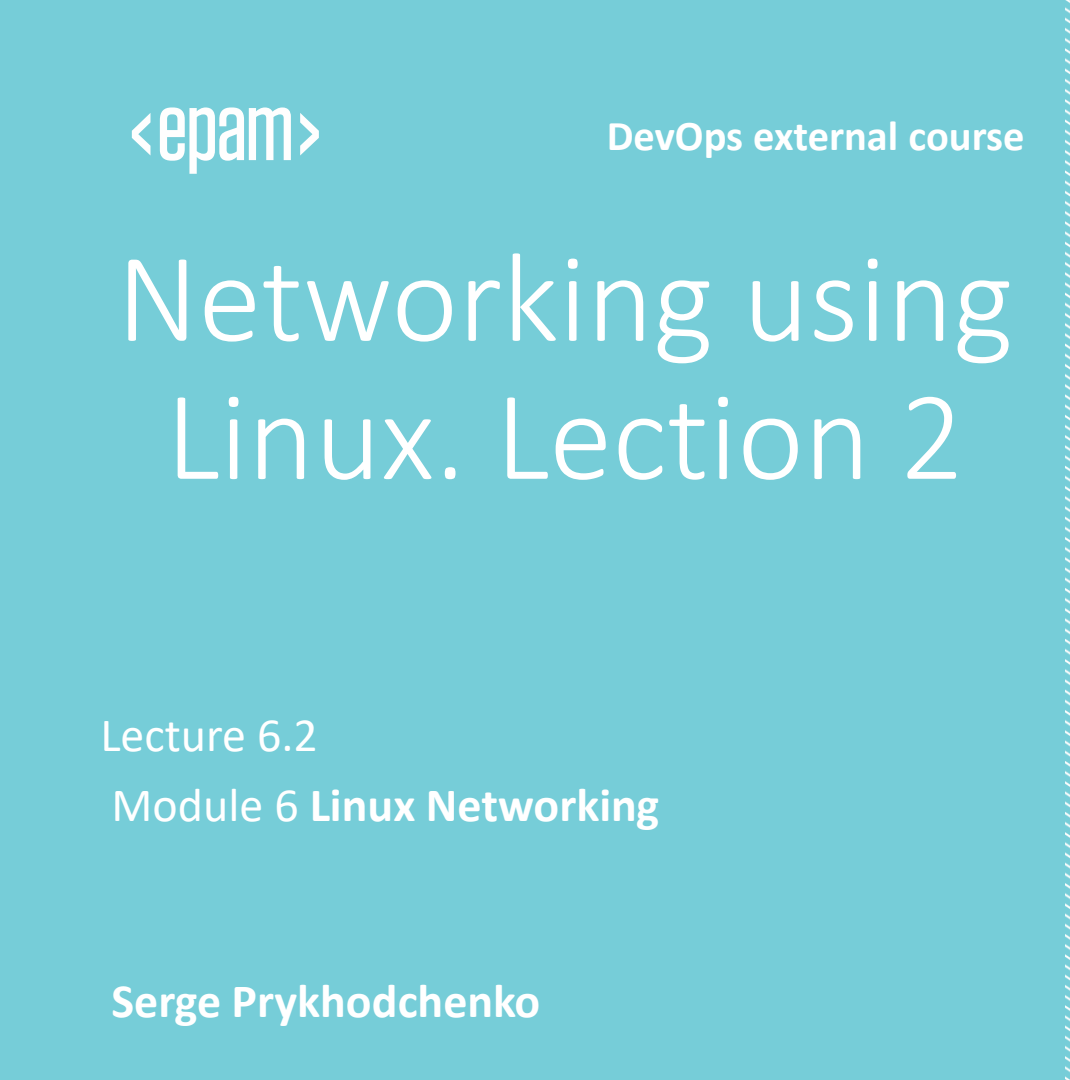


DevOps external course

Networking using Linux. Lection 2

Lecture 6.2

Module 6 **Linux Networking**

Serge Prykhodchenko

SELinux Security Policies

Security Policies are implemented using:

- **Type Enforcement® (TE)**

  (introduced in 1985 by Boebert and Kain)
- **Role-based access control (RBAC)**
- **Multi-level Security**

Users & Roles

- First and second component of a security context
- SELinux usernames and DAC usernames are not synonymous
- Semanage is used to maintain mappings of DAC to SELinux usernames.
- Roles are collections of types geared towards a purpose
- Roles can be used to further restrict actions on the system
- SELinux usernames are granted roles in the system

Each user gets a set of roles

Each role is assigned a set of TE domains.

