

Lab № 3

During this lab I will use Linux's firewall iptables. It is usually installed by default, let us check the version of iptables:

```
root@elba-vm:~# iptables --version
iptables v1.8.5 (nf_tables)
root@elba-vm:~#
```

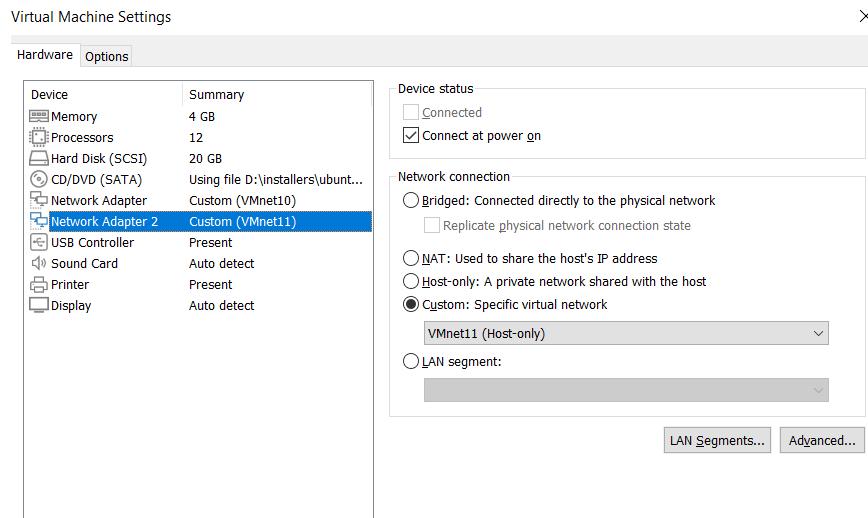
- 1) Set up two network interfaces: internal and external.
- 2) Assign IPv4 class C address for internal interface and B class address for external interface.

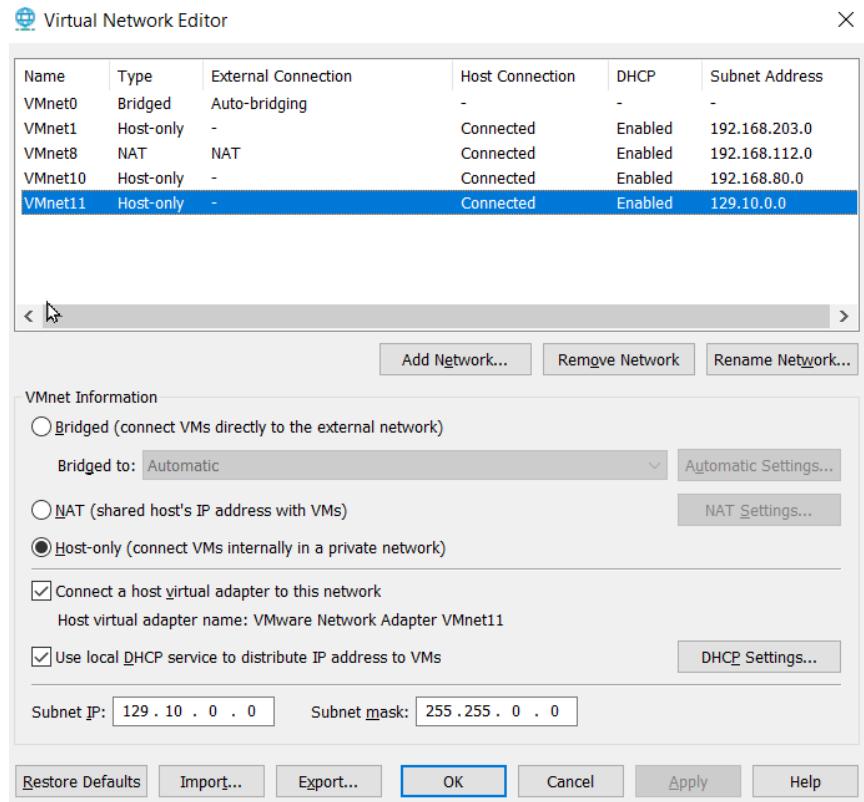
Firstly, two virtual networks were created with following subnet addresses and masks:

Vmnet10 192.168.80.0/24 – for internal interface.

