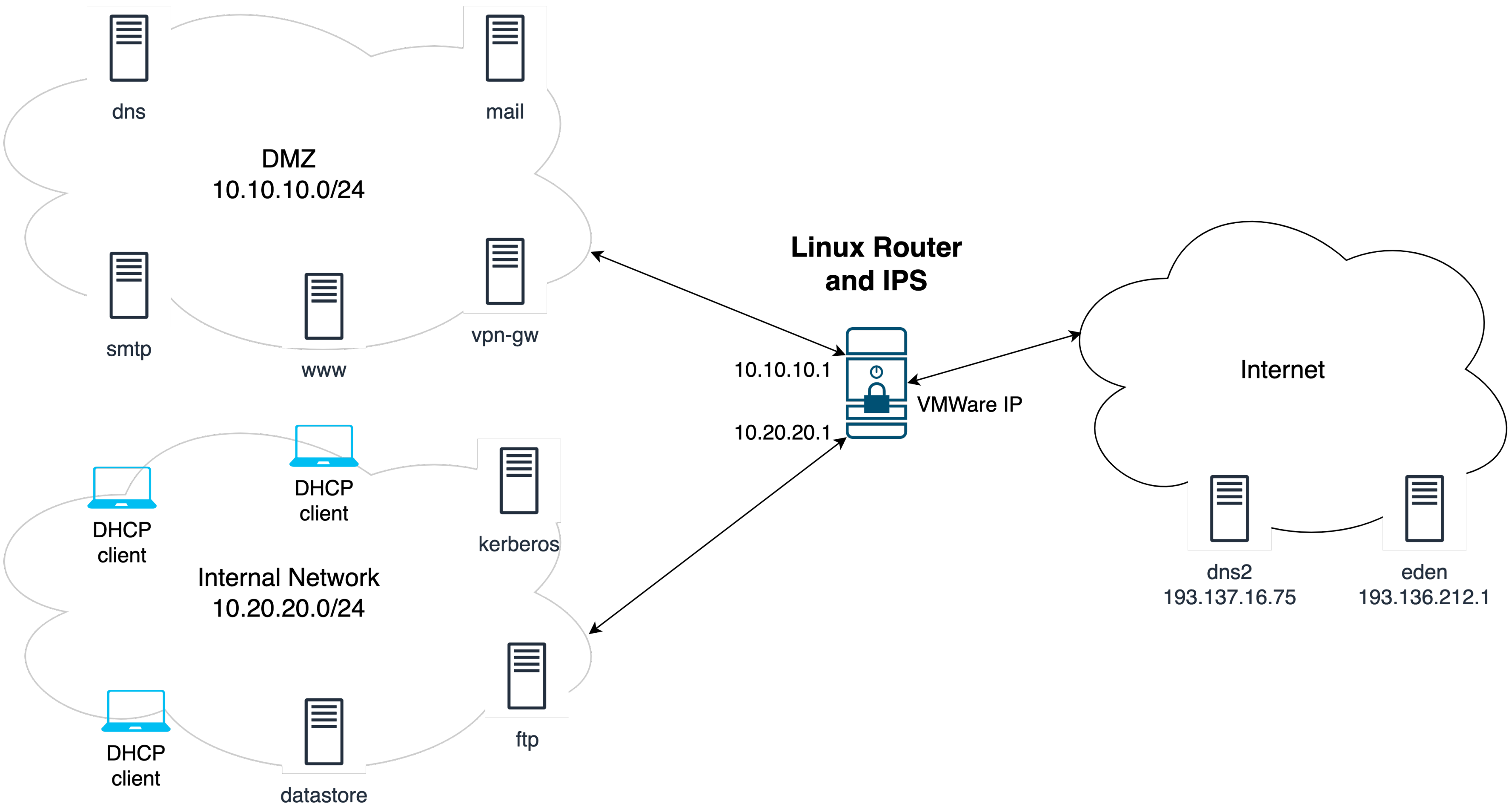
STI
MEI/MIEBOM
2021/2022

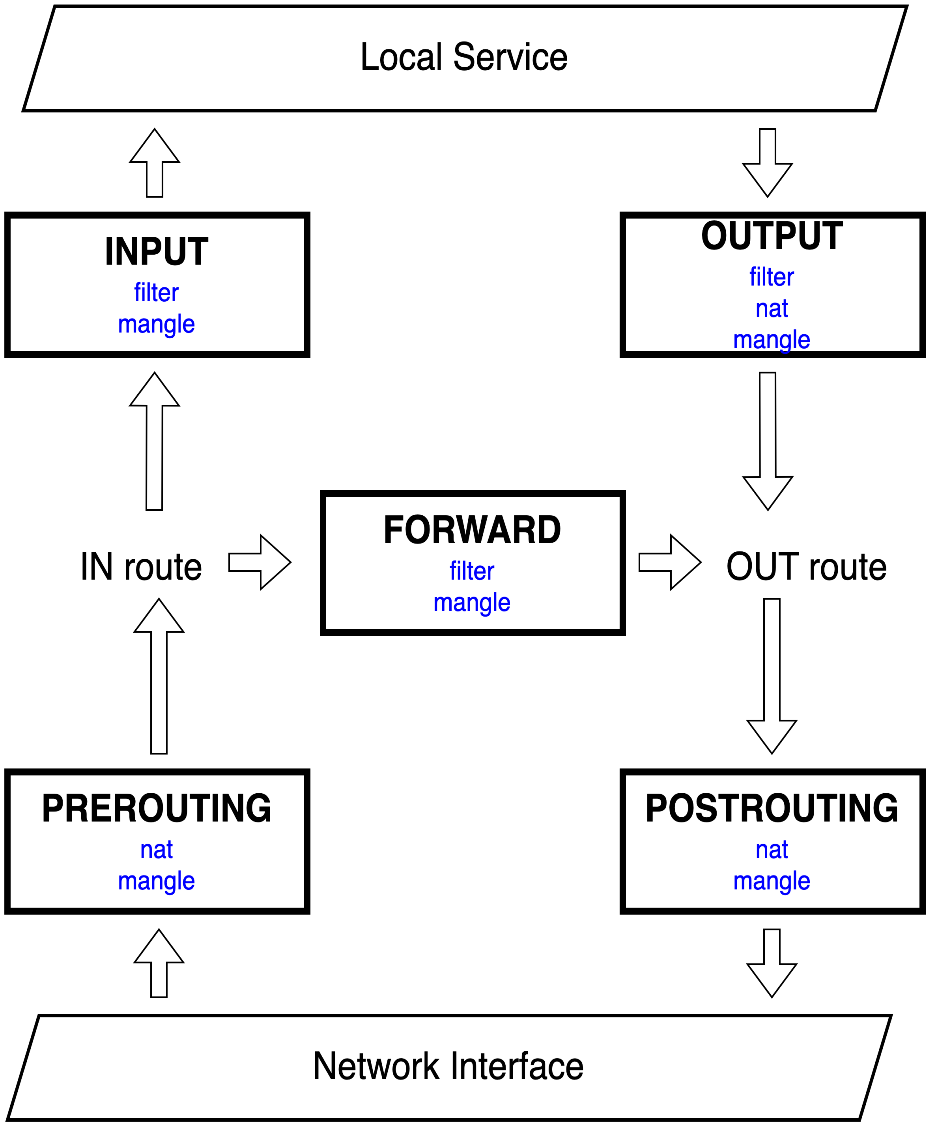
Practical Assignment #2

- Firewalls using IPTables
