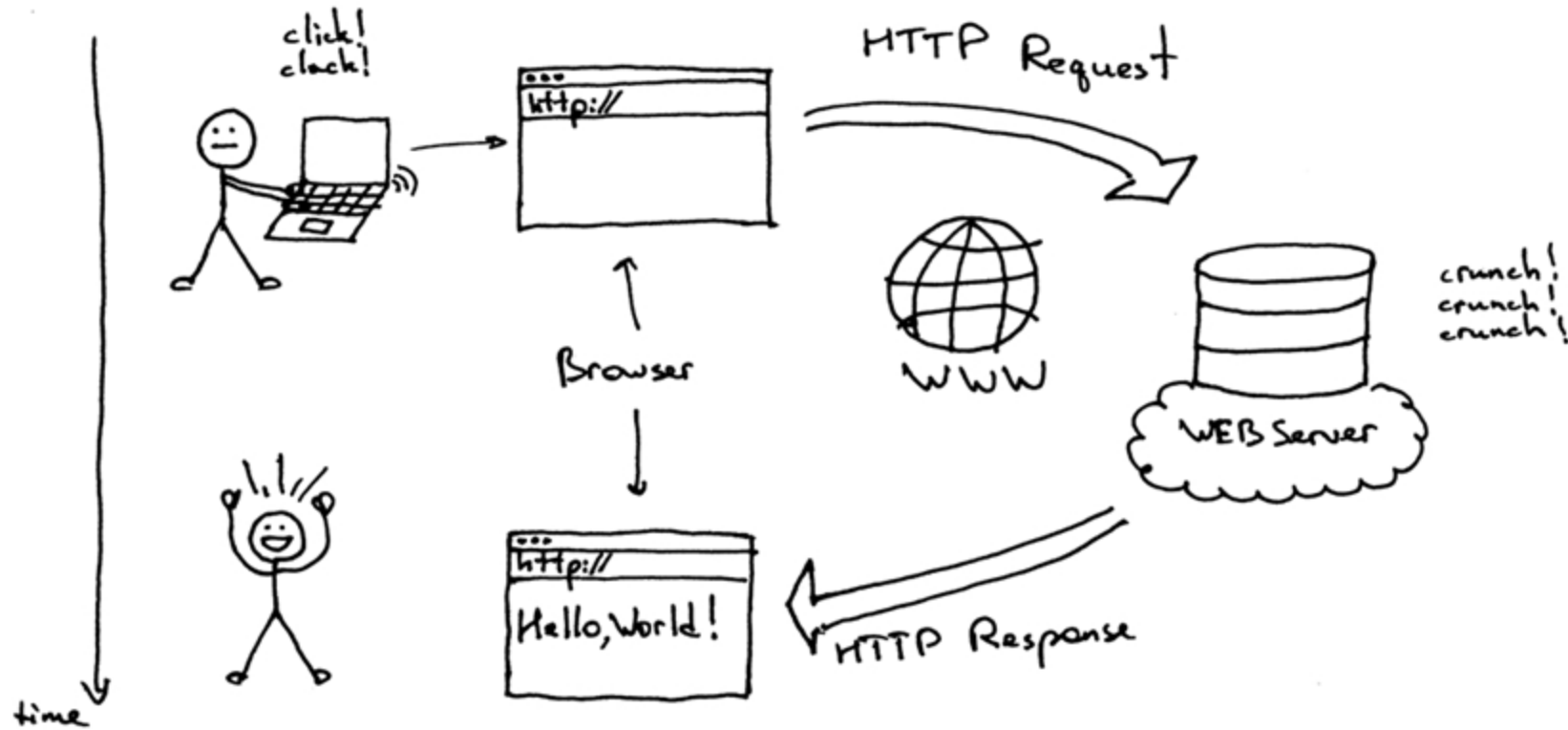

```
python sqlmap.py -u "http://localhost/sqlmap/mysql/get_1st.php?id=1" --batch
(1.3.6.40dev)
http://sqlmap.org

[!] legal disclaimer: usage of sqlmap for attacking targets without prior mutual consent is
illegal. It is the end user's responsibility to obey all applicable local, state and fed
eral laws. Developers assume no liability and are not responsible for any misuse or damage
caused by this program.

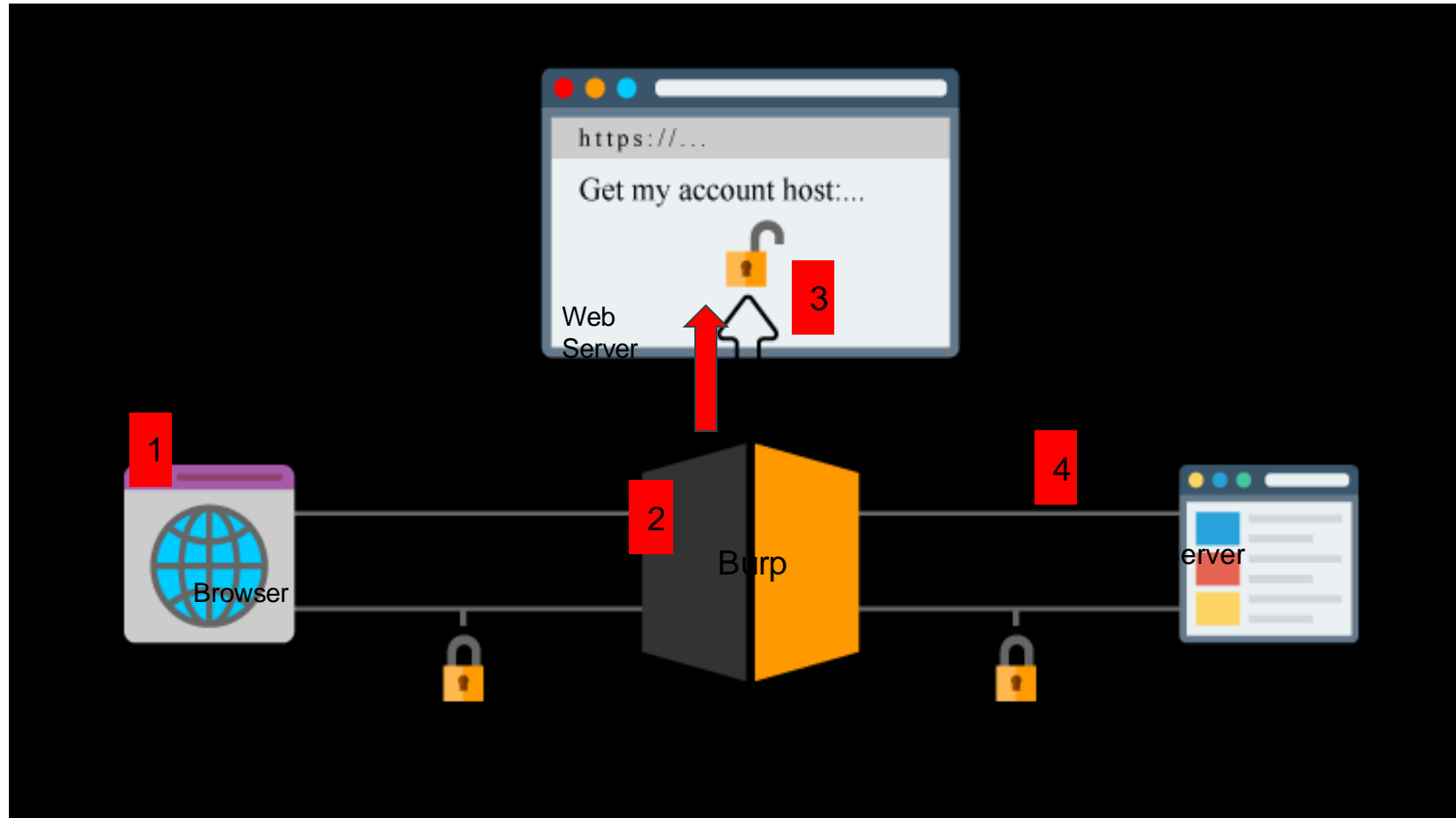
[*] starting @ 20:44:52 /2020-04-30/

[20:44:52] [INFO] testing connection to the target URL
[20:44:52] [INFO] headers/detected web page charset 'ascii'
[20:44:52] [INFO] checking if the target is protected by some kind of WAF/IPS
[20:44:52] [INFO] testing if the target URL content is stable
[20:44:52] [INFO] target URL content is stable
[20:44:52] [INFO] testing if GET parameter 'id' is dynamic
[20:44:52] [INFO] GET parameter 'id' appears to be dynamic
[20:44:52] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable
[possible RCE]: 'mysql'!
```

