Program: Cybersecurity

Course: Hardening Modern Ops Sys App – Midterm Project

Submitted by:

Alcides Gomez

Leoj Muyco

Luis Landa

Mohammad Kaif

To:

Professor Travis Czech

Introduction

Focusing on online transactions and financial consulting, Swifty Financial Solutions is a rapidly growing financial services company. After experiencing data breaches and financial losses due to cyberattacks, Swifty Financial hired our team to strengthen its virtual network infrastructure, enforce strict monitoring, and enhance its security measures against potential threats.

Our project entails identifying and fixing vulnerabilities in the Metasploitable 3 virtual machine, conducting simulated attacks, installing and configuring the Wazuh server and agent for comprehensive monitoring, and monitoring network traffic to verify the effectiveness of our security measures in order to address these security concerns. This report describes the actions that Swifty Financial Solutions took to improve its security posture as well as the remediations that came from our evaluations and interventions.

Objectives

The main goal is to strengthen the organization's security posture through a comprehensive penetration test and vulnerability assessment. There are several phases involved in this, all of which are meant to detect and address potential vulnerabilities. The primary goals of this evaluation are as follows:

1. **Set up Wazuh Agent**

   Installing the Wazuh agent on every VirtualBox virtual machine will allow for thorough security incident monitoring and alerting.

2. **Identify and secure the Vulnerabilities**

   Pay attention to Metasploitable 3, figuring out and fixing at least four vulnerabilities.

3. **Simulate Attacks and monitor traffic**

   Utilize the fortified Metasploitable 3 system to launch simulated attacks, and utilize Wireshark to record and examine malicious communications.

4. **Prepare an Incident report**

   Create a thorough report that is documented.

Scope

In order to guarantee a thorough approach to strengthening the security of Swifty Financial Solutions' virtual network architecture, the project's scope includes the following tasks and elements:

1. **Installation and Configuration of Security Tools**
   - To enable centralized security monitoring and incident detection, deploy and install the Wazuh server and Wazuh agents among all VirtualBox virtual machines.

2. **Vulnerability Identification and Assessment**
   - To find security flaws in the Metasploitable 3 virtual machine, use vulnerability scanning tools like OpenVAS, Nessus, Nikto, and OWASP.
   - Concentrate on locating a minimum of four significant vulnerabilities that an attacker might be able to exploit.

3. **Fixing the Vulnerabilities That Have Been Found**
   - Create and put into action corrective measures to resolve the vulnerabilities found in Metasploitable 3.
   - To verify that security vulnerabilities have been resolved, make sure the remediation procedures are carefully tested and validated.

4. **Controlled simulated attacks and Traffic Tracking**
   - Evaluating the efficacy of the installed security measures, conduct controlled simulated attacks on the hardened Metasploitable 3 system.
   - Utilize tools such as Wireshark and other network monitoring programs to record and examine network traffic in order to spot any indications of malicious activity and confirm the strength of the security measures.

5. **Documentation and Reporting**
   - Create a thorough incident report that includes information on vulnerabilities found, remediation actions taken, and outcomes of traffic analysis and simulated assaults.
   - Provide suggestions for best practices and additional security upgrades to keep a virtual network environment safe.

Wazuh Server and Agent Installation Steps

Wazuh is a free and open-source security platform that unifies XDR and SIEM protection for endpoints and cloud workloads (Wazuh, 2024).

- XDR (extended detection and response) collects and automatically correlates data across multiple security layers – email, endpoint, server, cloud workload, and network. This allows for faster detection of threats and improved investigation and response times through security analysis. (Trend Micro, n,d)
- SIEM (Security information and event management), is a solution that helps organizations detect, analyze, and respond to security threats before they harm business operations. SIEM combines security information management (SIM) and security event management (SEM) into one security management system. SIEM technology collects event log data from various sources, identifies activity that deviates from the norm with real-time analysis, and takes appropriate action (Microsoft, n,d).

To effectively implement Wazuh for centralized security monitoring, the Wazuh server and Wazuh agent are essential components. The Wazuh server is the central hub where all security data is collected, analyzed, and managed. On the other hand, the Wazuh agent is software that must be installed on each endpoint or system you wish to monitor.

The following screenshots will outline how to install Wazuh server:

1. As we are running on VrtualBox hypervisor, we will install Wazuh using Open Virtual Appliance (OVA) file. The Wazuh OVA can be downloaded from this link: https://packages.wazuh.com/4.x/vm/wazuh-4.7.5.ova
2. Once the file has been downloaded, import the OVA and let the installation run.
3. Once installation is complete, on the VM console, login using wazuh-user/wazuh credentials.

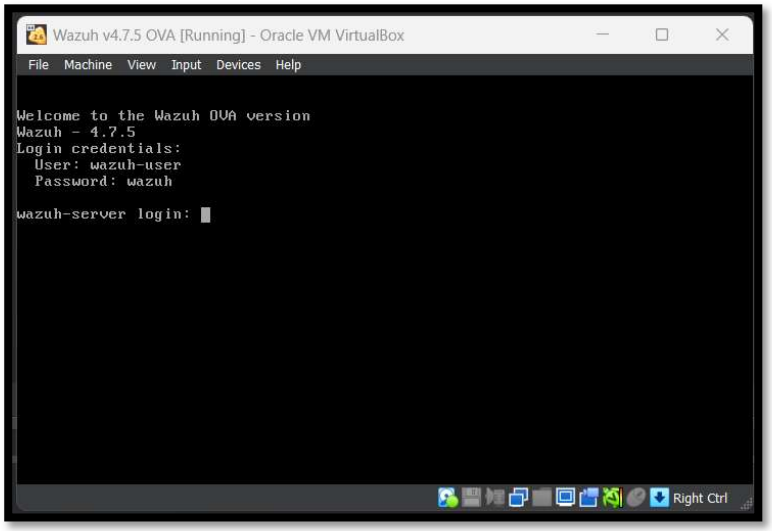


Figure 1. Wazuh VM Console

4. Once logged in, verify the IP address by issuing “ifconfig”. In this case, our server IP is 192.168.1.115 as shown below.

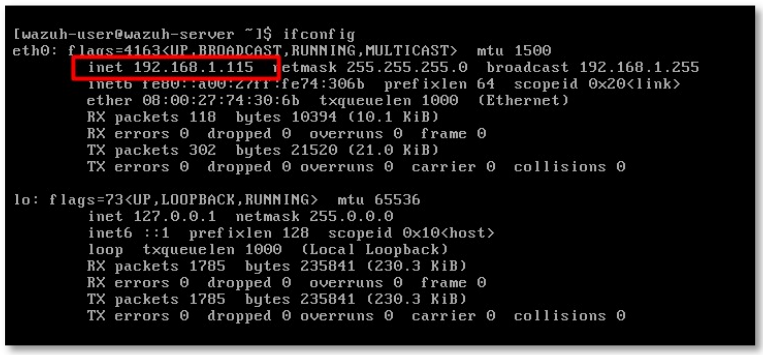


Figure 2. Wazuh IP Address

5. Navigate to Wazuh's Web Portal by opening a web browser and entering the address - https://192.168.1.115/. The default username/password is admin/admin.

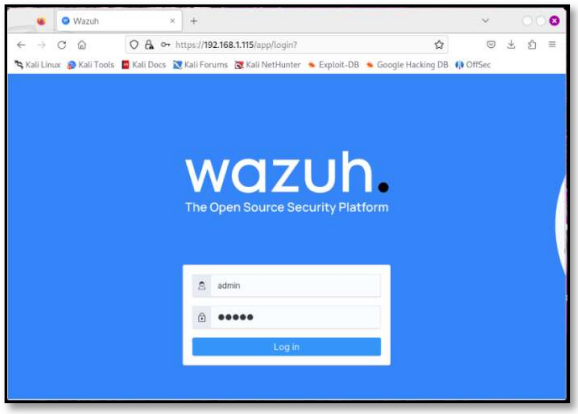


Figure 3. Wazuh Server Web Portal

6. Once logged in, it will display the list of endpoints connected to the Wazuh server.

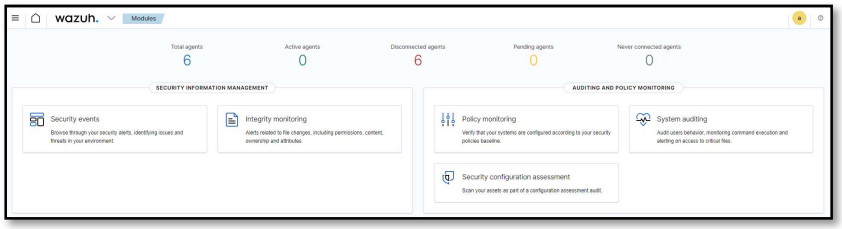


Figure 4. Wazuh Dashboard

7. Under this Wazuh server, we have all VMs connected.

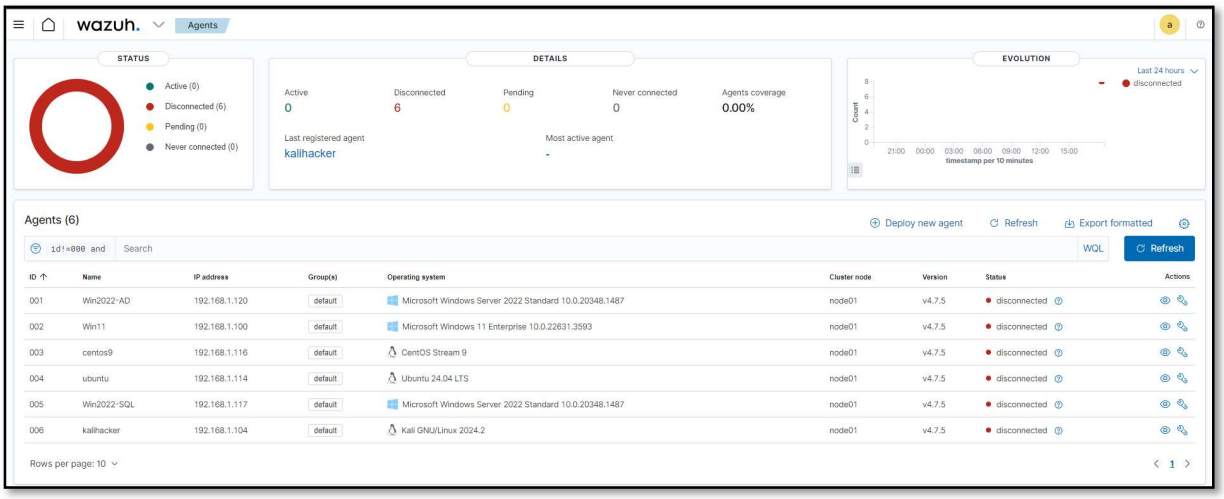


Figure 5. Wazuh Agents

From the screenshots above, all Virtual Machines (VM) except for Metasploitable 3 have already been added to Wazuh server.