- Intrusion detection and prevention using Snort

Scenario

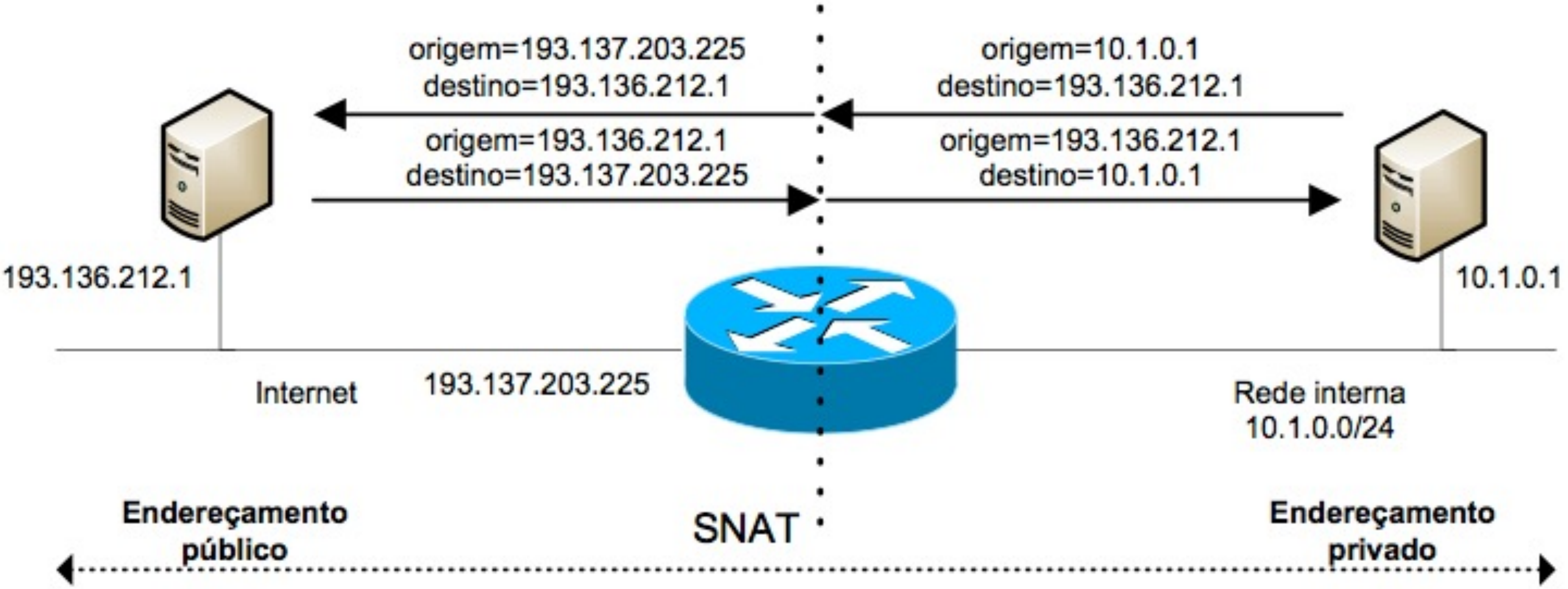

dns
mail
DMZ
10.10.10.0/24
smtp
www
vpn-gw
DHCP client
DHCP client
kerberos
Internal Network
10.20.20.0/24
DHCP client
datastore
ftp
Linux Router and IPS
10.10.10.1
10.20.20.1
VMWare IP
Internet
dns2
193.137.16.75
eden
193.136.212.1

IPTables (filtering and NAT)



