BUG

Sri Lanka Institute of Information Technology

Assignment 1- Year 2 semester 2

My Bug Bounty Journal

Student Name – Wanasinghe N.K

Student ID – IT23221000

IE2062 - Web Security

B.Sc. (Hons) in information Technology Specializing in Cyber Security

Table of contents

Introduction to my Bug Bounty Journal

With the increasing digital evolution and massive rise in need for cybersecurity, bug bounty hunting has become a key component for ethical hacking, or rather, white hat hacking to help protect systems. Under bug bounty programs, a company will reward anyone who finds and reports security vulnerabilities within their systems. This journal documents my adventure of learning through bugs, documenting the challenges, learning moments, and progress I make along the way.

This journal will be a record of my growth as a bug bounty hunter, through all the tools, techniques, and platforms I learn along the way. It will help me track my progress, learn from my mistakes, and grow as a cybersecurity professional. I hope my experience will offer useful information and inspiration for future bug bounty hunters.

Acknowledgement

First, I would like to extend my deepest gratitude to all the people who supported me to successfully complete this assignment. I would like to thank my esteemed mentor, Ms. Menaka Moonamaldeniya, for her unwavering guidance and insightful supervision throughout the course of this module. I would also owe thanks to my lab instructor Ms. Ruwani Dayananda for her guidance throughout the practical labs of this module. The knowledge from the labs has helped me to do my work successfully

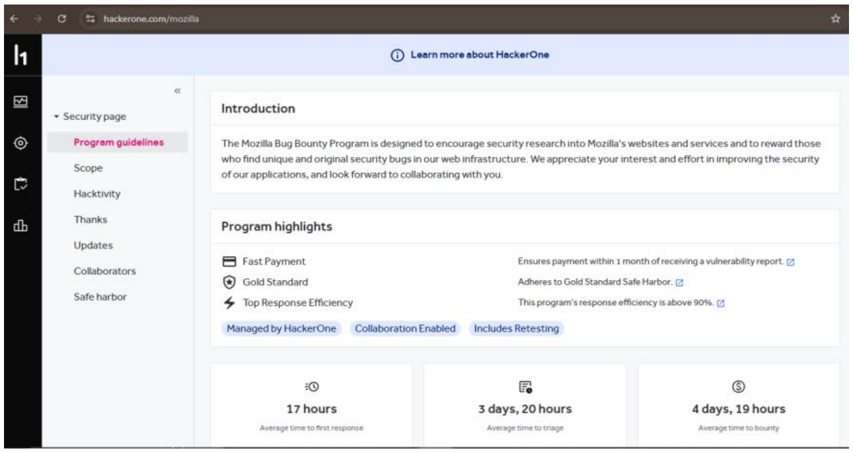
Moreover, I would like to extend my gratitude to all my friends and fellow bug bounty hunters for their shared experiences and discussions, and to all the cybersecurity communities for sharing resources and knowledge that helped me to learn more. Special thanks to hackerone, intigity and bugcrowd for offering well-structured bug bounty program platforms that allow me to dive into bug bounty hunting and enhance my skills.

What is Bug Bounty Hunting?

A Bug bounty program is an excellent crowd-sourcing program through which organizations can offer rewards or bounties to ethical hackers or researchers in order to find and discover flaws within their own machinery, software, and websites. The primary reason organizations would do this is to find bugs or flaws before the hackers exploit them, thus creating security risks in their products.

Important Features of Bug Bounty Programs

Program guidelines - Program guidelines explain the terms of participation, including what is and isn't allowed, and the ethical standards that researchers need to maintain and follow while hunting for vulnerabilities.



Introduction

The Mozilla Bug Bounty Program is designed to encourage security research into Mozilla's websites and services and to reward those who find unique and original security bugs in our web infrastructure. We appreciate your interest and effort in improving the security of our applications, and look forward to collaborating with you.

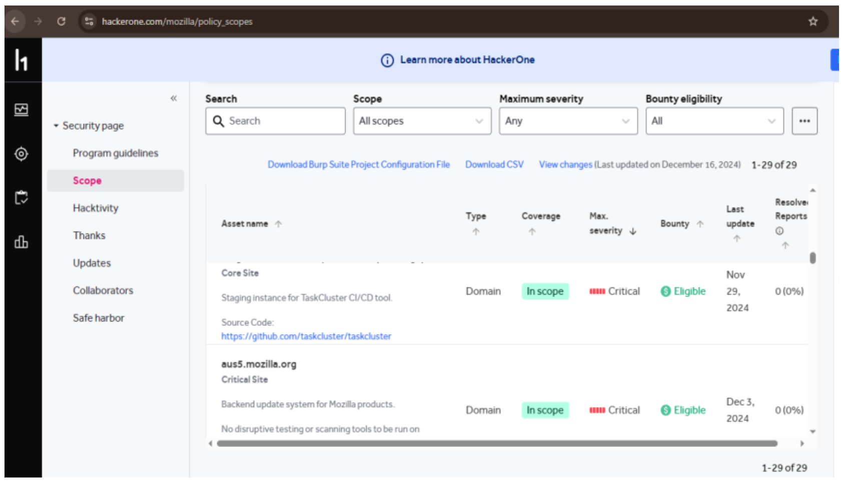
Program highlights

Fast Payment	Ensures payment within 1 month of receiving a vulnerability report.
Gold Standard	Adheres to Gold Standard Safe Harbor.
Top Response Efficiency	This program's response efficiency is above 90%.

Managed by HackerOne Collaboration Enabled Includes Retesting

17 hours	3 days, 20 hours	4 days, 19 hours
Average time to first response	Average time to triage	Average time to bounty

Scope – This defines eligible systems, applications, or services for testing. It helps ensure that researchers do not wander into unauthorized areas and are concentrating on only authorized targets.



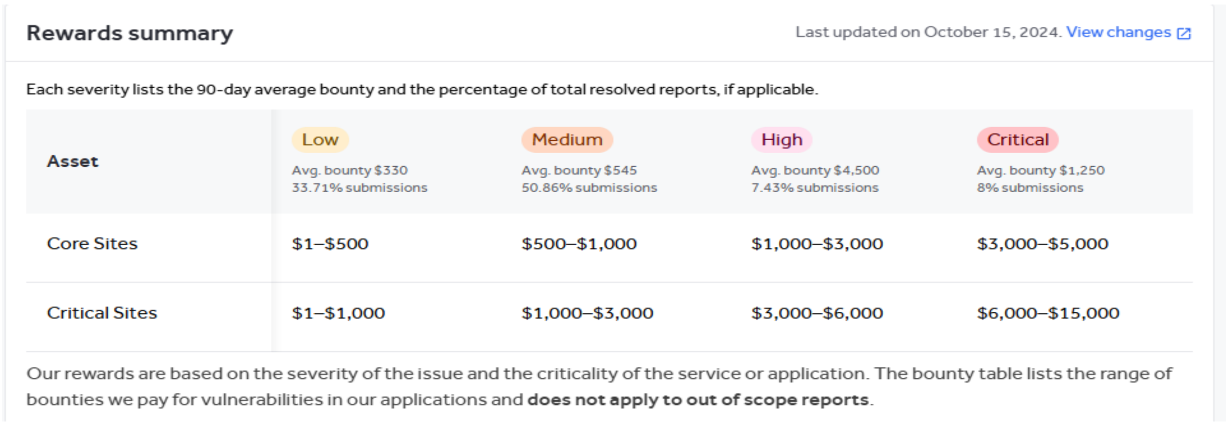
Search: Search | Scope: All scopes | Maximum severity: Any | Bounty eligibility: All

Download Burp Suite Project Configuration File Download CSV View changes (Last updated on December 16, 2024) 1-29 of 29

Asset name	Type	Coverage	Max. severity	Bounty	Last update	Resolved Reports
Core Site Staging instance for TaskCluster CI/CD tool. Source Code: https://github.com/taskcluster/taskcluster	Domain	In scope	Critical	Eligible	Nov 29, 2024	0 (0%)
aus5.mozilla.org Critical Site Backend update system for Mozilla products. No disruptive testing or scanning tools to be run on	Domain	In scope	Critical	Eligible	Dec 3, 2024	0 (0%)

1-29 of 29

Reward Systems - This mentions how a researcher is rewarded for identifying a vulnerability. Rewards may vary from lower to higher depending on the criticality of the program. (critical vulnerabilities will earn more pay).



Rewards summary

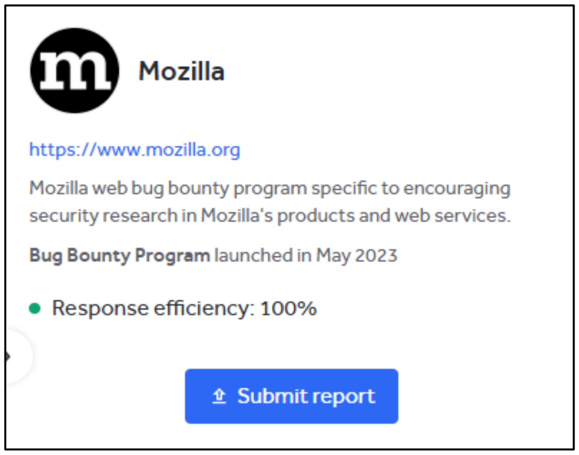
Last updated on October 15, 2024. View changes

Each severity lists the 90-day average bounty and the percentage of total resolved reports, if applicable.

Asset	Low Avg. bounty $330 33.71% submissions	Medium Avg. bounty $545 50.86% submissions	High Avg. bounty $4,500 7.43% submissions	Critical Avg. bounty $1,250 8% submissions
Core Sites	$1–$500	$500–$1,000	$1,000–$3,000	$3,000–$5,000
Critical Sites	$1–$1,000	$1,000–$3,000	$3,000–$6,000	$6,000–$15,000

Our rewards are based on the severity of the issue and the criticality of the service or application. The bounty table lists the range of bounties we pay for vulnerabilities in our applications and **does not apply to out of scope reports**.

Reporting - Researchers would report their findings via a platform (like HackerOne or Bugcrowd). The report contains a detailed description of the vulnerability, steps to reproduce it, and an impact assessment so that the organization can look the case.



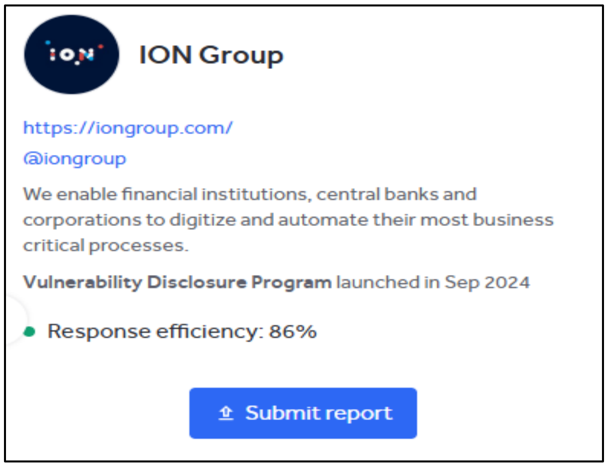
Mozilla

https://www.mozilla.org

Mozilla web bug bounty program specific to encouraging security research in Mozilla's products and web services.

Bug Bounty Program launched in May 2023

● Response efficiency: 100%



ION Group

https://iongroup.com/
@iongroup

We enable financial institutions, central banks and corporations to digitize and automate their most business critical processes.

Vulnerability Disclosure Program launched in Sep 2024

● Response efficiency: 86%

Methodology

The best practice is to follow a methodology to efficiently discover vulnerabilities and report them to the relevant platform.

Reconnaissance (Information Gathering)

- Identify target's information such as domains, subdomains, applications and IP addresses.
- Use tools like **Amass, Sublist3r, Shodan, and Censys** to gather information.
- Look for exposed services, open ports, and leaked credentials.

Enumeration and Mapping

- Analyze endpoints, APIs, and directories using **Burp Suite, Dirb, or Nmap**.
- Identify technologies in use such as frameworks and backend services.
- Check for old and vulnerable versions of software.