The following screenshots will outline how to install Wazuh agent onto Metasploitable 3 host:

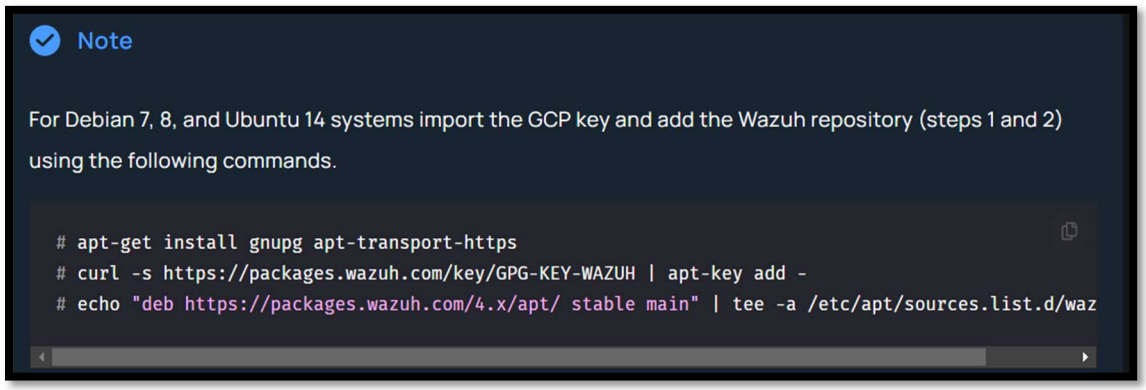


Figure 6. Wazuh Agent Installation Commands

https://documentation.wazuh.com/current/installation-guide/wazuh-agent/wazuh-agent-package-linux.html

The Metasploitable 3 is running on Ubuntu 14 platform, and as indicated on the Wazuh's official installation guide as shown above, we need to perform the commands mentioned.

1. Open and login Metasploitable VM's console in VirtualBox. Default username/password is vagrant/vagrant.

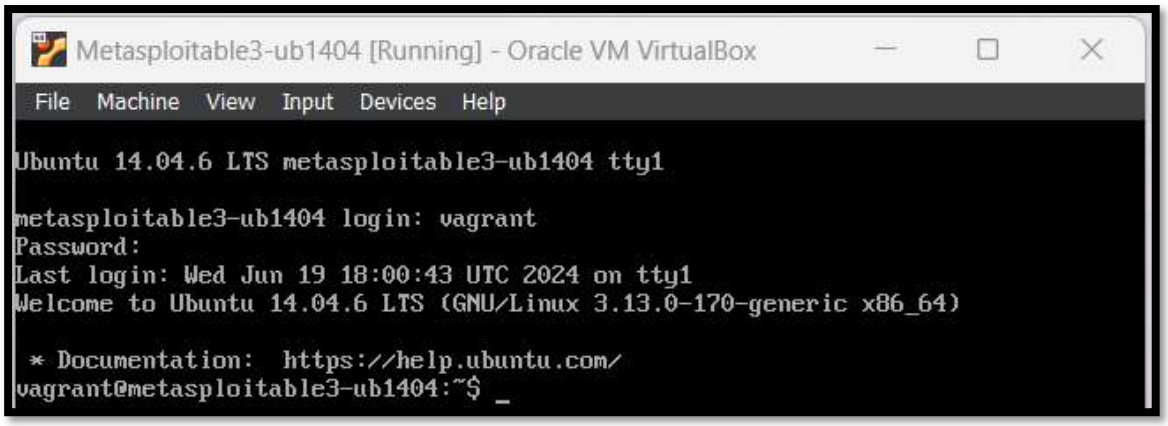


Figure 7. Wazuh VM Console Login

2. To perform the commands, we need to be a super user. Once logged in, enter "sudo su".

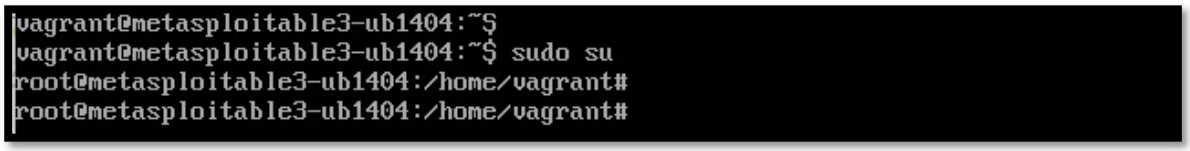


Figure 8. Wazuh Super User

3. Enter the following commands:
    a. apt-get install gnupg apt-transport-https
    b. curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | apt-key add -
    c. echo "deb https://packages.wazuh.com/4.x/apt/ stable main" | tee -a/etc/apt/sources.list.d/wazuh.list

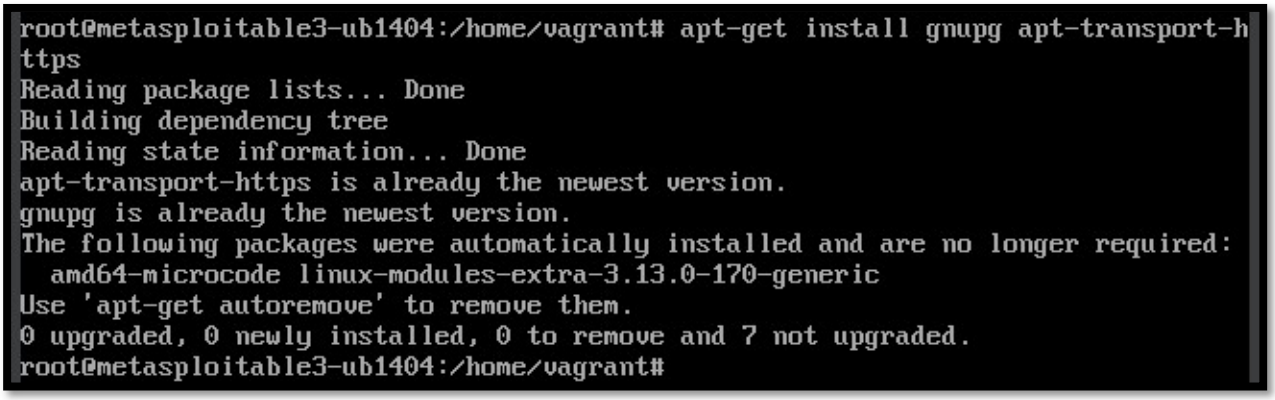


Figure 9. Wazuh 3.a Command

```
root@metasploitable3-ub1404:/home/vagrant# curl -s https://packages.wazuh.com/ke
y/GPG-KEY-WAZUH | apt-key add -
OK
root@metasploitable3-ub1404:/home/vagrant# _
```

Figure 10. Wazuh 3.b Command

```
root@metasploitable3-ub1404:/home/vagrant# echo "deb https://packages.wazuh.com/
4.x/apt/ stable main" | tee -a /etc/apt/sources.list.d/wazuh.list
deb https://packages.wazuh.com/4.x/apt/ stable main
root@metasploitable3-ub1404:/home/vagrant#
```

Figure 11. Wazuh 3.c Command

4. Once the above is done, we can now connect Metasploitable 3 to the Wazuh server. We need to perform the following commands:

    a. wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.7.5-1_amd64.deb
    b. WAZUH_MANAGER='192.168.1.115' dpkg -i ./wazuh-agent_4.7.5-1_amd64.deb
    c. /var/ossec/bin/wazuh-control start

```
root@metasploitable3-ub1404:/home/vagrant# wget https://packages.wazuh.com/4.x/a
pt/pool/main/w/wazuh-agent/wazuh-agent_4.7.5-1_amd64.deb
--2024-06-19 23:09:32--  https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-ag
ent/wazuh-agent_4.7.5-1_amd64.deb
Resolving packages.wazuh.com (packages.wazuh.com)... 13.33.165.51, 13.33.165.123
, 13.33.165.110, ...
Connecting to packages.wazuh.com (packages.wazuh.com)|13.33.165.51|:443... conne
cted.
HTTP request sent, awaiting response... 200 OK
Length: 9378818 (8.9M) [binary/octet-stream]
Saving to: 'wazuh-agent_4.7.5-1_amd64.deb.1'

100%[=====================================>] 9,378,818   2.49MB/s   in 3.5s

2024-06-19 23:09:41 (2.53 MB/s) - 'wazuh-agent_4.7.5-1_amd64.deb.1' saved [93788
18/9378818]
```

Figure 12. Wazuh 4.a Command

```
root@metasploitable3-ub1404:/home/vagrant#
root@metasploitable3-ub1404:/home/vagrant# WAZUH_MANAGER='192.168.1.115' dpkg -i
 ./wazuh-agent_4.7.5-1_amd64.deb
(Reading database ... 128868 files and directories currently installed.)
Preparing to unpack .../wazuh-agent_4.7.5-1_amd64.deb ...
Unpacking wazuh-agent (4.7.5-1) over (4.7.5-1) ...
Setting up wazuh-agent (4.7.5-1) ...
Processing triggers for ureadahead (0.100.0-16) ...
ureadahead will be reprofiled on next reboot
root@metasploitable3-ub1404:/home/vagrant#
```

Figure 13. Wazuh 4.b Command

```
root@metasploitable3-ub1404:/home/vagrant# /var/ossec/bin/wazuh-control start
Starting Wazuh v4.7.5...
Started wazuh-execd...
Started wazuh-agentd...
Started wazuh-syscheckd...
Started wazuh-logcollector...
Started wazuh-modulesd...
Completed.
root@metasploitable3-ub1404:/home/vagrant#
```

Figure 14. Wazuh 4.c Command

Now, we can check from Wazuh server's web portal whether Metasploitable 3 has successfully connected to Wazuh server.