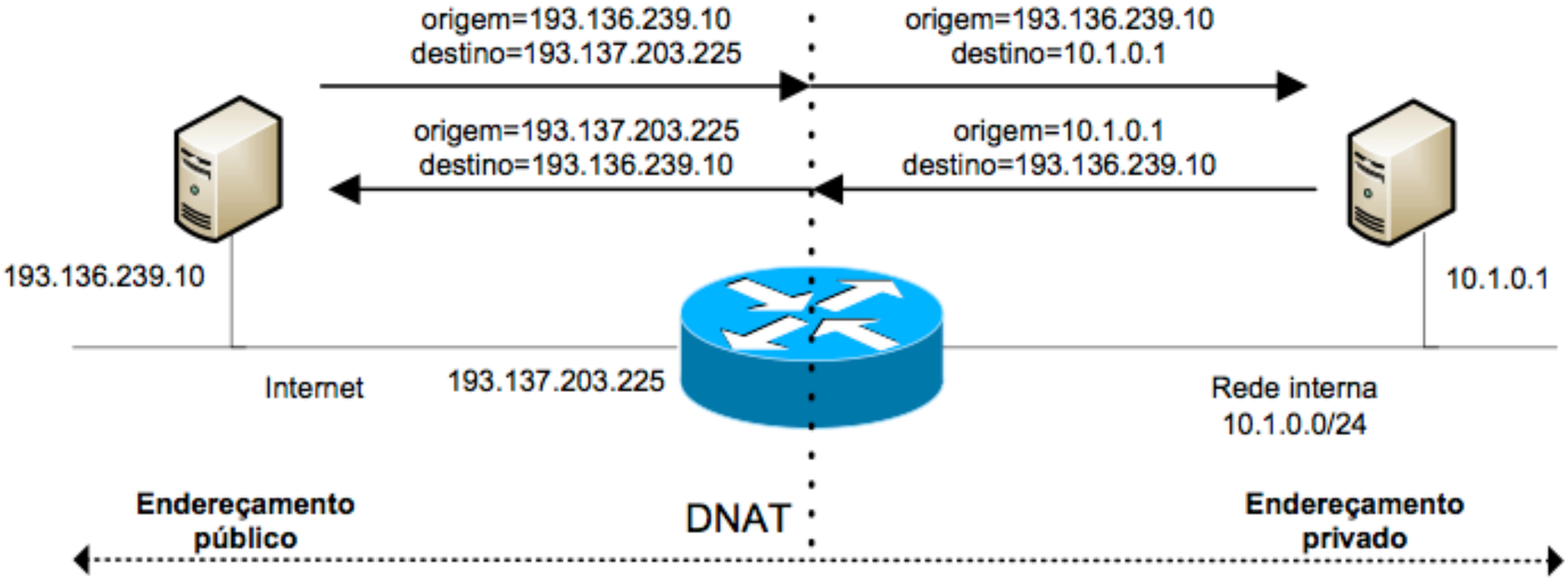
NAT (SNAT)

```
iptables -t nat -A POSTROUTING -s 10.1.0.0/24 -d 193.136.212.1 -j SNAT --to-source 193.137.203.225
```