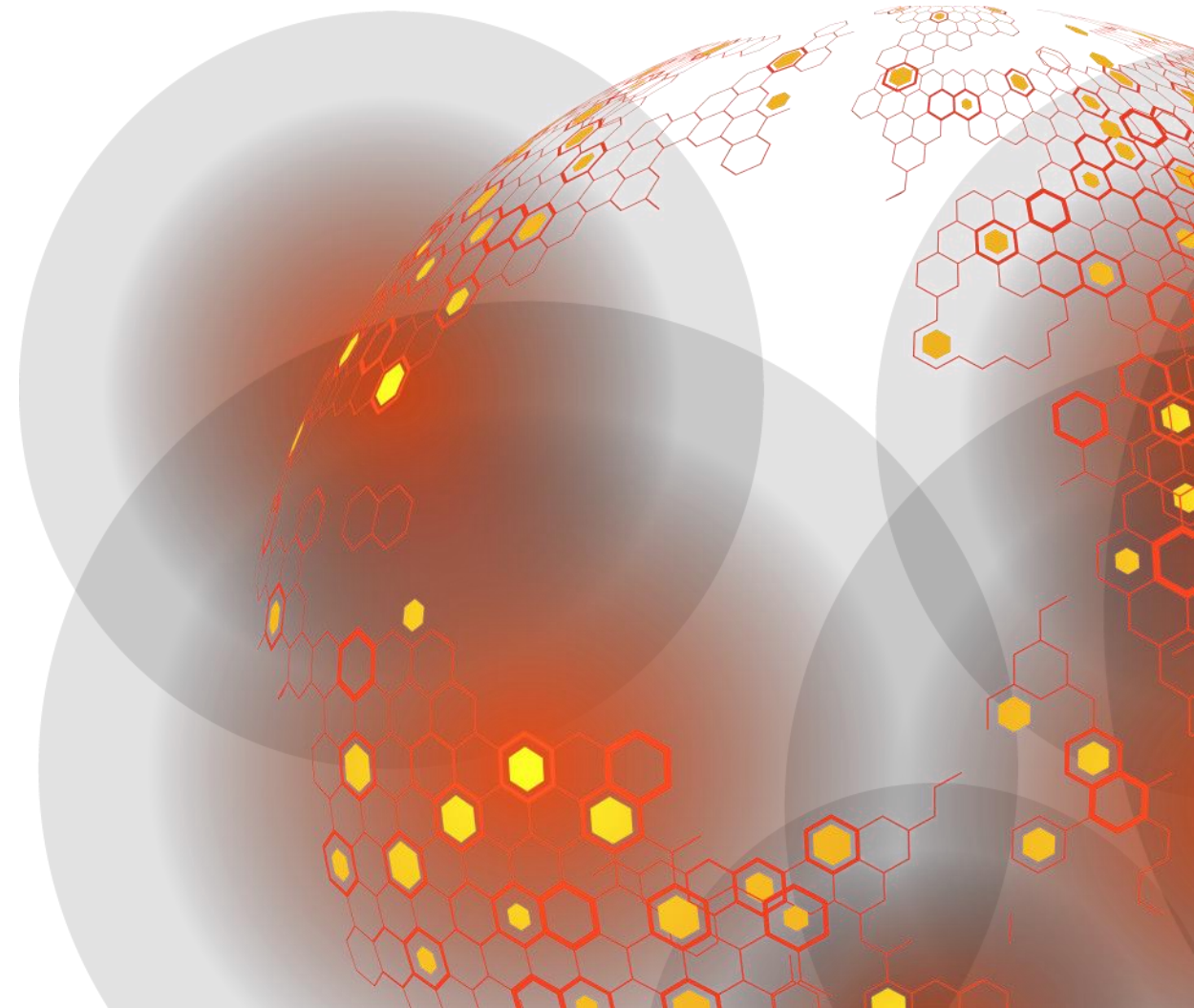


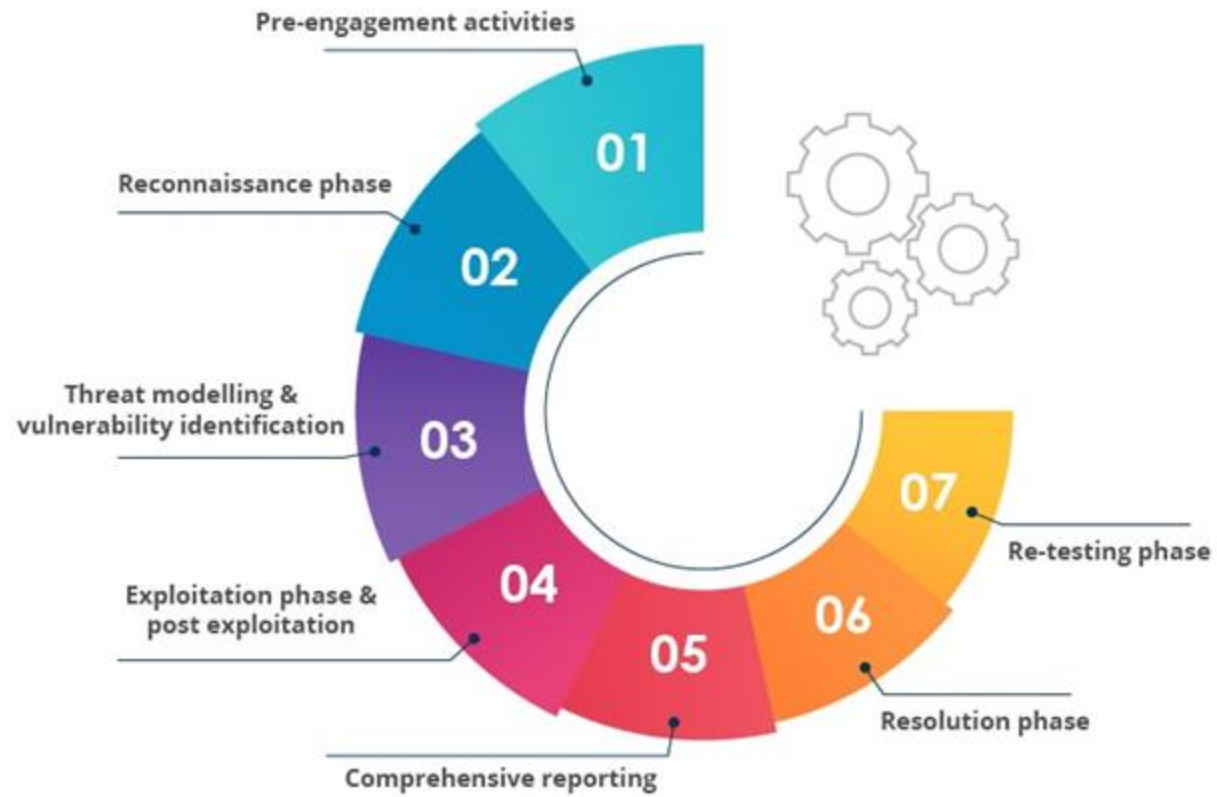


Burpsuite Application Proxy

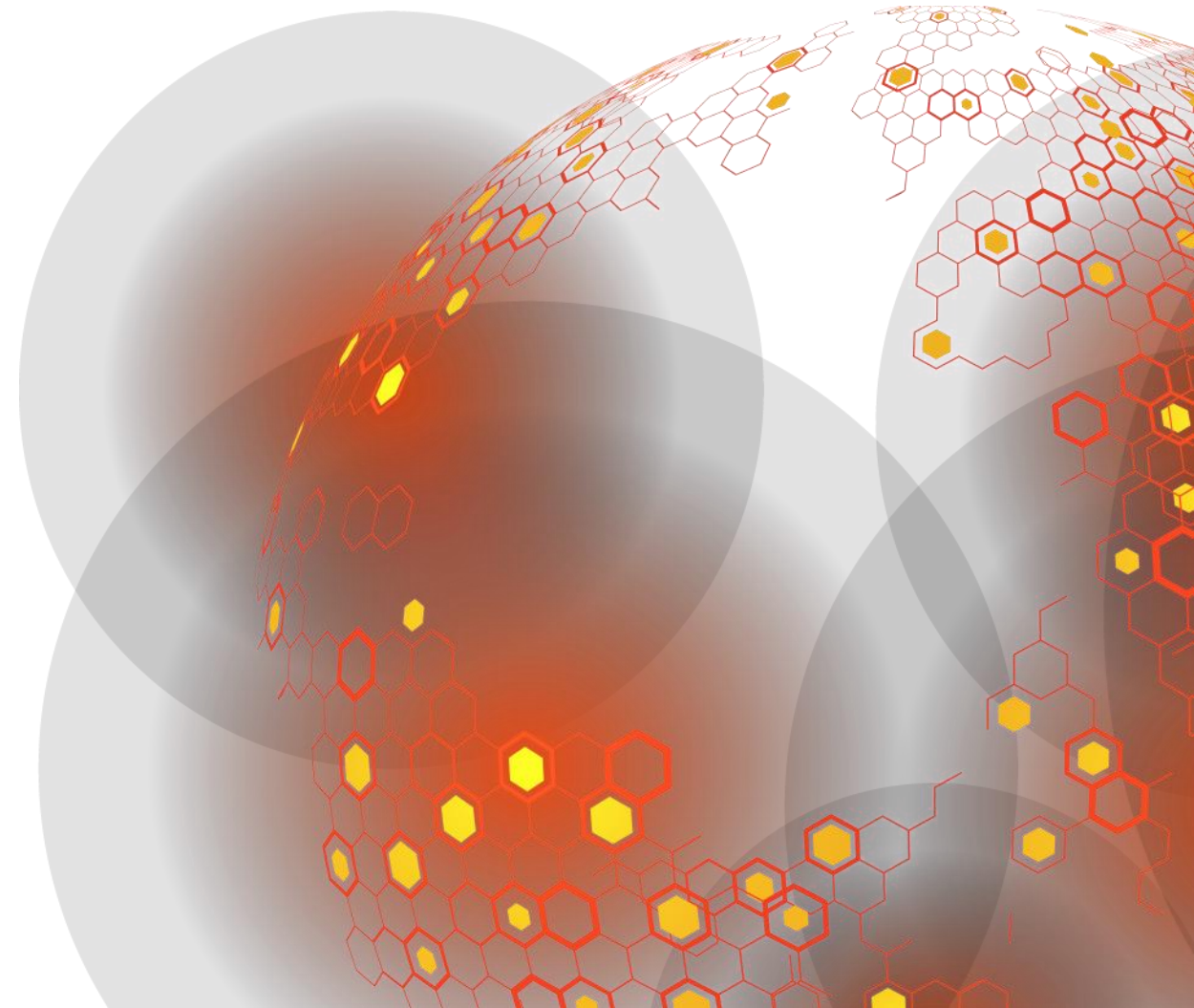


Security Testing Cycle



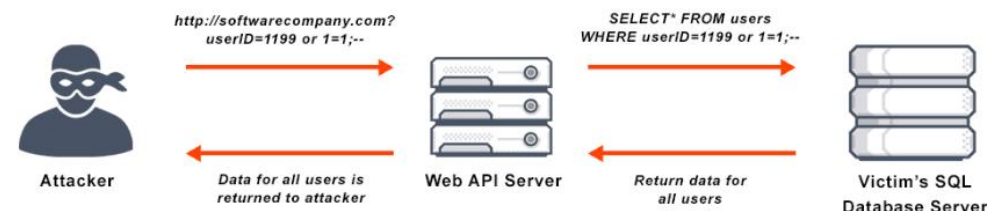
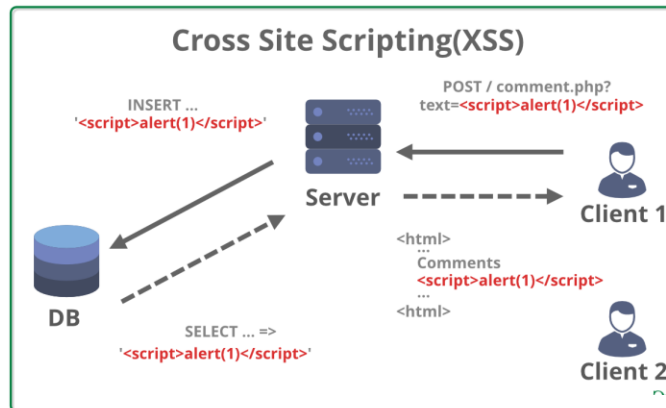
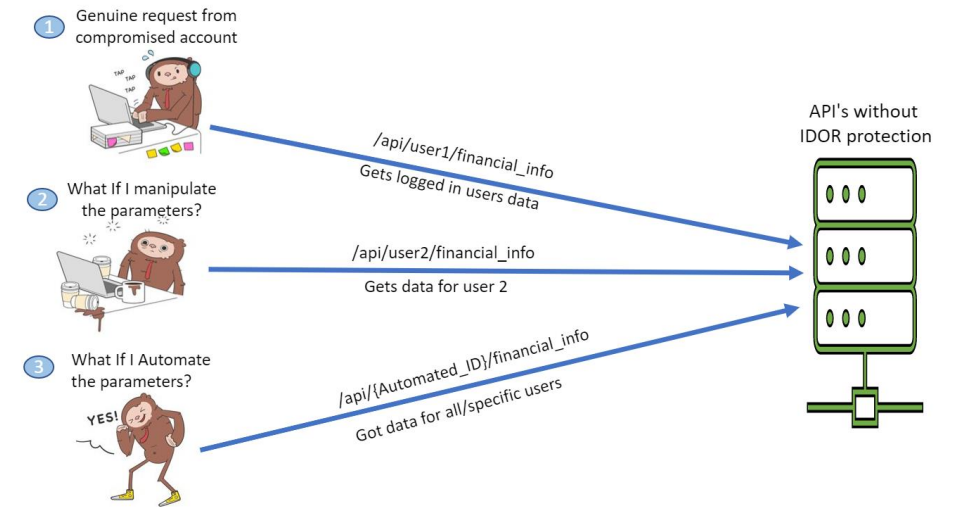


Common Vulnerabilities

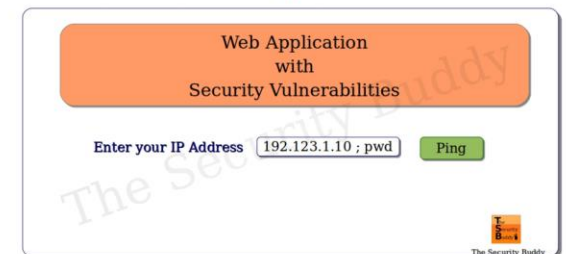


2021 OWASP Top 10

- A01 Broken Access Control
- A02 Cryptographic Failures
- A03 Injection
