

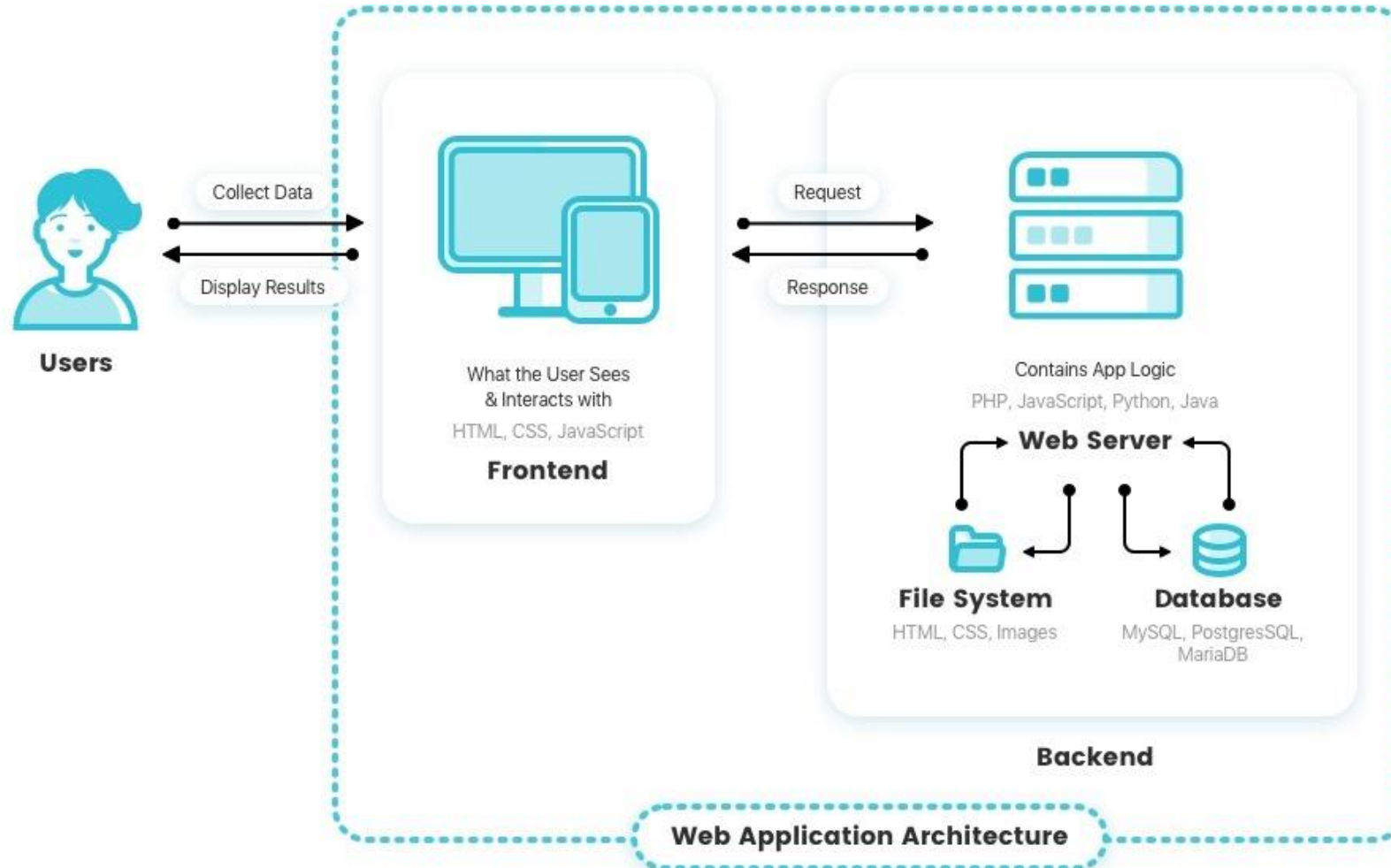
Ethical hacking and Penetration Testing

Web applications security

Identifying the goals of lesson

- Understanding basic terms
- Reviewing types of web application vulnerabilities
- Practicing with Webgoat and DVWA