Vmnet11 129.10.0.0/16 – for external interface.

Both adapters on VM were connected to these virtual networks.





IP addresses assigned by DHCP service, but not to let them alter and survive
restart we can save manual configuration in netplan, running netplan apply will
check configuration and apply assignments:

```
elbayi@elba-vm:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.80.128 netmask 255.255.255.0 broadcast 192.168.80.255
        inet6 fe80::ae8a:c6c:ad77:4791 prefixlen 64 scopeid 0x20<link>
          ether 00:0c:29:1c:52:dd txqueuelen 1000 (Ethernet)
            RX packets 32 bytes 7576 (7.5 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 62 bytes 7451 (7.4 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

ens38: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 129.10.128.0 netmask 255.255.0.0 broadcast 129.10.255.255
        inet6 fe80::b0a8:3134:acb6:880e prefixlen 64 scopeid 0x20<link>
          ether 00:0c:29:1c:52:e7 txqueuelen 1000 (Ethernet)
            RX packets 31 bytes 7446 (7.4 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 63 bytes 7546 (7.5 KB)
            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 115 bytes 9305 (9.3 KB)
          RX errors 0 dropped 0 overruns 0 frame 0
          TX packets 115 bytes 9305 (9.3 KB)
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

elbayi@elba-vm:~$
```

```
: GNU nano 5.2          /etc/netplan/01-network-manager-all.yaml
# Let NetworkManager manage all devices on this system
network:
  version: 2
  renderer: NetworkManager
  ethernets:
    ens33:
      dhcp4: no
      addresses: [192.168.80.128/24, ]
      gateway4: 192.168.80.1
      nameservers:
        addresses: [8.8.8.8,8.8.4.4]
    ens38:
      dhcp4: no
      addresses: [129.10.0.0/16, ]
      gateway4: 129.10.128.0
      nameservers:
        addresses: [8.8.8.8,8.8.4.4]
```

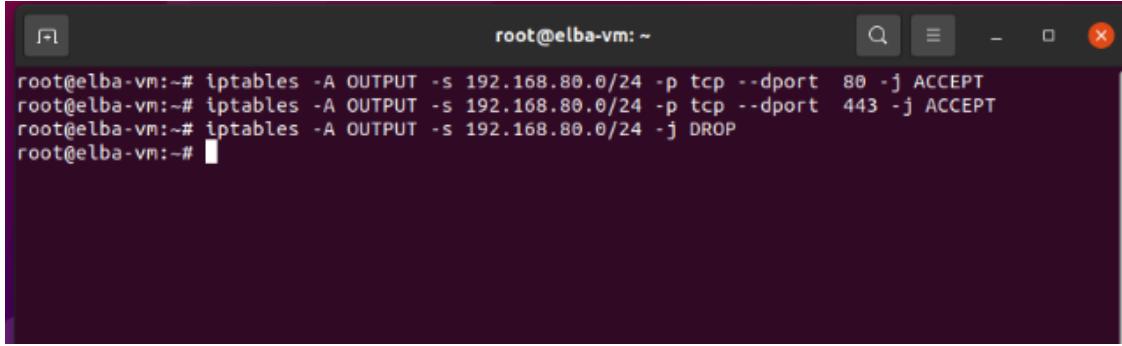
- 3) Deny all inbound traffic.
- 4) Setup port forwarding of 80 port to one of the internal addresses.

```
root@elba-vm:~# iptables -P INPUT DROP
root@elba-vm:~# iptables -L
Chain INPUT (policy DROP)
target     prot opt source                 destination
Chain FORWARD (policy ACCEPT)
target     prot opt source                 destination
Chain OUTPUT (policy ACCEPT)
target     prot opt source                 destination
root@elba-vm:~#
```

Prerouting rule defines how the packet's info will be changed after it will be received by the interface, postrouting before leaving an interface.