- A04 Insecure Design
- A05 Security Misconfiguration
- A06 Vulnerable and Outdated Components
- A07 Identification and Authentication Failures
- A08 Software and Data Integrity Failures
- A09 Security Logging and Monitoring Failures
- A10 Server-Side Request Forgery (SSRF)



Command Injection Attack



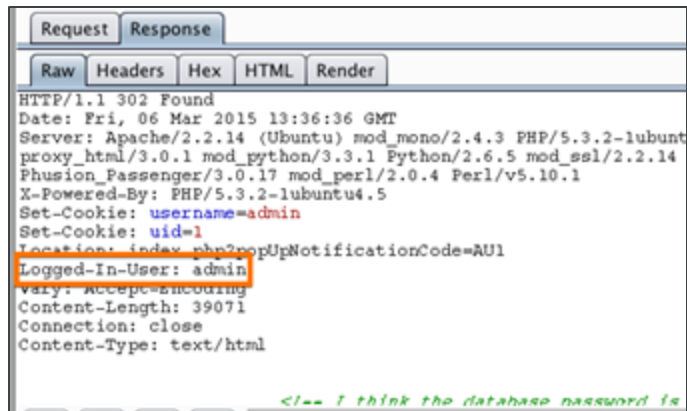
BAE SYSTEMS

Brute Force

A technique to repeatedly guess userid and password combinations until a valid response is found