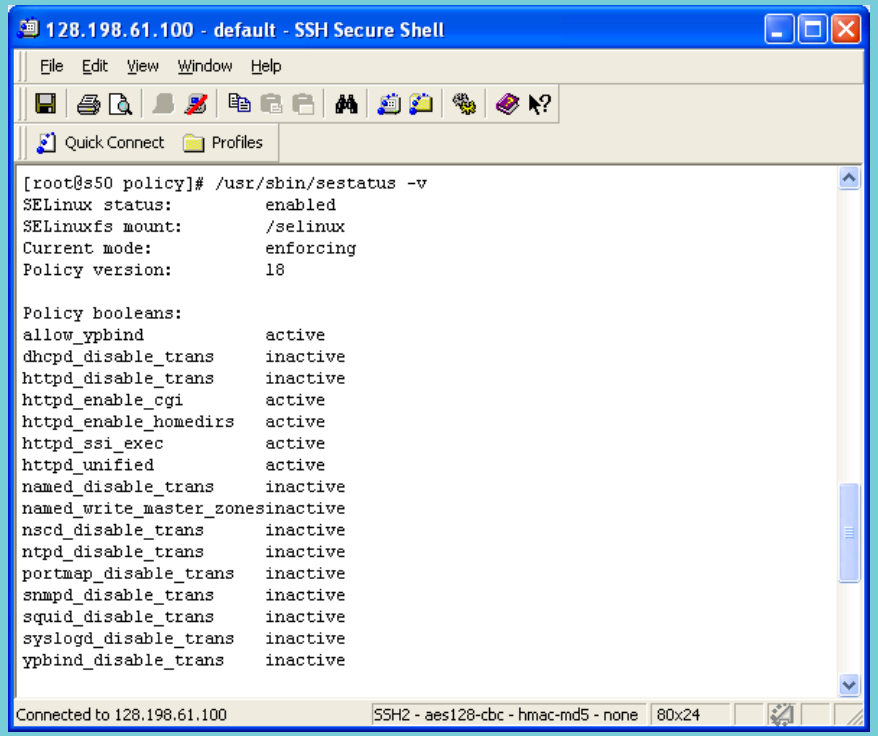
Note: users are not identified by Linux uids; instead a user identity attribute is used in the security context.

RBAC model

- Traditional RBAC model
  - authorizes users to act in certain roles and assigns a set of permissions to each role
- SELinux RBAC model
  - authorizes each user for a set of roles, each role for a set of TE domains
  - maintains a role attribute in the security context of each process

Configuration consists of:

- Flask definitions
- TE and RBAC declarations and rules
- User declarations
- Constraint definitions
- Security context specifications.



RBAC

- Adds 2 components to security context
  - **`user`**
  - **`role`**
- Adds 3 policy language keywords
  - **`allow`** (different than AVC allow)
  - **`role_transition`** (similar to type_transition)
  - **`dominance`**

RBAC Example

```
#valid security context
joe:user_r:passwd_t
#role user_r assigned to user joe
user joe roles { user_r };
#equivalent to this one
role user_r types { user_t passwd_t };
allow staff_r sysadm_r;
role_transition sysadm_r http_exec_t system_r;
#super_r inherits all types from sysadm_r and secadm_r
dominance { role super_r { role sysadm_r; role secadm_r; }}
```

Agenda

- NET-TOOLS vs IPROUTE
- Network administration
- iptables
- DHCP
- Q&A

NET-TOOLS VS IPROUTE

NET-TOOLS vs IPROUTE

Comparing NET-TOOLS vs IPROUTE Package Commands (ip vs ifconfig command comparison)