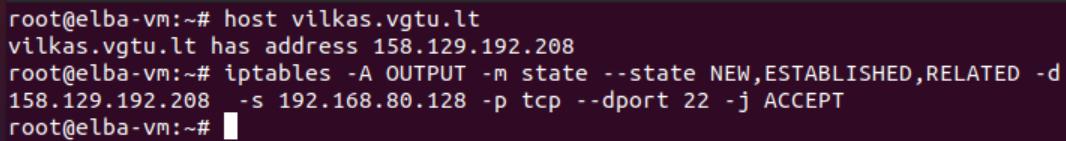
```
root@elba-vm:~# iptables -t nat -A PREROUTING -i ens38 -p tcp --dport 80 -j DNAT --to-destination 192.168.80.129:80
root@elba-vm:~# iptables -t nat -A POSTROUTING -j MASQUERADE
root@elba-vm:~# iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source                 destination
DNAT      tcp  --  anywhere            anywhere           tcp dpt:http to:192.168.80.129:80
Chain INPUT (policy ACCEPT)
target     prot opt source                 destination
Chain POSTROUTING (policy ACCEPT)
target     prot opt source                 destination
MASQUERADE  all  --  anywhere            anywhere
Chain OUTPUT (policy ACCEPT)
target     prot opt source                 destination
root@elba-vm:~#
```

- 5) Allow egress traffic from internal addresses only through the 80 and 443 ports.



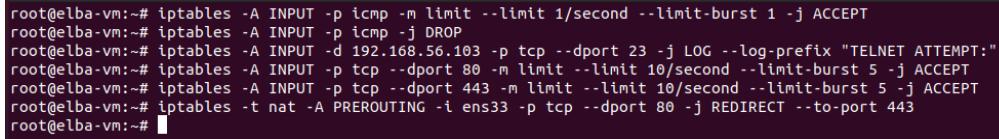
```
root@elba-vm:~# iptables -A OUTPUT -s 192.168.80.0/24 -p tcp --dport 80 -j ACCEPT
root@elba-vm:~# iptables -A OUTPUT -s 192.168.80.0/24 -p tcp --dport 443 -j ACCEPT
root@elba-vm:~# iptables -A OUTPUT -s 192.168.80.0/24 -j DROP
root@elba-vm:~#
```

- 6) Allow access to fmf.vgtu.lt(vilkas.vgtu.lt) through 22 port for one of the internal addresses.



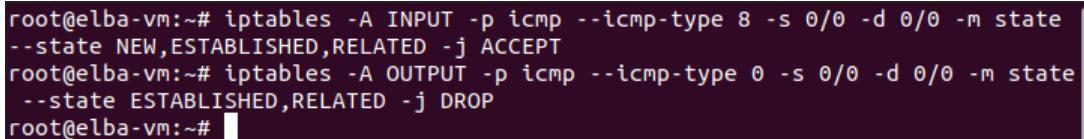
```
root@elba-vm:~# host vilkas.vgtu.lt
vilkas.vgtu.lt has address 158.129.192.208
root@elba-vm:~# iptables -A OUTPUT -m state --state NEW,ESTABLISHED,RELATED -d 158.129.192.208 -s 192.168.80.128 -p tcp --dport 22 -j ACCEPT
root@elba-vm:~#
```

