



Name(s): 1. DANIEL BIN ALI 2. SHAHIR AIMAAN BIN SHAHRUNISAM 3. M. ZAIDI FAHMI BIN ZAINUDIN		
ID Number(s): 1. AM2311015360 2. AM2311015363 3. AM2311015290		
Lecturer Name: Sir AKMAL	Lab group / Tutorial group / Tutor (if applicable): SECTION 01	
Course Name and Course Code: ETHICAL HACKING NWC4233	Submission Date:	
Assignment No. / Title Proposal: GROUP PROJECT	Extension & Late Submission: *Allowed / Disallowed	
Assignment Type: Group Project Assignment	% of Assignment Mark:	Returning Date:
Penalties: 1. 10% of the original mark will be deducted for every one-week period after the submission date. 2. No work will be accepted after two weeks of the deadline. 3. If you were unable to submit the coursework on time due to extenuating circumstances, you may be eligible for an extension. 4. Extension will not exceed one week.		
Declaration: We the undersigned confirm that we have read and agree to abide by these regulations on plagiarism and cheating. We confirm that this work is our own. We consent to appropriate storage of our work for checking to ensure that there is no plagiarism/academic cheating.		
<div style="display: flex; justify-content: space-between; align-items: flex-end;"> <div style="text-align: center;"> Signature(s):<i>Daniel</i>..... Full Name: (DANIEL BIN ALI) </div> <div style="text-align: center;"> ...<i>Shahir</i>.... (SHAHIR AIMAAN) </div> <div style="text-align: center;"> ...<i>Zaidi</i>..... (M. ZAIDI FAHMI) </div> </div>		
This section may be used for feedback or other information		



Ethical Hacking Pledge for Student Project

I, M. Zaid Fahmi Bin Zainudin pledge to conduct my ethical hacking project with the utmost integrity and adherence to ethical guidelines. I understand the importance of responsible and legal hacking practices, and I commit to upholding the following principles throughout the duration of my project:

· **Respect for Laws and Regulations:** I will strictly adhere to all local, national, and international laws and regulations governing computer security and hacking activities. I will not engage in any activity that is illegal or unauthorized.

Non-Disclosure: I will treat all information obtained during my testing as confidential and proprietary. I will not disclose any sensitive information, vulnerabilities, or data breaches to unauthorized parties.

Responsible Disclosure: In the event that I discover a security vulnerability, I will follow responsible disclosure practices. I will promptly report the vulnerability to the appropriate stakeholders and provide them with a reasonable amount of time to address the issue before making it public.

Limited Scope: I will only test the systems and networks explicitly defined within the scope of my project. I will not attempt to access or manipulate any systems outside of this scope.

No Harm: I will not intentionally or maliciously damage, disrupt, or compromise the availability, integrity, or confidentiality of any systems, networks, or data. My actions will be focused on identifying and mitigating vulnerabilities.

Documentation: I will maintain accurate and detailed records of my testing activities, methodologies, findings, and communications with stakeholders. This documentation will help demonstrate my commitment to ethical practices.

Continuous Learning: I will stay informed about the latest ethical hacking practices, security trends, and relevant laws. I will continuously improve my knowledge and skills to ensure the highest level of professionalism.

Education and Awareness: I will use the knowledge gained from my project to educate others about the importance of ethical hacking, responsible disclosure, and cybersecurity best practices.

Accountability: I understand that my actions reflect not only on myself but also on the ethical hacking community as a whole. I will hold myself accountable for maintaining the highest standards of ethics and professionalism.

By signing this pledge, I commit to conducting my student project with integrity, responsibility, and respect for ethical hacking principles.

Signature: *Zaidi*

Date: 10/04/2025

**to be printed on the first page of the report after cover page*

Table of contents

Table of contents	4
Chapter 1	5
Introduction to The Project	5
Plan for Penetration Testing	5
Overview of the Target	6
Overview of the Target Architecture	6
Review of Possible Threats	6
Chapter 2	7
[Reconnaissance]	7
Introduction	8
Implementation of Reconnaissance	10
1. DNS Information Gathering	10
2. Web Technology Fingerprinting	12
3. Port Scanning	13
Countermeasures to Reconnaissance	15
Chapter 3	16
[Scanning]	16
Introduction of Scanning	17
Implementation of Scanning	18
Port and Service Scanning	18
1. Nmap Scan – Port & Service Detection	18
2. Web Technology Detection (WhatWeb)	21
3. Vulnerability Scanning (Nikto)	22
Countermeasures to Scanning and Vulnerabilities	25
Chapter 4	27
[Gaining Access]	27
Introduction of Gaining Access	28
Implementation of Gaining Access	29
Analyze and Justification	35
Countermeasures	37
Conclusion	39

Chapter 1

Introduction to The Project

Penetration testing, often known as ethical hacking, is a simulated cyberattack on a computer system, network, or online application with the goal of detecting and correcting any security weaknesses before they are exploited by bad actors. It is a proactive security evaluation that reflects real-world attackers' tactics, methods, and procedures (TTPs). The primary purpose is to identify holes in the system's defenses and deliver actionable insights to improve its overall security posture.