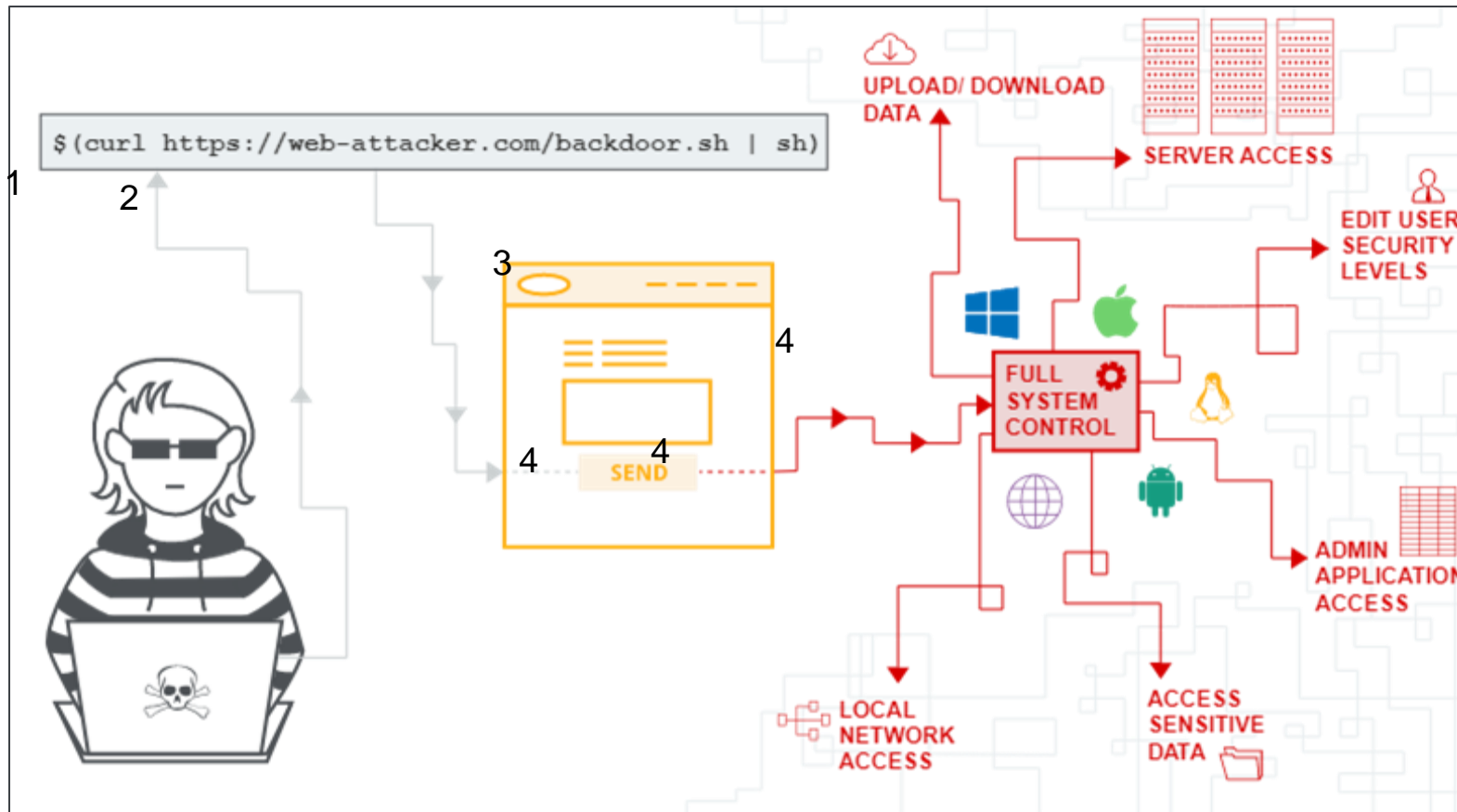


2



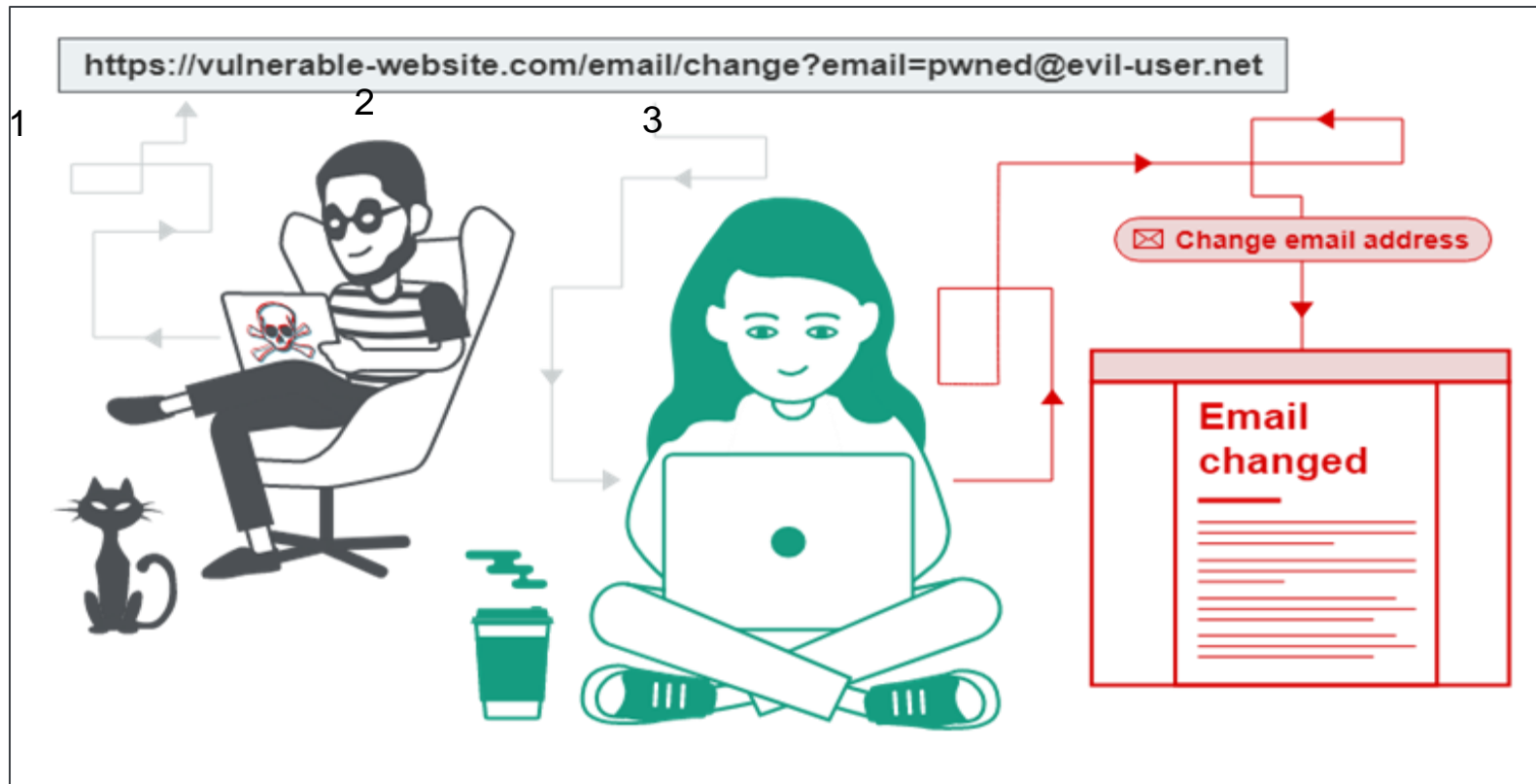
Command Injection

An attacker executes OS commands to the server via a parameter in the Web request.

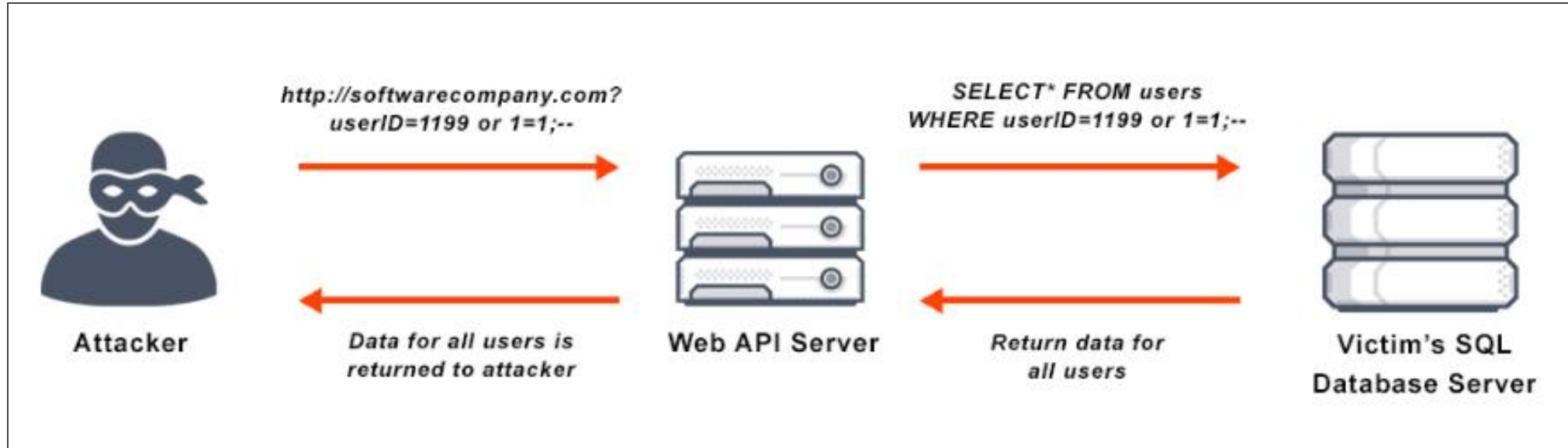


Client Side Request Forgery (CSRF)