NET-TOOLS COMMANDS	IPROUTE COMMANDS
arp -a	ip neigh
arp -v	ip -s neigh
arp -s 192.168.1.1 1:2:3:4:5:6	ip neigh add 192.168.1.1 lladdr 1:2:3:4:5:6 dev eth1
arp -i eth1 -d 192.168.1.1	ip neigh del 192.168.1.1 dev eth1
ifconfig -a	ip addr
ifconfig eth0 down	ip link set eth0 down
ifconfig eth0 up	ip link set eth0 up
ifconfig eth0 192.168.1.1	ip addr add 192.168.1.1/24 dev eth0
ifconfig eth0 netmask 255.255.255.0	ip addr add 192.168.1.1/24 dev eth0
ifconfig eth0 mtu 9000	ip link set eth0 mtu 9000
ifconfig eth0:0 192.168.1.2	ip addr add 192.168.1.2/24 dev eth0
netstat	ss
netstat -neopa	ss -neopa
netstat -g	ip maddr
route	ip route
route add -net 192.168.1.0 netmask 255.255.255.0 dev eth0 i	ip route add 192.168.1.0/24 dev eth0
route add default gw 192.168.1.1	ip route add default via 192.168.1.1

Networking. Ubuntu 18+. Working with NetPlan

A Static IP address

• An IP address is an identifier held by each device that connects to the Internet or a computer network. In the case of the Internet, it must be unique to avoid connection conflicts.

• On the one hand, there are dynamic IP addresses that change their value from time to time. Normally, these addresses are assigned by a DHCP server. The sysadmin does not have to worry about the assigned address as they are renewed from time to time.

• On the other hand, we have static or fixed IP addresses, which unlike dynamical ones, do not change over time. In this case, it must be assigned and configured manually in the system.

• Each of them has its own advantages, however, for internal networks, it is convenient to have equipment with static IP addresses. This facilitates the administration and routing of packets within the network. It is also easier to maintain the network.

Networking. Ubuntu 18+. Working with NetPlan

```
student@ubuntu18:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 10
00
    link/ether 08:00:27:3f:11:bd brd ff:ff:ff:ff:ff:ff
    inet 192.168.88.120/24 brd 192.168.88.255 scope global dynamic enp0s3
       valid_lft 423sec preferred_lft 423sec
    inet6 fe80::a00:27ff:fe3f:11bd/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 08:00:27:7d:1c:6e brd ff:ff:ff:ff:ff:ff
4: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:35:30:11:50 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
student@ubuntu18:~$
```

Networking. Ubuntu 18+. Working with NetPlan

```
student@ubuntu18:~$ sudo cp /etc/netplan/00-installer-config.yaml /etc/netplan/00-installer-config.
yaml.bak
[sudo] password for student:
student@ubuntu18:~$ sudo nano /etc/netplan/00-installer-config.yaml
student@ubuntu18:~$
```

```
# This is the network config written by 'subiquity'
network:
  ethernets:
    enp0s3:
      dhcp4: true
  version: 2
```

Networking. Ubuntu 18+. Working with NetPlan

```
# Let NetworkManager manage all devices on this
system
network:
  ethernets:
    enp0s3:
      dhcp4: yes
      dhcp6: yes
  ethernets:
    enp0s8:
      dhcp4: no
      addresses: [10.0.0.1/24]
      gateway4: 10.0.0.1
      nameservers:
        addresses: [8.8.8.8]
  version: 2
  renderer: NetworkManager
```

```
> sudo netplan apply
```

Networking. Ubuntu 18+. Working with NetPlan

```
student@ubuntu18:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 10
00
    link/ether 08:00:27:3f:11:bd brd ff:ff:ff:ff:ff:ff
    inet 192.168.88.120/24 brd 192.168.88.255 scope global dynamic enp0s3
       valid_lft 362sec preferred_lft 362sec
    inet6 fe80::a00:27ff:fe3f:11bd/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 10
00
    link/ether 08:00:27:7d:1c:6e brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.2/24 brd 10.0.0.255 scope global enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe7d:1c6e/64 scope link
       valid_lft forever preferred_lft forever
4: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:35:30:11:50 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
student@ubuntu18:~$
```