Vulnerability Discovery

- Test for **common web vulnerabilities** like:
    - **Injection attacks** (SQLi, XSS, Command Injection).
    - **Authentication flaws** (Weak passwords, 2FA bypass).
    - **IDOR (Insecure Direct Object References)**.
    - **Security misconfigurations** (Improper CORS, Open Redirects).
- Use tools like **Burp Suite, SQLmap, OWASP ZAP, and Nikto**.

Exploitation and Proof-of-Concept (PoC)

- Validate the impact of the vulnerability by safely exploiting it.
- Capture screenshots, logs, or response data as evidence.
- Ensure the attack does not cause harm (follow responsible disclosure).

Report Submission

- Write a clear, concise, and detailed report including:
  - **Bug description** and affected endpoint.
  - **Steps to reproduce** (detailed PoC).
  - **Impact assessment** (how it affects security).
  - **Suggested fixes or mitigations**.
- Submit through platforms like **HackerOne, Bugcrowd, or the company's bug bounty portal**.

Engage with the Community

- Participate in **forums, social media groups, and online chat platforms** focused on bug bounty hunting.
- Share experiences, ask for advice, and collaborate with other researchers.
- Stay updated with the latest security trends and techniques.

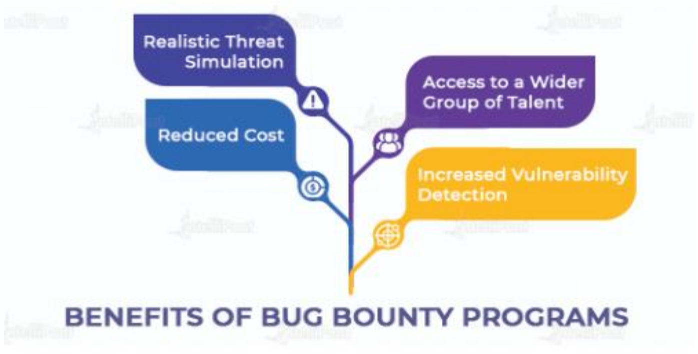
Learn and Adapt

- Continuously expand your knowledge through:
  - **Blog posts, tutorials, webinars, and security courses**.
  - **CTFs (Capture The Flag challenges) and hands-on practice**.
- Refine your methodology and toolset to become more efficient and effective.

Track Your Progress

- Maintain a record of:
  - **Submitted reports, accepted findings, and rewards earned**.
  - **Techniques that worked and areas for improvement**.
- Use this data to measure growth and optimize future bug-hunting efforts.

Advantages of Bug Bounty Hunting



Realistic Threat Simulation
Access to a Wider Group of Talent
Reduced Cost
Increased Vulnerability Detection
BENEFITS OF BUG BOUNTY PROGRAMS

❖ **Improved Security**

Bug bounty programs enable organizations to identify vulnerabilities of systems quickly and efficiently, resulting in improved security and fewer opportunities to exploit a vulnerability.

❖ **Cost-Effective**

Companies only reward valid bugs reports, making it cost effective than traditional security audits or hiring an internal security team.

❖ **Diverse Skillset**

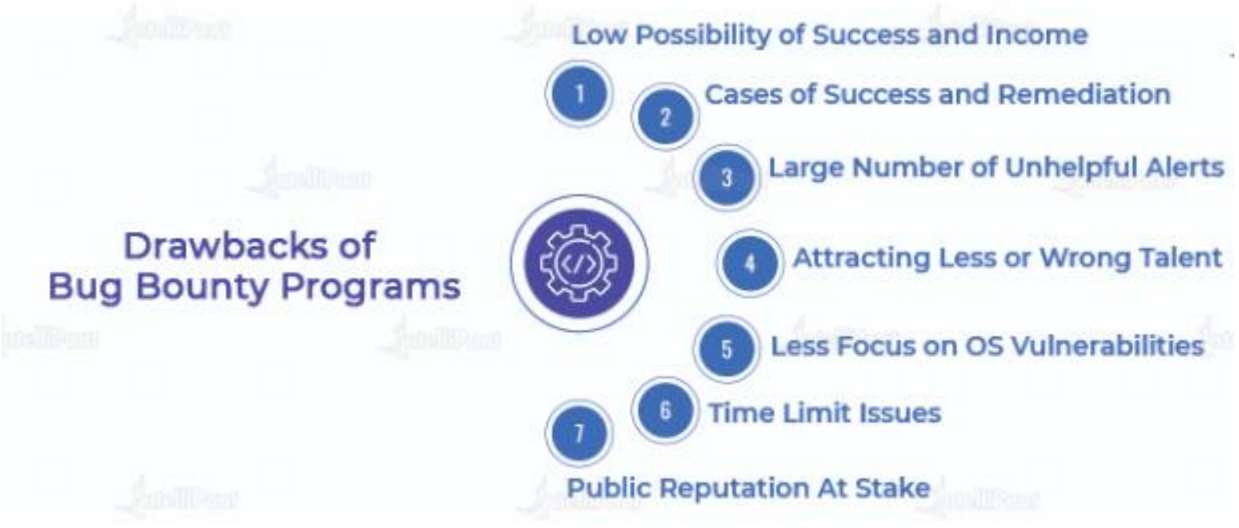
It offers access to a wide range of skilled ethical hackers, with diverse skills and a mindset to detect vulnerabilities that may be unknown to internal teams.

❖ **Real-World Testing**

The researchers conduct their assessments in a real-world environment to ultimately provide a more statistically accurate finding than using a vulnerability scanner or writing a theoretical assessment of a vulnerability.

❖ **Faster Detection**

Many hackers work simultaneously, the possibility of detecting and reporting a vulnerability is much quicker, thereby limiting the exposure time.

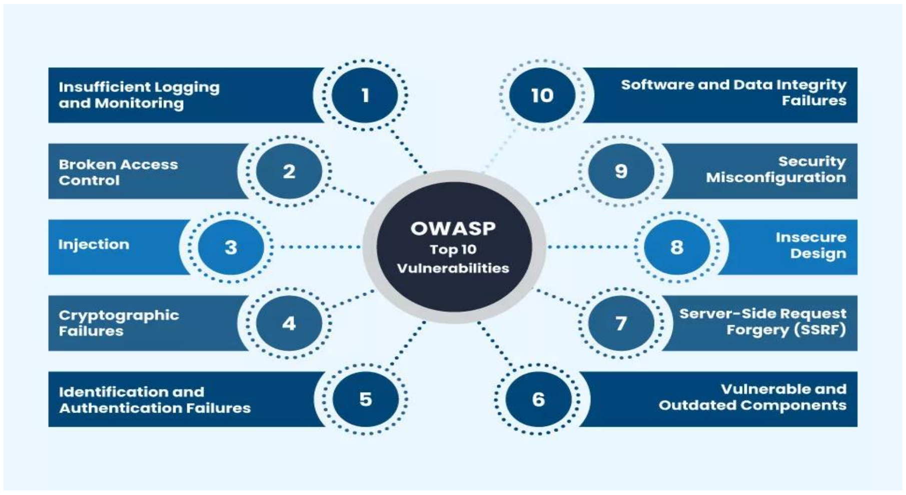


Drawbacks of Bug Bounty Programs
1 Low Possibility of Success and Income
2 Cases of Success and Remediation
3 Large Number of Unhelpful Alerts
4 Attracting Less or Wrong Talent
5 Less Focus on OS Vulnerabilities
6 Time Limit Issues
7 Public Reputation At Stake

OWASP Top 10 Vulnerabilities

Today, I took some time to go through the assignment and took some time to take notes about the fundamentals I need to cover before diving into actual bug bounty programs. As I noticed, I need to study about the OWASP Top 10 Vulnerabilities and essential tools to perform bug bounties.

The Open Web Application Security Project (OWASP) is renowned for the OWASP Top 10 list which outlines the most important security threats to web applications. This list helps developers, security experts, and even bug bounty hunters understand the common weaknesses that attackers take advantage of. The list of threats continues to evolve, making it relevant to web security, and as a result, remains beneficial to those in any web security field. The OWASP list of weaknesses not only enables me to effectively detect, report security vulnerabilities, but it also helps me understand secure coding and defensive strategies.


OWASP
Top 10
Vulnerabilities
Insufficient Logging and Monitoring
1
Broken Access Control
2
Injection
3
Cryptographic Failures
4
Identification and Authentication Failures
5
Vulnerable and Outdated Components
6
Server-Side Request Forgery (SSRF)
7
Insecure Design
8
Security Misconfiguration
9
Software and Data Integrity Failures
10

As I explore on my own the OWASP Top 10, I hope to understand the motives behind certain vulnerabilities and explain how they can be exploited, as well as implement strategies to block such actions from happening.

1. Broken Access Control

Motivation

Hackers try to force access to restricted resources to unauthorizedly access the content, modify them, view or delete them, as they prefer. This situation can be very critical as it can result in data breaches and be the cause of privilege escalation, or system compromise being taken.

Exploitation

✓ Creating changes in the URL parameters or API requests to take data that is not allowed.

✓ Bypassing access controls by changing cookies, tokens and JWTs.

✓ Exploiting misconfigured access control policies to grant high privileges.

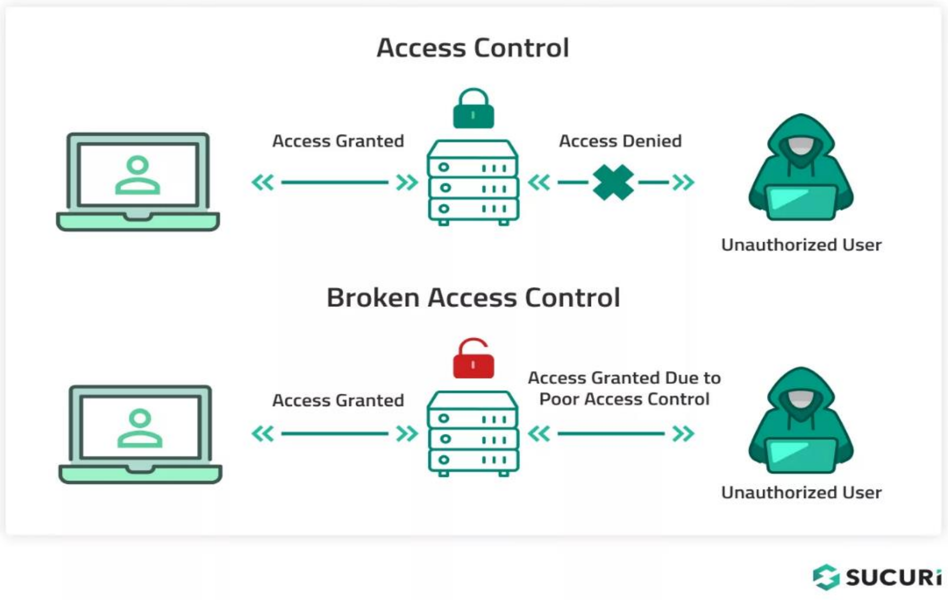
Mitigation Strategies

✓ The use of the principle of least privilege (PoLP) to limit user access.

✓ Server-side access controls are a must for critical operations.

✓ Try role-based access control (RBAC) that could manage user permissions.

✓ Regularly test for access control flaws using automated security tools and manual testing.



2. Cryptographic Failures

Motivation

Attack agents often take advantage of the weak or erroneously set up encryption with the aim of exploring sensitive data such as passwords, credit card data or users' data.

Exploitation

✓ Using obsolete encryption algorithms (MD5, SHA-1) which are easier to crack.

✓ The lack of key management practices such as the hardcoding of encryption keys in source code.

✓ Not using TLS (Transport Layer Security) for the data transfer, can cause a Man-in-the-Middle attack.