1. Login to Wazuh server: https://192.168.1.115
2. Click the Total Agents and check whether Metasploitable 3 is on the list.

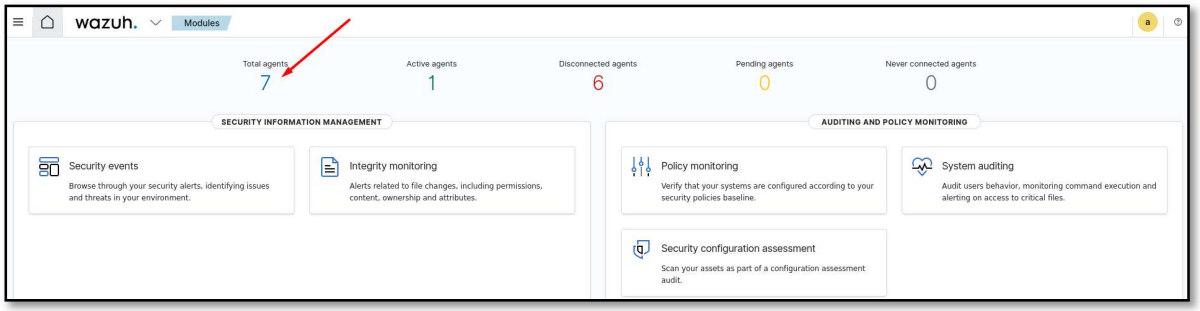


Figure 15. Wazuh Dashboard Portal

3. From the screenshot below, Metasploitable 3 is now on the list of connected agents.

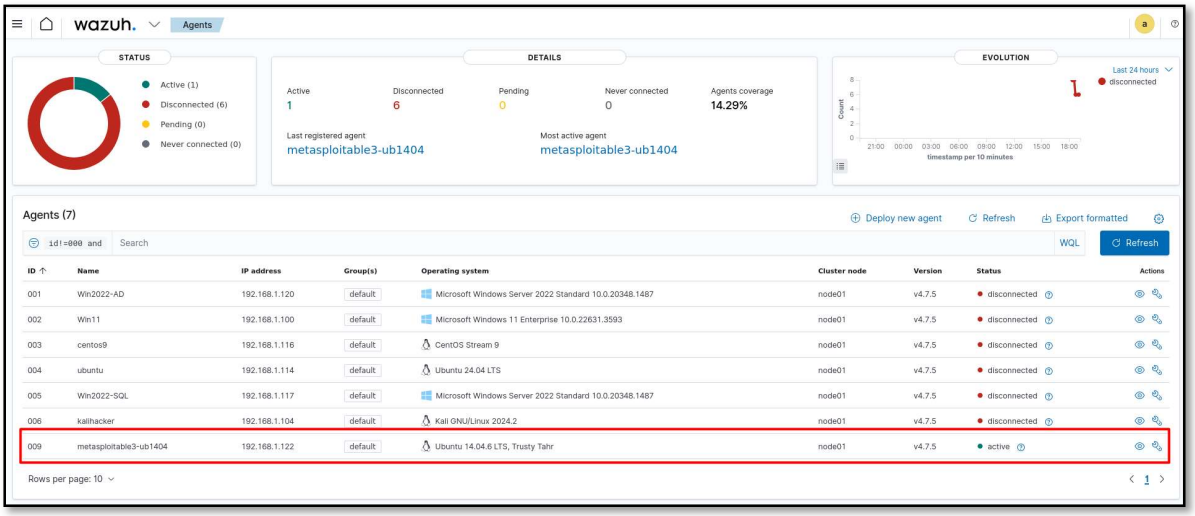


Figure 16. Wazuh Dashboard After Metasploitable has been added

4. We can now monitor and check Metasploitable 3 from our Wazuh server. All VMs are now connected to Wazuh server.

Vulnerability Identification and Remediation

Using vulnerability scanning tools such as OpenVAS, Nessus, Nitko, and OWASP, our team identified the following vulnerabilities in our target host – Metasploitable 3.

S/N	Port	Service	Version	Vulnerability	Sev	CVE/EDB No	Solution
1	21 / TCP	FTP	ProFTPD 1.3.5	Unauthorized users can copy data from one location on the same server to another due to the ProFTPD mod_copy vulnerability	10	CVE-2015-3306 EDB: 37262	- Upgrade to a secure version of ProFTPD. -Turn off the mod_copy module when not in use. - Allow access to administrator IP
2	22 / TCP	SSH	OpenSSH 6.6.1p1	This vulnerability is due to the software being configured with default root credentials. If left unaddressed, it allows unauthorized users to gain full access to the system, potentially enabling them to execute arbitrary code and access sensitive data.	9.8	CVE-2024-22902 EDB: 34993	-Limit access to SSH to a particular IP address only. - Change root credentials
3	80 / TCP	HTTP	Drupal 7.5	Vulnerability allows remote attackers to conduct SQL injection attacks via an array containing crafted keys. The script injects new Drupal administrator user via login form and then it attempts to log in as this user to determine if target is vulnerable. (nmap, nd)	10	CVE-2014-3704	- Convert every HTTP request to HTTPS. -Verify that SSL/TLS certificates are installed correctly.
4	631 / TCP	IPP	CUPS 1.7	Using outdated cipher suites can open HTTPS communications to attacks, according to the vulnerability "SSL/TLS: Report Vulnerable Cipher Suites for HTTPS".	7.5	CVE-2016-2183	- Disable weak cipher suites - Turn off CUPS service if not being used. - Create firewall rule to block incoming traffic to 631

Following are the remediations conducted on the above vulnerabilities found:

1. **Remediating SSH (Port 22)**

As we are aware of the credentials

1. Initiated a secure shell (SSH) connection to the metasploitable 3 by using '*ssh vagrant@192.168.1.113*' and metasploitable credentials.
2. Then we access the root privilages by '*sudo su*'

Figure 17. ssh connection

3. Now we are going to give access to specifi IP address by editing the SSH configuration files.
4. Open the SSH configuration file using suo '*nano /etc/ssh/sshd_config'* to open the ssh congig file.

```
root@metasploitable3-ub1404:/home/vagrant# sudo nano /etc/ssh/sshd_config
root@metasploitable3-ub1404:/home/vagrant#
```

5. Now we created '*AllowUsers vagrant@192.168.1.106'* by using this we allowed Vagrant user to access only from the kali vm, 192.168.1.106 is the ip of the Kali from where we have initaited the SSH session.

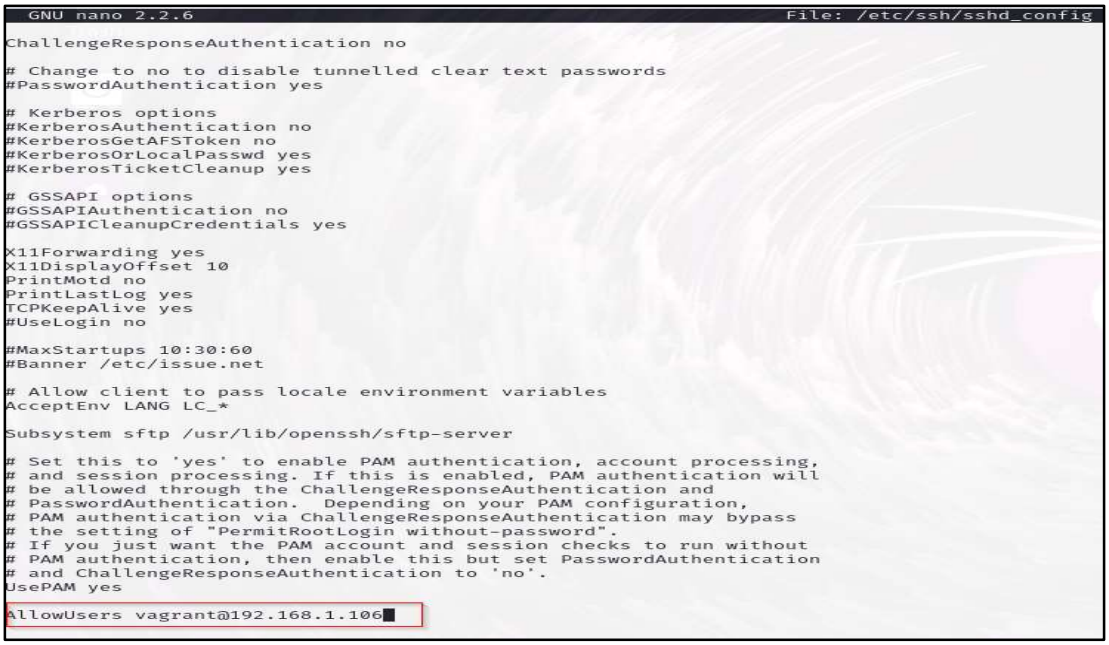


Figure 18. configure sshd file

6. Now we restart the ssh services on the metasploitable 3, just to make sure that the changes apply.

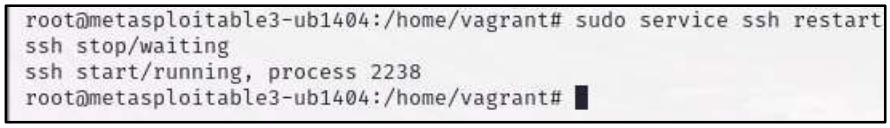


Figure 19. restart ssh services

7. Now check are we able to access to metasploitable 3 remotly through ssh. Seems like we are able to connect

Figure 20. ssh from kali machine

8. Now we will try from windows
9. Her we can see that access is denied which means the vagrnt user can access only from the kali ip not any other IP's