Basic Level Comparison Between SystemV & Systemd

Comments	SysVinit	Systemd
Start a service	service dummy start	systemctl start dummy.service
Stop a service	service dummy stop	systemctl stop dummy.service
Restart a service	service dummy restart	systemctl restart dummy.service
Reload a service	service dummy reload	systemctl reload dummy.service
Service status	service dummy status	systemctl status dummy.service
Restart a service if already running	service dummy condrestart	systemctl condrestart dummy.service
Enable service at startup	chkconfig dummy on	systemctl enable dummy.service
Disable service at startup	chkconfig dummy off	systemctl disable dummy.service
Check if a service is enabled at startup	chkconfig dummy	systemctl is-enabled dummy.service
Create a new service file or modify configuration	chkconfig dummy --add	systemctl daemon-reload
System halt	0	runlevel0.target, poweroff.target
Single user mode	1, s, single	runlevel1.target, rescue.target
Multi user	2	runlevel2.target, multi-user.target
Multi user with Network	3	runlevel3.target, multi-user.target
Experimental	4	runlevel4.target, multi-user.target
Multi user, with network, graphical mode	5	runlevel5.target, graphical.target
Reboot	6	runlevel6.target, reboot.target
Emergency Shell	emergency	emergency.target
Change to multi user runlevel/target	telinit 3	systemctl isolate multi-user.target (OR systemctl isolate runlevel3. target)
Set multi-user target on next boot	sed s/^id:.*:initdefault:/ id:3:initdefault:/	ln -sf /lib/systemd/system/multi-user.target /etc/systemd/system/ default.target
Check current runlevel	runlevel	systemctl get-default
Change default runlevel	sed s/^id:.*:initdefault:/ id:3:initdefault:/	systemctl set-default multi-user.target

Basic Level Comparison Between SystemV & Systemd

System halt	halt	systemctl halt
Power off the system	poweroff	systemctl poweroff
Restart the system	reboot	systemctl reboot
Suspend the system	pm-suspend	systemctl suspend
Hibernate	pm-hibernate	systemctl hibernate
Follow the system log file	tail -f /var/log/messages or tail -f /var/log/syslog	journalctl -f

Systemd new commands

Execute a systemd command on remote host	systemctl dummy.service start -H user@host
Check boot time	systemd-analyze or systemd-analyze time
Kill all processes related to a service	systemctl kill dummy
Get logs for events for today	journalctl --since=today
Hostname and other host related information	hostnamectl
Date and time of system with timezone and other information	timedatectl

Overview

- LINUX® Netfilter is a firewall engine built into the Linux kernel
- Sometimes called “iptables” for the command-line tool used to configure Netfilter

iptables

All modern operating systems come equipped with a firewall – a software application that regulates network traffic to a computer. Firewalls create a barrier between a trusted network (like an office network) and an untrusted one (like the internet). Firewalls work by defining rules that govern which traffic is allowed, and which is blocked. The utility firewall developed for Linux systems is iptables.

Prerequisites:
- A user account with sudo privileges
- Access to a terminal window/command line (Ctrl-Alt-T, Ctrl-Alt-F2)

Scenario: Linux

iptables - administration tool for IPv4 packet filtering and NAT.

… used to set up, maintain, and inspect the tables of IP packet filter rules in the Linux kernel.

Essentially, host based firewall for Linux. Filters, and does NAT.

Scenario: Linux

Block an incoming IP:

```
$ iptables –A INPUT –s 10.42.X.XXX –j DROP
```

Block outgoing IP:

```
$ iptables –A OUTPUT –d 10.42.X.XXX –j DROP
```

Block an incoming port:

```
$ iptables –A INPUT –s 10.42.X.XXX –p tcp –destination-port 80 –j drop
```

Persistence?

Debian

$ iptables-save >

/etc/iptables/rules.v4

/sbin/iptables-save

Redhat

$ Service iptables save

/etc/sysconfig/iptables

IPTables Flags

- -A Append one or more rule
- -D Delete a Rule
- -I Insert a Rule
- -R Replace
- -F FLUSH chain, delete rule one by one
- -j Jump
- -s Source IP
- -d Destination IP
- -p Protocol(TCP/IP)
- -L List all rules
- -N Numerically List
- -v Verbose (More information output)
- Need more? $ man iptables

UFW (Uncomplicated Firewall)

Front-end for iptables

$ sudo ufw allow from 1.1.1.1 to any port 22

$ sudo ufw deny from 1.1.1.1/24

$ sudo ufw deny http(80)