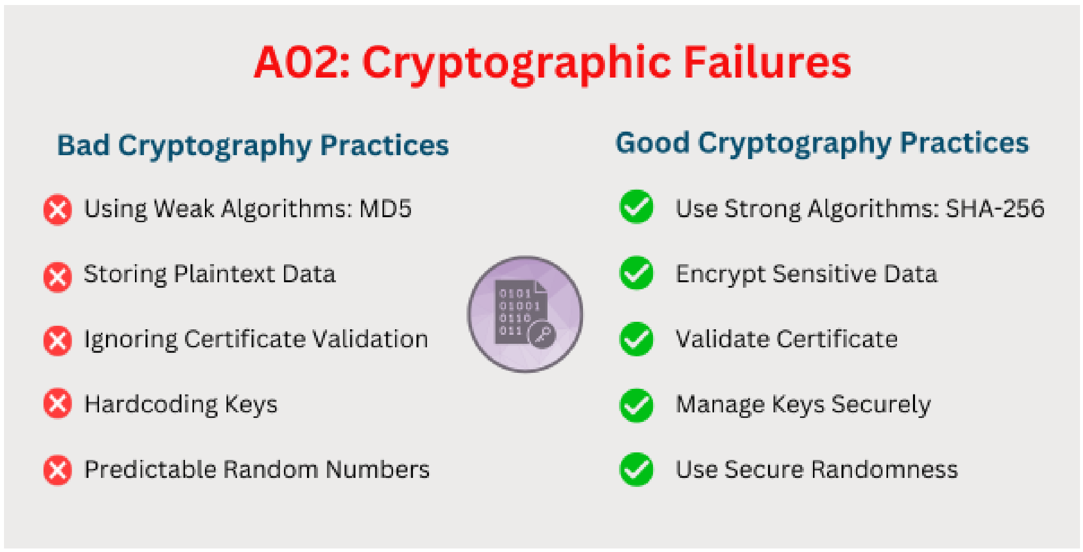
Mitigation Strategies

✓ Employ strong encryption algorithms such as AES-256 and a secure password generator.

✓ Encrypt all data with TLS 1.2 or higher.

✓ Keep control over cryptographic keys by utilizing HSMs (Hardware Security Modules).

✓ Information should not be stored or transmitted in the form of plain text.



A02: Cryptographic Failures

Bad Cryptography Practices	Good Cryptography Practices
✗ Using Weak Algorithms: MD5	✓ Use Strong Algorithms: SHA-256
✗ Storing Plaintext Data	✓ Encrypt Sensitive Data
✗ Ignoring Certificate Validation	✓ Validate Certificate
✗ Hardcoding Keys	✓ Manage Keys Securely
✗ Predictable Random Numbers	✓ Use Secure Randomness

3. Injection (SQL, XSS, Command Injection, etc.)

Motivation

When malicious users enter harmful data into the input fields, they manipulate databases to compromise, losing the data, and even getting the system controlled.

Exploitation

✓ SQL Injection- It's the act of inserting malicious queries into the database which is needed to obtain, extract, modify, or delete records from it.

✓ Cross-Site Scripting (XSS) - Inputting JavaScript code on a website to access session cookies or user's credentials.

✓ Command Injection- Running commands that is not allowed because of the improper validation of the given input.

Mitigation Strategies

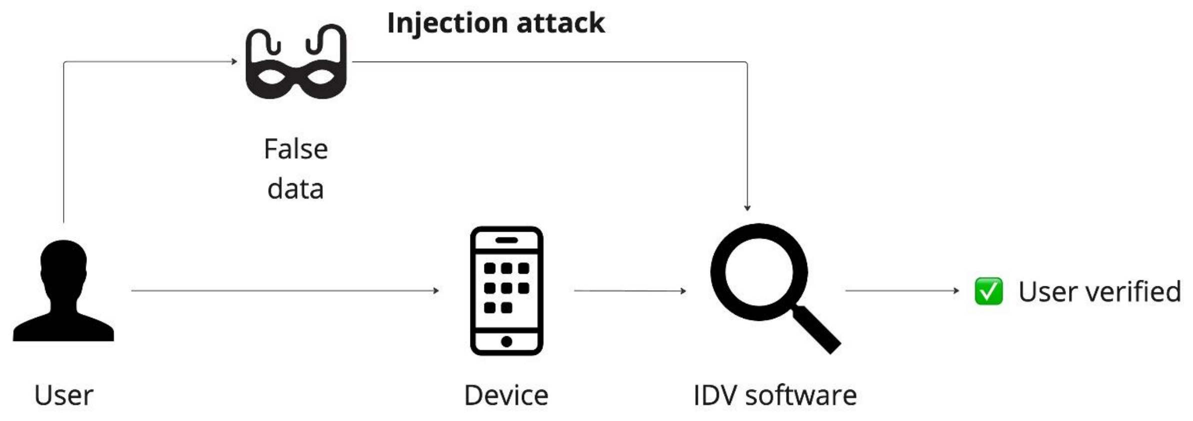
✓ Use parameterized queries and prepared statements to prevent SQL injection.

✓ Validate the input and encode output to protect against malicious scripts.

✓ Use content security policy to restrict untrusted scripts.

✓ Sanitize user data before processing them.

4. Insecure Design



Motivation

Attackers gain unauthorized access to an application by exploiting weaknesses within the application's logic design due to inherently flawed security architecture.

Exploitation

✓ Business logic flaws that allow one to bypass any security mechanism.

✓ Workflows containing no security validation such as initiating a password reset without verifying identity.

✓ Sensitive functionalities are exposed through unprotected APIs.

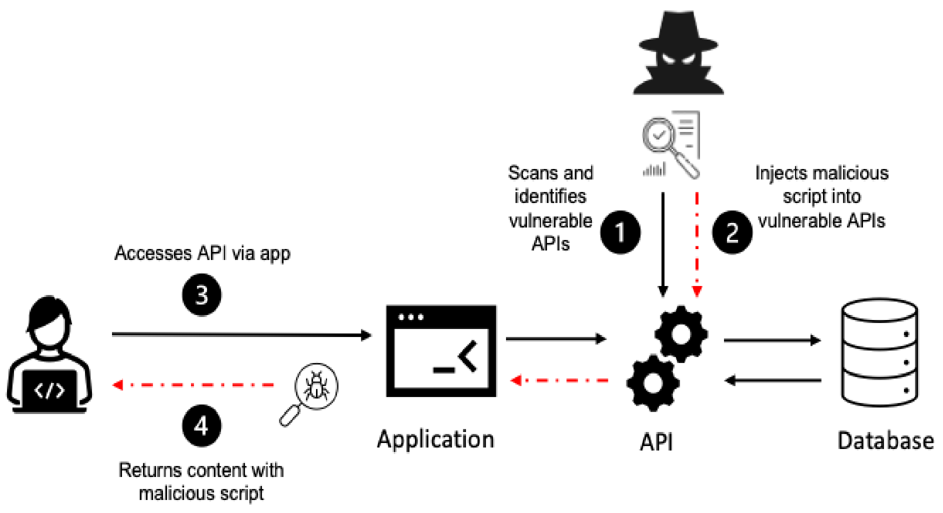
Mitigation Strategies

✓ Use secure design principles from the beginning of the development.

✓ Use threat modeling to influence possible attack vectors early.

✓ Repeat security code reviews so that flaws in logic can be found.

✓ Utilize security mitigations such as blocking repeated access, CAPTCHA, and MFA.


Scans and identifies vulnerable APIs
1
2
Injects malicious script into vulnerable APIs
Accesses API via app
3
4
Returns content with malicious script
Application
API
Database

5. Security Misconfiguration

Motivation

Attackers exploit default settings that were adopted, misconfigured security controls, or openly exposed admin panels.

Exploitation

✓The use of default credentials, that were never changed (admin/admin).

✓Abuse misconfigured CORS (sitting across Cross-Original Resource Sharing) policies that allowed unauthorized cross-site requests.

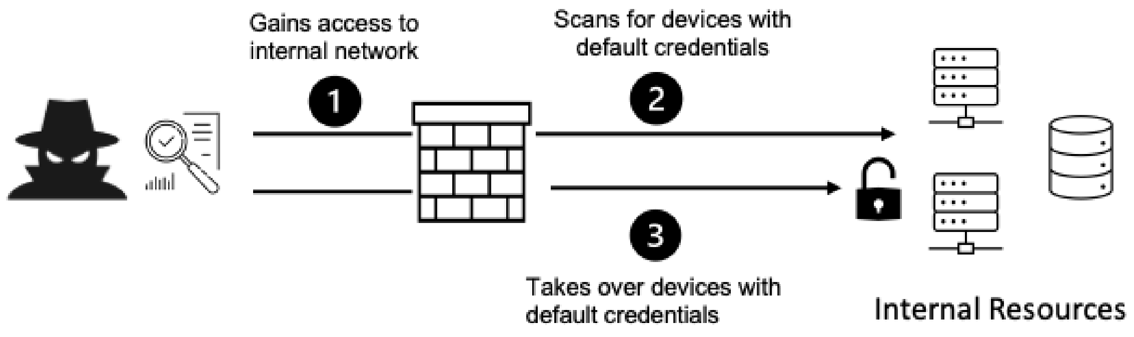
✓Collects sensitive information from stack traces or error messages unrestricted in public-facing environments.

Mitigation strategies

✓Changed default credentials immediately after the installation.

✓Disabled all unnecessary services and features to lessen chances of being attacked.
✓Configured error handling in such a way that no system details were exposed.

✓Regular manual auditing of security configuration should be done on a regular basis.


Gains access to internal network
1
Scans for devices with default credentials
2
3
Takes over devices with default credentials
Internal Resources

6. Vulnerable and Outdated Components

Motivation

The attackers take advantage of known vulnerabilities in outdated software, libraries, or plugins aimed at gaining control over applications.

Exploitation

✓Attackers exploit old frameworks, including outdated Apache Struts, jQuery, or WordPress plugins.

✓Uses publicly available exploits such as unpatched vulnerabilities. (Log4j Vulnerability)

✓Lauch dependency confusion attacks to inject the malicious code into the software supply chain.

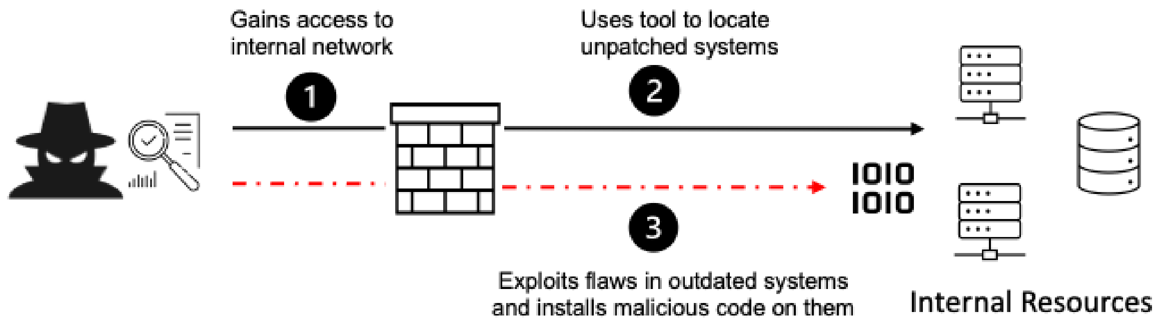
Mitigation strategies

✓The regular update of all libraries, dependencies, and frameworks.

✓SCA tools to identify outdated and old components.

✓Monitoring CVE databases and immediate application of security patches.

✓Removing unused plugins and modules that could introduce vulnerabilities.


Gains access to internal network
1
Uses tool to locate unpatched systems
2
1010
1010
3
Exploits flaws in outdated systems and installs malicious code on them
Internal Resources

7. Identification and Authentication Failures

Motivation

An attacker may target weak mechanisms for authenticating individuals by accessing resources with the intention of impersonating users or taking over accounts to use resources.

Exploitation

✓ Attacks on weak passwords by brute force.

✓ Credential stuffing with previously leaked passwords.

✓ Hijacking or replay of sessions due to improper session management.