Web applications structure



Choosing web browser

Function	Google Chrome	Mozilla Firefox	Edge/IE	Safari
Switching User Agents	✓	✓	✓	✓
Edit and Replay Requests	✗	✓	✗	✗
Editing Cookies	✓	✓	✓	✗
Editing Local Storage	✓	✓	✓	✗
Disable CSS	✓	✓	✓	✓
Disable Javascript	✓	✓	✗	✓
View Headers	✓	✓	✓	✓
Native screen-shot capture	✓	✓	✓	✗
Offline mode	✓	✓	✗	✗
Encode and Decode	✓	✓	✓	✓

Setting up Burp Suite

- **Burp Suite** is a proxy-based tool used to evaluate the security of web-based applications and do hands-on testing.
- Official website: <https://portswigger.net>
- Setting up:
<https://portswigger.net/burp/documentation/desktop/external-browser-config/browser-config-firefox>

Web Application Vulnerabilities

What is CIA?

- **The CIA Triad** (confidentiality, integrity, availability) is a model for information security. The three elements of the triad are considered the most crucial information security components and should be guaranteed in any secure system. Serious consequences can result if even one these elements is breached.

Confidentiality

- **Confidentiality** is "the property, that information is not made available or disclosed to unauthorized individuals, entities, or processes." In other words, confidentiality requires that unauthorized users should not be able to access sensitive resources.
- Confidentiality must be balanced with availability; authorized persons must still be able to access the resources they have been granted permissions for.

Integrity

- **Integrity** is "the property of accuracy and completeness." In other words, integrity means maintaining the consistency, accuracy and trustworthiness of data over its entire life cycle. Data must not be changed during transit and unauthorized entities should not be able to alter the data.

Availability

- **Availability** is "the property of being accessible and usable on demand by an authorized entity." In other words, authorized persons should have access to permitted resources at all times.

CVSS

- **The Common Vulnerability Scoring System (CVSS)** provides a way to capture the principal characteristics of a vulnerability and produce a numerical score reflecting its severity.
- Calculator - <https://www.first.org/cvss/>

OWASP TOP 10

- The OWASP Top 10 is a standard awareness document for developers and web application security. It represents a broad consensus about the most critical security risks to web applications.
- <https://owasp.org/www-project-top-ten/>

Vulnerability types

- **Server-Side** vulnerability – the final payload executes on servers software. Examples: Server-Side Request Forgery, SQL injection, Local File Inclusion, etc.
- **Client-Side** vulnerability – the final payload executes on clients software (for example, web browser). Examples: Cross-Site Scripting, Cross Site Request Forgery, etc.

Command injection

- **Command injection** is a cyber attack that involves executing arbitrary commands on a host operating system.
- For example, ping service uses such code:
 - `system("ping -c 1 "+ip);`
- Then attacker can request to ping such "IP":
 - `127.0.0.1;id`
- The resulting command is: `ping -c 1 127.0.0.1;id`

SQL injection

- **SQL injection** (also called SQLi) is one of the most common web hacking techniques.
- A SQL injection attack consists of insertion or "injection" of malicious code via the SQL query input from the client to the application.
- Example:
 - Query: `SELECT * FROM users WHERE id=USER_INPUT_ID;`
 - Send `USER_INPUT_ID=1`: `SELECT * FROM users WHERE id=1;`
 - Send `USER_INPUT_ID=1 OR 1=1`: `SELECT * FROM users WHERE id=1 OR 1=1;`

Path traversal

- **A path(directory) traversal** is a vulnerability where an attacker is able to access or store files and directories outside the location where the application is running. This may lead to reading files from other directories and in case of a file upload overwriting critical system files.
- Example:
 - Normal URL: `http://example.com/file=report.pdf`
 - Attacked URL: `http://example.com/file=../../../../etc/passwd`

Bruteforce

- **A brute force attack** uses trial-and-error to guess login info, encryption keys, or find a hidden web page, etc.

JWT token attacks

- Many application use JSON Web Tokens (JWT) to allow the client to indicate its identity for further exchange after authentication.
- Format: base64_data.base64_data.base64_data

The diagram illustrates the structure of a JWT token, which is a string composed of three parts separated by dots. The first part is the Header, the second is the Claims, and the third is the Signature. Each part is color-coded and labeled with an arrow pointing to it.