This testing procedure is usually divided into several phases: reconnaissance, scanning, exploitation, post-exploitation, and reporting. Each phase is crucial in simulating the attacker's path and learning how a true compromise could occur. For this assignment, we will perform a penetration test on a specific online application, investigating its components and interactions to determine its vulnerabilities and resilience to attacks. The results of this penetration test will help to improve security and risk mitigation methods.

Plan for Penetration Testing

1. **Reconnaissance** – Gather public and internal information about the target using passive and active methods to identify potential attack vectors.
2. **Scanning** – Perform network and web scanning to discover live hosts, open ports, running services, and potential vulnerabilities.
3. **Gaining Access** – Use exploitation tools and techniques to gain unauthorized access to the target based on information collected during the reconnaissance and scanning phases.

Overview of the Target

The chosen target: <http://megamegapom.free.nf/> is the first web application link provided in the project documentation. This application is a publicly accessible website used for demonstration and educational purposes in cybersecurity.

Overview of the Target Architecture

The target is a typical client-server architecture:

- **Frontend:** A web-based interface accessible via browser.
- **Backend:** A server that handles business logic and processes requests.
- **Database:** Stores application data such as user credentials, product data, and logs.
- **Technology stack:** The application appears to be developed using common web technologies like HTML, JavaScript, PHP/Python, and MySQL.

Review of Possible Threats

Threats were reviewed using the **STRIDE** threat modeling framework:

- **Spoofing:** Attackers may impersonate legitimate users.
- **Tampering:** Potential to alter data using injection attacks.
- **Repudiation:** Insufficient logging may allow malicious users to deny actions.
- **Information Disclosure:** Sensitive information may be exposed through misconfigured services or directories.
- **Denial of Service (DoS):** Overloading the server with traffic may cause unavailability.
- **Elevation of Privilege:** Attackers may attempt to gain higher access rights than intended.

Chapter 2

[Reconnaissance]

[Muhammad Zaidi Fahmi Bin Zainudin AM2311015290]

Introduction

Reconnaissance is the initial and most fundamental step of a penetration test, in which important information about the target system is gathered. This step is critical for developing an informed assault approach by analyzing the target's infrastructure, prospective entry points, and technologies. The reconnaissance method seeks to gather as much information as possible while remaining undiscovered, especially in the passive approach. It acts as a road map for the next steps of scanning and exploitation, decreasing guesswork and boosting the likelihood of a successful and efficient test.

Passive Reconnaissance

Passive reconnaissance involves collecting information from public sources without directly interacting with the target. It uses methods like OSINT to stay stealthy and avoid detection, helping build a basic profile of the target before any active probing.

Examples of passive recon tools:

- [nslookup](#) – DNS record lookup
- [theHarvester](#) – Collects emails, subdomains, etc.
- [WhatWeb](#) – Identifies web technologies
- [crt.sh](#) – Retrieves subdomains from certificate logs
- [whois](#) – Domain registration info

Active Reconnaissance

Active reconnaissance involves directly interacting with the target system to gather detailed technical information. This method can trigger security alerts and is usually done after passive recon to confirm or expand on initial findings.

Examples of active recon tools:

- [Nmap](#) – Port and service scanning
- [Wappalyzer](#) – Web technology detection (browser extension or CLI)
- [wafw00f](#) – Detects Web Application Firewalls
- [Nikto](#) – Web server vulnerability scanner
- [DNSrecon](#) – Performs DNS enumeration with zone transfer attempts