- 7) Add five additional advanced rules to your firewall on your own choice.



```
root@elba-vm:~# iptables -A INPUT -p icmp -m limit --limit 1/second --limit-burst 1 -j ACCEPT
root@elba-vm:~# iptables -A INPUT -p icmp -j DROP
root@elba-vm:~# iptables -A INPUT -d 192.168.56.103 -p tcp --dport 23 -j LOG --log-prefix "TELNET ATTEMPT:"
root@elba-vm:~# iptables -A INPUT -p tcp --dport 80 -m limit --limit 10/second --limit-burst 5 -j ACCEPT
root@elba-vm:~# iptables -A INPUT -p tcp --dport 443 -m limit --limit 10/second --limit-burst 5 -j ACCEPT
root@elba-vm:~# iptables -t nat -A PREROUTING -i ens33 -p tcp --dport 80 -j REDIRECT --to-port 443
root@elba-vm:~#
```

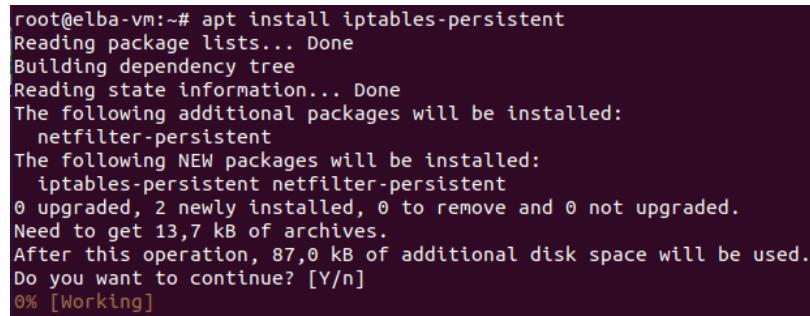
- 8) Apply ICMP policy when only „Echo Request“ is allowed.



```
root@elba-vm:~# iptables -A INPUT -p icmp --icmp-type 8 -s 0/0 -d 0/0 -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
root@elba-vm:~# iptables -A OUTPUT -p icmp --icmp-type 0 -s 0/0 -d 0/0 -m state --state ESTABLISHED,RELATED -j DROP
root@elba-vm:~#
```

- 9) Configure your Firewall that every time you start/restart it all configuration and rule set is loaded automatically.

To run iptables on set rules we install iptables-persistent, and then run iptables-save:



```
root@elba-vm:~# apt install iptables-persistent
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  netfilter-persistent
The following NEW packages will be installed:
  iptables-persistent netfilter-persistent
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 13,7 kB of archives.
After this operation, 87,0 kB of additional disk space will be used.
Do you want to continue? [Y/n]
0% [Working]
```

Configuring iptables-persistent

```
Current iptables rules can be saved to the configuration file
/etc/iptables/rules.v4. These rules will then be loaded automatically
during system startup.

Rules are only saved automatically during package installation. See the
manual page of iptables-save(8) for instructions on keeping the rules
file up-to-date.

Save current IPv4 rules?

    <Yes>          <No>
```

```
root@elba-vm:~# iptables-save
# Generated by iptables-save v1.8.5 on Sun May  9 09:27:07 2021
*filter
:INPUT ACCEPT [28951:9633356]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [9693:776346]
-A INPUT -p icmp -m icmp --icmp-type 8 -m state --state NEW,RELATED,ESTABLISHED
-j ACCEPT
-A OUTPUT -p icmp -m icmp --icmp-type 0 -m state --state RELATED,ESTABLISHED -j
DROP
COMMIT
# Completed on Sun May  9 09:27:07 2021
# Generated by iptables-save v1.8.5 on Sun May  9 09:27:07 2021
*nat
:PREROUTING ACCEPT [0:0]
:INPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
COMMIT
# Completed on Sun May  9 09:27:07 2021
root@elba-vm:~# reboot
```

