**Second month tasks**

**TryHackMe Labs – OWASP Top 10**

**Lab-1: Injection**

Exploited an SQL injection vulnerability by manipulating query parameters to access unauthorized data.

Mitigation techniques: Using parameterized queries, ORM Libraries and imput validation.

**Lab-2: Broken Authentication:=**

Identified weak authentication mechanisms and exploited them to gain unauthorized access.

Mitigation techniques: Use MFA, Strong password policies, regular password changes and secure session handling.

**Lab-3: Sensitive Data Exposure**

Discovered sensitive information stored without encryption and demonstrated how attackers could extract and misuse this data.

Mitigation techniques: Use strong encryption, secure transport protocols (e.g., HTTPS), close unused ports and avoid storing sensitive data unnecessarily.

**Lab-4: XML External Entities (XXE)**

Exploited XXE by injecting external entities and accessing restricted data.

Mitigation techniques: Disable XML external entity processing, use libraries that do not support XXE, and sanitize XML inputs.

**Lab 5:  Broken Access Control**

Manipulated URL parameters and request data to access restricted functionality.

Mitigation techniques: Enforce least privilege policy, implemet access control on the server side, and avoid client-side security checks.

**Lab-6: Security Misconfiguration**

Identified misconfigured security headers and default credentials that allowed unauthorized access.

Mitigation techniques: Regularly update software, implement secure configurations, and enforce security best practices.

**Lab-7: Cross-Site Scripting (XSS)**

Injected malicious JavaScript to demonstrate how XSS can compromise user sessions.

Mitigation techniques: Encode output data, use CSP (Content Security Policy), and validate user input.

**Lab-8: Insecure Deserialization**

Insecure deserialization can lead to remote code execution or elevation of privileges when untrusted data is deserialized.

Mitigation techniques: Avoid deserialization of untrusted data, enforce strict input validation, and use digital signatures to verify serialized objects.

**Lab-9: Using Components with Known Vulnerabilities**

Discovered and exploited outdated libraries within the application to compromise system security.

Mitigation techniques: Regularly update software dependencies and use vulnerability management tools to identify and remediate outdated components.

**Lab-10: Insufficient Logging and Monitoring**

Explored the lack of effective logging, demonstrating how attackers could act without alerting administrators.

Mitigation techniques: Implement centralized logging, monitor access control failures, and establish incident response protocols.

**By completing this learning path, I have learned much about OWASP Top 10 Vulnerabilities.**

**PicoCTF Challenges**

In this task I have completed the following PicoCTF challenges from different topics such as Forensics, General skills, web exploitation and Cryptogtaphy.

Verify- forensics

Scan surprise- forensics

Binary search -general skills

Web decode -web exploitation

Unminify -web exploitation

Super ssh- general skills

Secret of the polyglot- forensics

Intro to burp -web exploitation

Interencdec -cryptography

Endianness -general skills

CanYouSee -forensics

Bookmarklet -web exploitation

Blame game -general skills

Binhexa -general skills

Repetitions -general skills

Big zip -general skills

First find -General skills

Local authority -web exploitation

Inspect HTML -web exploitation

Includes web -exploitation

Runme.py -general skills

PW Crack 1- general skills

PW Crack 2 -general skills

HashingJobApp -general skills

Fixme1.py -general skills

Fixme2.py- general skills

Convertme.py -general skills

Codebook -general skills

Cookies web -exploitation

Wave a flag -general skills

Static ain't always noise -general skills

Scavenger hunt -web exploitation

Obedient cat -general skills

Mod 26 -Cryptography

GET aHead -web exploitation

2warm -general skills

Don't-use-client-side -web exploitation

Logon web -exploitation

Insp3ct0r -web exploitation

Where are the robots -web exploitation

Findme -web exploitation

After doing these challenges I have gained an immense amount of knowledge about Hashing, developer options in browsers, cookies, burpsuite and many other topics.

**Bug Bounty**

For the bug bounty task, I am testing a website from BugCrowd’s bug bounty program.

Target website:

[www.upwork.com](https://www.upwork.com/)

I have read and understood the scope of the program and possible legal implications if I go out of scope.

Scope: [[www.upwork.com](https://www.upwork.com/)]

**Information gathering:**

Tools Used:

1. DNS Enumeration: Tools like dnsmap, Amass, Sublist3r, or online services (e.g., SecurityTrails).

2. IP and Hosting Information: Tools like dig, nslookup, or online services (e.g., Shodan, Censys)

3. Website Fingerprinting: Tools like Wappalyzer or WhatWeb.

4. Vulnerability Scanners: Tools like Burp Suite, OWASP ZAP, Nikto (only passive scans within scope).

Findings:

dnsmap 0.36 - DNS Network Mapper

[+] searching (sub)domains for upwork.com using built-in wordlist

[+] using maximum random delay of 10 millisecond(s) between requests

beta.upwork.com

IP address #1: 104.18.90.237

IP address #2: 104.18.89.237

blog.upwork.com

IP address #1: 104.18.89.237

IP address #2: 104.18.90.237

developers.upwork.com

IP address #1: 104.18.89.237

IP address #2: 104.18.90.237

events.upwork.com

IP address #1: 18.239.142.40

IP address #2: 18.239.142.128

IP address #3: 18.239.142.24

IP address #4: 18.239.142.127

go.upwork.com

IP address #1: 104.17.72.206

IP address #2: 104.17.74.206

IP address #3: 104.17.70.206

IP address #4: 104.17.73.206

IP address #5: 104.17.71.206

hp.upwork.com

IP address #1: 104.18.89.237

IP address #2: 104.18.90.237

jj.upwork.com

IP address #1: 104.18.89.237

IP address #2: 104.18.90.237

kb.upwork.com

IP address #1: 104.18.89.237

IP address #2: 104.18.90.237

ox.upwork.com

IP address #1: 104.18.89.237

IP address #2: 104.18.90.237

research.upwork.com

IP address #1: 104.18.90.237

IP address #2: 104.18.89.237

rh.upwork.com

IP address #1: 104.18.89.237

IP address #2: 104.18.90.237

secure.upwork.com

IP address #1: 104.18.90.237

IP address #2: 104.18.89.237

shop.upwork.com

IPv6 address #1: 2606:4700:90d2:c0d9:6d42:56e:5a46:e3f9

shop.upwork.com

IP address #1: 104.18.1.116

IP address #2: 104.18.0.116

su.upwork.com

IP address #1: 104.18.89.237

IP address #2: 104.18.90.237

support.upwork.com

IP address #1: 216.198.53.1

IP address #2: 216.198.54.1

te.upwork.com

IP address #1: 104.18.90.237

IP address #2: 104.18.89.237

tl.upwork.com

IP address #1: 104.18.89.237

IP address #2: 104.18.90.237

www.upwork.com

IP address #1: 104.18.90.237

IP address #2: 104.18.89.237

[+] 19 (sub)domains and 42 IP address(es) found

[+] completion time: 213 second(s)

**Vulnerability Assessment:**

Potential Bugs Tested

1. Cross-Site Scripting (XSS):

Tested input fields and reflected output points.

Tools: Manual payload injection and Burp Suite Repeater.

2. SQL Injection:

Tested login forms and parameters

Tools: Manual payload testing with SQLMap for verification.

3. Subdomain Takeover:

Identified unclaimed CNAME records.

Verified claims on services like GitHub Pages, AWS S3, or Heroku.

4. Open Redirects:

Explored URL parameters for manipulation.

Tools: Manual testing with encoded/decoded payloads.

As a beginner I was not able to find any bugs but this experience helped me learn many things about different types of scans and vulnerabilities.