Routerlab

Summer semester 2018

Worksheet 5
Group 08

Valentin Franck, 364066
Nikhil Singh, 387694

Pages: 29

Submission Date: June 7, 2018

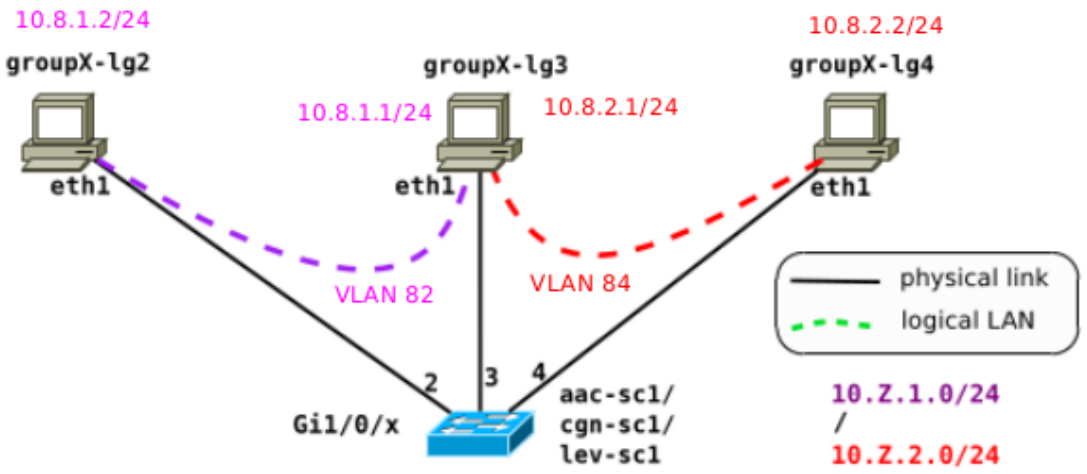

Figure 1: The topology we use for Question 1.

Question 1

1a

We are using the topology as shown in Figure 1.

1b

Q1(b) - We used this command to enable IP forwarding on lg3. Below mentioned is the output on lg3:

```
root@group08-lg3:~# cat /proc/sys/net/ipv4/ip_forward
0
root@group08-lg3:~# echo 1 > /proc/sys/net/ipv4/ip_forward
root@group08-lg3:~# cat /proc/sys/net/ipv4/ip_forward
1
```

We have created vlan interfaces on lg3

```
root@group08-lg3:~# modprobe 8021q
root@group08-lg3:~# echo 8021q | tee -a /etc/modules
8021q
root@group08-lg3:~# cat /etc/modules
# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with "#" are ignored.

8021q


root@group08-lg3:~# vconfig add eth1 82
Added VLAN with VID == 82 to IF -:eth1:-
root@group08-lg3:~# ifconfig eth1.82 10.8.1.1/24
root@group08-lg3:~# ifconfig -a
eth1.82: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.8.1.1  netmask 255.255.255.0  broadcast 10.8.1.255
        inet6 fe80::216:3eff:feaf:831  prefixlen 64  scopeid 0x20<link>
```

```
        ether 00:16:3e:af:08:31  txqueuelen 0  (Ethernet)
        RX packets 721  bytes 41472 (40.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 33  bytes 2762 (2.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@group08-lg3:~# vconfig add eth1 84
Added VLAN with VID == 84 to IF -:eth1:-
root@group08-lg3:~# ifconfig eth1.84 10.8.2.1/24
root@group08-lg3:~# ifconfig -a
eth1.84: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.8.2.1  netmask 255.255.255.0  broadcast 10.8.2.255
        inet6 fe80::216:3eff:feaf:831  prefixlen 64  scopeid 0x20<link>
        ether 00:16:3e:af:08:31  txqueuelen 0  (Ethernet)
        RX packets 783  bytes 43640 (42.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 28  bytes 2272 (2.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

We configured the switch and used vlan 82 on interface Gi1/0/2, vlan 84 on Gi1/0/4 and both on Gi1/0/3:

```
interface GigabitEthernet1/0/2
 description lg2
 switchport access vlan 82
 switchport mode access
 no cdp enable
!
interface GigabitEthernet1/0/3
 description lg3
 switchport trunk allowed vlan 82,84
 switchport mode trunk
 no cdp enable
!
interface GigabitEthernet1/0/4
 description lg4
 switchport access vlan 84
 switchport mode access
 no cdp enable
```

We assigned IP address 10.8.1.2/24 on lg2

```
root@group08-lg2:~# ip addr add 10.8.1.2/24 dev eth1
root@group08-lg2:~# ip a s
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
    state UP group default qlen 1000
    link/ether 00:16:3e:af:08:21 brd ff:ff:ff:ff:ff:ff
    inet 10.8.1.2/24 scope global eth1
       valid_lft forever preferred_lft forever
    inet6 fe80::216:3eff:feaf:821/64 scope link
       valid_lft forever preferred_lft forever
```

and 10.8.2.2/24 on lg4

```
root@group08-lg4:~# ip addr add 10.8.2.2/24 dev eth1
root@group08-lg4:~# ip a s
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
    state UP group default qlen 1000
    link/ether 00:16:3e:af:08:41 brd ff:ff:ff:ff:ff:ff
```

```
    inet 10.8.2.2/24 scope global eth1
       valid_lft forever preferred_lft forever
    inet6 fe80::216:3eff:feaf:841/64 scope link
       valid_lft forever preferred_lft forever
```

Then we added the route on lg2 and lg4

```
root@group08-lg2:~# ip route add 10.8.2.0/24 via 10.8.1.1 dev eth1
root@group08-lg4:~# ip route add 10.8.1.0/24 via 10.8.2.1 dev eth1
```

And pinged the devices from lg2 and lg4 using lg3 as a router

```
root@group08-lg2:~# ping 10.8.2.2
PING 10.8.2.2 (10.8.2.2) 56(84) bytes of data.
64 bytes from 10.8.2.2: icmp_seq=1 ttl=63 time=1.21 ms
64 bytes from 10.8.2.2: icmp_seq=2 ttl=63 time=1.28 ms
64 bytes from 10.8.2.2: icmp_seq=3 ttl=63 time=1.03 ms
64 bytes from 10.8.2.2: icmp_seq=4 ttl=63 time=1.36 ms
^C
--- 10.8.2.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 1.037/1.226/1.368/0.121 ms

root@group08-lg4:~# ping 10.8.1.2
PING 10.8.1.2 (10.8.1.2) 56(84) bytes of data.
64 bytes from 10.8.1.2: icmp_seq=1 ttl=63 time=1.13 ms
64 bytes from 10.8.1.2: icmp_seq=2 ttl=63 time=1.23 ms
64 bytes from 10.8.1.2: icmp_seq=3 ttl=63 time=1.34 ms
64 bytes from 10.8.1.2: icmp_seq=4 ttl=63 time=1.12 ms
^C
--- 10.8.1.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 1.128/1.210/1.345/0.096 ms
```

Question 2

2a

(i) In Linux 2.4.x and later kernel series netfilter is the packet filtering framework. The software which is commonly associated with it is iptables. Netfilter provides many functionalities for packet filtering, port translation and NAT which prohibits unwanted packets within a computer network. Iptables is the kernel module which is an important part of the Netfilter hook system which manages packet filtering and NAT rules. It comes with all Linux distributions. It helps us manage the Linux firewall effectively by managing the Linux firewall rules.

(ii) IP tables by default consists of several tables such as Filter Table, NAT Table, Mangle Table, Raw Table, Security Table which are used to alter, forward, inspect, drop or redirect IPv4 packets. And each table has built in chains in it which are the roster of rules which are followed in order. For instance Filter Table has INPUT chain which is for the packets coming to the firewall, OUTPUT chain for the packets which are generated locally and going out, FORWARD chain is for the packets routed through the local server. The structure is like this — iptables > Tables > Chains > Rules. Hence, the difference is hierarchically Chains come under Tables.

(iii) A chain Policy is used to change the policy chain rules. For example if we want to accept the connections by default we can use the below mentioned commands

```
iptables --policy INPUT ACCEPT
iptables --policy OUTPUT ACCEPT
iptables --policy FORWARD ACCEPT
```

and if we would like to deny all the connections we can replace ACCEPT with DROP. The policy defines how packets are handled, that get to the end of the chain (i.e. no specific rule has targeted them in the chain), so the default policy for chain is applied. We can set different connection specific responses by using ACCEPT, DROP, REJECT(Don't allow the connection but replies with an error) etc.

(iv) All five default tables exist in our loadgens (filter, mangle, nat, security and raw), as shown in the output below:

```
root@group08-lg4:~# iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
root@group08-lg4:~# iptables -t filter -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@group08-lg4:~# iptables -t mangle -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
root@group08-lg4:~# iptables -t raw -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@group08-lg4:~# iptables -t security -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
```

```
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

2b

```
root@group08-lg4:~# cat /etc/xinetd.d/echo
# default: off
# description: An xinetd internal service which echo's characters back
    to
# clients.
# This is the tcp version.
service echo
{
        disable         = no
        type            = INTERNAL
        id              = echo-stream
        socket_type     = stream
        protocol        = tcp
        user            = root
        wait            = no
}

# This is the udp version.
service echo
{
        disable         = yes
        type            = INTERNAL
        id              = echo-dgram
        socket_type     = dgram
        protocol        = udp
        user            = root
        wait            = yes
}
root@group08-lg4:~# /etc/init.d/xinetd restart
Restarting xinetd (via systemctl): xinetd.service.
```

Here we provide the output from our telnet session with the echo server:

```
root@group08-lg2:~# telnet 10.8.2.2 7
Trying 10.8.2.2...
Connected to 10.8.2.2.
Escape character is '^]'.
hallo
hallo
why is this responding?
why is this responding?
hu
hu

telnet> q
Connection closed.
```

2c

We have installed and familiarized ourselves with iptables tool on our Linux router loadgen which is root@group08-lg3. For the application of rules see the following question.