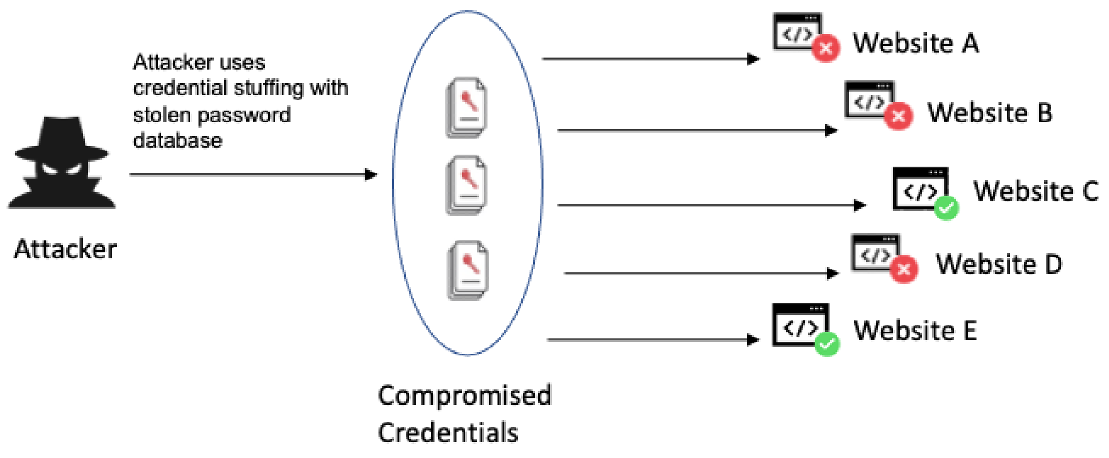
Mitigation strategies

✓ Multi-factor authentication (MFA) for sensitive operations.

✓ Strong password policies (including complexity requirements) should be enforced.

✓ Secure Session Management (Session ids should be regenerated after logging in).

✓ Rate-limiting and CAPTCHA are integrated to prevent automated attacks.



8. Software and Data Integrity Failures

Motivation

Attackers insert malicious code, or modified software components, into the target in order to gain control of application integrity.

Exploitation

✓ They exploit insecure software updates with no integrity verification.

✓ They perform code injection attacks through unverified third-party libraries.

✓ They launch supply chain attacks by inserting malicious dependencies into software packages.

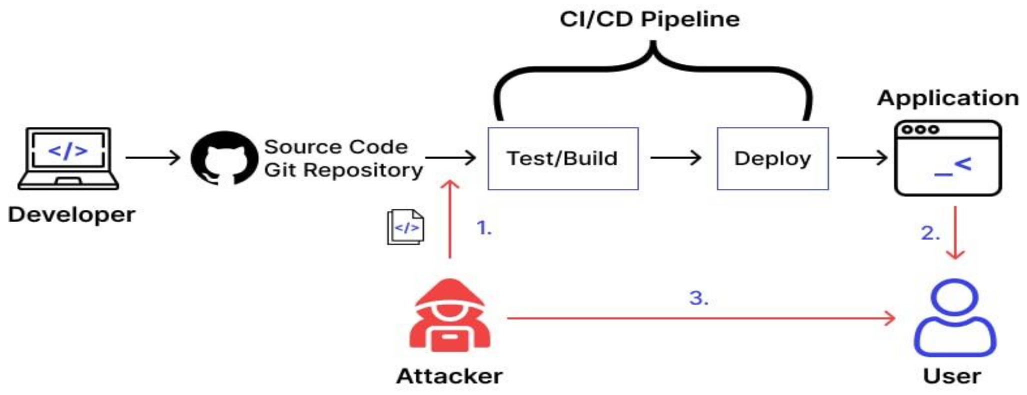
Mitigation strategies

✓ All software updates and dependencies were digitally signed and verified.

✓ Made use of subresource integrity for external scripts.

✓ Secure CI/CD pipelines with integrity checks.

✓ Audited third-party components before integrating them.



9. Security Logging and Monitoring Failures

Motivation

Attackers make better use of weak logging and monitoring to make their attacks unnoticed.

Exploitation

✓ They disable or bypass logging mechanisms to erase any trace of an intrusion.

✓ They exploit weak or absent alerting mechanisms for suspicious activities.

✓ They conduct persistent attacks over time without being detected.

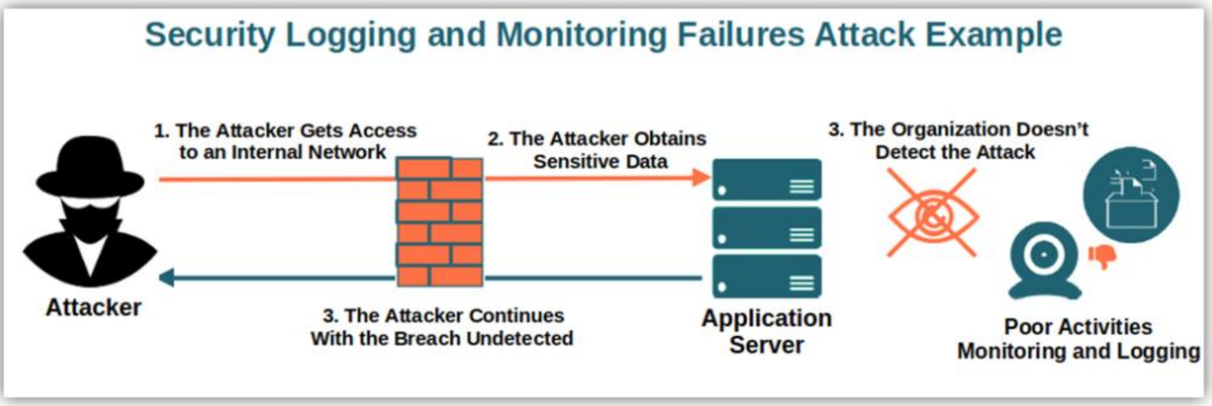
Mitigating Strategies

✓ Detailed logging of authentication attempts, failed logins and other critical system events were enabled.

✓ SIEM (Security Information and Event Management) solutions implemented for real-time threat detection.

✓ Automated alerting of suspicious activities was implemented.

✓ Regular review of the audit logs for unusual patterns.


Security Logging and Monitoring Failures Attack Example
1. The Attacker Gets Access to an Internal Network
2. The Attacker Obtains Sensitive Data
3. The Organization Doesn't Detect the Attack
Attacker
3. The Attacker Continues With the Breach Undetected
Application Server
Poor Activities Monitoring and Logging

10.Server-Side Request Forgery (SSRF)

Motivation

Attackers exploit server-side requests to access internal resources or perform malicious operations.

Exploitation

✓ They force the server to fetch internal URLs, which then leaked some private data.

✓ They gain access to the cloud metadata APIs to extract sensitive information.

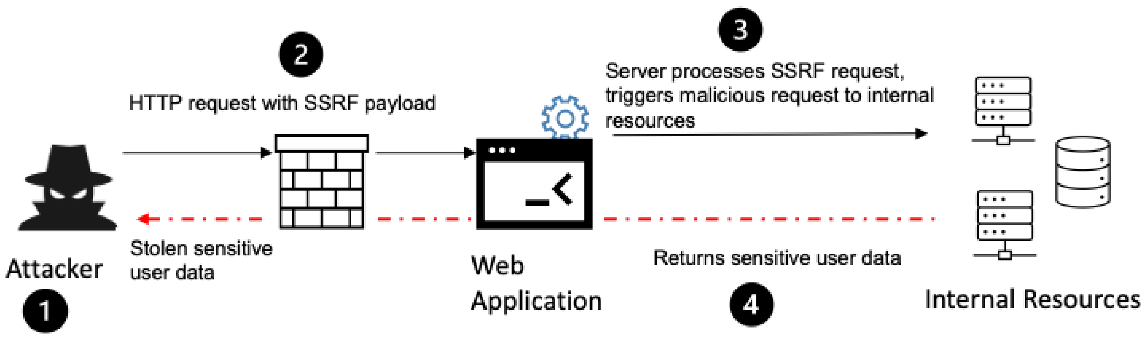
✓ They use SSRF to evade firewalls and mount attacks against internal network services.

Mitigating strategies

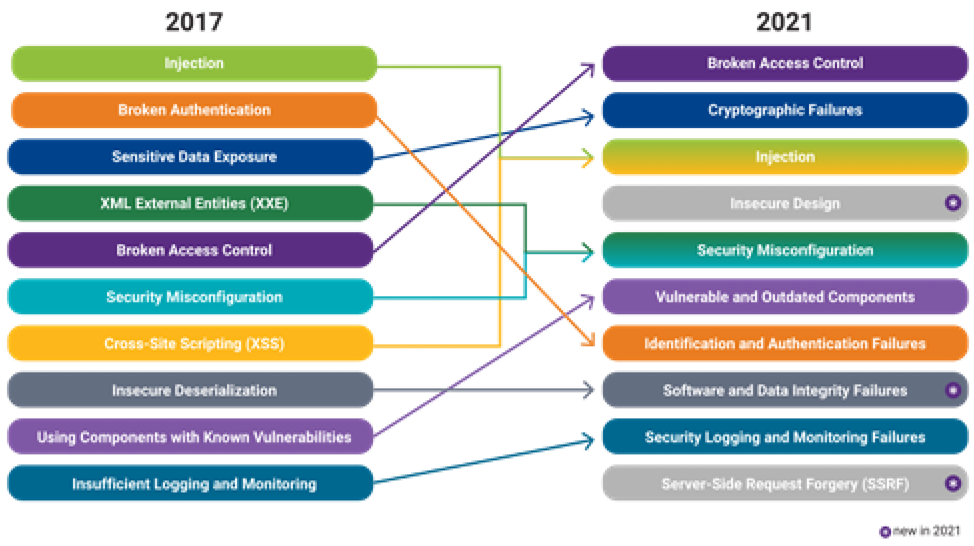
✓ Traffic to only known, whitelisted domains was allowed.

✓ Access from arbitrary user input to internal resources was denied.

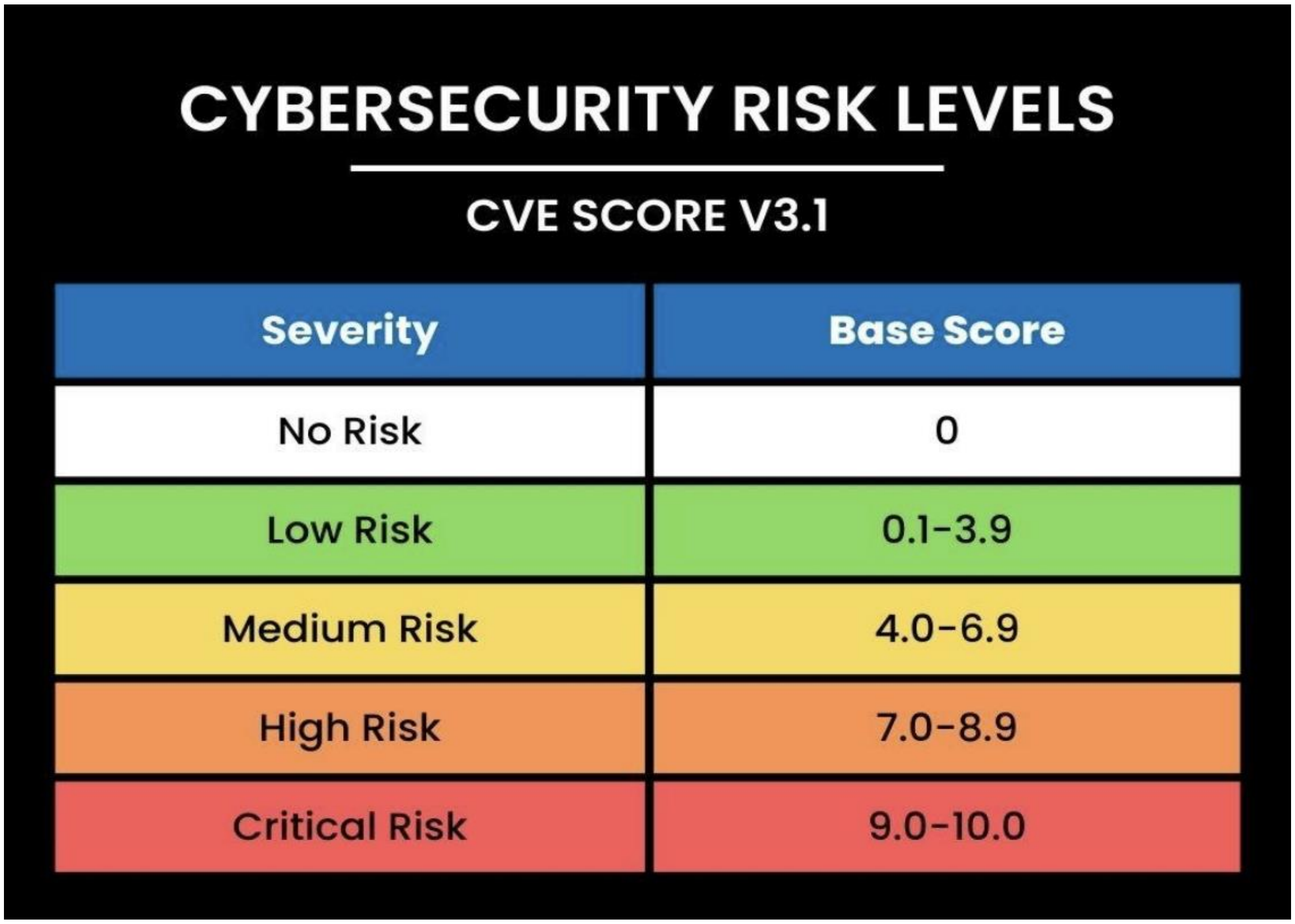
✓ All user-controllable URLs were properly validated and sanitized.