An attacker crafts a malicious hyperlink and sends it to the victim. If the victim is currently logged into the same system and clicks on the hacker provided link, victims' account will be automatically updated

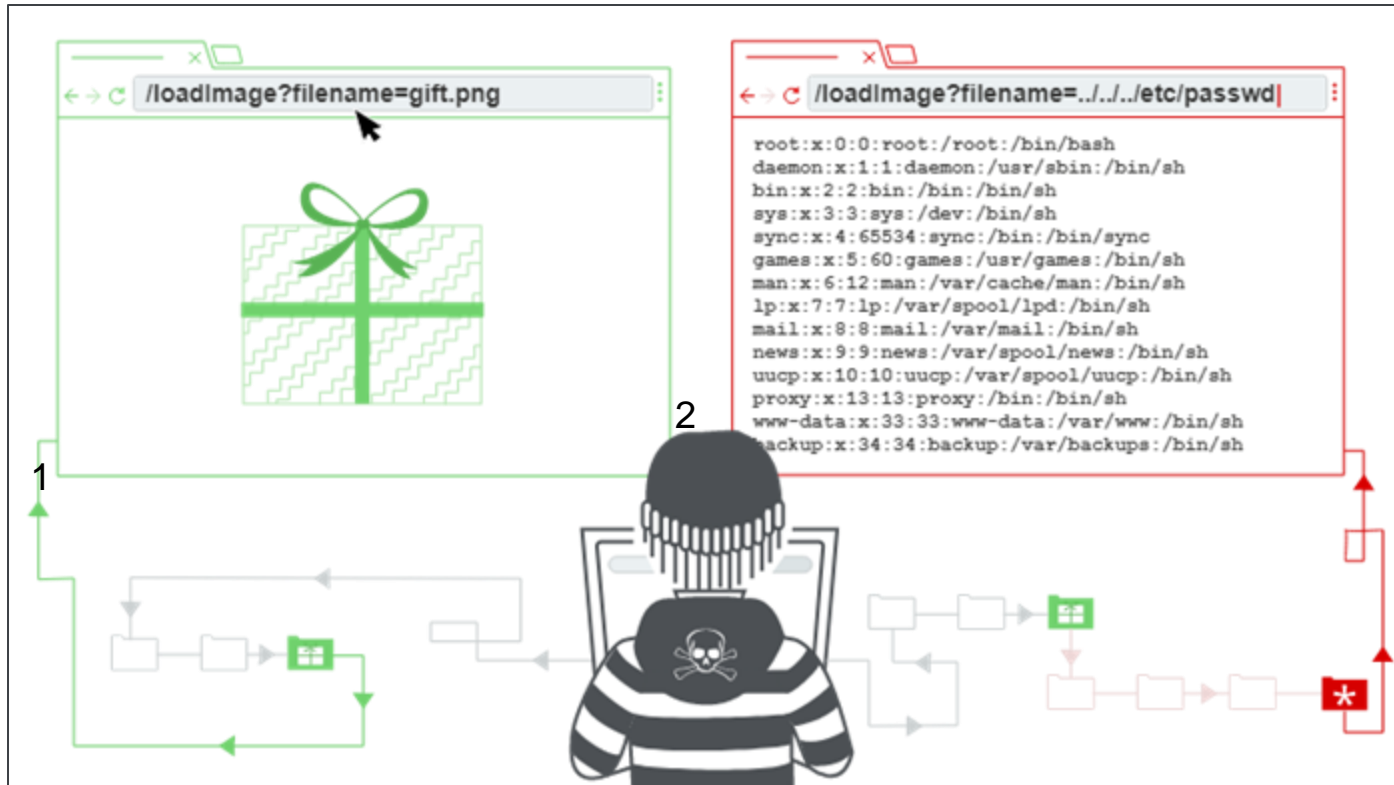


Solution: SQL Injection



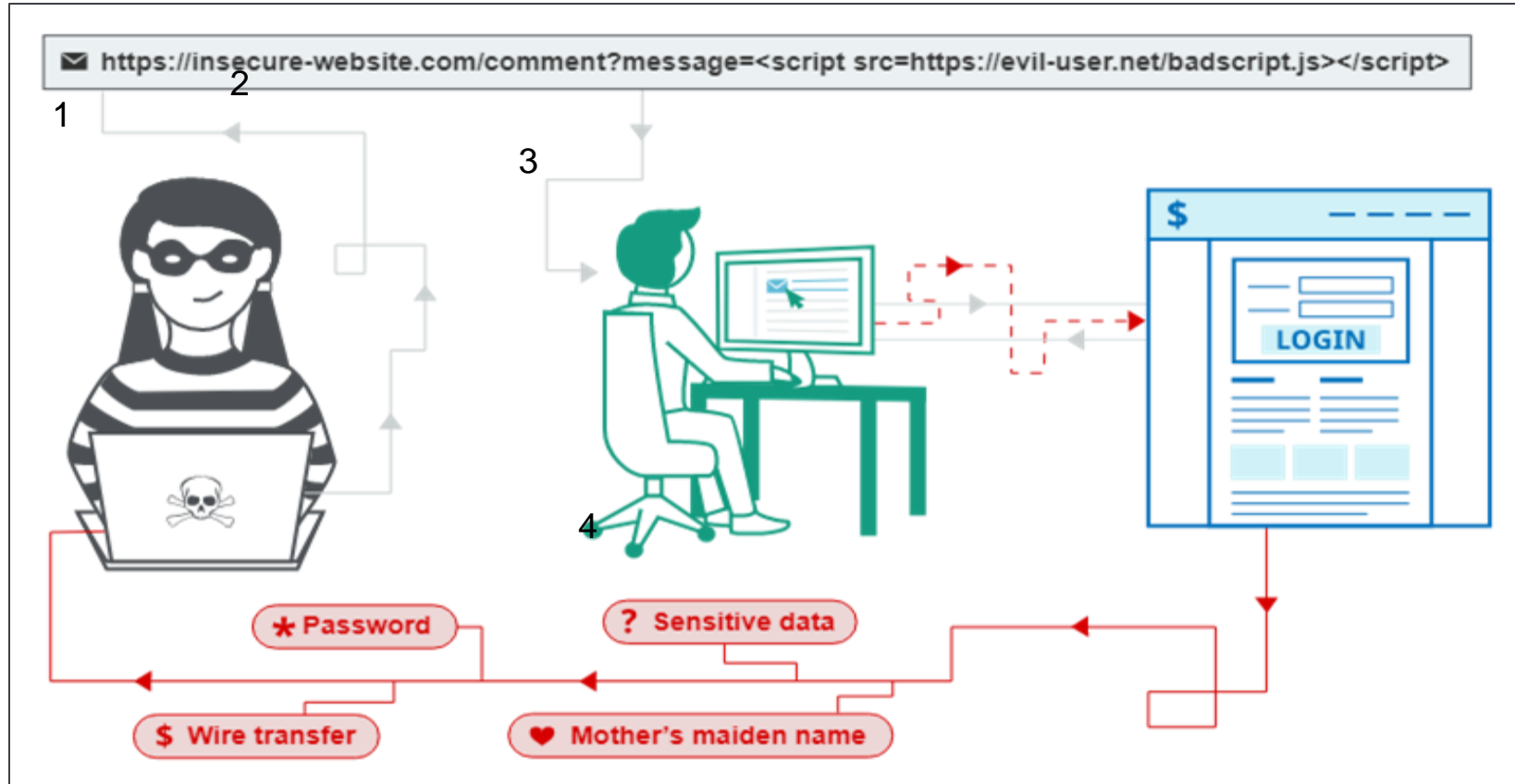
File Inclusion

Also known as dir traversal attack, allows an attacker to view OS files via submitting "../..." characters in Web requests.