2d

echo blocked in both directions

The rule we applied:

```
root@group08-lg3:~# iptables -A FORWARD -p tcp --dport 7 -j REJECT
root@group08-lg3:~# iptables-save
# Generated by iptables-save v1.6.0 on Sat Jun  2 13:10:55 2018
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -p tcp -m tcp --dport 7 -j REJECT --reject-with icmp-port-
    unreachable
COMMIT
# Completed on Sat Jun  2 13:10:55 2018
```

Before applying the rule:

```
root@group08-lg3:~# tcpdump -i eth1 port 7
[3130050.359344] device eth1 entered promiscuous mode
tcpdump: verbose output suppressed, use -v or -vv for full protocol
    decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144
    bytes
13:24:35.755208 IP 10.8.1.2.36996 > 10.8.2.2.echo: Flags [S], seq
    3230326158, win 29200, options [mss 1460,sackOK,TS val 20848506 ecr
     0,nop,wscale 7], length 0
13:24:35.755761 IP 10.8.2.2.echo > 10.8.1.2.36996: Flags [S.], seq
    4178751648, ack 3230326159, win 28960, options [mss 1460,sackOK,TS
    val 131455920 ecr 20848506,nop,wscale 7], length 0
13:24:35.756041 IP 10.8.1.2.36996 > 10.8.2.2.echo: Flags [.], ack 1,
    win 229, options [nop,nop,TS val 20848506 ecr 131455920], length 0
13:24:38.337228 IP 10.8.1.2.36988 > 10.8.2.2.echo: Flags [P.], seq
    1256830958:1256830960, ack 628023777, win 229, options [nop,nop,TS
    val 20849152 ecr 131083828], length 2
13:24:38.337972 IP 10.8.2.2.echo > 10.8.1.2.36988: Flags [P.], seq 1:3,
     ack 2, win 227, options [nop,nop,TS val 131456566 ecr 20849152],
    length 2
13:24:38.338502 IP 10.8.1.2.36988 > 10.8.2.2.echo: Flags [P.], seq
    2:24, ack 3, win 229, options [nop,nop,TS val 20849152 ecr
    131456566], length 22
13:24:38.339046 IP 10.8.2.2.echo > 10.8.1.2.36988: Flags [P.], seq
    3:25, ack 24, win 227, options [nop,nop,TS val 131456566 ecr
    20849152], length 22
13:24:38.339406 IP 10.8.1.2.36988 > 10.8.2.2.echo: Flags [P.], seq
    24:30, ack 25, win 229, options [nop,nop,TS val 20849152 ecr
    131456566], length 6
13:24:38.339818 IP 10.8.2.2.echo > 10.8.1.2.36988: Flags [P.], seq
    25:31, ack 30, win 227, options [nop,nop,TS val 131456566 ecr
    20849152], length 6
13:24:38.377160 IP 10.8.1.2.36988 > 10.8.2.2.echo: Flags [.], ack 31,
    win 229, options [nop,nop,TS val 20849162 ecr 131456566], length 0
13:24:38.533684 IP 10.8.1.2.36996 > 10.8.2.2.echo: Flags [P.], seq 1:8,
     ack 1, win 229, options [nop,nop,TS val 20849201 ecr 131455920],
    length 7
13:24:38.534300 IP 10.8.2.2.echo > 10.8.1.2.36996: Flags [.], ack 8,
    win 227, options [nop,nop,TS val 131456615 ecr 20849201], length 0
```

```
13:24:38.534337 IP 10.8.2.2.echo > 10.8.1.2.36996: Flags [P.], seq 1:8,
    ack 8, win 227, options [nop,nop,TS val 131456615 ecr 20849201],
    length 7
13:24:38.534652 IP 10.8.1.2.36996 > 10.8.2.2.echo: Flags [.], ack 8,
    win 229, options [nop,nop,TS val 20849201 ecr 131456615], length 0
13:24:46.008317 IP 10.8.1.2.36996 > 10.8.2.2.echo: Flags [F.], seq 8,
    ack 8, win 229, options [nop,nop,TS val 20851069 ecr 131456615],
    length 0
13:24:46.009363 IP 10.8.2.2.echo > 10.8.1.2.36996: Flags [F.], seq 8,
    ack 9, win 227, options [nop,nop,TS val 131458484 ecr 20851069],
    length 0
13:24:46.009659 IP 10.8.1.2.36996 > 10.8.2.2.echo: Flags [.], ack 9,
    win 229, options [nop,nop,TS val 20851070 ecr 131458484], length 0
^C
17 packets captured
17 packets received by filter
0 packets dropped by kernel
[3130307.169503] device eth1 left promiscuous mode
```

After applying the Rule:

```
root@group08-lg3:~# tcpdump -i eth1 port 7
[3129825.412659] device eth1 entered promiscuous mode
tcpdump: verbose output suppressed, use -v or -vv for full protocol
    decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144
    bytes
13:20:50.053523 IP 10.8.1.2.36994 > 10.8.2.2.echo: Flags [S], seq
    3289436981, win 29200, options [mss 1460,sackOK,TS val 20792082 ecr
     0,nop,wscale 7], length 0
^C
1 packet captured
1 packet received by filter
0 packets dropped by kernel
[3129882.448860] device eth1 left promiscuous mode
```

Block everything except ICMP in both directions

The rule we applied:

```
root@group08-lg3:~# iptables -P FORWARD DROP
root@group08-lg3:~# iptables -A FORWARD -p icmp -j ACCEPT
root@group08-lg3:~# iptables-save
# Generated by iptables-save v1.6.0 on Sat Jun  2 13:30:54 2018
*filter
:INPUT ACCEPT [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -p icmp -j ACCEPT
COMMIT
# Completed on Sat Jun  2 13:30:54 2018
```

Before applying the rule:

```
root@group08-lg3:~# tcpdump -i eth1 net 10.8.1.2
[3130851.793361] device eth1 entered promiscuous mode
tcpdump: verbose output suppressed, use -v or -vv for full protocol
    decode
```

```
listening on eth1, link-type EN10MB (Ethernet), capture size 262144
    bytes
13:37:53.238884 IP 10.8.1.2.36998 > 10.8.2.2.echo: Flags [S], seq
    856152643, win 29200, options [mss 1460,sackOK,TS val 21047873 ecr
    0,nop,wscale 7], length 0
13:37:53.239436 IP 10.8.2.2.echo > 10.8.1.2.36998: Flags [S.], seq
    3013194836, ack 856152644, win 28960, options [mss 1460,sackOK,TS
    val 131655291 ecr 21047873,nop,wscale 7], length 0
13:37:53.239772 IP 10.8.1.2.36998 > 10.8.2.2.echo: Flags [.], ack 1,
    win 229, options [nop,nop,TS val 21047873 ecr 131655291], length 0
13:37:55.802326 IP 10.8.1.2.36998 > 10.8.2.2.echo: Flags [P.], seq 1:8,
     ack 1, win 229, options [nop,nop,TS val 21048514 ecr 131655291],
    length 7
13:37:55.803355 IP 10.8.2.2.echo > 10.8.1.2.36998: Flags [.], ack 8,
    win 227, options [nop,nop,TS val 131655932 ecr 21048514], length 0
13:37:55.803393 IP 10.8.2.2.echo > 10.8.1.2.36998: Flags [P.], seq 1:8,
     ack 8, win 227, options [nop,nop,TS val 131655932 ecr 21048514],
    length 7
13:37:55.804095 IP 10.8.1.2.36998 > 10.8.2.2.echo: Flags [.], ack 8,
    win 229, options [nop,nop,TS val 21048514 ecr 131655932], length 0
13:37:58.252952 ARP, Reply 10.8.1.2 is-at 00:16:3e:af:08:21 (oui
    Unknown), length 46
13:37:59.053618 IP 10.8.1.2.36998 > 10.8.2.2.echo: Flags [F.], seq 8,
    ack 8, win 229, options [nop,nop,TS val 21049327 ecr 131655932],
    length 0
13:37:59.054876 IP 10.8.2.2.echo > 10.8.1.2.36998: Flags [F.], seq 8,
    ack 9, win 227, options [nop,nop,TS val 131656745 ecr 21049327],
    length 0
13:37:59.055208 IP 10.8.1.2.36998 > 10.8.2.2.echo: Flags [.], ack 9,
    win 229, options [nop,nop,TS val 21049327 ecr 131656745], length 0
13:38:05.919739 IP 10.8.1.2 > 10.8.2.2: ICMP echo request, id 1733, seq
     1, length 64
13:38:05.920282 IP 10.8.2.2 > 10.8.1.2: ICMP echo reply, id 1733, seq
    1, length 64
13:38:06.921327 IP 10.8.1.2 > 10.8.2.2: ICMP echo request, id 1733, seq
     2, length 64
13:38:06.922154 IP 10.8.2.2 > 10.8.1.2: ICMP echo reply, id 1733, seq
    2, length 64
13:38:07.923211 IP 10.8.1.2 > 10.8.2.2: ICMP echo request, id 1733, seq
     3, length 64
13:38:07.924092 IP 10.8.2.2 > 10.8.1.2: ICMP echo reply, id 1733, seq
    3, length 64
^C
17 packets captured
17 packets received by filter
0 packets dropped by kernel
[3130874.475718] device eth1 left promiscuous mode
```