2
HTTP request with SSRF payload
3
Server processes SSRF request, triggers malicious request to internal resources
Attacker
1
Stolen sensitive user data
Web Application
Returns sensitive user data
4
Internal Resources

Comparison


2017
Injection
Broken Authentication
Sensitive Data Exposure
XML External Entities (XXE)
Broken Access Control
Security Misconfiguration
Cross-Site Scripting (XSS)
Insecure Deserialization
Using Components with Known Vulnerabilities
Insufficient Logging and Monitoring
2021
Broken Access Control
Cryptographic Failures
Injection
Insecure Design
Security Misconfiguration
Vulnerable and Outdated Components
Identification and Authentication Failures
Software and Data Integrity Failures
Security Logging and Monitoring Failures
Server-Side Request Forgery (SSRF)
new in 2021

Risk levels of Vulnerabilities



CYBERSECURITY RISK LEVELS

CVE SCORE V3.1

Severity	Base Score
No Risk	0
Low Risk	0.1-3.9
Medium Risk	4.0-6.9
High Risk	7.0-8.9
Critical Risk	9.0-10.0

Critical (9.0 - 10.0)

Impact: Provides complete system hijack, data exposure, or remote code execution.

Examples: Remote Code Execution (RCE), Privilege Escalation, Unrestricted Access to Sensitive Data.

Action: Should be addressed immediately and patched.

High (7.0 - 8.9)

Impact: Allows attackers to gain unauthorized access or deny services.

Examples: SQL Injection (SQLi), Server-Side Request Forgery (SSRF), Authentication Bypass.

Action: Should be addressed as soon as possible.

Medium (4.0 - 6.9)

Impact: Potentially able to reveal sensitive data or weaken security defenses.

Examples: Cross-Site Scripting (XSS), Security Misconfigurations, Open Redirects.

Action: Needs to be remediated but is less urgent than high/critical ones.

Low (0.1 - 3.9)

Impact: Low-level security flaws that require special conditions to be exploited.

Examples: Clickjacking, Information Disclosure (leakage of non-sensitive information).

Action: Remediation is recommended but not urgent.

No Risk (0.0)

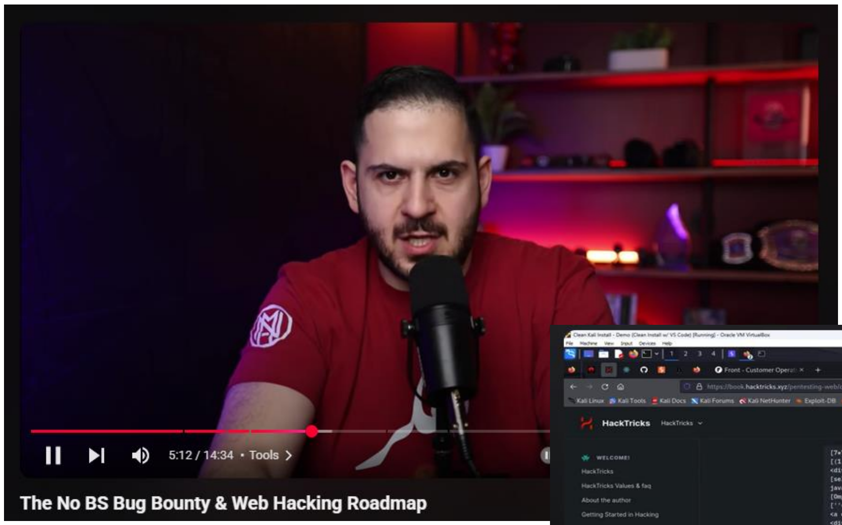
Impact: No actual security threat but potentially at risk if combined with other issues.

Examples: Outdated software versions, exposed HTTP headers.

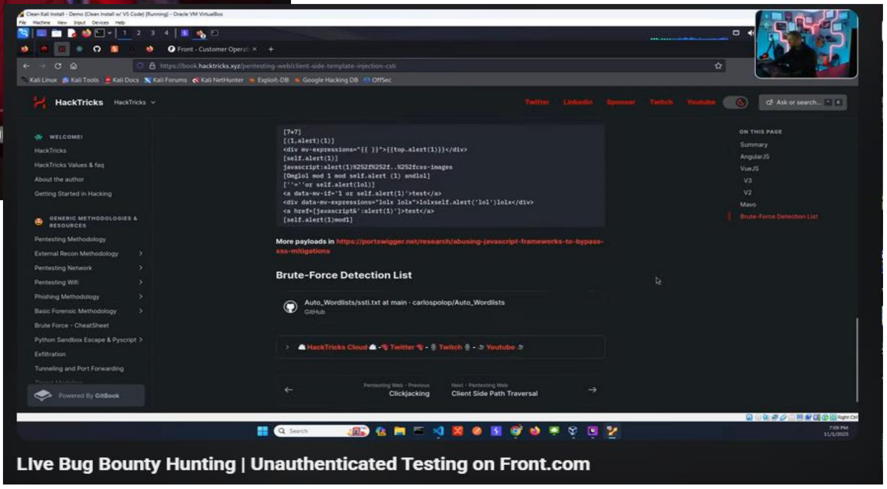
Action: Recorded for alert but generally no correction is necessary urgently.

Learning Process

Watched many youtube videos to gain knowledge about bug bounty



The No BS Bug Bounty & Web Hacking Roadmap



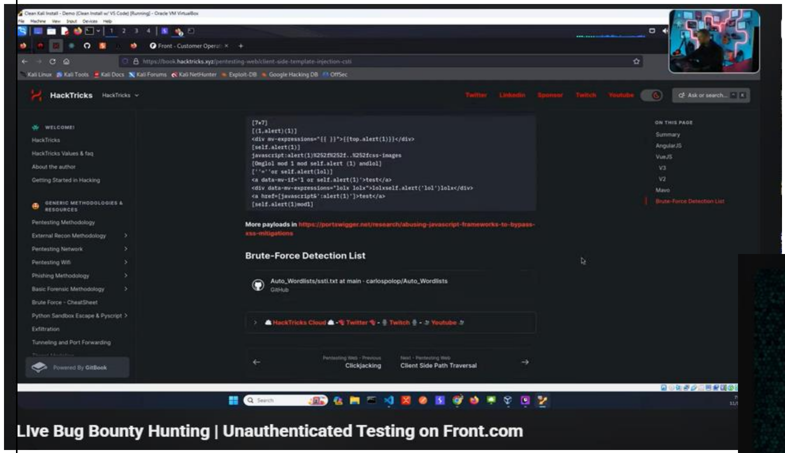
```
[7*7]
[(1,alert)(1)]
<div mv-expressions="{{ }}">{{top.alert(1)}}</div>
[self.alert(1)]
javascript:alert(1)%252f%252f..%252fcss-images
[Omglol mod 1 mod self.alert (1) andlol]
[''='' or self.alert(lol)]
<a data-mv-if='1 or self.alert(1)'>test</a>
<div data-mv-expressions="lolx lolx">lolxself.alert('lol')lolx</div>
<a href=[javascript&':alert(1)']>test</a>
[self.alert(1)mod1]
```

More payloads in https://portswigger.net/research/abusing-javascript-frameworks-to-bypass-xss-mitigations

Brute-Force Detection List

Auto_Wordlists/ssti.txt at main · carlospolop/Auto_Wordlists

Live Bug Bounty Hunting | Unauthenticated Testing on Front.com



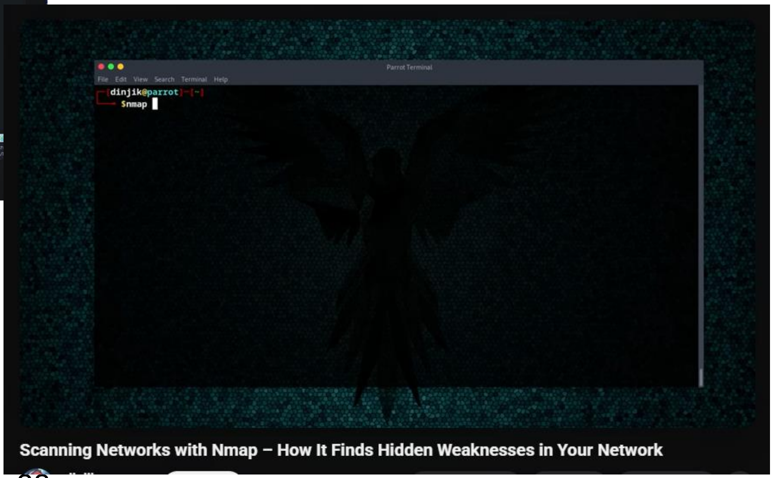
```
[7*7]
[(1,alert)(1)]
<div mv-expressions="{{ }}">{{top.alert(1)}}</div>
[self.alert(1)]
javascript:alert(1)%252f%252f..%252fcss-images
[Omglol mod 1 mod self.alert (1) andlol]
[''='' or self.alert(lol)]
<a data-mv-if='1 or self.alert(1)'>test</a>
<div data-mv-expressions="lolx lolx">lolxself.alert('lol')lolx</div>
<a href=[javascript&':alert(1)']>test</a>
[self.alert(1)mod1]
```

More payloads in https://portswigger.net/research/abusing-javascript-frameworks-to-bypass-xss-mitigations

Brute-Force Detection List

Auto_Wordlists/ssti.txt at main · carlospolop/Auto_Wordlists

Live Bug Bounty Hunting | Unauthenticated Testing on Front.com

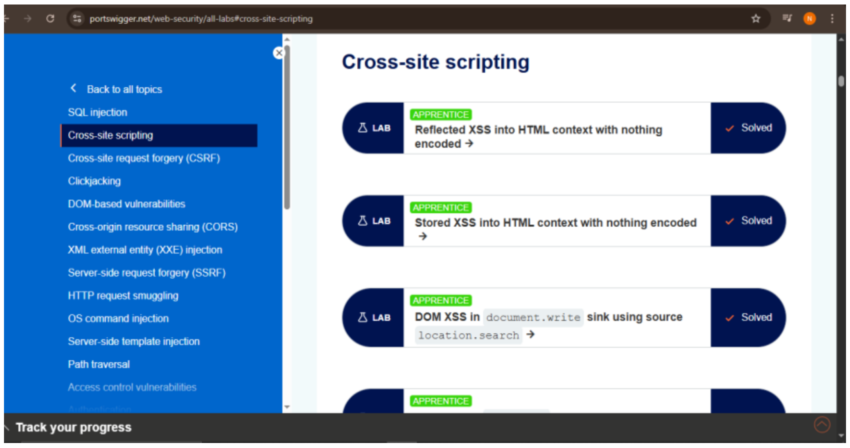


```
┌─[dinjik@parrot]─[~]
└──╼ $nmap
```

Scanning Networks with Nmap – How It Finds Hidden Weaknesses in Your Network

Portswigger lab completion

Completed portswigger labs to gain knowledge about certain vulnerabilities.