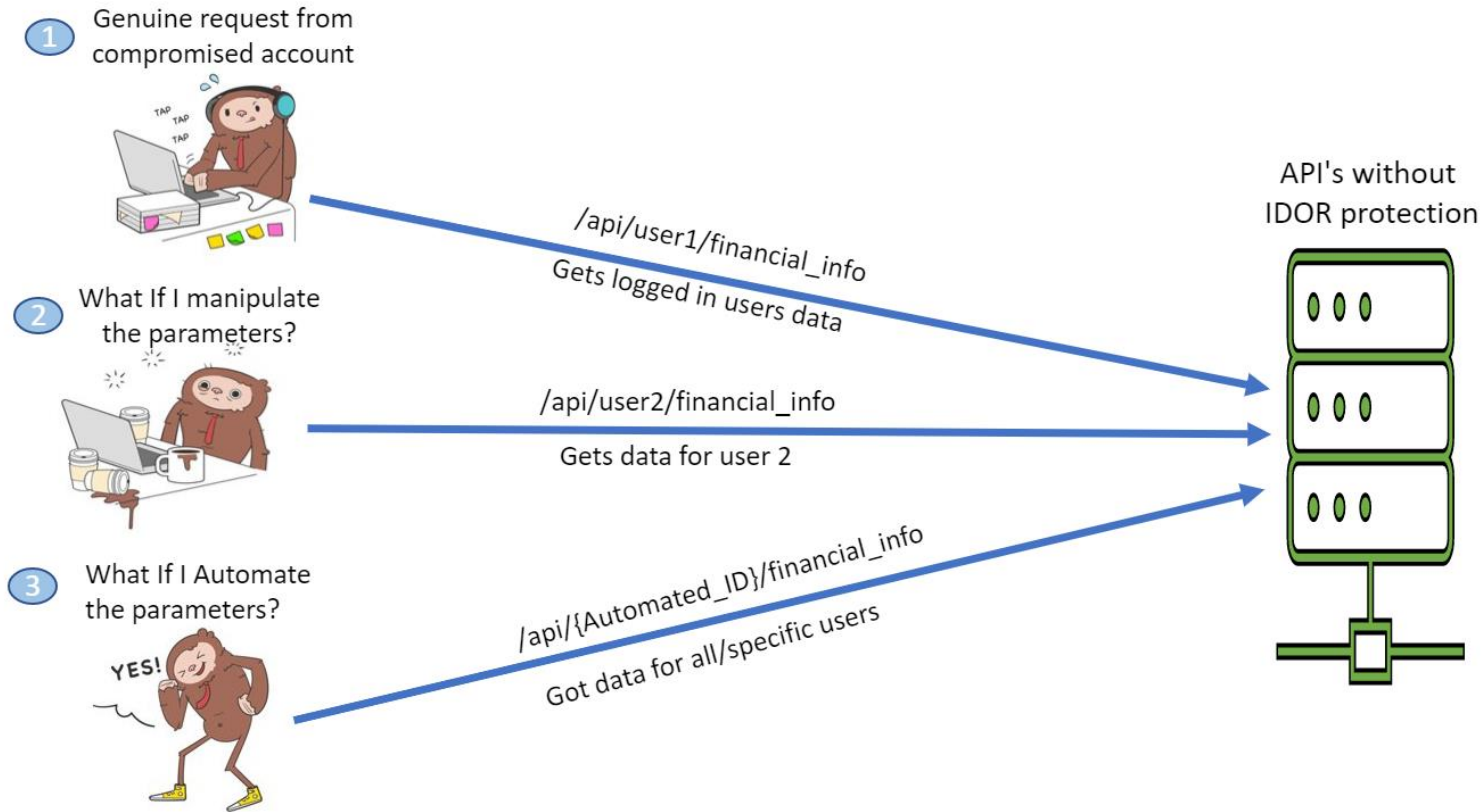


Cross Site Scripting (XSS)

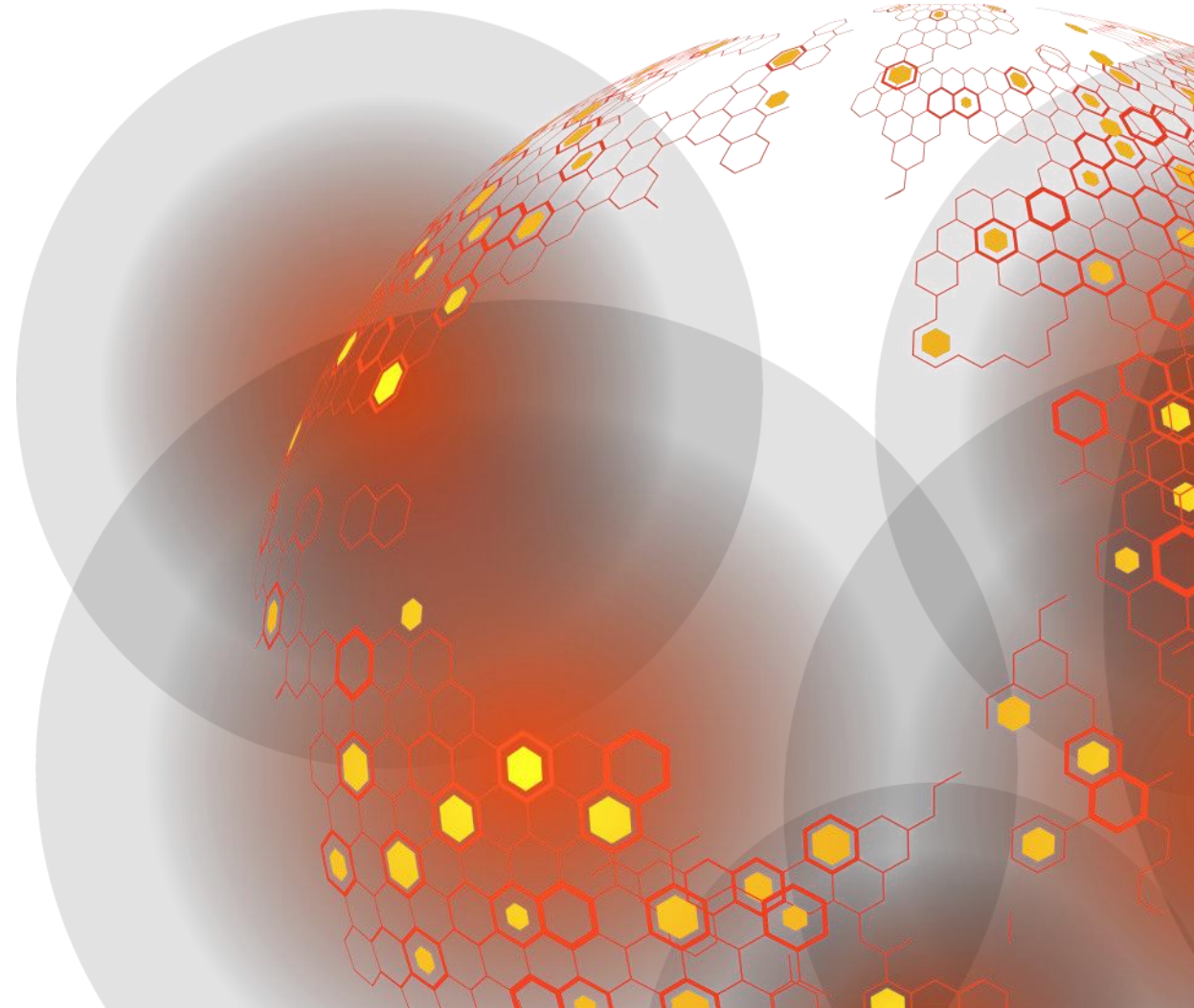
A technique used to execute code on the client/victim web browser. An attacker can use js scripts to siphon cookies, keystrokes or credentials to a attacker controlled webserver