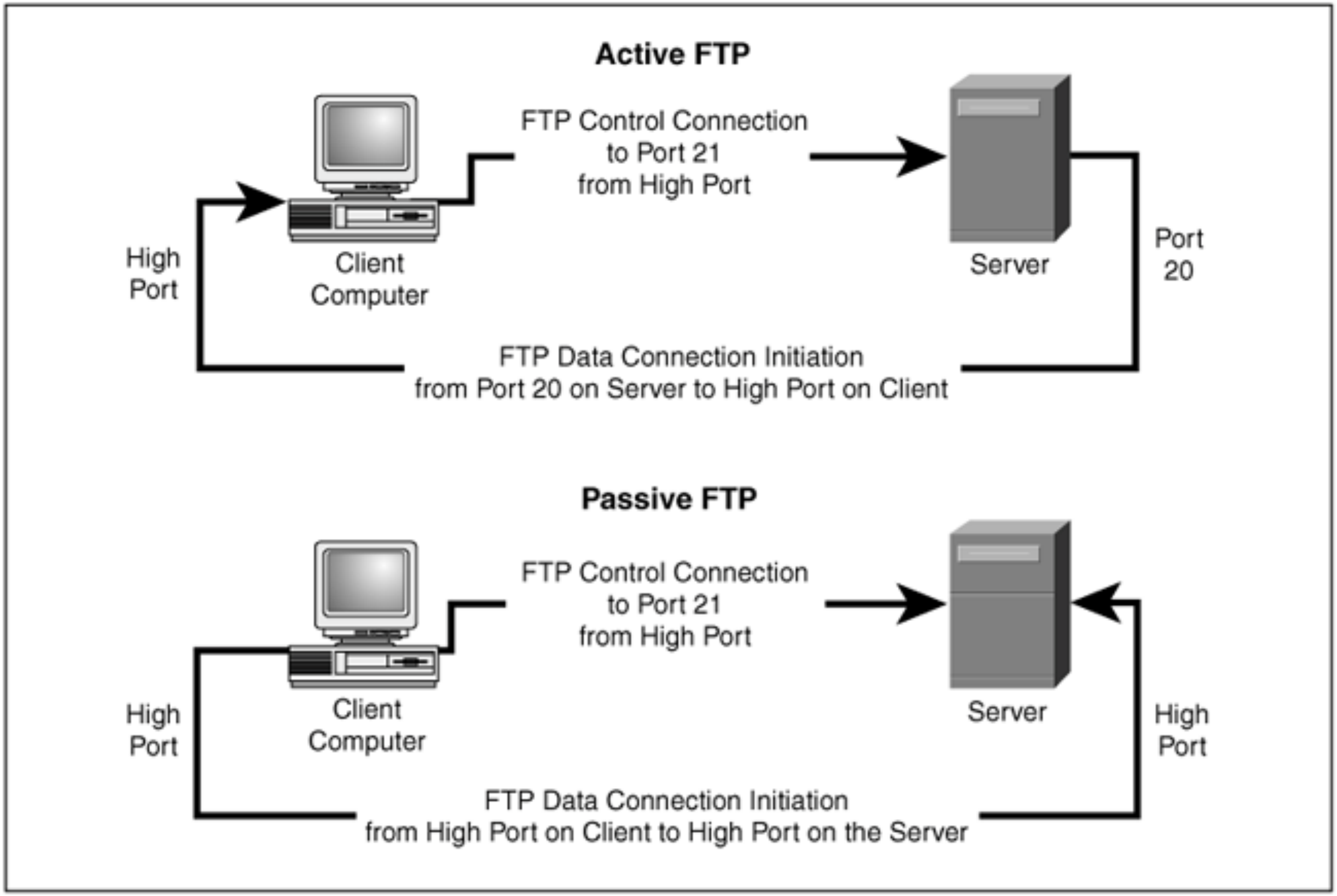


NAT (DNAT)

```
iptables -t nat -A PREROUTING -s 193.136.239.10 -d 193.137.203.225 -j DNAT --to-destination 10.1.0.1
```



FTP (Active and Passive modes)



http://codeidol.com/unix/linux-fix/Linux-FTP-Server-Setup/FTP-Overview/

Snort

Snort is:

- A lightweight network IDS
- Real-time traffic logging, content searching/matching and alerting
- Logs in *tcpdump* binary format and ASCII
- Integrates with IPTables for intrusion detection and prevention (inline mode)

Snort may work in three modes:

- Packet sniffer (as *tcpdump* or *tethereal/wireshark*)
- Packet logging mode (useful for network traffic debugging)
- Network intrusion detection (from rules defined on a configuration file)

Snort in **packet sniffer mode**, examples:

```
# Print TCP/IP packet headers on the screen
snort –v
```

```
# Print TCP/IP packet headers and also the application data
snort –vd
```

```
# A more descriptive description of the packets (including data link layer headers)
snort –vde
```

Snort

Snort in **packet logger mode**, examples:

```
snort -vde -K ascii -l ./log

# Log in binary (tcpdump) format
snort -b -l ./log

# Snort in “playback mode” from log file
snort -vd -r snort.log

# Reading the packet log file using tcpdump
tcpdump -r snort.log
```

Snort in **network intrusion detection system mode**, examples:

```
# Read configuration file for intrusion detection rules
snort -dev -l log -c snort.conf
```