After applying the Rule we first see the unsuccessful telnet connection attempts and then the successful ping:

```
root@group08-lg3:~# tcpdump -i eth1 net 10.8.1.2
[3130658.244076] device eth1 entered promiscuous mode
tcpdump: verbose output suppressed, use -v or -vv for full protocol
    decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144
    bytes
```

```
13:34:42.631771 IP 10.8.1.2.36997 > 10.8.2.2.echo: Flags [S], seq
    1380147961, win 29200, options [mss 1460,sackOK,TS val 21000222 ecr
     0,nop,wscale 7], length 0
13:34:43.629556 IP 10.8.1.2.36997 > 10.8.2.2.echo: Flags [S], seq
    1380147961, win 29200, options [mss 1460,sackOK,TS val 21000472 ecr
     0,nop,wscale 7], length 0
13:34:45.633507 IP 10.8.1.2.36997 > 10.8.2.2.echo: Flags [S], seq
    1380147961, win 29200, options [mss 1460,sackOK,TS val 21000973 ecr
     0,nop,wscale 7], length 0
13:34:47.645596 ARP, Request who-has 10.8.1.1 tell 10.8.1.2, length 46
13:34:49.645634 IP 10.8.1.2.36997 > 10.8.2.2.echo: Flags [S], seq
    1380147961, win 29200, options [mss 1460,sackOK,TS val 21001976 ecr
     0,nop,wscale 7], length 0
13:34:57.661727 IP 10.8.1.2.36997 > 10.8.2.2.echo: Flags [S], seq
    1380147961, win 29200, options [mss 1460,sackOK,TS val 21003980 ecr
     0,nop,wscale 7], length 0
13:35:13.862811 IP 10.8.1.2 > 10.8.2.2: ICMP echo request, id 1728, seq
     1, length 64
13:35:13.863390 IP 10.8.2.2 > 10.8.1.2: ICMP echo reply, id 1728, seq
    1, length 64
13:35:14.864285 IP 10.8.1.2 > 10.8.2.2: ICMP echo request, id 1728, seq
     2, length 64
13:35:14.864793 IP 10.8.2.2 > 10.8.1.2: ICMP echo reply, id 1728, seq
    2, length 64
13:35:15.865838 IP 10.8.1.2 > 10.8.2.2: ICMP echo request, id 1728, seq
     3, length 64
13:35:15.866446 IP 10.8.2.2 > 10.8.1.2: ICMP echo reply, id 1728, seq
    3, length 64
13:35:16.867550 IP 10.8.1.2 > 10.8.2.2: ICMP echo request, id 1728, seq
     4, length 64
13:35:16.868121 IP 10.8.2.2 > 10.8.1.2: ICMP echo reply, id 1728, seq
    4, length 64
13:35:18.876885 ARP, Reply 10.8.1.2 is-at 00:16:3e:af:08:21 (oui
    Unknown), length 46
^C
15 packets captured
15 packets received by filter
0 packets dropped by kernel
[3130771.693682] device eth1 left promiscuous mode
```

Ping only works from Host A to Host B

The rule we applied:

```
root@group08-lg3:~# iptables -P FORWARD DROP
root@group08-lg3:~# iptables -A FORWARD -p icmp --icmp-type echo-
    request -s 10.8.1.2 -j ACCEPT
root@group08-lg3:~# iptables -A FORWARD -p icmp --icmp-type echo-reply
    -d 10.8.1.2 -j ACCEPT
root@group08-lg3:~# iptables-save
# Generated by iptables-save v1.6.0 on Sat Jun  2 13:50:16 2018
*filter
:INPUT ACCEPT [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -s 10.8.1.2/32 -p icmp -m icmp --icmp-type 8 -j ACCEPT
```

```
-A FORWARD -d 10.8.1.2/32 -p icmp -m icmp --icmp-type 0 -j ACCEPT
COMMIT
# Completed on Sat Jun  2 13:50:16 2018
```

Before applying the rule:

```
root@group08-lg3:~# tcpdump -i eth1 net 10.8.1.2
[3131061.068345] device eth1 entered promiscuous mode
tcpdump: verbose output suppressed, use -v or -vv for full protocol
    decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144
    bytes
13:41:23.274932 IP 10.8.1.2 > 10.8.2.2: ICMP echo request, id 1737, seq
     1, length 64
13:41:23.275451 IP 10.8.2.2 > 10.8.1.2: ICMP echo reply, id 1737, seq
    1, length 64
13:41:24.276247 IP 10.8.1.2 > 10.8.2.2: ICMP echo request, id 1737, seq
     2, length 64
13:41:24.276777 IP 10.8.2.2 > 10.8.1.2: ICMP echo reply, id 1737, seq
    2, length 64
13:41:25.278237 IP 10.8.1.2 > 10.8.2.2: ICMP echo request, id 1737, seq
     3, length 64
13:41:25.279004 IP 10.8.2.2 > 10.8.1.2: ICMP echo reply, id 1737, seq
    3, length 64
13:41:28.277790 ARP, Request who-has 10.8.1.1 tell 10.8.1.2, length 46
13:41:35.566239 IP 10.8.2.2 > 10.8.1.2: ICMP echo request, id 14907,
    seq 1, length 64
13:41:35.566802 IP 10.8.1.2 > 10.8.2.2: ICMP echo reply, id 14907, seq
    1, length 64
13:41:36.567650 IP 10.8.2.2 > 10.8.1.2: ICMP echo request, id 14907,
    seq 2, length 64
13:41:36.568167 IP 10.8.1.2 > 10.8.2.2: ICMP echo reply, id 14907, seq
    2, length 64
13:41:37.569173 IP 10.8.2.2 > 10.8.1.2: ICMP echo request, id 14907,
    seq 3, length 64
13:41:37.569849 IP 10.8.1.2 > 10.8.2.2: ICMP echo reply, id 14907, seq
    3, length 64
13:41:38.570758 IP 10.8.2.2 > 10.8.1.2: ICMP echo request, id 14907,
    seq 4, length 64
13:41:38.571388 IP 10.8.1.2 > 10.8.2.2: ICMP echo reply, id 14907, seq
    4, length 64
13:41:40.573229 ARP, Reply 10.8.1.2 is-at 00:16:3e:af:08:21 (oui
    Unknown), length 46
^C
16 packets captured
16 packets received by filter
0 packets dropped by kernel
[3131084.960320] device eth1 left promiscuous mode
```