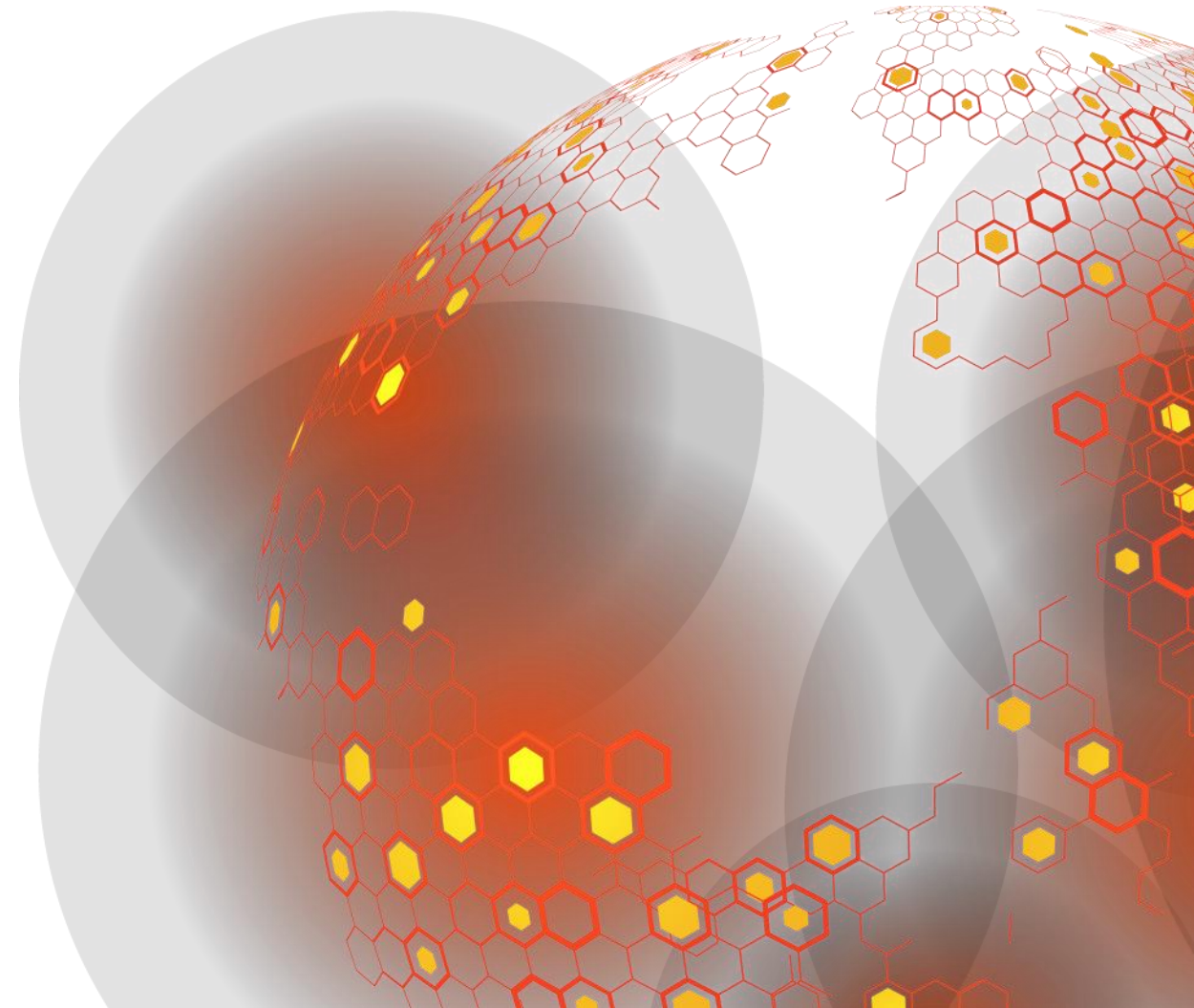
Insecure Document Object Reference (IDOR)



Intro To Burpsuite Professional



Lab Practice



Command Execution

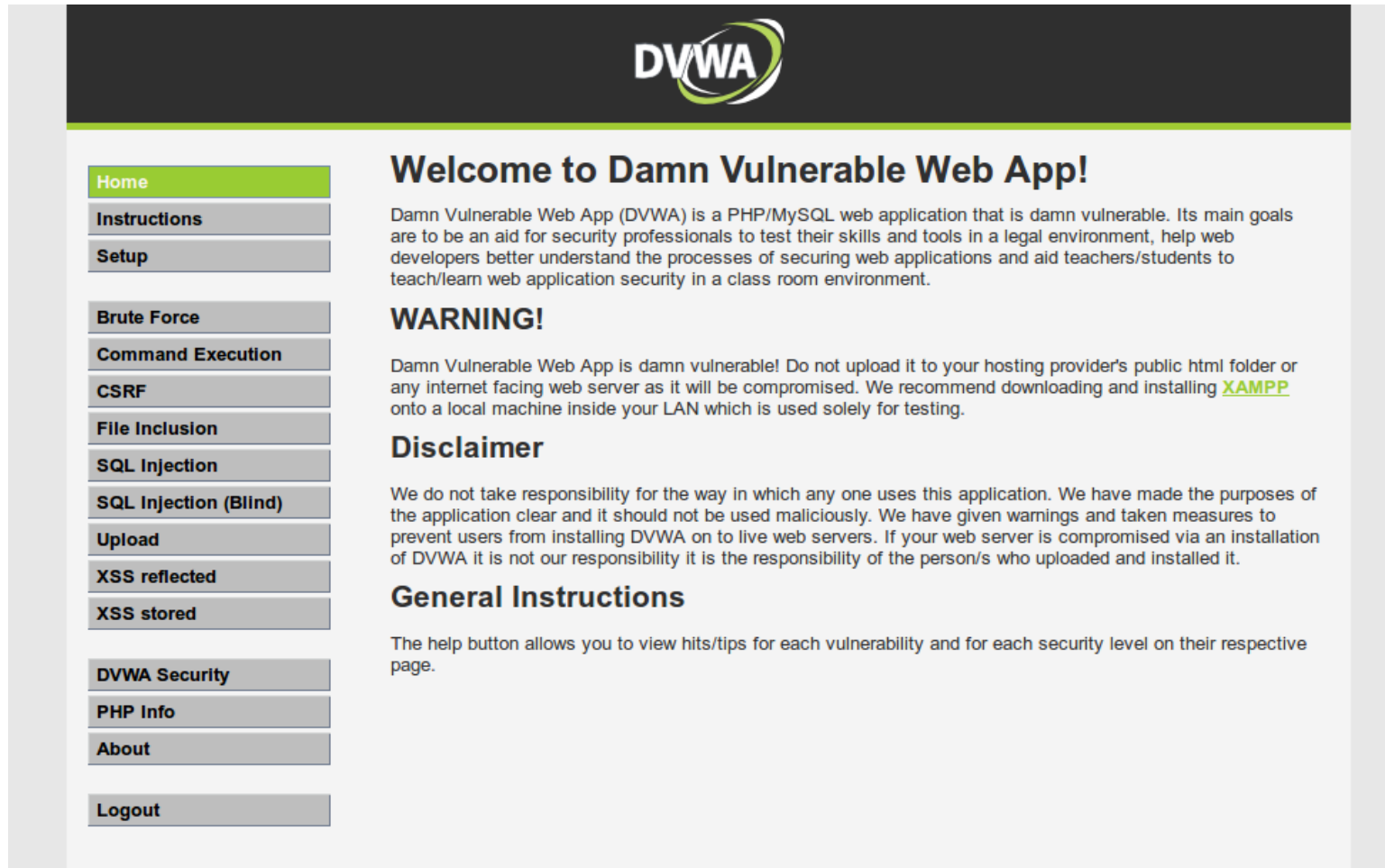
<code>exec</code>	- Returns last line of commands output
<code>passthru</code>	- Passes commands output directly to the browser
<code>system</code>	- Passes commands output directly to the browser and returns last line
<code>shell_exec</code>	- Returns commands output
<code>``` (backticks)</code>	- Same as <code>shell_exec()</code>
<code>popen</code>	- Opens read or write pipe to process of a command
<code>proc_open</code>	- Similar to <code>popen()</code> but greater degree of control
<code>pcntl_exec</code>	- Executes a program

PHP Code Execution

Apart from `eval` there are other ways to execute PHP code: `include/require` can be used for remote code execution in the form of Local File Include and Remote File Include vulnerabilities.

```
assert() - identical to eval()
preg_replace('/./e',...) - /e does an eval() on the match
create_function()
include()
include_once()
require()
require_once()
$_GET['func_name']($_GET['argument']);
$func = new ReflectionFunction($_GET['func_name']); $func->invoke(); or $func->invokeArgs(array());
```

DVWA : Damn Vulnerable Web Application



DVWA

Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

DVWA Security

PHP Info

About

Logout

Welcome to Damn Vulnerable Web App!

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and aid teachers/students to teach/learn web application security in a class room environment.

WARNING!

Damn Vulnerable Web App is damn vulnerable! Do not upload it to your hosting provider's public html folder or any internet facing web server as it will be compromised. We recommend downloading and installing [XAMPP](#) onto a local machine inside your LAN which is used solely for testing.

Disclaimer

We do not take responsibility for the way in which any one uses this application. We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.

General Instructions

The help button allows you to view hits/tips for each vulnerability and for each security level on their respective page.