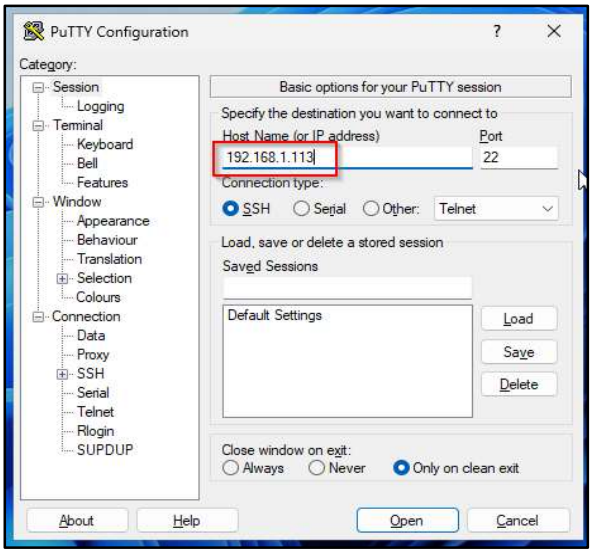


Figure 21. putty windows Part 1

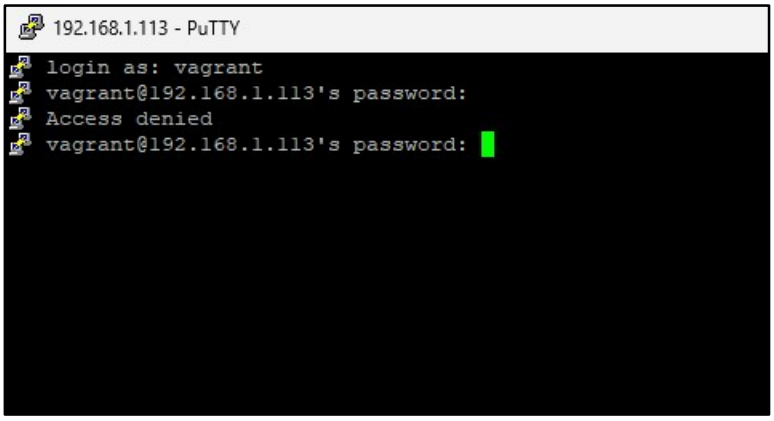


Figure 22. Putty Access denied windows Part 2

2. Remediating HTTP vulnerability on Metasploitable 3

1. Created a SSH session into metasploitable 3 by '*ssh vagrant@192.168.1.113'* . Now we have a remote access to metasploitable 3.

Figure 23. SSH into metasploitable 3

2. Generating the SSL self signed certificate on metasploitable 3
   This stage involved creating a self-signed SSL certificate for Metasploitable 3's Apache web server. The certificate contains a 2048-bit RSA key and has a 265-day validity period.

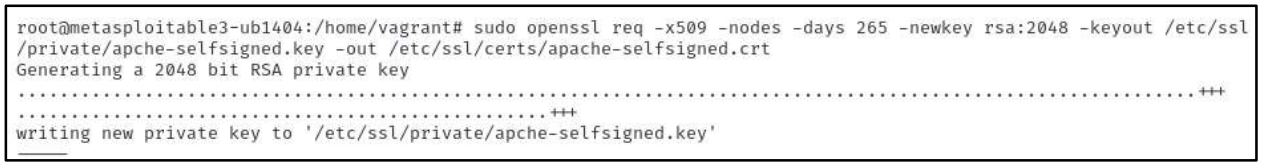


Figure 24. Generating SSL self-signed certificate

3. Filling the information for the certificate
   In this stage, you provide the data required to create a self-signed SSL certificate. In order to ensure that clients connecting to the server can authenticate it, the information entered helps uniquely identify the server and its owner. In order to establish secure communication via HTTPS, this information is essential.

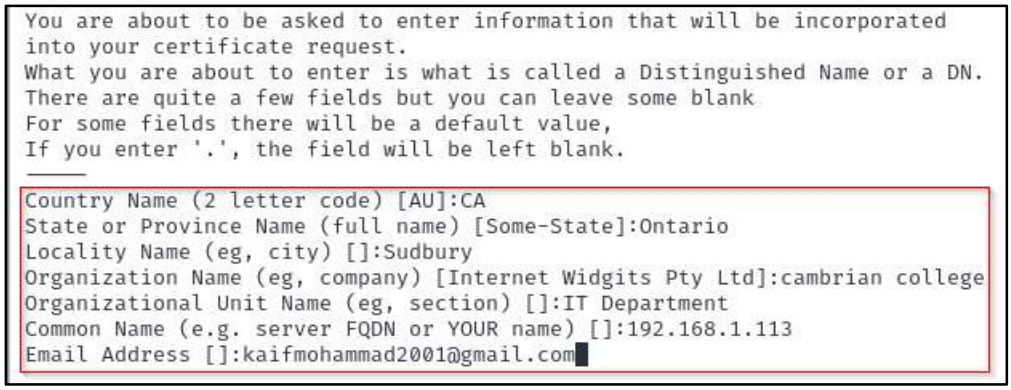


Figure 25. Filling info. for the certificate

4. Configure the default-ssl.config file
    - Virtual host that listens on port 443 is defined here.
    - indicates the file that contains the access and error messages linked to this virtual host.
    - For this virtual host, turns on SSL.
    - Both the private key and SSL certificate are mentioned.
    - Security is improved through the configuration of the protocol and cipher.

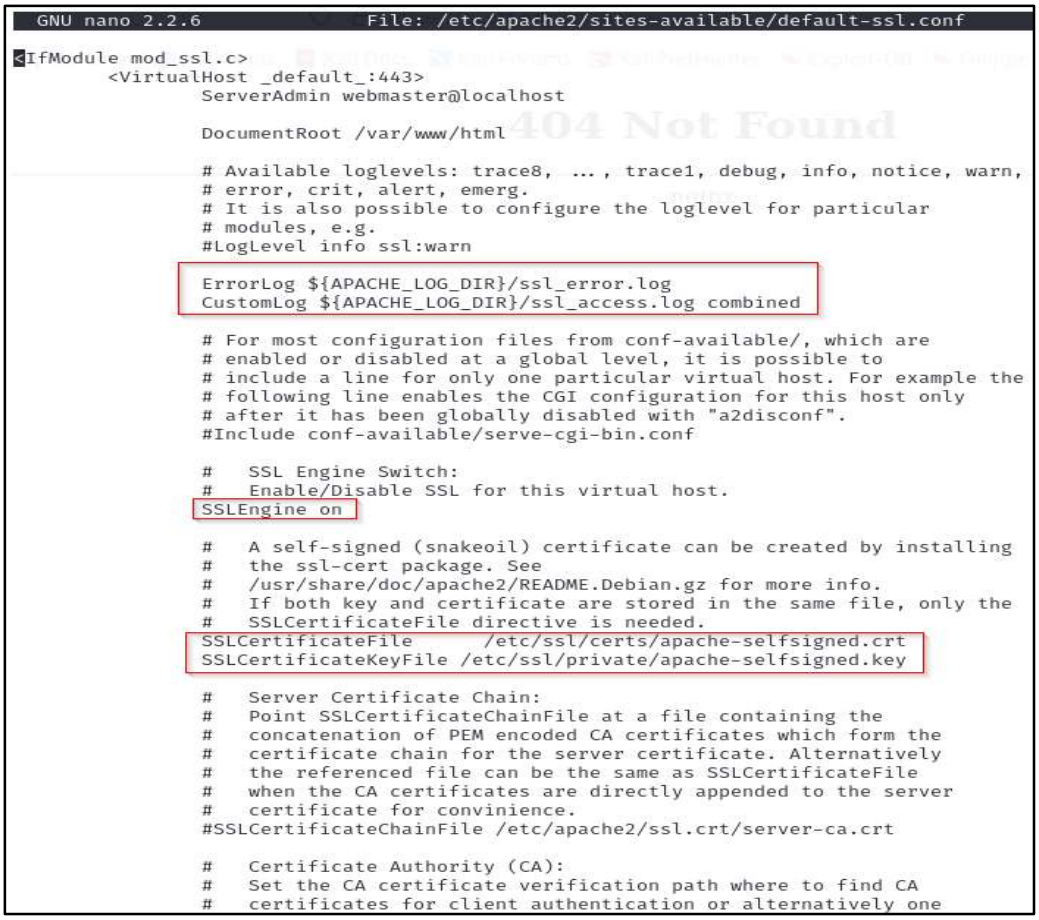


Figure 26. Configure default ssl config file part 1

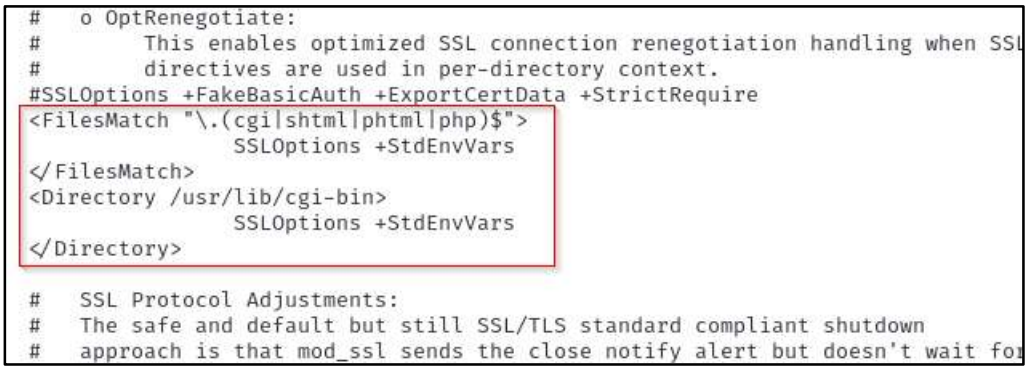


Figure 27. Configure default ssl config file part 2

```
                BrowserMatch "MSIE [2-6]" \
                        nokeepalive ssl-unclean-shutdown \
                        downgrade-1.0 force-response-1.0
                # MSIE 7 and newer should be able to use keepalive
                BrowserMatch "MSIE [17-9]" ssl-unclean-shutdown

        </VirtualHost>
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Figure 28. Configure default ssl config file part 3

5. Now enabling the SSL module as well as the SSL site module
   In this step, Apache's default-ssl site configuration and the SSL module are enabled. By turning them on, you can be sure that Apache can manage SSL/TLS traffic with the configuration options found in the default-ssl.conf file. Once the Apache service has been restarted or reloaded, the modifications will take effect.

```
root@metasploitable3-ub1404:/home/vagrant# sudo a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create self-signed certificates.
To activate the new configuration, you need to run:
  service apache2 restart
root@metasploitable3-ub1404:/home/vagrant# sudo a2ensite default-ssl
Enabling site default-ssl.
To activate the new configuration, you need to run:
  service apache2 reload
root@metasploitable3-ub1404:/home/vagrant#
```

Figure 29. Enabling ssl module and ssl site module

6. We have created the key and the self-signed SSL certificate, as shown in the screenshots, and they are both in the appropriate directories with the appropriate permissions.