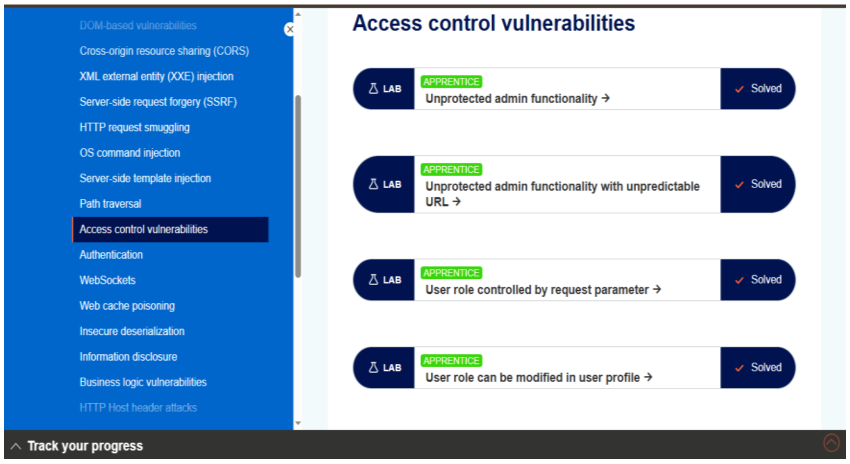
portswigger.net/web-security/all-labs#cross-site-scripting

Back to all topics
SQL injection
Cross-site scripting
Cross-site request forgery (CSRF)
Clickjacking
DOM-based vulnerabilities
Cross-origin resource sharing (CORS)
XML external entity (XXE) injection
Server-side request forgery (SSRF)
HTTP request smuggling
OS command injection
Server-side template injection
Path traversal
Access control vulnerabilities

Cross-site scripting

LAB	APPRENTICE Reflected XSS into HTML context with nothing encoded →	✓ Solved
LAB	APPRENTICE Stored XSS into HTML context with nothing encoded →	✓ Solved
LAB	APPRENTICE DOM XSS in `document.write` sink using source `location.search` →	✓ Solved
	APPRENTICE	

Track your progress



DOM-based vulnerabilities
Cross-origin resource sharing (CORS)
XML external entity (XXE) injection
Server-side request forgery (SSRF)
HTTP request smuggling
OS command injection
Server-side template injection
Path traversal
Access control vulnerabilities
Authentication
WebSockets
Web cache poisoning
Insecure deserialization
Information disclosure
Business logic vulnerabilities
HTTP Host header attacks

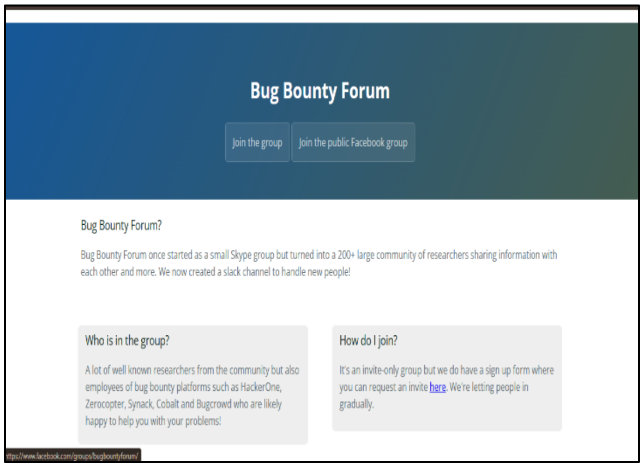
Access control vulnerabilities

LAB	APPRENTICE Unprotected admin functionality →	✓ Solved
LAB	APPRENTICE Unprotected admin functionality with unpredictable URL →	✓ Solved
LAB	APPRENTICE User role controlled by request parameter →	✓ Solved
LAB	APPRENTICE User role can be modified in user profile →	✓ Solved

Track your progress

Bug bounty forums and communities

Read about bug bounty forum and communities ols for Bug Hunting



Bug Bounty Forum

Join the group | Join the public Facebook group

Bug Bounty Forum?

Bug Bounty Forum once started as a small Skype group but turned into a 200+ large community of researchers sharing information with each other and more. We now created a slack channel to handle new people!

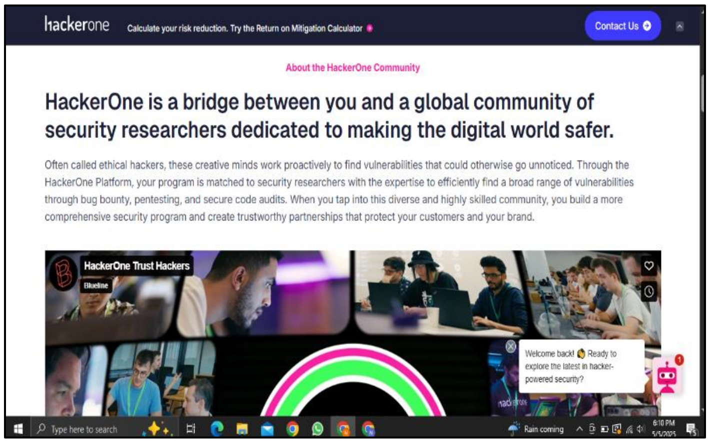
Who is in the group?

A lot of well known researchers from the community but also employees of bug bounty platforms such as HackerOne, Zerocopter, Synack, Cobalt and Bugcrowd who are likely happy to help you with your problems!

How do I join?

It's an invite-only group but we do have a sign up form where you can request an invite here. We're letting people in gradually.

https://www.facebook.com/groups/bugbountyforum/



hackerone Calculate your risk reduction. Try the Return on Mitigation Calculator | Contact Us

About the HackerOne Community

HackerOne is a bridge between you and a global community of security researchers dedicated to making the digital world safer.

Often called ethical hackers, these creative minds work proactively to find vulnerabilities that could otherwise go unnoticed. Through the HackerOne Platform, your program is matched to security researchers with the expertise to efficiently find a broad range of vulnerabilities through bug bounty, pentesting, and secure code audits. When you tap into this diverse and highly skilled community, you build a more comprehensive security program and create trustworthy partnerships that protect your customers and your brand.

Tools for bug bounty

After understanding the OWASP Top 10 vulnerabilities, I found out that it is equally important to use the right tools as it is to know the vulnerabilities. Bug bounty hunters often use a variety of tools for automation of reconnaissance, vulnerability identification, and exploiting them efficiently.

These tools allow to make the whole testing process smooth, which means to find a security flaw in web applications easily.

I will cover some of the most common tools used for bug bounty hunting. I will explain the purpose and how they work as well as the importance of these tools in finding defects.

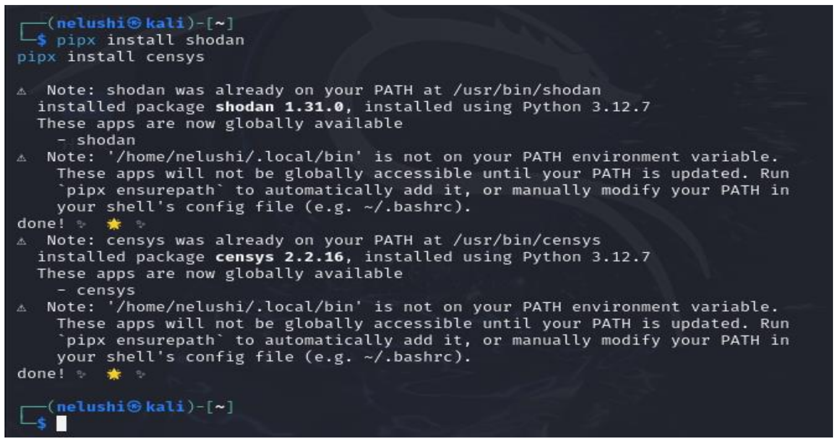
Reconnaissance Tools (Information Gathering)

Shodan

Purpose - Queries devices, services, and vulnerabilities that are exposed to the internet.

How it works - It scans the internet for open ports, misconfigured security, exposed services like databases and webcams, and industrial systems.

Why it's useful - It eases finding misconfigured servers and other potential entry points before launching the active tests.



```
┌──(nelushi㉿kali)-[~]
└─$ pipx install shodan
pipx install censys

⚠  Note: shodan was already on your PATH at /usr/bin/shodan
  installed package shodan 1.31.0, installed using Python 3.12.7
  These apps are now globally available
    - shodan
⚠  Note: '/home/nelushi/.local/bin' is not on your PATH environment variable.
    These apps will not be globally accessible until your PATH is updated. Run
    `pipx ensurepath` to automatically add it, or manually modify your PATH in
    your shell's config file (e.g. ~/.bashrc).
done! ✨ 🌟 ✨
⚠  Note: censys was already on your PATH at /usr/bin/censys
  installed package censys 2.2.16, installed using Python 3.12.7
  These apps are now globally available
    - censys
⚠  Note: '/home/nelushi/.local/bin' is not on your PATH environment variable.
    These apps will not be globally accessible until your PATH is updated. Run
    `pipx ensurepath` to automatically add it, or manually modify your PATH in
    your shell's config file (e.g. ~/.bashrc).
done! ✨ 🌟 ✨

┌──(nelushi㉿kali)-[~]
└─$ 
```

Censys

Purpose - same as Shodan, allows researchers to seek exposed assets over the internet.

Method - It indexes publicly available network services while showing properties like SSL certificates, open ports, and vulnerable services.

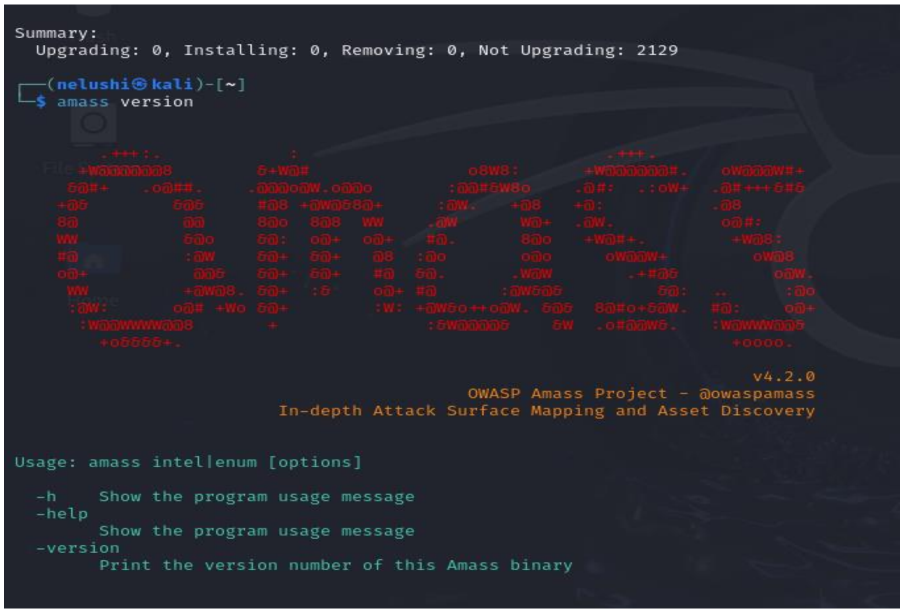
Why it's useful - It can flag out-of-date software versions or weak security configurations.