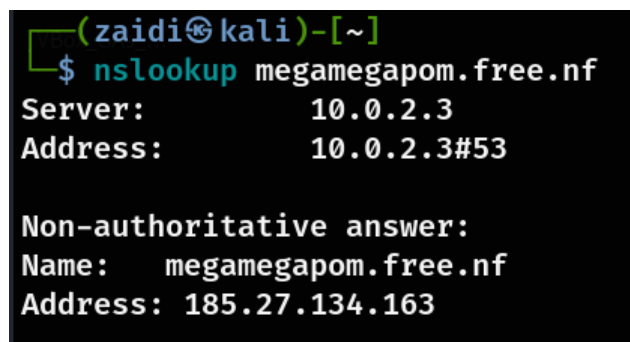
For this project, reconnaissance will be conducted to gather preliminary information about the target environment. This method involves collecting data from publicly accessible sources without directly engaging with the target systems in a way that would trigger security alerts. Tools such as nslookup will be used to retrieve DNS records and domain-related information, while WhatWeb will help identify technologies used by web applications, including server details and frameworks. In addition, Nmap will be utilized to perform port scanning to identify open ports and the services running on them, which can provide deeper insights into the target's attack surface. These tools collectively offer valuable information while maintaining a relatively low profile, helping to develop an initial understanding of the target without direct intrusion.

Implementation of Reconnaissance

1. DNS Information Gathering

For the purposes of this assessment, the target website was hosted at the subdomain: `megamegapom.free.nf`. The organization utilizes a **free web hosting provider** rather than a private infrastructure, which limits the availability of WHOIS data or DNS records for subdomains. The intent was to simulate an external adversary perspective with no internal knowledge of the infrastructure.

To begin identifying the potential attack surface, we used DNS resolution tools to extract information about the subdomain. Using the `nslookup` utility, we resolved the domain to its associated IP address (Figure 1).



```
(zaidi@kali)-[~]
$ nslookup megamegapom.free.nf
Server:      10.0.2.3
Address:     10.0.2.3#53

Non-authoritative answer:
Name:   megamegapom.free.nf
Address: 185.27.134.163
```

Figure 1: Resolving the IP address of megamegapom.free.nf using nslookup.

With the IP address identified, we then performed an IP information lookup to gather metadata about the infrastructure. The IP `185.27.134.163` is associated with the hosting provider I FastNet LTD, located in Lincoln, England (Figure 2). The IP is hosted in a datacenter and falls under ASN 34119, which is commonly used by free hosting providers such as InfinityFree.

This information suggests that the subdomain is hosted on a shared hosting environment, where multiple sites share the same server and IP. As such, deeper enumeration or infrastructure mapping (e.g., zone transfers) is restricted and likely sandboxed by the provider.

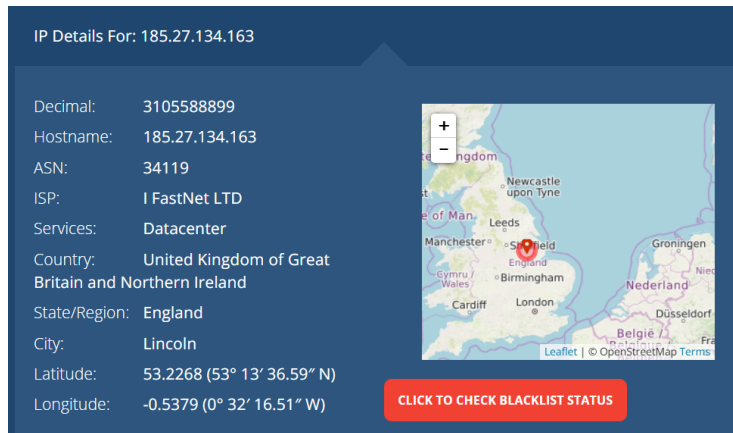


Figure 2: IP geolocation and metadata information for 185.27.134.163 using whatismyipaddress.com

To expand DNS enumeration efforts, a `nslookup -type=NS` query was issued against `megamegapom.free.nf`. While the response included some authoritative information, the lack of a direct answer for the NS record query indicates that the domain's DNS settings are either misconfigured or intentionally restrictive due to the nature of shared hosting.

The output also revealed that the origin of the DNS zone is hosted by `ns1.infinityfree.com`, further confirming that the target utilizes InfinityFree's managed infrastructure.

```
(zaidi@kali)-[~]
$ nslookup -type=NS megamegapom.free.nf
Server:      10.0.2.3
Address:     10.0.2.3#53

Non-authoritative answer:
*** Can't find megamegapom.free.nf: No answer

Authoritative answers can be found from:
megamegapom.free.nf
  origin = ns1.infinityfree.com
  mail addr = support.megamegapom.free.nf
  serial = 2006112402
  refresh = 28800
  retry = 7200
  expire = 604800
  minimum = 86400
```

Figure 3: Attempting to query NS records and SOA for `megamegapom.free.nf`

This information may help in timing subsequent DNS queries or understanding caching behavior. The results demonstrate a classic limitation of external enumeration against domains hosted under free providers — while certain DNS metadata (like SOA) is visible, actual zone transfers, subdomain enumeration, and in-depth NS record mapping are either restricted or entirely disabled. As such, attackers would likely shift focus toward probing the resolved IP directly or conducting application-layer enumeration on the hosted content.