10) Implement at least six different advanced „Match“ and four different advanced „Target“ conditions.

```
root@ubuntu:~# iptables -A INPUT -p tcp --syn --dport 80 -m connlimit --connlimit-above 10 --connlimit-mask 24 -j REJECT
root@ubuntu:~# iptables -A INPUT -m state --state INVALID -j DROP
root@ubuntu:~# ipset create port_scanners hash:ip family inet hashsize 32768 maxelem 65536 timeout 600
root@ubuntu:~# ipset create scanned_ports hash:ip,port family inet hashsize 32768 maxelem 65536 timeout 60
root@ubuntu:~# iptables -A INPUT -m state --state NEW -m set ! --match-set scanned_ports src,dst -m hashlimit --hashlimit-above 1/hour --hashlimit-burst 5 --hashlimit-mode srip --hashlimit-name portscan --hashlimit-table-expire 10000 -j SET --add-set port_scanners src --exist
root@ubuntu:~# iptables -A INPUT -m state --state NEW -m set --match-set port_scanners src -j DROP
root@ubuntu:~# iptables -A INPUT -m state --state NEW -m set --match-set port_scanners src -j LOG --log-prefix "SCANNER DETECTED"
root@ubuntu:~# iptables -A INPUT -m state --state NEW -j SET --add-set scanned_ports src,dst
root@ubuntu:~# iptables -A INPUT -p tcp --syn -m multiport --dports 22,23 -m connlimit --connlimit-above 2 -j REJECT
root@ubuntu:~#
```

11) Allow rpcinfo (Remote Procedure Call) from specific host/interface.

```
root@ubuntu:~# iptables -A INPUT -s 192.168.80.129 -i ens38 -p udp --dport 111
-j ACCEPT
root@ubuntu:~#
```

12) Implement Single Packet Authorization (SPA) or alternative solution.

I used fwknop-server and client to implement SPA:

```
root@elba-vm:~# apt install fwknop-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  fwknop-apparmor-profile
The following NEW packages will be installed:
  fwknop-server
0 upgraded, 1 newly installed, 0 to remove and 3 not upgraded.
Need to get 84,3 kB of archives.
After this operation, 252 kB of additional disk space will be used.
Get:1 http://lt.archive.ubuntu.com/ubuntu groovy/universe amd64 fwknop-server amd64 2.6.10-11 [84,3 kB]
Fetched 84,3 kB in 1s (75,2 kB/s)
Selecting previously unselected package fwknop-server.
(Reading database ... 198305 files and directories currently installed.)
Preparing to unpack .../fwknop-server_2.6.10-11_amd64.deb ...
Unpacking fwknop-server (2.6.10-11) ...
Setting up fwknop-server (2.6.10-11) ...
fwknop-server.service is a disabled or a static unit not running, not starting it.
Processing triggers for man-db (2.9.3-2) ...
Processing triggers for systemd (246.6-1ubuntu1.3) ...
root@elba-vm:~#
```

The screenshot shows a Kali Linux terminal window with several tabs open. The current tab displays the command-line interface for managing the fwknop package. It starts with the command `apt install fwknop-server`, followed by the output of the package manager showing dependencies and download progress. Below this, the user runs `fwknop -A` to generate an initial rc file at `/root/.fwknoprc`. The terminal then shows the contents of this file, which includes configuration parameters like `ALLOW_IP`, `ACCESS`, `SPA_SERVER`, `KEY_BASE64`, `HMAC_KEY_BASE64`, and `USE_HMAC`. Finally, the user opens the `fwknopd.conf` configuration file in nano editor, where they can see detailed settings for the fwknop daemon, including network interfaces, packet filtering, and security keys.

```
root@kali:~# fwknop -A tcp/22 -a 192.168.80.131 -D 192.168.80.128 --key-gen --use-hmac --save-rc-stanza
[*] Creating initial rc file: /root/.fwknoprc.
[+] Wrote Rijndael and HMAC keys to rc file: /root/.fwknoprc

root@kali:~# cat ~/.fwknoprc
[default]

[192.168.80.128]
ALLOW_IP          192.168.80.131
ACCESS            tcp/22
SPA_SERVER        192.168.80.128
KEY_BASE64        Zk0$1JJtUNL7YifJcp5Y07HTVzhasoH6iBchAsCVd7c=
HMAC_KEY_BASE64   gYmUZAXYA5jIYZuPsjjfL9jVjjejTHo/zbpTc83gfsiHpi7FQy0yK3/8P+CS3TJRw6bWaeYecYwoVaxWZoK/Og==
USE_HMAC          Y

GNU nano 5.2                               /etc/fwknop/fwknopd.conf
#VERBOSE                         0;

# Define the ethernet interface on which we will sniff packets.
# Default if not set is eth0. The '-i <intf>' command line option overrides
# the PCAP_INTF setting.
#
PCAP_INTF           ens33;

# By default fwknopd does not put the pcap interface into promiscuous mode.
# Set this to 'Y' to enable promiscuous sniffing.
#
#ENABLE_PCAP_PROMISC      N;

# Define the filter used for PCAP modes; we default to udp port 62201.
# However, if an fwknop client uses the --rand-port option to send the
# SPA packet over a random port, then this variable should be updated to
# something like "udp dst portrange 10000-65535;".
# Default is "udp port 62201".
#
PCAP_FILTER         udp port 62201;
```