Header (orange text): `eyJhbGciOiJIUzI1NiJ9.`


Claims (green text): `eyJleHAiOjE0MTY0NzE5MzQsInVzZXJfbmFtZSI6InVzZXIiLCJzY29wZSI6WyJyZWZkIiwid3JpdGUiXSwiYXV0aG9yaXRpZXMiOlsiuk9MRV9BRE1JTtIsIlJPTEVfVVFUijJdLCJqdGkiOiI5YmM5MmE0NC0wYjFhLTRjNWUtYmU3MC1kYTUyMDC1YjIhODQiLCJjbGllbnRfawQioiJteS1jbGllbnQtd210ac1zZWNYZXQifQ.`

Signature (blue text): `qxNjYSPIKSURZEMqLQQPw1Zdk6Le2FdGHRYZG7SQnNk`

XML External Entity (XXE)

- **An XML External Entity attack** is a type of attack against an application that parses XML input.
- This attack occurs when XML input containing a reference to an external entity is processed by a weakly configured XML parser.
- This attack may lead to the disclosure of confidential data, denial of service, server side request forgery, port scanning from the perspective of the machine where the parser is located, and other system impacts.

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE author [
  <!ENTITY showme SYSTEM "file:///etc/passwd">
]>
<author>&showme</author>
```



```
graph LR
    A["<!ENTITY showme SYSTEM \"file:///etc/passwd\">"] --> B["cat /etc/passwd"]
    B --> C["root:x:0:0:root:/root:/bin/bash  
daemon:x:1:1:daemon:/usr/sbin:/bin/sh  
bin:x:2:2:bin:/bin:/bin/sh  
sys:x:3:3:sys:/dev:/bin/sh  
sync:x:4:65534:sync:/bin:/bin/sync  
games:x:5:60:games:/usr/games:/bin/sh  
man:x:6:12:man:/var/cache/man:/bin/sh  
lp:x:7:7:lp:/var/spool/lpd:/bin/sh  
mail:x:8:8:mail:/var/mail:/bin/sh"]
    C --> D["root:x:0:0:root...  
daemon:x:1:1:daemon...  
..."]
```

```
<author>
  root:x:0:0:root...
  daemon:x:1:1:daemon...
  ...
</author>
```

```
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
```

Cross-Site Scripting (XSS)

- **Cross-Site Scripting (XSS)** is a vulnerability/ flaw that combines the allowance of html/script tags as input that are rendered into a browser without encoding or sanitization
- Payload example: "><script>alert(document.cookie)</script>

Session fixation attack

- **Session fixation attack** is attempt to exploit the vulnerability of a system that allows one person to fixate (find or set) another person's session identifier.
- Imagine that someone text you:
 - “Hey, look at this discounts at book shop: <http://book-shop.local/?session=a135e1cd36f6a4>”
- If this book is vulnerable, it will set this sessions as yours. After your registration, attacker who sent you link can use your account, while session is alive.

A little bit about defense

Web application firewall

- **A web application firewall (WAF)** is a specific form of application firewall that filters, monitors, and blocks HTTP traffic to and from a web service.
- <https://geekflare.com/open-source-web-application-firewall/>
- <https://serverguy.com/security/open-source-web-application-firewall/>

Backdoor scanners

- Webshell analyzer
 - ImunifyAV
 - Ai-bolit
 - Any antivirus
-
- Web shells - <https://github.com/tennc/webshell>

Yara rules

- YARA is a tool aimed at (but not limited to) helping malware researchers to identify and classify malware samples.
 - Yara - <http://virustotal.github.io/yara/>
 - Signatures - <https://github.com/Neo23x0/signature-base>
- * In practice, Yara helped to find Chinese encrypted Cobalt Strike payload.

Any questions?

Homework for the next lesson

- Complete manually (without using automatic tools) SQL injection task in DVWA (use **low** security level). Final payload should get the version of MySQL.
- Complete manually (without using automatic tools) XSS Reflected task in DVWA (use **low** security level). Final payload should get the value of document.cookie param.
- Complete manually (without using automatic tools) Command Execution task in DVWA (use **medium** security level). Final payload should execute whoami command.

Feedback: did we achieved the goals of our lesson?

- Discussion 5-10 minutes

Useful links

- <https://reinvently.com/blog/fundamentals-web-application-architecture/>
- [https://owasp.org/www-project-web-security-testing-guide/stable/4-Web Application Security Testing/11-Client-side Testing/README](https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/11-Client-side_Testing/README)
- <https://pwning.owasp-juice.shop/part1/rules.html>
- <https://geekflare.com/open-source-web-application-firewall/>
- <https://serverguy.com/security/open-source-web-application-firewall/>
- <https://www.imperva.com/learn/application-security/command-injection/>