After applying the Rule, we can see, how the ping requests from 10.8.2.2 get no reply because they are dropped at lg3, while it works fine the other way round:

```
root@group08-lg3:~# tcpdump -i eth1 net 10.8.1.2
[3131646.618921] device eth1 entered promiscuous mode
tcpdump: verbose output suppressed, use -v or -vv for full protocol
    decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144
    bytes
```

```
13:51:08.409657 IP 10.8.2.2 > 10.8.1.2: ICMP echo request, id 14909,
    seq 1, length 64
13:51:09.416947 IP 10.8.2.2 > 10.8.1.2: ICMP echo request, id 14909,
    seq 2, length 64
13:51:10.425013 IP 10.8.2.2 > 10.8.1.2: ICMP echo request, id 14909,
    seq 3, length 64
13:51:11.433034 IP 10.8.2.2 > 10.8.1.2: ICMP echo request, id 14909,
    seq 4, length 64
13:51:12.440977 IP 10.8.2.2 > 10.8.1.2: ICMP echo request, id 14909,
    seq 5, length 64
13:51:13.448795 IP 10.8.2.2 > 10.8.1.2: ICMP echo request, id 14909,
    seq 6, length 64
13:51:14.456895 IP 10.8.2.2 > 10.8.1.2: ICMP echo request, id 14909,
    seq 7, length 64
13:51:18.302346 IP 10.8.1.2 > 10.8.2.2: ICMP echo request, id 1743, seq
     1, length 64
13:51:18.302885 IP 10.8.2.2 > 10.8.1.2: ICMP echo reply, id 1743, seq
    1, length 64
13:51:19.304027 IP 10.8.1.2 > 10.8.2.2: ICMP echo request, id 1743, seq
     2, length 64
13:51:19.304755 IP 10.8.2.2 > 10.8.1.2: ICMP echo reply, id 1743, seq
    2, length 64
13:51:20.305972 IP 10.8.1.2 > 10.8.2.2: ICMP echo request, id 1743, seq
     3, length 64
13:51:20.306550 IP 10.8.2.2 > 10.8.1.2: ICMP echo reply, id 1743, seq
    3, length 64
^C
13 packets captured
13 packets received by filter
0 packets dropped by kernel
[3131666.402177] device eth1 left promiscuous mode
```

Everything except ssh connections initiated from Host A to Host B is blocked

The rule we applied:

```
root@group08-lg3:~# iptables -P FORWARD DROP
root@group08-lg3:~# iptables -A FORWARD -p tcp -d 10.8.2.2 --dport 22 -
    j ACCEPT
root@group08-lg3:~# iptables -A FORWARD -p tcp -s 10.8.2.2 --sport 22 -
    j ACCEPT
root@group08-lg3:~# iptables-save
# Generated by iptables-save v1.6.0 on Sat Jun  2 14:08:31 2018
*filter
:INPUT ACCEPT [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -d 10.8.2.2/32 -p tcp -m tcp --dport 22 -j ACCEPT
-A FORWARD -s 10.8.2.2/32 -p tcp -m tcp --sport 22 -j ACCEPT
COMMIT
# Completed on Sat Jun  2 14:08:31 2018
```

Before applying the rule, we can see how ssh'ing from lg4 into lg2 works and also the telnet connection in the other direction is successful:

```
root@group08-lg3:~# tcpdump -i eth1 net 10.8.1.2
[3132951.412943] device eth1 entered promiscuous mode
```

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol
    decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144
    bytes
14:12:53.682607 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [S], seq
    3860880608, win 29200, options [mss 1460,sackOK,TS val 132180403
    ecr 0,nop,wscale 7], length 0
14:12:53.683294 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [S.], seq
    1531441765, ack 3860880609, win 28960, options [mss 1460,sackOK,TS
    val 21572973 ecr 132180403,nop,wscale 7], length 0
14:12:53.683861 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 1, win
     229, options [nop,nop,TS val 132180403 ecr 21572973], length 0
14:12:53.684334 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [P.], seq 1:41,
     ack 1, win 229, options [nop,nop,TS val 132180403 ecr 21572973],
    length 40
14:12:53.684601 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [.], ack 41,
    win 227, options [nop,nop,TS val 21572974 ecr 132180403], length 0
14:12:53.700996 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq 1:40,
     ack 41, win 227, options [nop,nop,TS val 21572978 ecr 132180403],
    length 39
14:12:53.701562 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 40,
    win 229, options [nop,nop,TS val 132180408 ecr 21572978], length 0
14:12:53.702100 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [P.], seq
    41:1473, ack 40, win 229, options [nop,nop,TS val 132180408 ecr
    21572978], length 1432
14:12:53.703843 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    40:1120, ack 1473, win 249, options [nop,nop,TS val 21572979 ecr
    132180408], length 1080
14:12:53.710659 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [P.], seq
    1473:1521, ack 1120, win 245, options [nop,nop,TS val 132180410 ecr
     21572979], length 48
14:12:53.725256 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    1120:1540, ack 1521, win 249, options [nop,nop,TS val 21572984 ecr
    132180410], length 420
14:12:53.764776 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 1540,
    win 262, options [nop,nop,TS val 132180424 ecr 21572984], length 0
14:12:55.844837 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [P.], seq
    1521:1537, ack 1540, win 262, options [nop,nop,TS val 132180943 ecr
     21572984], length 16
14:12:55.884006 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [.], ack 1537,
    win 249, options [nop,nop,TS val 21573524 ecr 132180943], length 0
14:12:55.884517 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [P.], seq
    1537:1581, ack 1540, win 262, options [nop,nop,TS val 132180953 ecr
     21573524], length 44
14:12:55.884921 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [.], ack 1581,
    win 249, options [nop,nop,TS val 21573524 ecr 132180953], length 0
14:12:55.885049 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    1540:1584, ack 1581, win 249, options [nop,nop,TS val 21573524 ecr
    132180953], length 44
14:12:55.885296 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 1584,
    win 262, options [nop,nop,TS val 132180954 ecr 21573524], length 0
14:12:55.885421 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [P.], seq
    1581:1641, ack 1584, win 262, options [nop,nop,TS val 132180954 ecr
     21573524], length 60
14:12:55.889684 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    1584:1636, ack 1641, win 249, options [nop,nop,TS val 21573525 ecr
```

```
    132180954], length 52
14:12:55.928743 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 1636,
    win 262, options [nop,nop,TS val 132180965 ecr 21573525], length 0
14:12:58.684885 ARP, Reply 10.8.1.2 is-at 00:16:3e:af:08:21 (oui
    Unknown), length 46
14:13:00.353458 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [P.], seq
    1641:1789, ack 1636, win 262, options [nop,nop,TS val 132182071 ecr
     21573525], length 148
14:13:00.374296 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    1636:1664, ack 1789, win 271, options [nop,nop,TS val 21574646 ecr
    132182071], length 28
14:13:00.374936 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 1664,
    win 262, options [nop,nop,TS val 132182076 ecr 21574646], length 0
14:13:00.375273 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [P.], seq
    1789:1901, ack 1664, win 262, options [nop,nop,TS val 132182076 ecr
     21574646], length 112
14:13:00.392561 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    1664:2164, ack 1901, win 271, options [nop,nop,TS val 21574651 ecr
    132182076], length 500
14:13:00.432767 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 2164,
    win 279, options [nop,nop,TS val 132182091 ecr 21574651], length 0
14:13:00.433248 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    2164:2208, ack 1901, win 271, options [nop,nop,TS val 21574661 ecr
    132182091], length 44
14:13:00.433706 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 2208,
    win 279, options [nop,nop,TS val 132182091 ecr 21574661], length 0
14:13:00.434002 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [P.], seq
    1901:2329, ack 2208, win 279, options [nop,nop,TS val 132182091 ecr
     21574661], length 428
14:13:00.437254 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    2208:2316, ack 2329, win 294, options [nop,nop,TS val 21574662 ecr
    132182091], length 108
14:13:00.439356 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    2316:2432, ack 2329, win 294, options [nop,nop,TS val 21574662 ecr
    132182091], length 116
14:13:00.439509 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    2432:2468, ack 2329, win 294, options [nop,nop,TS val 21574662 ecr
    132182091], length 36
14:13:00.439656 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    2468:2568, ack 2329, win 294, options [nop,nop,TS val 21574662 ecr
    132182091], length 100
14:13:00.439822 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    2568:2704, ack 2329, win 294, options [nop,nop,TS val 21574662 ecr
    132182091], length 136
14:13:00.439860 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 2432,
    win 279, options [nop,nop,TS val 132182092 ecr 21574662], length 0
14:13:00.440186 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    2704:3012, ack 2329, win 294, options [nop,nop,TS val 21574663 ecr
    132182091], length 308
14:13:00.440219 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 2704,
    win 296, options [nop,nop,TS val 132182092 ecr 21574662], length 0
14:13:00.440380 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    3012:3148, ack 2329, win 294, options [nop,nop,TS val 21574663 ecr
    132182092], length 136
14:13:00.440579 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    3148:3376, ack 2329, win 294, options [nop,nop,TS val 21574663 ecr
```