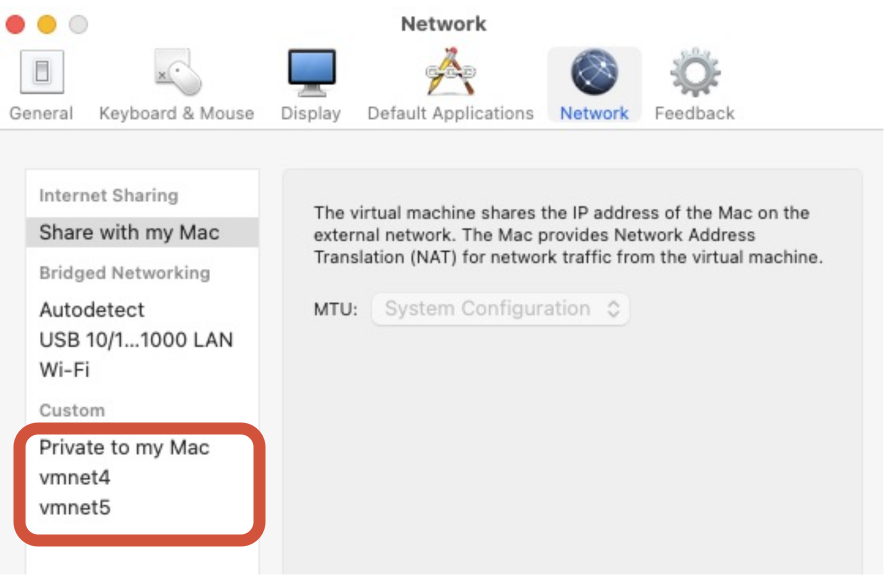
Detection rules, a few simple examples:

```
alert tcp any any -> 10.254.0.0/24 80 (msg:”HTTP packet”;)

var MY_NETS [10.254.0.0/24,10.1.0.0/24]
log tcp any any -> $MY_NETS any (flags:S; msg:”SYN packet”;)

alert tcp any any -> any 80 (content:”GET”;)
```

Starting the exercise

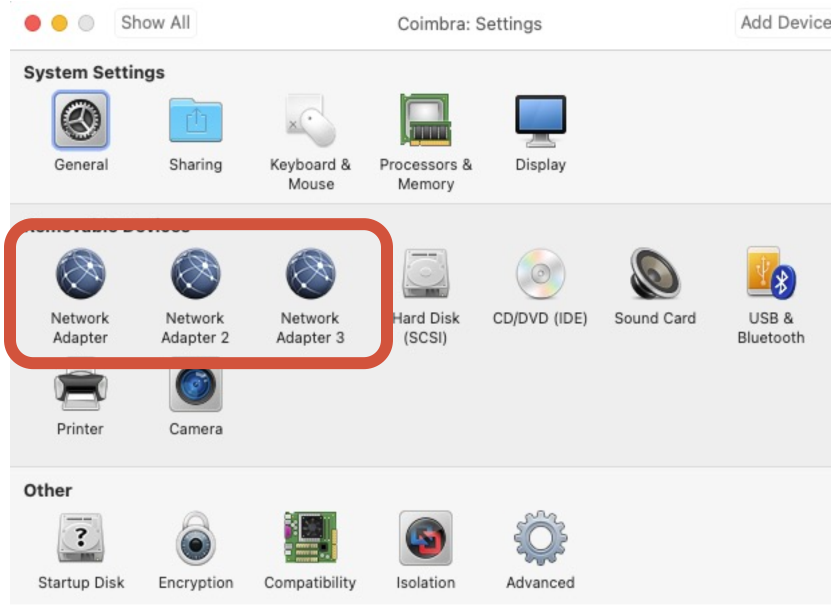
In the **Linux Router and IPS**:

1) Configure 3 network interfaces (use VMWare workstation/Fusion for this purpose)
2) Create networks using VMWare workstation/Fusion
3) Configure the addresses according to the scenario



The virtual machine shares the IP address of the Mac on the external network. The Mac provides Network Address Translation (NAT) for network traffic from the virtual machine.

MTU: System Configuration



Starting the exercise

In the **Linux Router and IPS**:

1) Configure 3 network interfaces (use VMWare workstation/Fusion for this purpose)
2) Configure the addresses according to the scenario

To ‘enable’ the **networks** consider:

1) One VM emulating the DMZ network (it can be Lisboa VM used in TP1)
2) Another VM emulating the internal network (it can be the Road Warrior VM of TP1)

The diverse services in the networks can be ‘emulated’ using the **netcat** utility, example:

```
$ nc -l -p 8080  # acts as server, binding to port 8080
```

To connect (as client): **nc 8080**