2. Web Technology Fingerprinting

Web technology fingerprinting was used to determine the underlying technologies utilized by the target website. This stage is critical throughout the information-gathering portion of a security assessment because it provides insight into the server software, frameworks, and client-side technologies being used. Understanding these components aids in the identification of potential vulnerabilities and the selection of appropriate tools and procedures in subsequent testing phases.

The WhatWeb tool was used to identify the web server's technology. WhatWeb is a popular reconnaissance tool for online application assessments that uses passive fingerprinting. It operates by evaluating a target website's HTTP headers, HTML text, and other response metadata without interacting in a way that could interrupt the server or cause security alerts.



```
(zaidi@kali)-[~]  
$ whatweb http://megamegapom.free.nf  
http://megamegapom.free.nf [200 OK] Country[UNITED KINGDOM][GB], HTTPServer[openresty], IP[185.27.134.163], Script[text/javascript]
```

**Figure 4: Web Technology Fingerprinting using WhatWeb on
megamegapom.free.nf**

The results show that the web server is OpenResty, a dynamic web platform built on nginx that is often used to manage high-performance online applications. This information can be used to focus additional inquiry on the platform's recognized flaws.

The resolved IP address (185.27.134.163) is consistent with the DNS lookup results, indicating that the findings are reliable. Furthermore, the use of JavaScript implies the presence of client-side interactions, which might be investigated further for exposed logic or vulnerabilities. The server location is identified to be the United Kingdom, which is consistent with earlier IP intelligence. This information may also be useful in understanding hosting policies, geographical compliance, and regulatory limits.

3. Port Scanning

Port scanning was conducted to identify open ports and services running on the target web server, megamegapom.free.nf (185.27.134.163). This step is a fundamental part of the information-gathering phase in a security assessment. By discovering which ports are open and what services are running behind them, security analysts can map the server's potential attack surface and determine the next steps for vulnerability enumeration or exploitation.

The Nmap tool was utilized for this process. Nmap is one of the most powerful and widely used tools in penetration testing for host discovery and service enumeration. It was executed with different flags to extract various levels of information from the target server.

```
(zaidi@kali)-[~]
$ nmap -sS megamegapom.free.nf
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-10 12:28 +08
Nmap scan report for megamegapom.free.nf (185.27.134.163)
Host is up (0.0036s latency).
Not shown: 961 filtered tcp ports (no-response), 37 closed tcp ports (reset)
PORT      STATE SERVICE
80/tcp    open  http
443/tcp    open  https

Nmap done: 1 IP address (1 host up) scanned in 88.49 seconds

(zaidi@kali)-[~]
$ nmap -sS -sV megamegapom.free.nf
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-10 12:31 +08
Nmap scan report for megamegapom.free.nf (185.27.134.163)
Host is up (0.0088s latency).
Not shown: 946 filtered tcp ports (no-response), 52 closed tcp ports (reset)
PORT      STATE SERVICE  VERSION
80/tcp    open  tcpwrapped
443/tcp    open  tcpwrapped

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 66.90 seconds
```

Figure 5: Port Scanning with Nmap on megamegapom.free.nf

The first scan used the `-sS` flag for a TCP SYN (stealth) scan, which quickly probes ports without completing the TCP handshake. The results showed that port 80 (HTTP) and port 443 (HTTPS) were open. This indicates that the target is running web services over both HTTP and HTTPS protocols.

In the second scan, the `-sS -sV` flags were used. The `-sV` flag enables service version detection, which tries to determine the exact service and version running on each open port. However, in this case, both ports returned `tcpwrapped`, which means that the server is configured to block or restrict version detection.

The tcpwrapped results suggest that the server is using defensive measures like TCP wrappers or firewall rules to block detailed service detection. Despite this, the host was responsive with low latency, confirming it's active. The large number of filtered ports indicates the presence of a firewall, reducing exposure to basic scans and highlighting the need for more advanced enumeration techniques.

```

[~](zaidi@kali)-[~]
└─$ openssl s_client -connect megamegapom.free.nf:443
Connecting to 185.27.134.163
CONNECTED(00000003)
405702ED8E7F0000:error:0A000438:SSL routines:ssl3_read_bytes:tlsv1 alert internal error:../ssl/record/rec_layer_s3.c:908:SSL alert number 80
no peer certificate available
No client certificate CA names sent
SSL handshake has read 7 bytes and written 466 bytes
Verification: OK
New, (NONE), Cipher is (NONE)
Protocol: TLSv1.3
This TLS version forbids renegotiation.
Compression: NONE
Expansion: NONE
No ALPN negotiated
Early data was not sent
Verify return code: 0 (ok)

```

Figure 6: Open SSL Scan

The SSL/TLS inspection using OpenSSL revealed critical issues with the target server's security configuration, despite a failed handshake. The server is confirmed to be listening on port 443, as shown by the initial "CONNECTED" status, but it abruptly terminated the handshake with a TLSv1 "internal error" (code 80), indicating misconfiguration or strict security controls.