Amass

Purpose - This is a very aggressive subdomain enumeration tool used for mapping the attack surface of an organization.

How it works - It collects subdomains using different methodologies like through DNS lookups, web scraping, and by passive data sources.

Why it's useful - It helps discover unknown insecure domains.



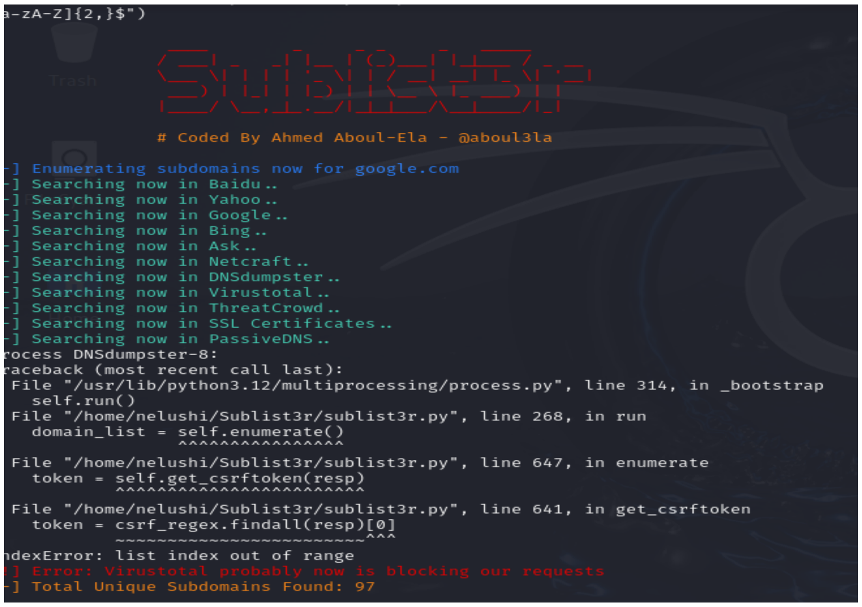
```
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 2129

┌──(nelushi㉿kali)-[~]
└─$ amass version

        .+++:.            :                             .+++.
      +W@@@@@@8        &+W@#               o8W8:      +W@@@@@@#.   oW@@@W#+
     &@#+   .o@##.    .@@@o@W.o@@o       :@@#&W8o    .@#:  .:oW+  .@#+++&#&
    +@&        &@&     #@8 +@W@&8@+     :@W.   +@8   +@:          .@8
    8@          @@     8@o  8@8  WW    .@W      W@+  .@W.          o@#:
    WW          &@o    &@:  o@+  o@+   #@.      8@o   +W@#+.        +W@8:
    #@          :@W    &@+  &@+   @8  :@o       o@o     oW@@W+        oW@8
    o@+          @@&   &@+  &@+   #@  &@.      .W@W       .+#@&         o@W.
     WW         +@W@8. &@+  :&    o@+ #@      :@W&@&         &@:  ..     :@o
     :@W:      o@# +Wo &@+        :W: +@W&o++o@W. &@&  8@#o+&@W.  #@:    o@+
      :W@@WWWW@@8       +              :&W@@@@&    &W  .o#@@W&.   :W@WWW@@&
        +o&&&&+.                                                 +oooo.

                                                                      v4.2.0
                                         OWASP Amass Project - @owaspamass
                                In-depth Attack Surface Mapping and Asset Discovery

Usage: amass intel|enum [options]

  -h    Show the program usage message
  -help
        Show the program usage message
  -version
        Print the version number of this Amass binary
```

Sublist3r

Purpose - Sublist3r is another subdomain enumeration tool that finds subdomains through public search engines.

How it works - It queries search engines of such color as Google, Yahoo by VirusTotal, to collect subdomain data.

Why it's useful - It's a quick and effective method for finding the right subdomains to test.



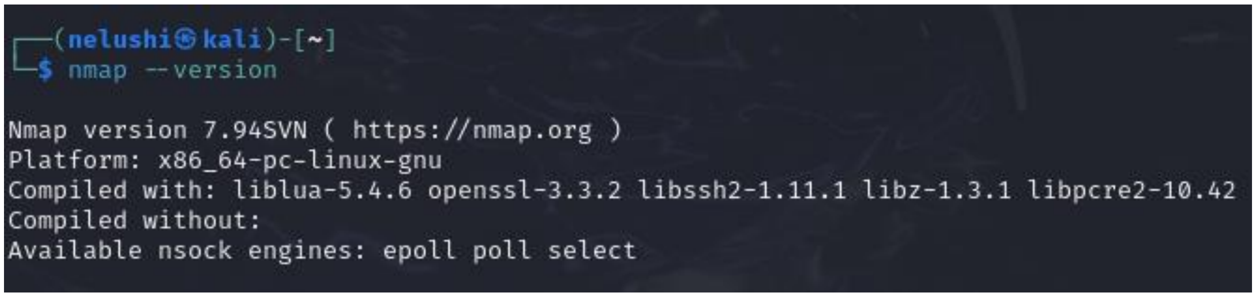
```
a-zA-Z]{2,}$")

                # Coded By Ahmed Aboul-Ela - @aboul3la

-] Enumerating subdomains now for google.com
-] Searching now in Baidu..
-] Searching now in Yahoo..
-] Searching now in Google..
-] Searching now in Bing..
-] Searching now in Ask..
-] Searching now in Netcraft..
-] Searching now in DNSdumpster..
-] Searching now in Virustotal..
-] Searching now in ThreatCrowd..
-] Searching now in SSL Certificates..
-] Searching now in PassiveDNS..
rocess DNSdumpster-8:
raceback (most recent call last):
 File "/usr/lib/python3.12/multiprocessing/process.py", line 314, in _bootstrap
   self.run()
 File "/home/nelushi/Sublist3r/sublist3r.py", line 268, in run
   domain_list = self.enumerate()
                 ^^^^^^^^^^^^^^^^

 File "/home/nelushi/Sublist3r/sublist3r.py", line 647, in enumerate
   token = self.get_csrftoken(resp)
           ^^^^^^^^^^^^^^^^^^^^^^^^

 File "/home/nelushi/Sublist3r/sublist3r.py", line 641, in get_csrftoken
   token = csrf_regex.findall(resp)[0]
           ~~~~~~~~~~~~~~~~~~~~~~~~^^^
ndexError: list index out of range
!] Error: Virustotal probably now is blocking our requests
-] Total Unique Subdomains Found: 97
```

Nmap

Purpose - A network scan tool that can open ports, along with running services, and firewall configurations.

How it works - It sends crafted packets to a target and analyzes the responses to determine what services are running.

Why it's useful - Bug bounty hunters could utilize the same to expose services and attack vectors.



```
┌──(nelushi㉿kali)-[~]
└─$ nmap --version

Nmap version 7.94SVN ( https://nmap.org )
Platform: x86_64-pc-linux-gnu
Compiled with: liblua-5.4.6 openssl-3.3.2 libssh2-1.11.1 libz-1.3.1 libpcre2-10.42
Compiled without:
Available nsock engines: epoll poll select
```

How to download and install:

rpm -vhU https://nmap.org/dist/nmap-7.95-3.x86_64.rpm

Basic Commands:

nmap <target>

nmap -p 22,80,443 <target>

nmap -p- <target>

nmap -sV <target>

nmap -oN <outputfile.txt> <target>

Web Application Scanning Tools (Vulnerability Identification)

Burp Suite

Purpose - An extremely popular web security testing tool to intercept, modify, and analyze HTTP/S requests.

How it works - Acts as a proxy between the user and the site, thus allowing security testers to inspect and modify requests/responses in real time.

Why it’s useful - Helps find security weaknesses such as injections, bypass authentication, or misconfigurations.

How to install:

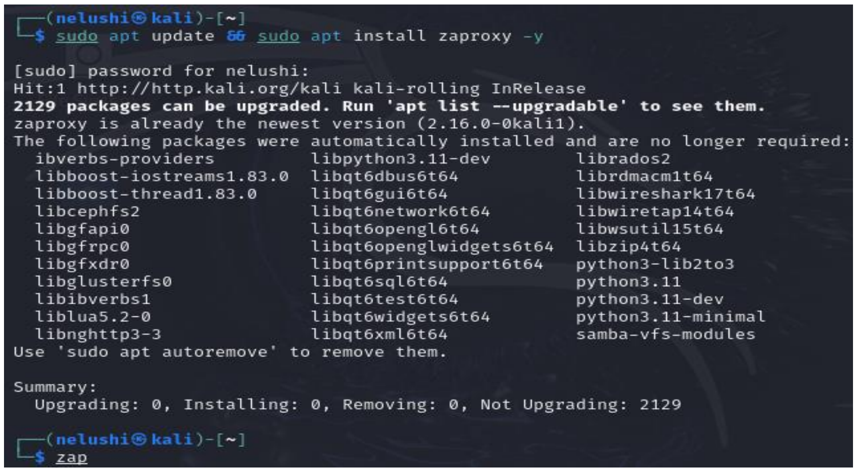
sudo apt install burpsuite

OWASP ZAP (Zed Attack Proxy)

Purpose - It is a completely automated vulnerability scanner for use on web applications.

How it works - It actively scans a website and reports potential security risks such as injection, broken authentication, and cross-site scripting.

Why it’s useful - It is another free, beginner-friendly penetration testing alternative to Burp Suite.



```
┌──(nelushi㉿kali)-[~]
└─$ sudo apt update && sudo apt install zaproxy -y

[sudo] password for nelushi:
Hit:1 http://http.kali.org/kali kali-rolling InRelease
2129 packages can be upgraded. Run 'apt list --upgradable' to see them.
zaproxy is already the newest version (2.16.0-0kali1).
The following packages were automatically installed and are no longer required:
  ibverbs-providers         libpython3.11-dev         librados2
  libboost-iostreams1.83.0  libqt6dbus6t64            librdmacm1t64
  libboost-thread1.83.0     libqt6gui6t64             libwireshark17t64
  libcephfs2                libqt6network6t64         libwiretap14t64
  libgfapi0                 libqt6opengl6t64          libwsutil15t64
  libgfrpc0                 libqt6openglwidgets6t64   libzip4t64
  libgfxdr0                 libqt6printsupport6t64    python3-lib2to3
  libglusterfs0             libqt6sql6t64             python3.11
  libibverbs1               libqt6test6t64            python3.11-dev
  liblua5.2-0               libqt6widgets6t64         python3.11-minimal
  libnghttp3-3              libqt6xml6t64             samba-vfs-modules
Use 'sudo apt autoremove' to remove them.

Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 2129

┌──(nelushi㉿kali)-[~]
└─$ zap
```

Nikto

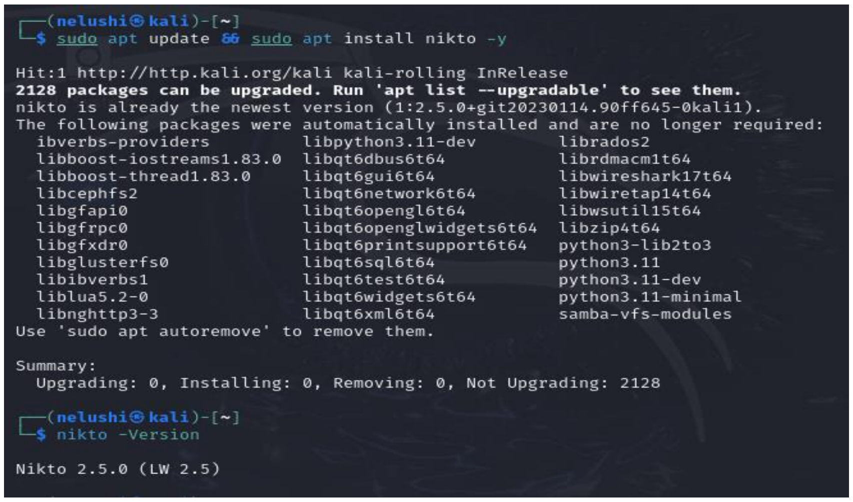
Purpose - It's a web server vulnerability scanner to check against known security problems.