root@kali: ~

File Actions Edit View Help

GNU nano 5.4 /root/.fwknoprc [default]

[192.168.80.128]

ALLOW_IP	192.168.80.131
ACCESS	tcp/22
SPA_SERVER	192.168.80.128
KEY_BASE64	bgmH3cP0g97pXavYZxlccTmB4C61Movh1M4gX8yvK98=
HMAC_KEY_BASE64	DBMOiabunImi83hdToC38yf+0qsYzu5mo/LFQvjWY7ygQLn/WN1U5rbwq9Dzq2ShrhCX>
USE_HMAC	Y

fwknopd access.conf stanzas

SOURCE ANY
#KEY_BASE64 __CHANGEME__
#HMAC_KEY_BASE64 __CHANGEME__

REQUIRE_SOURCE_ADDRESS Y

KEY_BASE64 Zk0S1JJtUNl7Y1fJcp5Y07HTVzhosoH6iBchAsCVd7c=
HMAC_KEY_BASE64 gYmUZAXYA5jIYZuPsjjfL9jVjjejTHo/zbpTc83gfsiHpi7FQy0yK3/8P+CS3TJRw6bWaeYecYwoVaxWZoK/Og==

CMD_CYCLE_OPEN iptables -A INPUT -p \$PROTO --dport \$PORT -j ACCEPT
CMD_CYCLE_CLOSE iptables -D INPUT -p \$PROTO --dport \$PORT -j ACCEPT

CMD_CYCLE_TIMER 180
If you want to use GnuPG keys then define the following variables

elbayi@ubuntu: ~

File Actions Edit View Help

(root㉿kali)-[~] # fwknop -n 192.168.80.128

(root㉿kali)-[~] # ssh elbayi@192.168.80.128

elbayi@192.168.80.128's password:

Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 5.4.0-73-generic x86_64)

* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage

* Canonical Livepatch is available for installation.
- Reduce system reboots and improve kernel security. Activate at:
https://ubuntu.com/livepatch

Your Hardware Enablement Stack (HWE) is supported until April 2023.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