While the server claims to support TLS 1.3, no cipher suite was successfully negotiated ("cipher is (NONE)"), suggesting a broken TLS stack or intentional blocking of standard negotiation. No peer certificate was exchanged, possibly due to missing certificate configuration or a strict policy against unauthenticated connections.

Diagnostic output identified the failure point in the `ssl3_read_bytes` function and noted the amount of data exchanged. Although the verification result was "OK," the handshake still failed, pointing to issues with the server's TLS implementation rather than certificate validation. Overall, the server's HTTPS service is reachable but unable to establish secure connections, posing a significant security risk.

Countermeasures to Reconnaissance

During the reconnaissance phase, several tools were utilized to gather information about the target system, including nslookup, whatweb, nmap, and OpenSSL scanning. Each tool revealed different types of information, which can be useful for an attacker to plan further exploitation. To mitigate these risks, appropriate countermeasures must be implemented to strengthen the system's security posture.

- 1. Restrict DNS Zone Transfers.**

DNS zone transfers allow replication of DNS databases between DNS servers. If unrestricted, attackers can retrieve the full list of hostnames and IP addresses for a domain, giving them a detailed map of your network. By configuring DNS servers to allow zone transfers only to specific, trusted IP addresses (usually internal secondary DNS servers), you can prevent tools like [nslookup](#) from harvesting sensitive DNS data. This minimizes the information an attacker can gather about internal systems or services.

- 2. Hide or Modify HTTP Response Headers.**

Web servers often expose detailed information such as software type, version (e.g., [openresty](#)), and language frameworks in HTTP headers. Tools like [whatweb](#) utilize this metadata to fingerprint the underlying technology stack. By modifying server settings (e.g., in Nginx, Apache, or OpenResty), you can suppress or alter these headers—such as [Server](#), [X-Powered-By](#), and [Via](#)—to limit what reconnaissance tools can detect. This prevents attackers from tailoring their exploits to known vulnerabilities in those software versions.

- 3. Close Unused Ports and Limit Service Exposure.**

During a port scan with tools like [nmap](#), any open ports and associated services (e.g., FTP, SSH, HTTP) are identified, which helps attackers find potential entry points. You should conduct internal scans to identify which ports are open and close any that are unnecessary using firewalls or host-based filtering tools (like [iptables](#) or Windows Firewall). Additionally, for services that must remain open, restrict access to specific IPs or networks. This significantly reduces the available surface for attack.

- 4. Use Strong SSL/TLS Configurations and Disable Weak Protocols.**

OpenSSL scanning can detect expired certificates, weak ciphers, and support for outdated SSL/TLS protocols, all of which are exploitable. To counter this, configure your web server to support only strong cipher suites and modern protocols (TLS 1.2 or TLS 1.3), and disable deprecated ones like SSLv2, SSLv3, and early versions of TLS. You should also enforce HTTP Strict Transport Security (HSTS) to mandate encrypted connections. This protects sensitive communications from interception and reduces the risk of downgrade or MITM attacks.

Conclusion

The Gaining Access phase represents one of the most critical stages in the penetration testing process, where identified vulnerabilities are actively exploited to assess the security posture of a system. Through practical demonstrations such as Brute Force, Cross-Site Scripting (XSS), SQL Injection (SQLi), and Metasploit-based exploits, we highlighted how common weaknesses can be leveraged to compromise target systems.

Despite encountering some blocked attempts due to security controls like WAFs and input validation, these simulations underscore the importance of continuous security assessments and proactive defense mechanisms. The findings demonstrate that even simple attacks, if left unmitigated, can pose a serious threat to web applications and networked systems.

Ultimately, this phase reinforced the value of ethical hacking in identifying real-world risks, helping organizations patch vulnerabilities before they are exploited by malicious attackers. By implementing proper countermeasures such as strong input validation, secure authentication protocols, regular patching, and intrusion detection systems, organizations can significantly reduce their exposure to unauthorized access attempts.