```
    132182092], length 228
14:13:00.440797 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    3376:3536, ack 2329, win 294, options [nop,nop,TS val 21574663 ecr
    132182092], length 160
14:13:00.440985 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 3148,
    win 330, options [nop,nop,TS val 132182093 ecr 21574663], length 0
14:13:00.441012 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    3536:3664, ack 2329, win 294, options [nop,nop,TS val 21574663 ecr
    132182092], length 128
14:13:00.441020 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    3664:3700, ack 2329, win 294, options [nop,nop,TS val 21574663 ecr
    132182092], length 36
14:13:00.441238 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 3536,
    win 364, options [nop,nop,TS val 132182093 ecr 21574663], length 0
14:13:00.441479 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 3700,
    win 380, options [nop,nop,TS val 132182093 ecr 21574663], length 0
14:13:00.466226 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    3700:3760, ack 2329, win 294, options [nop,nop,TS val 21574669 ecr
    132182093], length 60
14:13:00.504775 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 3760,
    win 380, options [nop,nop,TS val 132182109 ecr 21574669], length 0
14:13:01.589117 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [P.], seq
    2329:2365, ack 3760, win 380, options [nop,nop,TS val 132182380 ecr
     21574669], length 36
14:13:01.590084 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    3760:3804, ack 2365, win 294, options [nop,nop,TS val 21574950 ecr
    132182380], length 44
14:13:01.590124 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    3804:3840, ack 2365, win 294, options [nop,nop,TS val 21574950 ecr
    132182380], length 36
14:13:01.590491 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 3804,
    win 380, options [nop,nop,TS val 132182380 ecr 21574950], length 0
14:13:01.590522 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 3840,
    win 380, options [nop,nop,TS val 132182380 ecr 21574950], length 0
14:13:01.590918 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    3840:3876, ack 2365, win 294, options [nop,nop,TS val 21574950 ecr
    132182380], length 36
14:13:01.591198 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 3876,
    win 380, options [nop,nop,TS val 132182380 ecr 21574950], length 0
14:13:01.591638 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [P.], seq
    3876:4016, ack 2365, win 294, options [nop,nop,TS val 21574950 ecr
    132182380], length 140
14:13:01.591900 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 4016,
    win 397, options [nop,nop,TS val 132182380 ecr 21574950], length 0
14:13:01.592037 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [P.], seq
    2365:2401, ack 4016, win 397, options [nop,nop,TS val 132182380 ecr
     21574950], length 36
14:13:01.592092 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [P.], seq
    2401:2461, ack 4016, win 397, options [nop,nop,TS val 132182380 ecr
     21574950], length 60
14:13:01.592151 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [F.], seq 2461,
     ack 4016, win 397, options [nop,nop,TS val 132182380 ecr
    21574950], length 0
14:13:01.592401 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [.], ack 2462,
    win 294, options [nop,nop,TS val 21574951 ecr 132182380], length 0
14:13:01.595635 IP 10.8.1.2.ssh > 10.8.2.2.37278: Flags [F.], seq 4016,
```

```
    ack 2462, win 294, options [nop,nop,TS val 21574951 ecr
    132182380], length 0
14:13:01.595874 IP 10.8.2.2.37278 > 10.8.1.2.ssh: Flags [.], ack 4017,
    win 397, options [nop,nop,TS val 132182381 ecr 21574951], length 0
14:13:05.851894 IP 10.8.1.2.37003 > 10.8.2.2.echo: Flags [S], seq
    851879398, win 29200, options [mss 1460,sackOK,TS val 21576015 ecr
    0,nop,wscale 7], length 0
14:13:05.852519 IP 10.8.2.2.echo > 10.8.1.2.37003: Flags [S.], seq
    34246357, ack 851879399, win 28960, options [mss 1460,sackOK,TS val
    132183445 ecr 21576015,nop,wscale 7], length 0
14:13:05.853554 IP 10.8.1.2.37003 > 10.8.2.2.echo: Flags [.], ack 1,
    win 229, options [nop,nop,TS val 21576016 ecr 132183445], length 0
14:13:07.909009 IP 10.8.1.2.37003 > 10.8.2.2.echo: Flags [P.], seq
    1:10, ack 1, win 229, options [nop,nop,TS val 21576530 ecr
    132183445], length 9
14:13:07.909592 IP 10.8.2.2.echo > 10.8.1.2.37003: Flags [.], ack 10,
    win 227, options [nop,nop,TS val 132183960 ecr 21576530], length 0
14:13:07.909627 IP 10.8.2.2.echo > 10.8.1.2.37003: Flags [P.], seq
    1:10, ack 10, win 227, options [nop,nop,TS val 132183960 ecr
    21576530], length 9
14:13:07.910125 IP 10.8.1.2.37003 > 10.8.2.2.echo: Flags [.], ack 10,
    win 229, options [nop,nop,TS val 21576530 ecr 132183960], length 0
14:13:12.146855 IP 10.8.1.2.37003 > 10.8.2.2.echo: Flags [F.], seq 10,
    ack 10, win 229, options [nop,nop,TS val 21577589 ecr 132183960],
    length 0
14:13:12.148054 IP 10.8.2.2.echo > 10.8.1.2.37003: Flags [F.], seq 10,
    ack 11, win 227, options [nop,nop,TS val 132185019 ecr 21577589],
    length 0
14:13:12.148426 IP 10.8.1.2.37003 > 10.8.2.2.echo: Flags [.], ack 11,
    win 229, options [nop,nop,TS val 21577590 ecr 132185019], length 0
^C
74 packets captured
74 packets received by filter
0 packets dropped by kernel
[3132978.867792] device eth1 left promiscuous mode
```

After applying the Rule we see the successful connection from lg2 to lg4 and the unsuccessful attempt the other way round:

```
root@group08-lg3:~# tcpdump -i eth1 net 10.8.1.2
[3132807.267813] device eth1 entered promiscuous mode
tcpdump: verbose output suppressed, use -v or -vv for full protocol
    decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144
    bytes
14:10:28.253600 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [S], seq
    2956215231, win 29200, options [mss 1460,sackOK,TS val 21536617 ecr
    0,nop,wscale 7], length 0
14:10:28.254314 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [S.], seq
    2760038621, ack 2956215232, win 28960, options [mss 1460,sackOK,TS
    val 132144046 ecr 21536617,nop,wscale 7], length 0
14:10:28.255123 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 1, win
    229, options [nop,nop,TS val 21536617 ecr 132144046], length 0
14:10:28.255881 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [P.], seq 1:41,
    ack 1, win 229, options [nop,nop,TS val 21536617 ecr 132144046],
    length 40
14:10:28.256993 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [.], ack 41,
```

```
    win 227, options [nop,nop,TS val 132144046 ecr 21536617], length 0
14:10:28.273533 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq 1:40,
     ack 41, win 227, options [nop,nop,TS val 132144051 ecr 21536617],
    length 39
14:10:28.274078 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 40,
    win 229, options [nop,nop,TS val 21536622 ecr 132144051], length 0
14:10:28.275251 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [P.], seq
    41:1473, ack 40, win 229, options [nop,nop,TS val 21536622 ecr
    132144051], length 1432
14:10:28.276507 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    40:1120, ack 1473, win 249, options [nop,nop,TS val 132144051 ecr
    21536622], length 1080
14:10:28.283312 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [P.], seq
    1473:1521, ack 1120, win 245, options [nop,nop,TS val 21536624 ecr
    132144051], length 48
14:10:28.297913 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    1120:1540, ack 1521, win 249, options [nop,nop,TS val 132144057 ecr
     21536624], length 420
14:10:28.306534 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [P.], seq
    1521:1537, ack 1540, win 262, options [nop,nop,TS val 21536630 ecr
    132144057], length 16
14:10:28.344975 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [.], ack 1537,
    win 249, options [nop,nop,TS val 132144069 ecr 21536630], length 0
14:10:28.345378 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [P.], seq
    1537:1581, ack 1540, win 262, options [nop,nop,TS val 21536640 ecr
    132144069], length 44
14:10:28.345740 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [.], ack 1581,
    win 249, options [nop,nop,TS val 132144069 ecr 21536640], length 0
14:10:28.345938 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    1540:1584, ack 1581, win 249, options [nop,nop,TS val 132144069 ecr
     21536640], length 44
14:10:28.346360 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [P.], seq
    1581:1641, ack 1584, win 262, options [nop,nop,TS val 21536640 ecr
    132144069], length 60
14:10:28.351140 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    1584:1636, ack 1641, win 249, options [nop,nop,TS val 132144070 ecr
     21536640], length 52
14:10:28.388881 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 1636,
    win 262, options [nop,nop,TS val 21536651 ecr 132144070], length 0
14:10:34.225129 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [P.], seq
    1641:1789, ack 1636, win 262, options [nop,nop,TS val 21538110 ecr
    132144070], length 148
14:10:34.246045 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    1636:1664, ack 1789, win 271, options [nop,nop,TS val 132145544 ecr
     21538110], length 28
14:10:34.246450 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 1664,
    win 262, options [nop,nop,TS val 21538115 ecr 132145544], length 0
14:10:34.246680 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [P.], seq
    1789:1901, ack 1664, win 262, options [nop,nop,TS val 21538115 ecr
    132145544], length 112
14:10:34.264353 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    1664:2164, ack 1901, win 271, options [nop,nop,TS val 132145548 ecr
     21538115], length 500
14:10:34.301034 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 2164,
    win 279, options [nop,nop,TS val 21538129 ecr 132145548], length 0
14:10:34.301568 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
```