elbayi@ubuntu:~\$

```

May 12 09:22:30 ubuntu fwknopd[42314]: (stanza #1) SPA Packet from IP: 192.168.80.131 received with access source match
May 12 09:22:30 ubuntu fwknopd[42314]: [192.168.80.131] (stanza #1) Running CMD_CYCLE_OPEN command: iptables -A INPUT -p 6 --dport 22 -j ACCEPT
May 12 09:22:30 ubuntu fwknopd[42314]: [192.168.80.131] (stanza #1) Running CMD_CYCLE_CLOSE command in 180 seconds: iptables -D INPUT -p 6 --dport 22 -j ACCEPT
May 12 09:24:46 ubuntu dhclient[1221]: DHCPREQUEST of 192.168.80.128 on ens33 to 192.168.80.254 port 67 (xid=0x394953f9)
May 12 09:24:46 ubuntu dhclient[1221]: DHCPCACK of 192.168.80.128 from 192.168.80.254
May 12 09:24:46 ubuntu dhclient[1221]: DHCPACK of 192.168.80.128 from 192.168.80.254
May 12 09:24:46 ubuntu dhclient[1221]: <Info> [1620836686.1815] dhcpc4 (ens33): address 192.168.80.128
May 12 09:24:46 ubuntu NetworkManager[1091]: <Info> [1620836686.1816] dhcpc4 (ens33): plen 24 (255.255.255.0)
May 12 09:24:46 ubuntu NetworkManager[1091]: <Info> [1620836686.1816] dhcpc4 (ens33): lease time 1800
May 12 09:24:46 ubuntu NetworkManager[1091]: <Info> [1620836686.1817] dhcpc4 (ens33): nameserver '192.168.80.1'
May 12 09:24:46 ubuntu NetworkManager[1091]: <Info> [1620836686.1817] dhcpc4 (ens33): domain name 'localdomain'
May 12 09:24:46 ubuntu NetworkManager[1091]: <Info> [1620836686.1817] dhcpc4 (ens33): state changed bound -> bound
May 12 09:24:46 ubuntu dbus-daemon[1040]: [system] Activating via systemd: service name='org.freedesktop.nm_dispatcher' unit='dbus-org.freedesktop.nm-dispatcher.service' requested by '/usr/sbin/NetworkManager --no-daemon' label='unconfined'
May 12 09:24:46 ubuntu dhclient[1221]: bound to 192.168.80.128 -- renewal in 885 seconds.
May 12 09:24:46 ubuntu systemd[1]: Starting Network Manager Script Dispatcher Service...
May 12 09:24:46 ubuntu dbus-daemon[1040]: [system] Successfully activated service 'org.freedesktop.nm_dispatcher'
May 12 09:24:46 ubuntu systemd[1]: Started Network Manager Script Dispatcher Service.
May 12 09:24:46 ubuntu nm-dispatcher: req:1 'dhcpc4-change' [ens33]: new request (1 scripts)
May 12 09:24:46 ubuntu nm-dispatcher: req:1 'dhcpc4-change' [ens33]: start running ordered scripts...
May 12 09:25:30 ubuntu fwknopd[42314]: [192.168.80.131] (stanza #1) Timer expired, running CMD_CYCLE_CLOSE command: iptables -D INPUT -p 6 --dport 22 -j ACCEPT

```

13) Test your firewall with local (Nmap or Nessus) or online* (ShieldsUP or similar) tools. Perform firewall tests with both interfaces (external to internal and vice versa).

```

└──(root💀kali㉿kali)-[~]
# nmap -v -sA -n 192.168.80.128 -oA firewallaudit
Starting Nmap 7.91 ( https://nmap.org ) at 2021-05-16 15:39 EEST
Initiating ARP Ping Scan at 15:39
Scanning 192.168.80.128 [1 port]
Completed ARP Ping Scan at 15:39, 0.07s elapsed (1 total hosts)
Initiating ACK Scan at 15:39
Scanning 192.168.80.128 [1000 ports]
Completed ACK Scan at 15:39, 21.51s elapsed (1000 total ports)
Nmap scan report for 192.168.80.128
Host is up (0.00052s latency).
All 1000 scanned ports on 192.168.80.128 are filtered
MAC Address: 00:0C:29:1C:52:DD (VMware)

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 21.75 seconds
    Raw packets sent: 2001 (80.028KB) | Rcvd: 1 (28B)

└──(root💀kali㉿kali)-[~]
# 

```

Sev	Name	Family	Count	
INFO	Ethernet Card Manufacturer Detection	Misc.	1	
INFO	Ethernet MAC Addresses	General	1	
INFO	Nessus Scan Information	Settings	1	
INFO	VMware Virtual Machine Detection	General	1	