Solution: Local & Remote File Inclusion

EXERCISE
30 MINS

<https://github.com/mrudnitsky/dvwa-guide-2019/blob/master/low/Challenge%204:%20File%20Inclusion.md>

Solution: Command Injection

EXERCISE
30 MINS

<https://github.com/mrudnitsky/dvwa-guide-2019/blob/master/low/Challenge%202:%20Command%20Injection.md>

Solution: File Upload to RCE (Remote Code Exec)

EXERCISE
40 MINS

<https://github.com/mrudnitsky/dvwa-guide-2019/blob/master/low/Challenge%2005:%20File%20Upload.md>

Solution: Brute Forcing

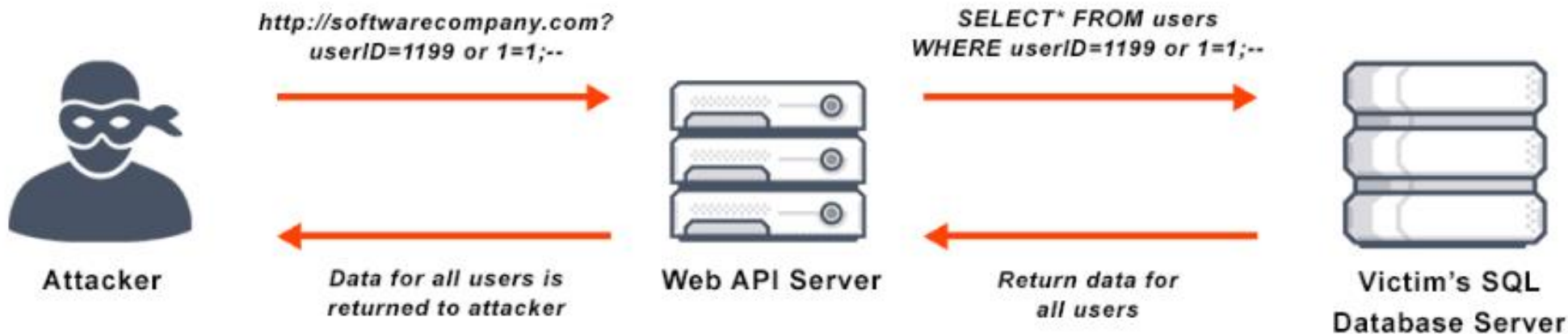
<https://github.com/mrudnitsky/dvwa-guide-2019/blob/master/low/Challenge%20I:%20Brute%20Force.md>

EXERCISE
30 MINS

Solution: SQL Injection

EXERCISE
40 MINS

<https://github.com/mrudnitsky/dvwa-guide-2019/blob/master/low/Challenge%2007:%20SQL%20Injection.md>



Solution: Cross Site Scripting (XSS)

EXERCISE
30 MINS

[https://github.com/mrudnitsky/dvwa-guide-2019/blob/master/low/Challenge%20I:%20XSS%20\(Reflected\).md](https://github.com/mrudnitsky/dvwa-guide-2019/blob/master/low/Challenge%20I:%20XSS%20(Reflected).md)

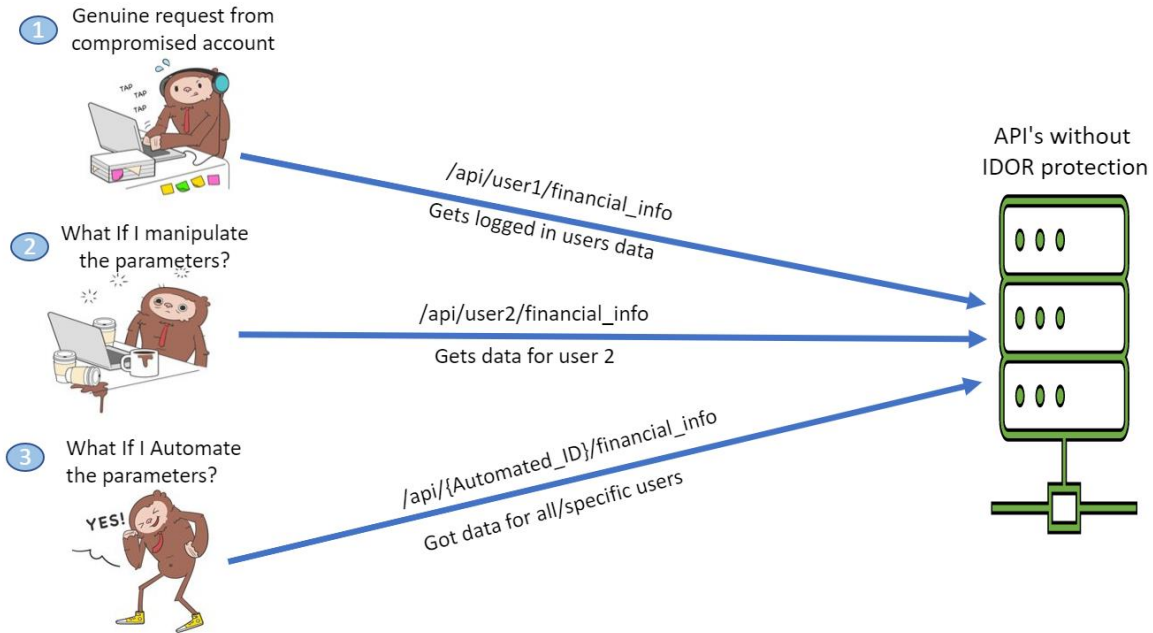
Client Side Request Forgery (CSRF)

EXERCISE
30 MINS

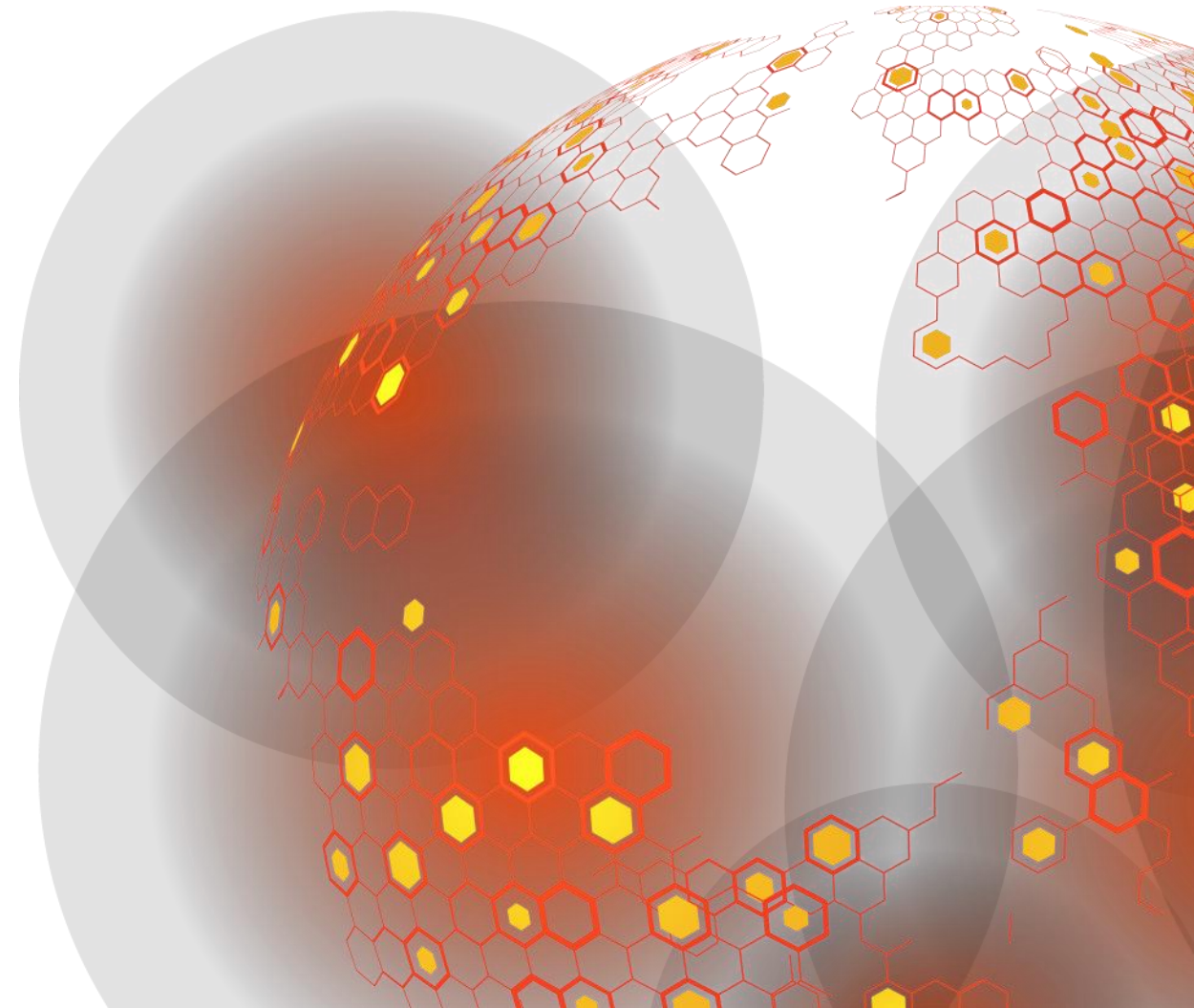
<https://github.com/mrudnitsky/dvwa-guide-2019/blob/master/low/Challenge%2003:%20CSRF.md>

Insecure Document Object Reference (IDOR)

EXERCISE
30 MINS



Summary Learnings



BAE SYSTEMS