```
    2164:2208, ack 1901, win 271, options [nop,nop,TS val 132145558 ecr
     21538129], length 44
14:10:34.301964 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 2208,
    win 279, options [nop,nop,TS val 21538129 ecr 132145558], length 0
14:10:34.302325 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [P.], seq
    1901:2897, ack 2208, win 279, options [nop,nop,TS val 21538129 ecr
    132145558], length 996
14:10:34.305135 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    2208:2316, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538129], length 108
14:10:34.307111 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    2316:2432, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538129], length 116
14:10:34.307280 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    2432:2468, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538129], length 36
14:10:34.307332 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    2468:2568, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538129], length 100
14:10:34.307552 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    2568:2604, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538129], length 36
14:10:34.307579 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    2604:2704, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538129], length 100
14:10:34.307652 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 2432,
    win 279, options [nop,nop,TS val 21538130 ecr 132145559], length 0
14:10:34.307728 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    2704:2740, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538129], length 36
14:10:34.307983 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    2740:2840, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538129], length 100
14:10:34.308352 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 2704,
    win 279, options [nop,nop,TS val 21538130 ecr 132145559], length 0
14:10:34.308441 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    2840:3112, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538130], length 272
14:10:34.308471 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    3112:3148, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538130], length 36
14:10:34.308485 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    3148:3248, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538130], length 100
14:10:34.308512 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    3248:3284, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538130], length 36
14:10:34.308526 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    3284:3376, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538130], length 92
14:10:34.308539 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 2840,
    win 279, options [nop,nop,TS val 21538130 ecr 132145559], length 0
14:10:34.308840 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
    3412:3572, ack 2897, win 294, options [nop,nop,TS val 132145559 ecr
     21538130], length 160
14:10:34.308876 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 3376,
```

```
   win 296, options [nop,nop,TS val 21538131 ecr 132145559], length 0
14:10:34.308885 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
   3572:3664, ack 2897, win 294, options [nop,nop,TS val 132145560 ecr
    21538130], length 92
14:10:34.309093 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
   3664:3700, ack 2897, win 294, options [nop,nop,TS val 132145560 ecr
    21538130], length 36
14:10:34.309307 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 3664,
   win 313, options [nop,nop,TS val 21538131 ecr 132145559], length 0
14:10:34.334419 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
   3700:3760, ack 2897, win 294, options [nop,nop,TS val 132145566 ecr
    21538131], length 60
14:10:34.335141 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 3760,
   win 313, options [nop,nop,TS val 21538137 ecr 132145560], length 0
14:10:36.472708 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [P.], seq
   2897:2933, ack 3760, win 313, options [nop,nop,TS val 21538671 ecr
   132145560], length 36
14:10:36.473773 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
   3760:3804, ack 2933, win 294, options [nop,nop,TS val 132146101 ecr
    21538671], length 44
14:10:36.473892 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
   3804:3840, ack 2933, win 294, options [nop,nop,TS val 132146101 ecr
    21538671], length 36
14:10:36.474355 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 3840,
   win 313, options [nop,nop,TS val 21538672 ecr 132146101], length 0
14:10:36.474653 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
   3840:3876, ack 2933, win 294, options [nop,nop,TS val 132146101 ecr
    21538671], length 36
14:10:36.475371 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [P.], seq
   3876:4016, ack 2933, win 294, options [nop,nop,TS val 132146101 ecr
    21538672], length 140
14:10:36.475708 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 4016,
   win 330, options [nop,nop,TS val 21538672 ecr 132146101], length 0
14:10:36.475826 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [P.], seq
   2933:2969, ack 4016, win 330, options [nop,nop,TS val 21538672 ecr
   132146101], length 36
14:10:36.475845 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [P.], seq
   2969:3029, ack 4016, win 330, options [nop,nop,TS val 21538672 ecr
   132146101], length 60
14:10:36.476157 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [F.], seq 3029,
    ack 4016, win 330, options [nop,nop,TS val 21538672 ecr
   132146101], length 0
14:10:36.476183 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [.], ack 3029,
   win 294, options [nop,nop,TS val 132146101 ecr 21538672], length 0
14:10:36.479635 IP 10.8.2.2.ssh > 10.8.1.2.43870: Flags [F.], seq 4016,
    ack 3030, win 294, options [nop,nop,TS val 132146102 ecr
   21538672], length 0
14:10:36.479985 IP 10.8.1.2.43870 > 10.8.2.2.ssh: Flags [.], ack 4017,
   win 330, options [nop,nop,TS val 21538673 ecr 132146102], length 0
14:10:40.728838 IP 10.8.2.2.37277 > 10.8.1.2.ssh: Flags [S], seq
   470153386, win 29200, options [mss 1460,sackOK,TS val 132147164 ecr
    0,nop,wscale 7], length 0
14:10:41.725097 IP 10.8.2.2.37277 > 10.8.1.2.ssh: Flags [S], seq
   470153386, win 29200, options [mss 1460,sackOK,TS val 132147414 ecr
    0,nop,wscale 7], length 0
14:10:43.729012 IP 10.8.2.2.37277 > 10.8.1.2.ssh: Flags [S], seq
```

```
    470153386, win 29200, options [mss 1460,sackOK,TS val 132147915 ecr
     0,nop,wscale 7], length 0
^C
67 packets captured
68 packets received by filter
1 packet dropped by kernel
[3132835.872959] device eth1 left promiscuous mode
```

Question 3

3a

We used the following rules. Note that we consider UDP and TCP for DNS and we allow packets to flow between the complete /24 subnets. (It could easily be reduced to just the IP addresses of host A and B, if that was required.).

```
root@group08-lg3:~# iptables-save
# Generated by iptables-save v1.6.0 on Mon Jun  4 15:05:01 2018
*filter
:INPUT ACCEPT [5:1640]
:FORWARD DROP [11:682]
:OUTPUT ACCEPT [0:0]
-A FORWARD -m state --state INVALID -j DROP
-A FORWARD -s 10.8.1.0/24 -d 10.8.2.0/24 -p tcp -m tcp --dport 22 -j
    ACCEPT
-A FORWARD -s 10.8.1.0/24 -d 10.8.2.0/24 -p tcp -m tcp --dport 80 -j
    ACCEPT
-A FORWARD -s 10.8.1.0/24 -d 10.8.2.0/24 -p tcp -m tcp --dport 53 -j
    ACCEPT
-A FORWARD -s 10.8.1.0/24 -d 10.8.2.0/24 -p udp -m udp --dport 53 -j
    ACCEPT
-A FORWARD -s 10.8.2.0/24 -d 10.8.1.0/24 -m state --state ESTABLISHED -
    j ACCEPT
-A FORWARD -j LOG
COMMIT
# Completed on Mon Jun  4 15:05:01 2018
```

We decided to display only one dropped and logged packet per service to make it easier to read:

```
root@group08-lg3:~# [3308657.819293] IN=eth1.84 OUT=eth1.82 MAC=00:16:3
    e:af:08:31:00:16:3e:af:08:41:08:00:45:10:00:3c SRC=10.8.2.2 DST
    =10.8.1.2 LEN=60 TOS=0x10 PREC=0x00 TTL=63 ID=43973 DF PROTO=TCP
    SPT=55903 DPT=80 WINDOW=29200 RES=0x00 SYN URGP=0
[3308674.427909] IN=eth1.84 OUT=eth1.82 MAC=00:16:3e:af:08:31:00:16:3e:
    af:08:41:08:00:45:10:00:3c SRC=10.8.2.2 DST=10.8.1.2 LEN=60 TOS=0
    x10 PREC=0x00 TTL=63 ID=34193 DF PROTO=TCP SPT=38602 DPT=53 WINDOW
    =29200 RES=0x00 SYN URGP=0
[3308781.909038] IN=eth1.84 OUT=eth1.82 MAC=00:16:3e:af:08:31:00:16:3e:
    af:08:41:08:00:45:00:00:22 SRC=10.8.2.2 DST=10.8.1.2 LEN=34 TOS=0
    x00 PREC=0x00 TTL=63 ID=30506 DF PROTO=UDP SPT=40159 DPT=53 LEN=14
[3309114.326371] IN=eth1.84 OUT=eth1.82 MAC=00:16:3e:af:08:31:00:16:3e:
    af:08:41:08:00:45:00:00:3c SRC=10.8.2.2 DST=10.8.1.2 LEN=60 TOS=0
    x00 PREC=0x00 TTL=63 ID=38982 DF PROTO=TCP SPT=37292 DPT=22 WINDOW
    =29200 RES=0x00 SYN URGP=0
```

3b

We used curl because the standard linux cli ftp client does not accept root credentials and sftp tunnels over ssh which avoids the issues we want to show here. The download works fine:

```
root@group08-lg3:~# curl -O ftp://10.8.2.2/test.txt
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
       Current
                                 Dload  Upload   Total   Spent    Left
                                    Speed
100     5  100     5    0     0    147      0 --:--:-- --:--:--
    --:--:--   151
```

The output when trying to access the ftp server from lg2 shows that a connection is never established. Obviously because the packet is dropped due to our iptables rules from 3a. Here we also show the ftp output:

```
root@group08-lg2:~# curl -O ftp://10.8.2.2/test.txt
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
       Current
                                 Dload  Upload   Total   Spent    Left
                                    Speed
  0     0    0     0    0     0      0      0 --:--:--  0:00:12
     --:--:--     0^C
root@group08-lg2:~# ftp 10.8.2.2
ftp: connect: Connection timed out
ftp> quit
```

After applying the following rule set in the FORWARD chain, ftp works. The reason is that ftp uses different ftp connections for control and data flow. iptables can recognize the new connection and allow it, if we configure related states to be accepted.