```
root@metasploitable3-ub1404:/home/vagrant# sudo chmod 600 /etc/ssl/private/apache-selfsigned.key
root@metasploitable3-ub1404:/home/vagrant# ls -l /etc/ssl/private/apache-selfsigned.key
-rw------- 1 root root 1704 Jun 18 21:24 /etc/ssl/private/apache-selfsigned.key
root@metasploitable3-ub1404:/home/vagrant# ls -l /etc/ssl/certs/apache-selfsigned.crt
-rw-r--r-- 1 root root 1497 Jun 18 21:24 /etc/ssl/certs/apache-selfsigned.crt
```

Figure 30. we have key and self-assigned

7. Now restart the apache service, apache has restarted with statis displaying OK

```
root@metasploitable3-ub1404:/home/vagrant# sudo service apache2 restart
 * Restarting web server apache2
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.0.1. Set the 'S
erverName' directive globally to suppress this message
                                                                                                    [ OK ]
root@metasploitable3-ub1404:/home/vagrant#
```

Figure 31. Restart apache service

8. Now opened the main Apache config file and add the IP of metasploitable 3 in the 'servername' directive

   We make sure that Apache recognizes Metasploitable 3 instance's IP address as the server's identity by setting the ServerName to that address. This avoids possible warnings and facilitates accurate server identification.

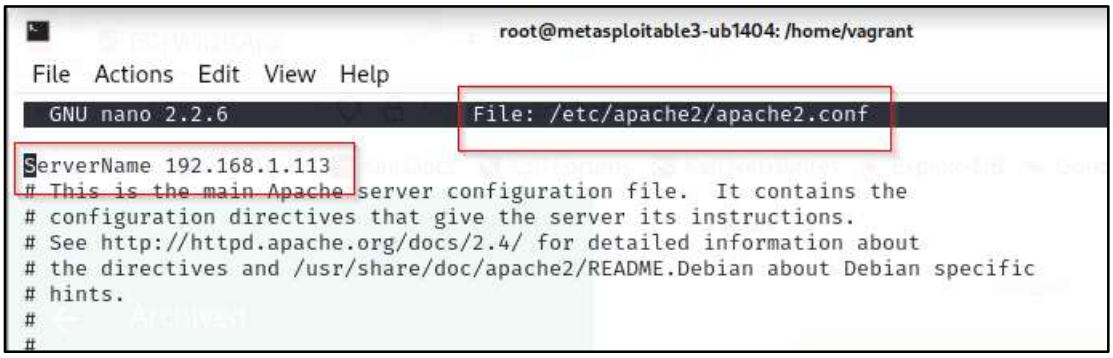


Figure 32. Configure the main apache file

9. Restart the Apache service

```
root@metasploitable3-ub1404:/home/vagrant# sudo service apache2 restart
 * Restarting web server apache2                                                    [ OK ]
root@metasploitable3-ub1404:/home/vagrant#
```

Figure 33. Restart the apache service

10. Directing all traffic to https

    We have configured a redirection from HTTP to HTTPS in this configuration. This is accomplished by creating a virtual host that is listening on port 80 and redirecting all traffic to the HTTPS URL with the 'Redirect directive'. Making sure that every connection to your server is encrypted and secure requires doing this step.

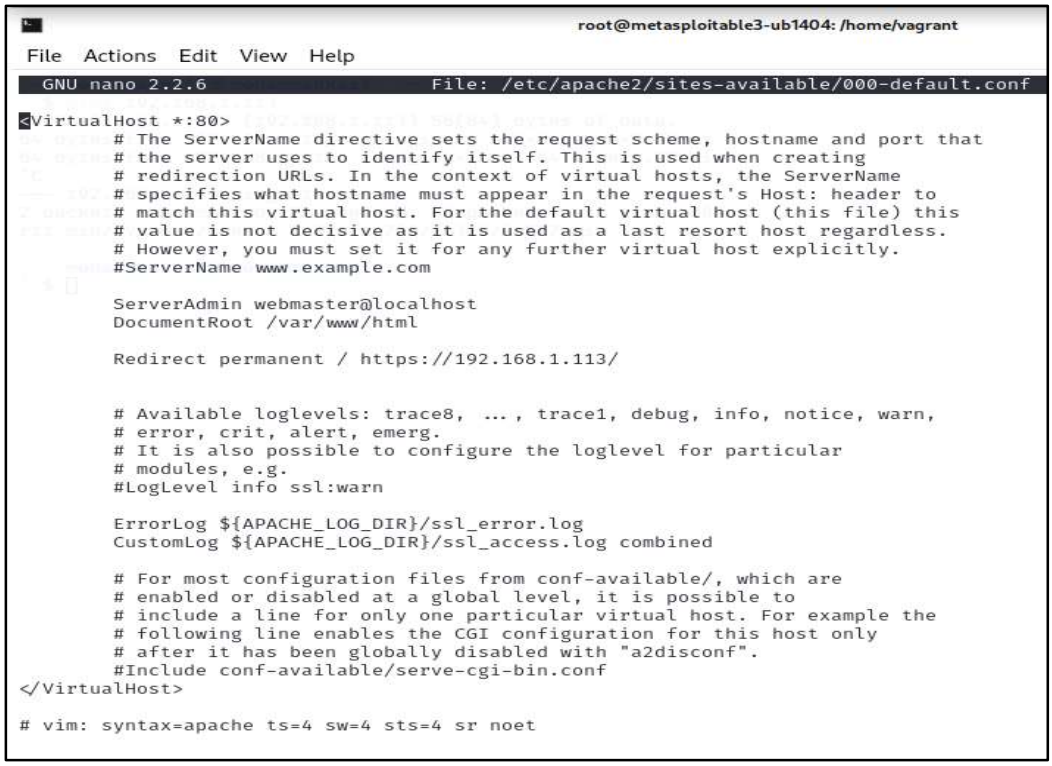


Figure 34. Directing all traffic to HTTPS

11. Since, Curl verifies that the server is properly configured for HTTPS since it can reach the HTTPS website. Most likely, the problem is with the client's browser settings or the network which is not letting me open the '*https://192.168.1.113*' on the browser

Figure 35. Curl -k result

3. Remediating FTP port 21 vulnerability on Metasploitable 3

1. We will convert FTP to SFTP
2. First of all getting the remote access to metasploitable 3 using ssh from kali. And running '*sudo service ssh status*' to check that the ssh connection is on. We need to build a directory first in order to store our FTP data. Launch a terminal window, type su to become the root user (and then enter the password for the root user when requested), and then execute the next two commands:

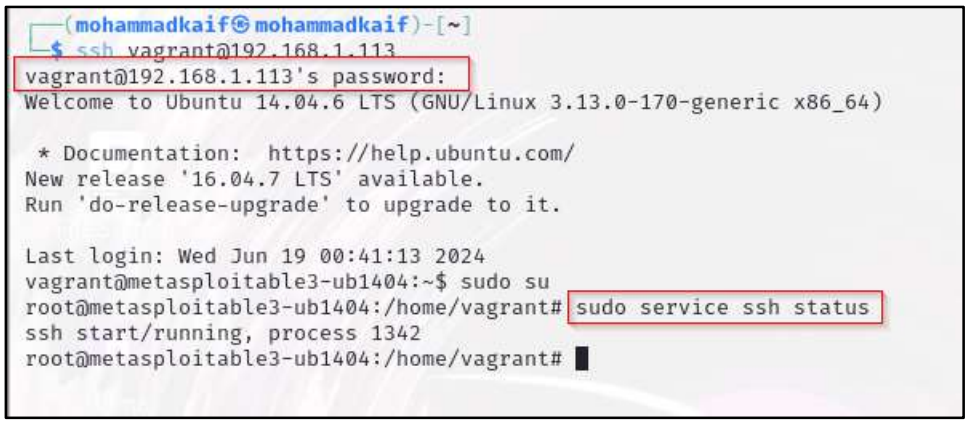


Figure 36. SSH connection to metasploitable 3

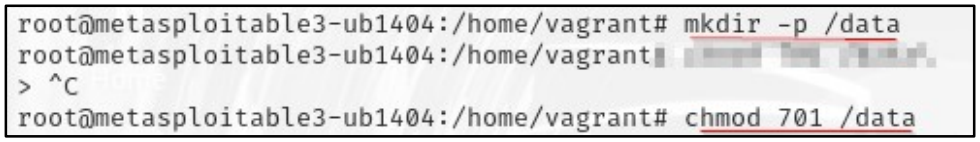


Figure 37. Creating directory to shift FTP data

3. Now we created a group named sftp_users, and we will add new users under this group. So we created a user named as '*KAIF'* and set the password as '*kaif*'. This user will be only restricted to SFTP only.

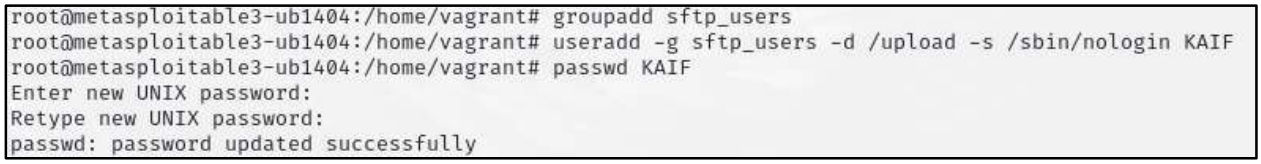


Figure 38. Create group, user and credentials

4. We will now make an upload directory that is unique to the new user and provide the directory the necessary permissions. The screenshots below listed the following commands
    - We are creating a unique folder in this stage for the user 'KAIF' to utilize when uploading files.
    - We ensure that only the system administrator has complete control over this directory and that users in the sftp_users group have restricted access by changing the owner to root.
    - You can allow the user KAIF to upload files to this directory while maintaining control over it through the sftp_users group by setting the owner to KAIF.