How it works - It does this through scanning outdated software, misconfigured headers, and exposed sensitive files.

Why it matters - Identifies many common security problems which are often missed very quickly.

How to Install:

sudo apt install nikto



```
┌──(nelushi㉿kali)-[~]
└─$ sudo apt update && sudo apt install nikto -y

Hit:1 http://http.kali.org/kali kali-rolling InRelease
2128 packages can be upgraded. Run 'apt list --upgradable' to see them.
nikto is already the newest version (1:2.5.0+git20230114.90ff645-0kali1).
The following packages were automatically installed and are no longer required:
  ibverbs-providers          libpython3.11-dev          librados2
  libboost-iostreams1.83.0   libqt6dbus6t64             librdmacm1t64
  libboost-thread1.83.0      libqt6gui6t64              libwireshark17t64
  libcephfs2                 libqt6network6t64          libwiretap14t64
  libgfapi0                  libqt6opengl6t64           libwsutil15t64
  libgfrpc0                  libqt6openglwidgets6t64    libzip4t64
  libgfxdr0                  libqt6printsupport6t64     python3-lib2to3
  libglusterfs0              libqt6sql6t64              python3.11
  libibverbs1                libqt6test6t64             python3.11-dev
  liblua5.2-0                libqt6widgets6t64          python3.11-minimal
  libnghttp3-3               libqt6xml6t64              samba-vfs-modules
Use 'sudo apt autoremove' to remove them.

Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 2128

┌──(nelushi㉿kali)-[~]
└─$ nikto -Version

Nikto 2.5.0 (LW 2.5)
```

Exploitation Tools (Attack & Testing Frameworks)

SQLMap

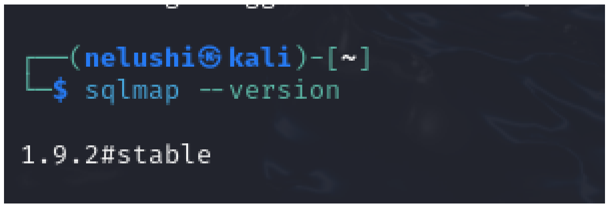
Purpose - An automated SQL Injection tool that helps detect and exploit SQL vulnerabilities.

How it works - It tests different payloads against an application's database to extract data, modify tables, or even gain shell access.

Why it's useful - Saves time when testing for SQL Injection vulnerabilities.

How to install:

sudo apt install sqlmap



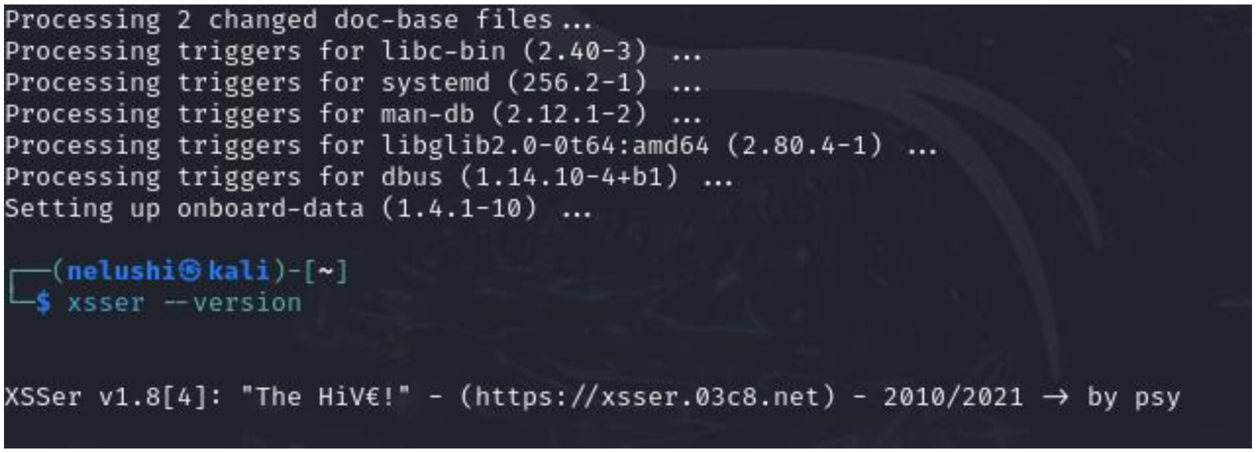
```
┌──(nelushi㉿kali)-[~]
└─$ sqlmap --version

1.9.2#stable
```

XSSer

Purpose - A tool specifically designed to detect and exploit Cross-Site Scripting vulnerabilities.

How it works - It injects malicious scripts into web forms, URLs, or input fields to test whether an application is vulnerable to XSS attacks.

Why it's useful - Helps security researchers confirm reflected, stored, and DOM-based XSS vulnerabilities.



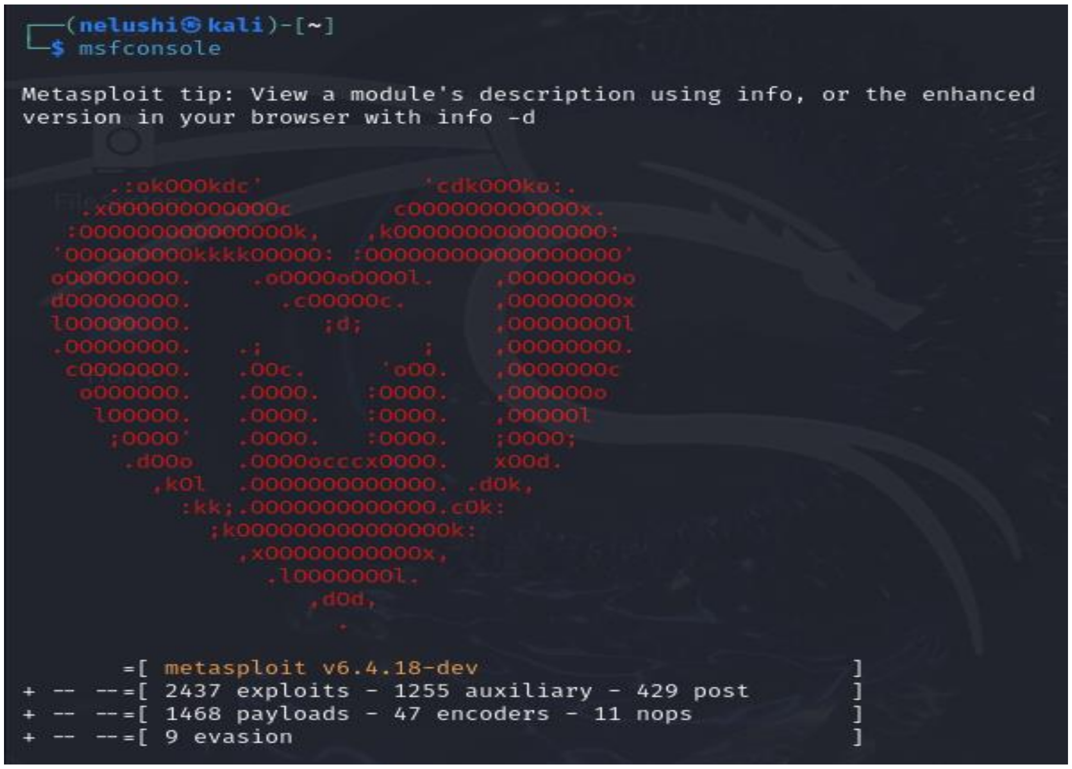
```
Processing 2 changed doc-base files...
Processing triggers for libc-bin (2.40-3) ...
Processing triggers for systemd (256.2-1) ...
Processing triggers for man-db (2.12.1-2) ...
Processing triggers for libglib2.0-0t64:amd64 (2.80.4-1) ...
Processing triggers for dbus (1.14.10-4+b1) ...
Setting up onboard-data (1.4.1-10) ...

┌──(nelushi㉿kali)-[~]
└─$ xsser --version


XSSer v1.8[4]: "The HiV€!" - (https://xsser.03c8.net) - 2010/2021 → by psy
```

Metasploit

Purpose - A penetration testing framework that provides pre-built exploits for various security vulnerabilities.

How it works - It allows security testers to launch attacks against known vulnerabilities, providing payloads that can compromise systems.

Why it's useful - Helps validate the severity of discovered vulnerabilities and simulate real-world cyberattacks.



```
┌──(nelushi㉿kali)-[~]
└─$ msfconsole

Metasploit tip: View a module's description using info, or the enhanced
version in your browser with info -d

      .:okOOOkdc'           'cdkOOOko:.
    .xOOOOOOOOOOOOc       cOOOOOOOOOOOOx.
   :OOOOOOOOOOOOOOOk,   ,kOOOOOOOOOOOOOOO:
  'OOOOOOOOOkkkkOOOOO: :OOOOOOOOOOOOOOOOOO'
  oOOOOOOOO.    .oOOOOoOOOOl.    ,OOOOOOOOo
  dOOOOOOOO.      .cOOOOOc.      ,OOOOOOOOx
  lOOOOOOOO.         ;d;         ,OOOOOOOOl
  .OOOOOOOO.   .;           ;    ,OOOOOOOO.
   cOOOOOOO.   .OOc.     'oOO.   ,OOOOOOOc
    oOOOOOO.   .OOOO.   :OOOO.   ,OOOOOOo
     lOOOOO.   .OOOO.   :OOOO.   ,OOOOOl
      ;OOOO'   .OOOO.   :OOOO.   ;OOOO;
       .dOOo   .OOOOocccxOOOO.   xOOd.
         ,kOl  .OOOOOOOOOOOOO. .dOk,
           :kk;.OOOOOOOOOOOOO.cOk:
             ;kOOOOOOOOOOOOOOOk:
               ,xOOOOOOOOOOOx,
                 .lOOOOOOOl.
                    ,dOd,
                      .

       =[ metasploit v6.4.18-dev                          ]
+ -- --=[ 2437 exploits - 1255 auxiliary - 429 post       ]
+ -- --=[ 1468 payloads - 47 encoders - 11 nops           ]
+ -- --=[ 9 evasion                                       ]
```

Challenges faced during Bug Bounty Hunting

Insufficient familiarity with bug bounty programs and resources

- **Challenge:** Knowing the basics of bug bounty schemes and selecting appropriate vulnerability testing tools.
- **Resolution:** I looked at some online tutorials and platforms like HackerOne and Bugcrowd and looked into online cybersecurity communities. I came across some essential tools like Burp Suite, OWASP ZAP, and Sublist3r, and I developed a bug-hunting technique and learnt something useful.

Learning OWASP ZAP for automated vulnerability scans

- **Challenge:** learning about OWASP ZAP and how it can be used to automate vulnerability scans.
- **Resolution:** concentrated on setting up passive and active scans, AJAX spider scans, and went over the official OWASP ZAP documentation and discovered vulnerabilities for the bug bounty reports after learning how to analyze the results.

Order and arranging bug bounty reports

- **Challenge:** Writing detailed and well-structured bug bounty reports that convey vulnerabilities clearly.
- **Resolution:** examined bug bounty community recommended practices and report templates. Practiced writing reports, making sure to include technical analysis, possible security impact, and clear reproduction instructions. With every submission, the reporting procedure was improved.

Identifying Real Vulnerabilities
Differentiating between genuine security issues and normal application behavior needed patience and experience.

Bypassing Security Filters
Encountered challenges when trying to exploit vulnerabilities due to modern protections like WAFs.

Handling Rejections
Some reports were marked as duplicates, informational, or out-of-scope, which was initially discouraging.

Technical Limitations
Limited knowledge of certain tools or advanced attack techniques slowed down the early stages of testing.

Time Management
Balancing bug bounty hunting with academic and personal commitments required better time organization.