```
root@group08-lg3:~# iptables-save
# Generated by iptables-save v1.6.0 on Mon Jun  4 15:57:37 2018
*filter
:INPUT ACCEPT [28:2427]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [27:1663]
-A FORWARD -m state --state INVALID -j DROP
-A FORWARD -s 10.8.1.0/24 -d 10.8.2.0/24 -p tcp -m tcp --dport 22 -j
    ACCEPT
-A FORWARD -s 10.8.1.0/24 -d 10.8.2.0/24 -p tcp -m tcp --dport 80 -j
    ACCEPT
-A FORWARD -s 10.8.1.0/24 -d 10.8.2.0/24 -p tcp -m tcp --dport 53 -j
    ACCEPT
-A FORWARD -s 10.8.1.0/24 -d 10.8.2.0/24 -p udp -m udp --dport 53 -j
    ACCEPT
-A FORWARD -s 10.8.1.0/24 -d 10.8.2.0/24 -p tcp -m tcp --dport 21 -j
    ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG
COMMIT
# Completed on Mon Jun  4 15:57:37 2018
```

3c

We specify the port range (which is also the default) and set -Pn parameter to skip host discovery, because we already know that lg4 is up and running. We do not need to specify TCP as it is the default protocol nmap will use. (In this case we also do not do anything fancy such as Synscans

or Xmasscans.) The output shows which ports are open and which ports are closed. We can see the closed state, because these ports are allowed to be accesses by our iptables rules, but there are no services listening on these ports. So lg2 will send an appropriate ICMP message to let us know. The first two ports are open, so we can speak to the services listening on the other side. Everything else is filtered by the firewall on lg3. The services nmap show are not 100% reliable. It is the standard services running on these ports, but of course theoretically people could run different things on these ports and this simple scan might not find out about that (for further information on how to find out about specific services listening on a port see below).

```
root@group08-lg2:~# nmap 10.8.2.2 -p0-1024 -Pn

Starting Nmap 7.40 ( https://nmap.org ) at 2018-06-04 16:02 UTC
Nmap scan report for 10.8.2.2
Host is up (0.014s latency).
Not shown: 1021 filtered ports
PORT   STATE  SERVICE
21/tcp open   ftp
22/tcp open   ssh
53/tcp closed domain
80/tcp closed http

Nmap done: 1 IP address (1 host up) scanned in 10.92 seconds
```

We use -O parameter to find out which OS is running on lg2 and it tells us which kernel we are running (Linux 3.10 - 4.2):

```
root@group08-lg2:~# nmap -O 10.8.2.2

Starting Nmap 7.40 ( https://nmap.org ) at 2018-06-04 16:05 UTC
Nmap scan report for 10.8.2.2
Host is up (0.0013s latency).
Not shown: 996 filtered ports
PORT   STATE  SERVICE
21/tcp open   ftp
22/tcp open   ssh
53/tcp closed domain
80/tcp closed http
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 3.10 - 4.2

OS detection performed. Please report any incorrect results at https://
    nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 22.85 seconds
```

However we can dig deeper by finding out something about the services listening on the ports that are not closed/filtered. The -A parameter also tells us, that we are running Debian 10+deb9u3. Of course these values do not always figure out the correct parameters, but still it can be useful (for an attacker):

```
root@group08-lg2:~# nmap -A 10.8.2.2

Starting Nmap 7.40 ( https://nmap.org ) at 2018-06-04 16:04 UTC
Nmap scan report for 10.8.2.2
Host is up (0.0064s latency).
Not shown: 996 filtered ports
PORT   STATE  SERVICE VERSION
21/tcp open   ftp     vsftpd 3.0.3
```

```
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-rw-r--r--    1 0        0               5 Jun 04 15:11 test.txt
22/tcp open  ssh     OpenSSH 7.4p1 Debian 10+deb9u3 (protocol 2.0)
| ssh-hostkey:
|   2048 af:66:61:33:6e:ed:74:63:e9:01:5e:94:29:55:62:f9 (RSA)
|_  256 12:5d:00:d9:d4:a2:13:20:29:14:4a:e3:1c:45:46:82 (ECDSA)
53/tcp closed domain
80/tcp closed http
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 3.10 - 4.2
Network Distance: 2 hops
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 80/tcp)
HOP RTT     ADDRESS
1   0.63 ms 10.8.1.1
2   0.97 ms 10.8.2.2

OS and Service detection performed. Please report any incorrect results
    at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 23.55 seconds
```

Question 4

4a

tc is the binary in the iproute2 package which is used for traffic control by performing configuration of the kernel structures. It takes qdisc, class or filter as its first non option argument. qdisc is also known as queuing discipline and it is a scheduler which is used to arrange or rearrange packets between input and output of a queue. Classful qdiscs as the name suggests contains classes which have filters attached to it which selects child class or the application of filter to drop or reclassify the traffic entering a particular class. We can basically set rules to filter packets into each class. For instance Hierarchical Token Bucket(HTB). Classless qdiscs as the name indicates are without classes and we can not attach filter to it as well. It is also not possible to classify because it has no children of any kind. It doesn't allow to add more qdiscs to it. It does the basic management of traffic by slowing, reordering or dropping packets. For example fifofast, Tocken Bucket Filter(TBF), Stochastic Fairness Queuing(SFQ), CoDel and Fair queuing CoDel. IPTables and tc both can be used together to classify and shape traffic by using the method of iptables known as fwmark which is used to mark packets across interfaces which then be processed by the class.

4b

First we will run iperf server on Host B by using this command 'iperf -s' and then we will run iperf client on Host A 'iperf -c 10.8.2.2' then we can get the output on our terminal with the bandwidth.

4c

The estimated average bandwidth between Host A and Host B is 47.86 Kbits/sec which we got by taking the average of the three iterations of the below mentioned output.

```
root@group08-lg4:~# iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
```

```
[  4] local 10.8.2.2 port 5001 connected with 10.8.1.2 port 50919
[  5] local 10.8.2.2 port 5001 connected with 10.8.1.2 port 50920
[  6] local 10.8.2.2 port 5001 connected with 10.8.1.2 port 50921

root@group08-lg2:~# iperf -c 10.8.2.2
```

```
Client connecting to 10.8.2.2, TCP port 5001
TCP window size: 85.0 KByte (default)
```

```
[  3] local 10.8.1.2 port 50919 connected with 10.8.2.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-15.0 sec  82.0 KBytes  44.8 Kbits/sec
root@group08-lg2:~# iperf -c 10.8.2.2
```

```
Client connecting to 10.8.2.2, TCP port 5001
TCP window size: 85.0 KByte (default)
```

```
[  3] local 10.8.1.2 port 50920 connected with 10.8.2.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-15.0 sec  90.5 KBytes  49.4 Kbits/sec
root@group08-lg2:~# iperf -c 10.8.2.2
```

```
Client connecting to 10.8.2.2, TCP port 5001
TCP window size: 85.0 KByte (default)
```

```
[  3] local 10.8.1.2 port 50921 connected with 10.8.2.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-15.0 sec  90.5 KBytes  49.4 Kbits/sec
```

4d

We ran iperf server on lg4 and client on lg2. We sent the testfile of size 100MB from lg2 to lg4 by using below mentioned command

```
root@group08-lg2:~# scp testfile root@10.8.2.2:/
root@10.8.2.2's password:
testfile                                      100%  100MB  21.7MB/s
   00:04
```

In the meantime we used screen and took three measurements to get the average bandwidth while transmitting a 100MB file via scp which is 400.5 Mbits/sec. Please check the below mentioned output for details.