```
root@metasploitable3-ub1404:/home/vagrant# mkdir -p /data/KAIF/upload
root@metasploitable3-ub1404:/home/vagrant# chown -R root:sftp_users /data/KAIF
root@metasploitable3-ub1404:/home/vagrant# chown -R KAIF:sftp_users /data/KAIF/upload
```

Figure 39. Configuring

5. Now we will configure the SSH configuration file by going into '*nano /etc/ssh/ssh_config*'

```
root@metasploitable3-ub1404:/home/vagrant# nano /etc/ssh/sshd_config
```

Figure 40. Configuring ssh_config file Part 1

```
Match Group sftp_users
ChrootDirectory /data/%u
ForceCommand internal-sftp
```

Figure 41. Configuring ssh_config file Part 2

- *Match Group sftp_users*: Ensuring that the configuration is restricted to SFTP users and doesn't impact other SSH users.
- *Chrootdirectory /data/%u*: By separating the user's files, this configuration makes sure that the user cannot access outside of the specified directory, improving security.
- *ForceCommand internal-sftp*: This means that users who are logged in as members of the sftp_users group can only utilize SFTP (file transfer) and cannot run additional SSH commands.

6. Now we restart the SSH by *'sudo service ssh restart'*
7. Now from the local machine like kali, login into by entering *'sftpKAIF@192.168.1.113'*. We have successfully loggined with via sftp.

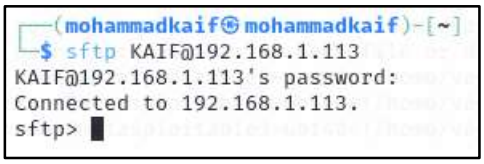


Figure 42. sftp into metasploitable from kali IP

8. Here you can see I have uploaded a txt file

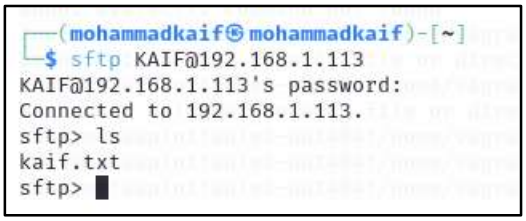


Figure 43. Uploading .txt file

9. Moreover, we can now turn off the proftpd service as well as the FTP

```
root@metasploitable3-ub1404:/home/vagrant# sudo service proftpd stop
Stopping proftpd.
```

Figure 44. Stop proftpd

4. Remediating Port 631

This port 631 vulnerability is also known as SWEET32 Birthday attack, where it exploits a weak cipher '3DES-CBC' in TLS encryption. By default, servers have the '3DES-CBC' cipher enabled in TLS.

We need to disable '3DES-CBC' cipher suites to remediate this vulnerability.

1. Login to Metasploitable 3 VM
2. Once logged in, go to /etc/cups, then edit cups.conf
3. Add SSLOptions DenyCBC MinTLS1.2

> This is to Deny CBC cipher and set the minimum TLS as 1.2

```
# Set SSLOptions
SSLOptions DenyCBC MinTLS1.2
```

Figure 45. Deny CBC

4. Restart cups service

```
root@metasploitable3-ub1404:/etc/cups# /etc/init.d/cups restart
cups stop/waiting
cups start/running, process 19687
```

Figure 46. Restart Cups service

5. Check whether 3DES_CBC is still being detected.
6. From the Nmap result, it is still showing there.

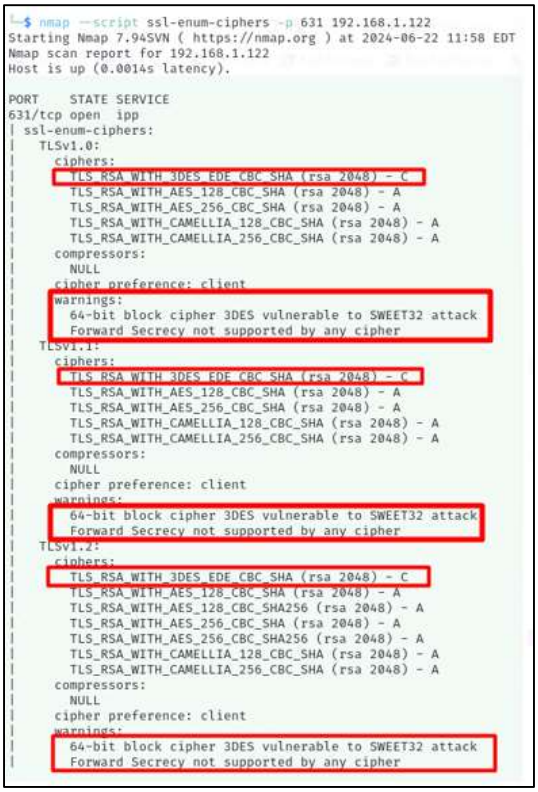


Figure 47. Still showing 3DES_CBC

7. Since the change did not take effect, as as workaround, we will disallow communications to port 631.

8. From the current rule, port 631/ipp is ACCEPT state.

```
root@metasploitable3-ub1404:/home/vagrant# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http-alt
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:3000
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ipp
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:mysql
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ftp
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:3500
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:microsoft-ds
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:8181
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:6697
DROP       all  --  anywhere             anywhere
```

Figure 48. Port 631/ipp is ACCEPT state

9. Deleted the ACCEPT rule temporarily.

```
root@metasploitable3-ub1404:/home/vagrant#
root@metasploitable3-ub1404:/home/vagrant# sudo iptables -D INPUT  -p tcp --dport ipp -j ACCEPT
root@metasploitable3-ub1404:/home/vagrant#
root@metasploitable3-ub1404:/home/vagrant#
root@metasploitable3-ub1404:/home/vagrant#
root@metasploitable3-ub1404:/home/vagrant# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere            ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:http-alt
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:3000
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:mysql
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:ftp
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:3500
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:microsoft-ds
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:8181
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:6697
DROP       all  --  anywhere             anywhere
```

Figure 49. Delete ACCEPT rule temporarliy

10. Added the DROP rule.

```
root@metasploitable3-ub1404:/home/vagrant#
root@metasploitable3-ub1404:/home/vagrant# sudo iptables -I INPUT 1 -p tcp --dport 631 -j ACCEPT
root@metasploitable3-ub1404:/home/vagrant#
root@metasploitable3-ub1404:/home/vagrant#
root@metasploitable3-ub1404:/home/vagrant# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:ipp
ACCEPT     all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere            ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:http-alt
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:3000
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:mysql
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:ftp
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:3500
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:microsoft-ds
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:8181
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:6697
DROP       all  --  anywhere             anywhere
```

Figure 50. Adding drop rule

11. 631 is on filtered status now. This means that the firewall is protecting the traffic to 631.

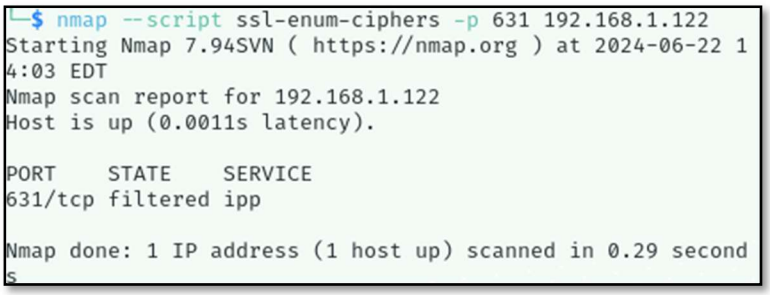


Figure 51. 631 is on filtered status

Attack Simulation and Traffic Monitoring

As seen on the vulnerability scanning conducted from OWASP and Nmap tools, the Metasploitable 3 is vulnerable to SQL injection. In the following screenshots, we will demonstrate an SQL injection attack, together with Wireshark captures to show that an SQL injection is possible and that the use of HTTP can reveal the details entered by the users.

1. SQL Injection: A code injection technique that might destroy a database.

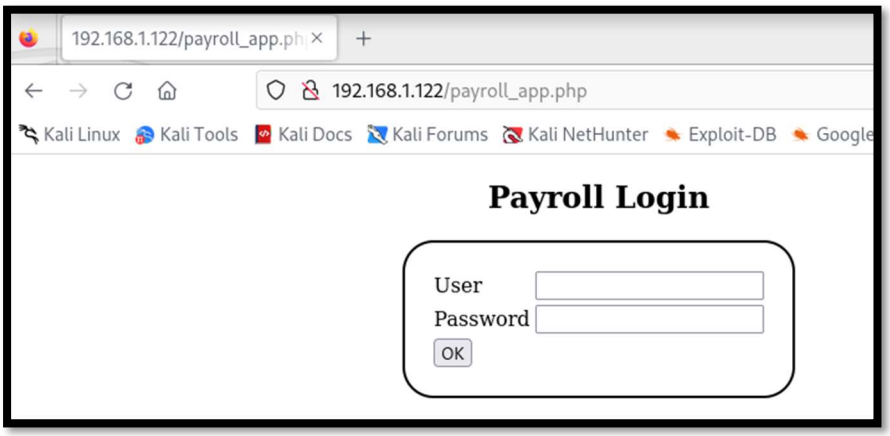


Figure 52. Payroll Website

We can check whether SQL injection is happening by inputting below as the username and password below:

The SQL query for logging in is:

```
SELECT username, first_name, last_name, salary
FROM users
WHERE username = '$user'
AND PASSWORD = '$pass'
```

Figure 53. Payroll Website Login SQL Command

Let's try to input below SQL injection statement:

Username: 'OR 1=1#
Password:

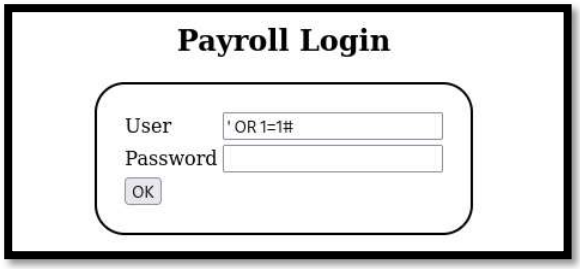


Figure 54. Payroll Login with SQL injection command

Then, the modified SQL query would be:

```
SELECT username, first_name, last_name, salary
FROM users
WHERE username = ''
OR 1 =1 # and password = '$pass'
```

Figure 55. SQL command when SQL is entered

In the screenshot above, this means that this statement '***username = '' OR 1 = 1'*** will always be true since the condition of "1=1" results in always true, and the ***# and password = '$pass'*** will be considered as a comment in the SQL query because of the symbol #.

Hence, the selected query will output ***username, first_name, last_name, and salary.***

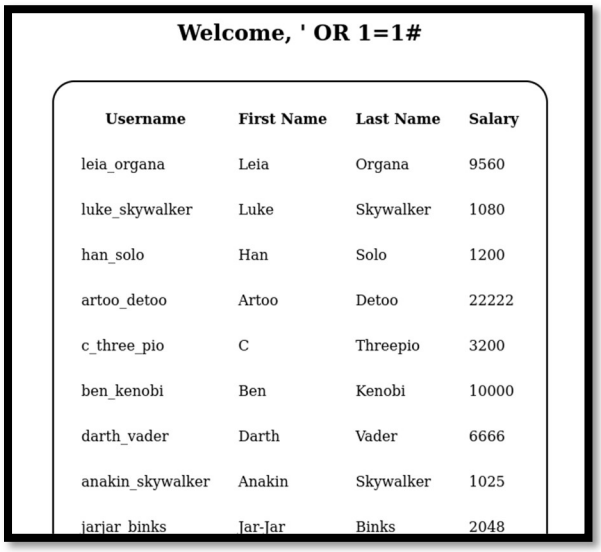


Figure 56. Payroll Website Result After Entering SQL injection

2. HTTP Traffic

Since the website uses HTTP, all traffic is transmitted in clear text. This means that any input the user enters, including sensitive information such as credentials, will be visible in plain text to anyone performing packet capture.

The screenshots below demonstrate the captured traffic using the Wireshark tool.

1. The user tries to log in to the Payroll website (http://192.168.1.122) using the below credentials:

   Username: testaccount
   Password:

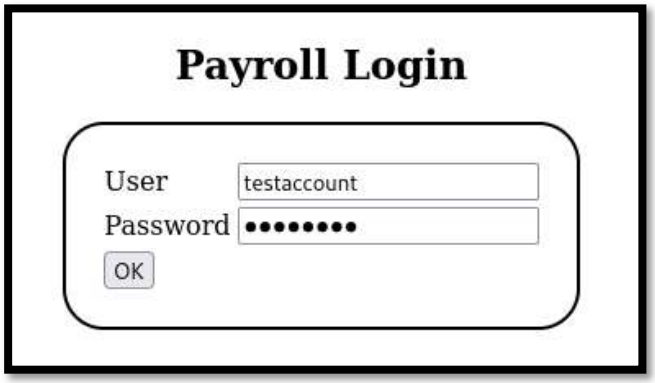


Figure 57. Payroll Login with Username and Password

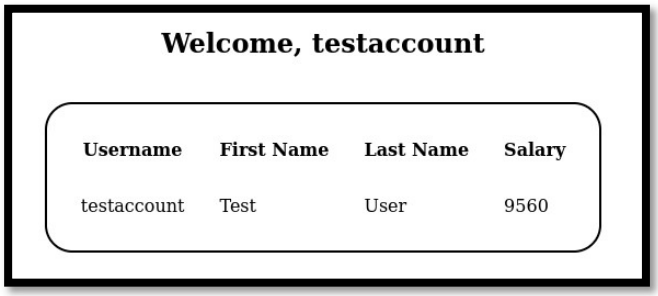


Figure 58. Payroll Login After Entering the Credentials

2. By using the Wireshark tool, let's try to monitor the traffic.

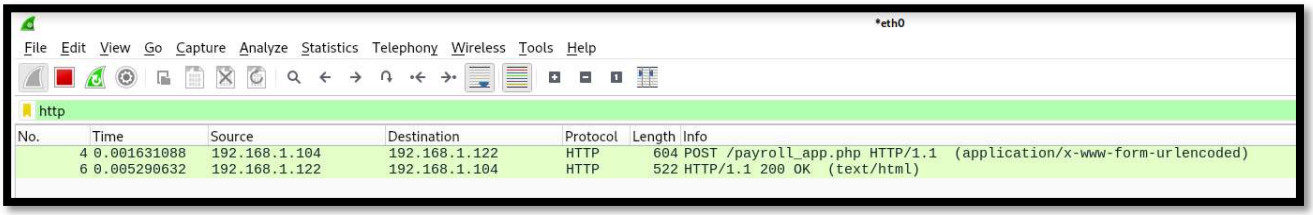


Figure 59. Wireshark Capture

3. From the capture above, let's analyze the traffic sent to 192.168.1.122.

Source IP: 192.168.1.104 / Source Port: 48262
Destination IP: 192.168.1.122 / Destination Port: 80

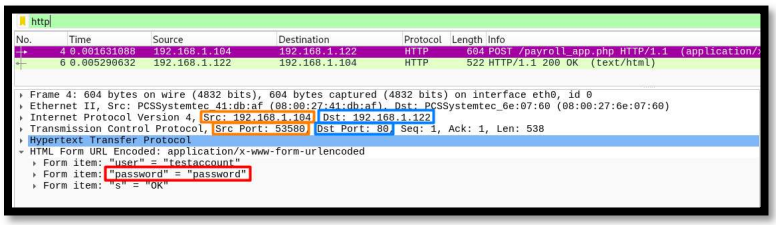


Figure 60. Wireshark Capture with the entered Password

Another way to check is by following the HTTP traffic flow:

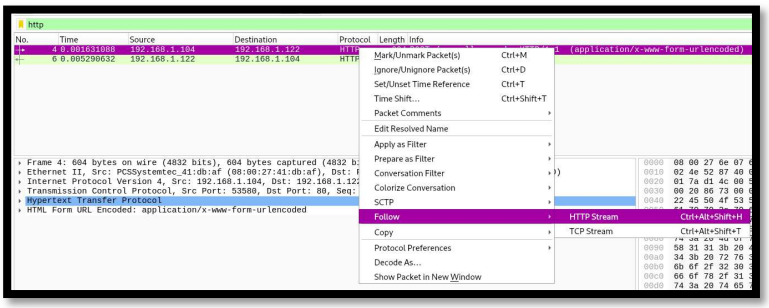


Figure 61. Wireshark Follow HTTP Stream

This shows that we can see from the packet capture the password entered by the user. It clearly displayed the credentials entered and the corresponding response of the web server.

Figure 62. Wireshark HTTP Stream

3. HTTPS Traffic and Fixed SQL Injection

After we have implemented from http to https and fixed the php commands for SQL injection, we are now using https when logging in to the Payroll web portal.

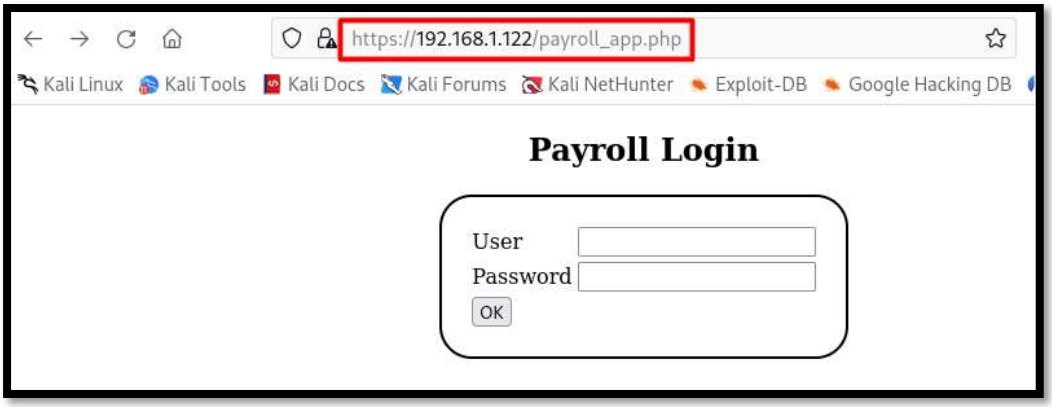


Figure 63. Payroll Login with HTTPS

1. After fixing the SQL injection attack, the Payroll web portal will now only accept valid user credentials.
2. If we try to put an SQL injection statement, the server will respond with “No results found”.

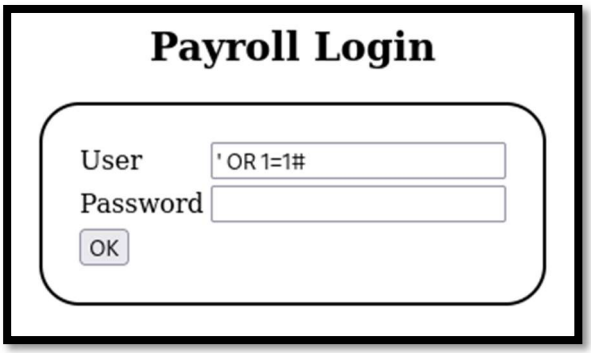


Figure 64. SQL injection command

Figure 65. No results found after fixing PHP commands

3. We can get the output if we put in the correct credential.
4. With regards to the Wireshark traffic, we cannot sniff the password anymore as the website is now using https, which is encrypted.

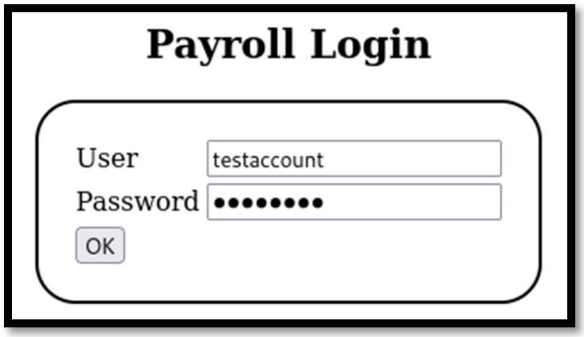


Figure 66. Real Account to Login

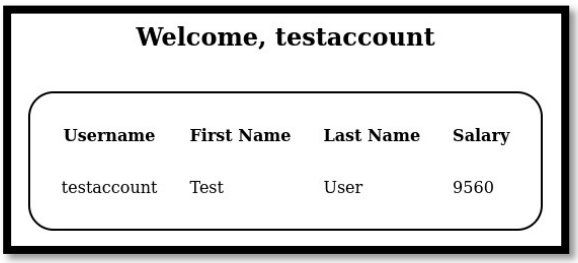


Figure 67. Results after logging correct credential

5. The screenshot below shows the traffic from Wireshark.

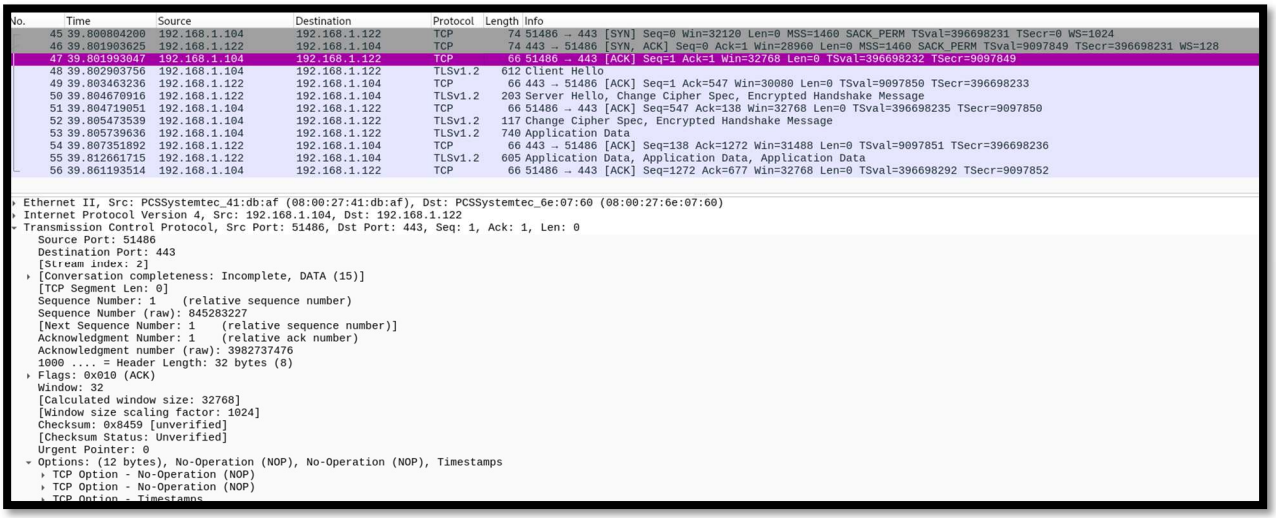


Figure 68. Wireshark Capture after implementing HTTPS

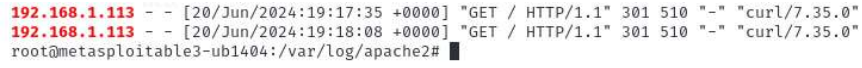
From the screenshot above, we are now using HTTPS (port 443), and credentials are passed on encrypted, and we cannot find it anymore in the capture.

Incident Report

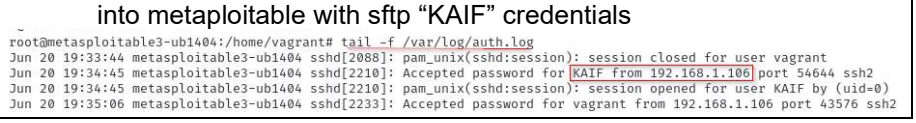
1. SSH Port (22)

Incident Details	
Incident Classification:	Unauthorized Access Incident. This can be related to the software's default root credential configuration, which gave unauthorized users complete access to the system.
Impact Analysis	
Affected Assets:	▫ System: Metasploitable 3 ▫ Data: Due to root access, every data on the Metasploitable 3 system is accessible. ▫ Process: Any programs that depend on the Metasploitable 3 system's security could be compromised.
Severity Assessment:	▫ 9.8 (CVSS score) ▫ Due to this vulnerability, unauthorized users can take complete control of the system and possibly run arbitrary code and access private information.
Incident Response Actions	
Challenges & Successes:	Challenges ▫ Figuring out the widely available default root credentials. ▫ Ensuring the SSH service was set up properly to avoid unwanted access. Success ▫ SSH access has been successfully limited to particular IP addresses. ▫ Made sure that only authorized users could use SSH to access the system. ▫ Stopped any potential illegal access by ensuring the SSH configurations were secure.
Findings & Recommendations	
Root Cause Analysis:	Using the default root credentials in the Metasploitable 3 virtual machine's SSH configuration was the primary cause of the problem. Due to this error, unauthorized individuals might have gained complete access to the system.
Preventive Measures:	▫ Verify and update SSH setups regularly to ensure default credentials are not being used. ▫ Restricted SSH access to specific IP addresses. ▫ Install a centralized monitoring and logging system to identify and stop illegal access attempts quickly.
Appendices	