```
root@group08-lg2:~# iperf -c 10.8.2.2
```

```
Client connecting to 10.8.2.2, TCP port 5001
TCP window size: 85.0 KByte (default)
```

```
[  3] local 10.8.1.2 port 34556 connected with 10.8.2.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   490 MBytes   411 Mbits/sec
Client connecting to 10.8.2.2, TCP port 5001
TCP window size: 85.0 KByte (default)
```

```
[  3] local 10.8.1.2 port 34557 connected with 10.8.2.2 port 5001
[ ID] Interval       Transfer     Bandwidth
```

```
[  3]  0.0-10.0 sec   487 MBytes   408 Mbits/sec
root@group08-lg2:~# iperf -c 10.8.2.2
------------------------------------------------------------
Client connecting to 10.8.2.2, TCP port 5001
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  3] local 10.8.1.2 port 34559 connected with 10.8.2.2 port 5001
^[[A^[[B^[[B^[[D[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   394 MBytes   330 Mbits/sec
root@group08-lg2:~# iperf -c 10.8.2.2
------------------------------------------------------------
Client connecting to 10.8.2.2, TCP port 5001
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  3] local 10.8.1.2 port 34560 connected with 10.8.2.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   540 MBytes   453 Mbits/sec
```

4e

To configure Linux router using tc we first execute this command 'apt-get install iproute2' because tc is bundled with iproute2 package in Debian. As per the hint provided in the question we assume we have to do it with TBF which is Tocken Bucket Filter and it is used to slow down the interface. We will execute the below mentioned command to limit the bandwidth between our two test hosts to 300 Mbps. "tc qdisc add dev eth1 root tbf rate 300mbps burst 24000kb latency 400ms"

4f

Typical values for ADSL are 8Mbps down- and 1Mbps upstream.

```
#first mark traffic for up and downstream
iptables -t mangle -A FORWARD -s 10.8.1.2 -j MARK --set-mark 10
iptables -t mangle -A FORWARD -d 10.8.1.2 -j MARK --set-mark 20
#check how to use different tbf based on this marking?!
tc qdisc add dev eth1 root tbf rate 8mbit burst 3000kb latency 400ms
tc qdisc add dev eth1 root tbf rate 1mbit burst 3000kb latency 400ms
```

4g

We cannot shape the incoming traffic with tc, because we do not control it. Our idea is to use HTB in the following way: We create a root class and give it the desired download rate as guaranteed and ceil rate 8000kbit/s. We then create a child class for downloading and give it the desired rate as guaranteed (ceil rate is the same). At the same hierarchy level (leaf) we create another child class to our first class and give it a very low guaranteed rate of just 1kbps, but a ceil rate of the maximum upstream: 1000kbit/s. This way we ensure that the upload link can only borrow tokens from the super class when the download is not using all of these tokens. Therefore the download always works at maximum ADSL speed while the upload only works as fast as it will not affect the download.

We are not 100% certain which are the correct commands. This is what we would do:

```
tc qdisc add dev eth1 root handle 1: htb
tc class add dev eth1 parent 1: classid 1:1 htb rate 8000kbit ceil 8000
   kbit
tc class add dev eth1 parent 1: classid 1:20 htb rate 8000kbit ceil
   8000kbit
tc class add dev eth1 parent 1: classid 1:30 htb rate 1kbit ceil 1000
   kbit
```

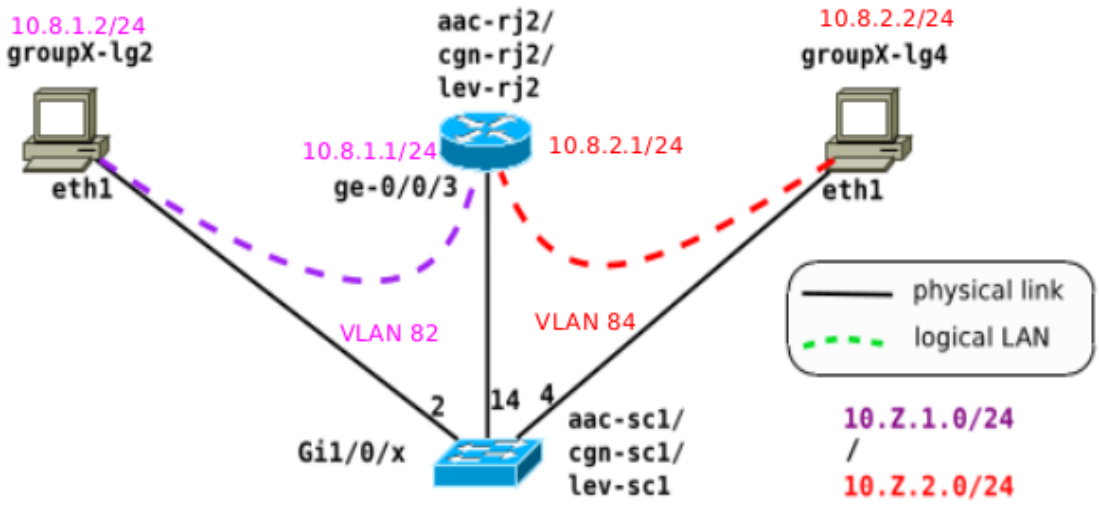

Figure 2: The topology we use for Question 5.

However, we did not find out how to apply the qdisc to both ingress and egress, because it seems it cannot be applied to ingress.

Question 5

We are using the topology as shown in Figure 2.

The configuration of sc1:

```
!
interface GigabitEthernet1/0/2
 description lg2
 switchport access vlan 82
 switchport mode access
!
interface GigabitEthernet1/0/4
 description lg4
 switchport access vlan 84
 switchport mode access
!
interface GigabitEthernet1/0/14
 description lev-rj2
 switchport mode trunk
!
```

The interface configuration of rj2:

```
ge-0/0/3 {
    description lev-sc1;
    vlan-tagging;
    unit 82 {
        vlan-id 82;
        family inet {
            address 10.8.1.1/24;
        }
    }
    unit 84 {
```

```
            vlan-id 84;
            family inet {
                address 10.8.2.1/24;
            }
        }
    }
```

The traceroute from lg2 to lg4 (forwarded by rj2):

```
root@group08-lg2:~# traceroute 10.8.2.2
traceroute to 10.8.2.2 (10.8.2.2), 30 hops max, 60 byte packets
 1  10.8.1.1 (10.8.1.1)  0.563 ms  0.508 ms  0.579 ms
 2  10.8.2.2 (10.8.2.2)  1.161 ms  1.127 ms  1.085 ms
```

5a

In Table 1 we provide an overview over the commands for access lists and the different existing types on Cisco routers. The ACL type is usually specified by the number we use for the access-list

5b

It is a bit weird to configure it without being able to test it. So here is, what we came up with. In the configuration mode:

```
ip access-list extended sshOnly
 permit tcp host 10.8.1.2 host 10.8.2.2 eq 22 reflect sshOnl timeout
    300
 deny   ip any any
```

Then we activate it in the interfaces where we need it by using these commands:

```
 ip access-group sshOnly in
 ip access-group sshOnly out
```

We might want to use different filter for in and outbound traffic.

5c

Here is how we configured the filter. It rejects all traffic except for icmp-requests from host A and for icmp-replies from host b. Of course we could also add the destination-address in both terms to be more secure, but given our current topology this configuration is actually enough.

```
root@lev-rj2# show firewall
family inet {
    filter test {
        term 1 {
            from {
                source-address {
                    10.8.1.2/32;
                }
                icmp-type echo-request;
            }
            then accept;
        }
        term 2 {
            from {
                source-address {
                    10.8.2.2/32;
                }
```

ACL	function	example command
Standard	allow filtering only based on the source address (except for 2 notable exceptions); are very efficient due to their simplicity	`access-list 2 deny any log`
Extended	allow filtering based on source, destination address/-port and used protocols. allow for much more complex filtering (more similar to iptables than the standard variant); allows complex filtering based on TCP flags, time etc.	`access-list 101 deny udp any any`
Reflexive	similar to stateful filtering in iptables; they allow filtering based on state, e.g. allowing connections only to be initiated from one side and not the other by keeping state of active connections. does not work with more complex protocols such as ftp which opens further connections from the remote site	`ip access-list extended 101; permit udp any any reflect no-udp`
Lock and Key Dynamics	allows to grant access based on user (using authentication); each user may use different traffic; works dynamically; useful if only a particular remote host is allowed to access a host within a network; it will then authenticate and be granted access although there generally is a firewall preventing it for unauthenticated hosts	`ip access-list extended 101; dynamic listname permit udp any any`
CBAC	works similar to the beforementioned ACLs, but allows inspection of router specific traffic, this ensures the protocol used has not been tampered with, which will protect the systems behind the firewall; CBAC also provides stateful packet filtering and can filter based on common application layer protocol, thus providing a minimal intrusion detection system; upon an attack it will reset the respective connections	`ip inspect name listname tcp alert on`

Table 1: Cisco access list overview.

```
                icmp-type echo-reply;
            }
            then accept;
        }
        term 3 {
            then {
                reject;
            }
        }
    }
}
```

[edit]

We tested this configuration and it behaves as intended. (The only traffic that is not dropped is when we ping host B from host A. Everything else does not work.)

Included Files

q05-config-rj2.txt