Logs & Data:	Here in the log, we can see that my Windows IP is been blocked to access ssh. Jun 20 01:02:39 metasploitable3-ub1404 sudo: pam_unix(sudo:session): session closed for user root Jun 20 01:03:06 metasploitable3-ub1404 sshd[2265]: User vagrant from 192.168.1.101 not allowed because not listed in AllowUsers Jun 20 01:03:06 metasploitable3-ub1404 sshd[2265]: input_userauth_request: invalid user vagrant [preauth]

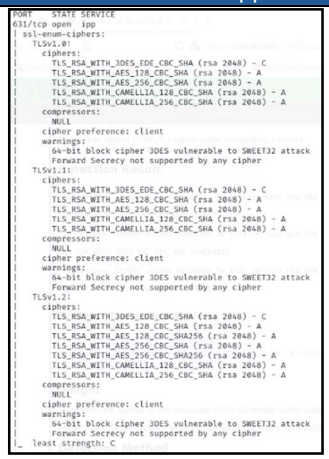
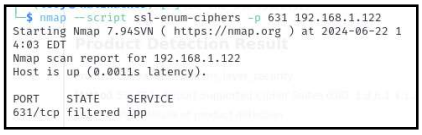
2. Port 80.

Incident Details	
Incident Classification:	Remote Code Execution (RCE) Vulnerability
Impact Analysis	
Affected Assets:	▫ System: Metasploitable 3 ▫ Data: All of the data accessed through the Apache server and the Drupal 7.5 installation ▫ Procedures: Any web apps that are utilizing port 80 on the Apache server
Severity Assessment:	▫ 7.5 (CVSS score) ▫ Potential data breaches and system compromise could result from attackers being able to execute arbitrary code due to the RCE vulnerability in Drupal and the outdated version of Apache.
Incident Response Actions	
Challenges & Successes:	Challenegs ▫ Facing difficulty while downloading the SSL certificate ▫ Wasnt able to open HTTPS version on the browser Success ▫ HTTP was changed to HTTPS to provide encrypted communication and lower the possibility of man-in-the-middle attacks. ▫ Got the result on curl –k 192.168.1.113
Findings & Recommendations	
Root Cause Analysis:	▫ The root cause was that port 80 works on http where there is no encryption. So, it becomes easy for someone to become man -in the- middle
Preventive Measures:	▫ Conduct regular security audits and vulnerability scans. ▫ Implemented HTTPS to secure communications. ▫ Regularly update and patch all web applications and servers. As, attackers can get their way through drupal and apache.
Appendices	
Logs & Data:	By '*grep "192.168.1.113" access log*' we can see the request made by the curl –k command 

3. Port 21.

Incident Details	
Incident Classification:	Unauthorized File Access Incident. This can be related to the software's default root credential configuration, which gave unauthorized users complete access to the system.
Impact Analysis	
Affected Assets:	▫ System: Metasploitable 3 ▫ Data: All information available through the ProFTPD service ▫ Processes: Any operations that depend on the FTP service to send data.
Severity Assessment:	▫ 9.8 (CVSS score) ▫ Unauthorized users may copy files from one place on the server to another without the required authentication due to the ProFTPD mod_copy vulnerability, which raises the possibility of data breaches and unauthorized data tampering.
Incident Response Actions	
Challenges & Successes:	Challenges ▫ Finding every incidence of the FTP configuration vulnerability. ▫ Found error while configuring the SSHD config file Success ▫ FTP to SFTP conversion completed successfully, improving file transmission security. ▫ ProFTPD service was turned down in order to stop the vulnerability from being exploited.
Findings & Recommendations	
Root Cause Analysis:	▫ Using an out-of-date version of ProFTPD (1.3.5) with the mod_copy vulnerability, which permitted unauthorized file manipulations, was the primary cause of the incident.
Preventive Measures:	▫ FTP was changed to SFTP for secure file transfers. ▫ turned off the vulnerable ProFTPD service. ▫ Perform frequent vulnerability assessments and security audits. ▫ Furthermore, for SFTP users, we can implement strict access to users.
Appendices	
Logs & Data:	▫ Here we can see in the logs that we successfully managed to log into metaploitable with sftp “KAIF” credentials 

4. Port 631.

Incident Details	
Incident Classification:	Port 631 / IPP port is affected by SWEET32 Birthday attack vulnerability. This attack can be used to exploit communication that uses a DES/3DES based cipher suite.
Impact Analysis	
Affected Assets:	▫ System: Metasploitable 3 ▫ Data: All information available through the Internet Printing Protocol (IPP). This port manages print jobs and queues.
Severity Assessment:	▫ 7.5 (CVSS Score) ▫ Using outdated cipher suites can open HTTPS communications to attacks, according to the vulnerability "SSL/TLS: Report Vulnerable Cipher Suites for HTTPS".
Incident Response Actions	
Challenges & Successes:	Success: ▫ Find the vulnerability by using OpenVAS, Nessus and Nmap scripts. ▫ Successfully blocked the traffic to port 631. Challenges: ▫ The stated permanent fix by manually changing the cupsd.conf file is not working.
Findings & Recommendations	
Root Cause Analysis:	Port 631 / CUPS IPP is using a weak cipher and as per recommendations by the vulnerability scanning tools, system should not use 3DES cipher to avoid this vulnerability.
Preventive Measures:	Temporarily block port 631 communication, while working on how to get the permanent fix.
Appendices	
Logs & Data:	  The nmap script shows that ciphers used are vulnerable to SWEET32 attack. Temporarily block traffic to port 631.

References

Wazuh. (2024). *Wazuh: The Open Source Security Platform*.

https://wazuh.com/

Trend Micro. (n.d.). *What is XDR?*

https://www.trendmicro.com/en_ca/what-is/xdr.html

Wazuh. (n.d.). *Wazuh agent - Installation guide · Wazuh documentation*.

https://documentation.wazuh.com/current/installation-guide/wazuh-agent/index.html