$ sudo ufw status numbered

$ sudo ufw delete 2

$ sudo ufw default deny incoming

iptables

How iptables work:

Network traffic is made up of packets. Data is broken up into smaller pieces (called packets), sent over a network, then put back together. Iptables identifies the packets received and then uses a set of rules to decide what to do with them.

iptables filters packets based on:

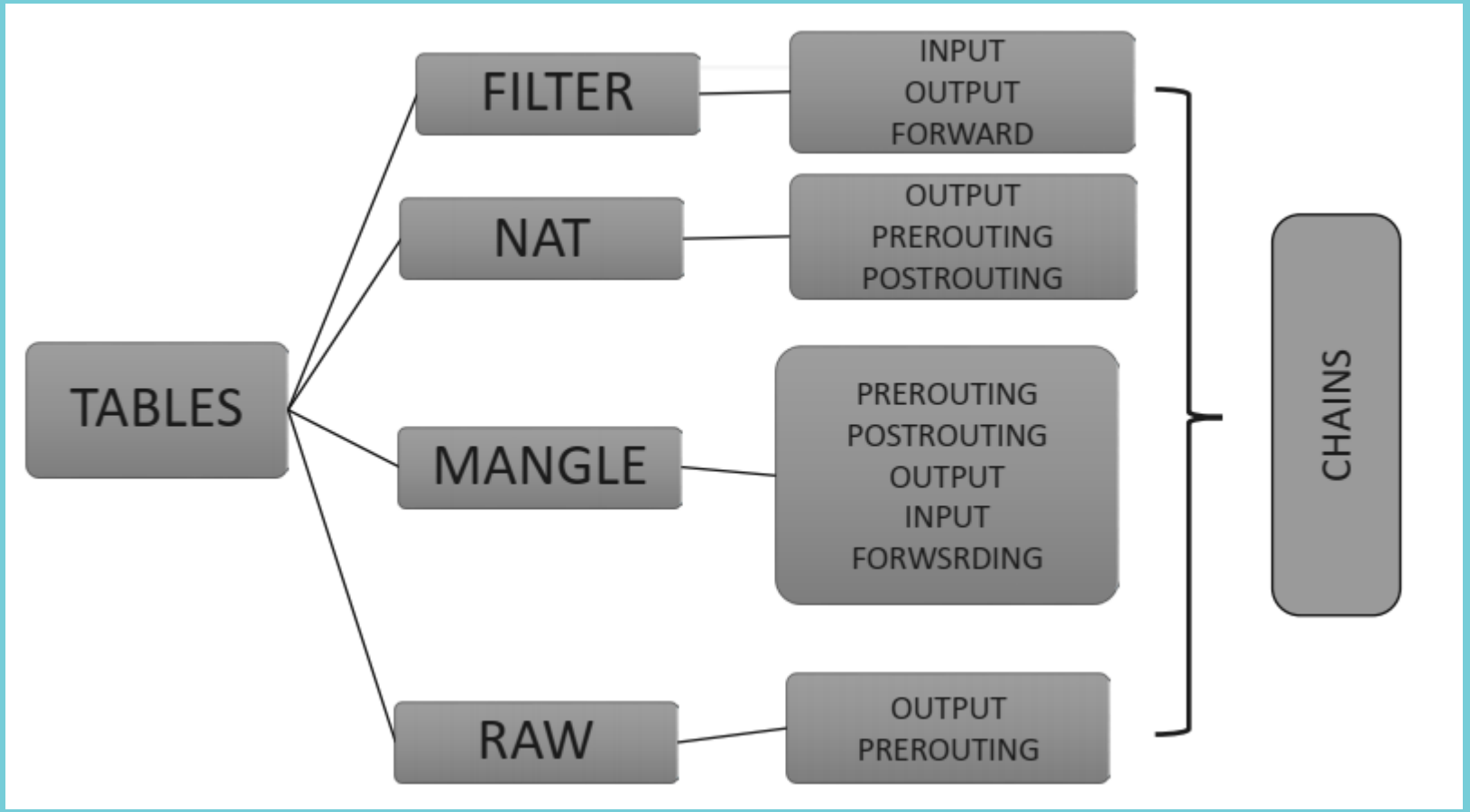
Tables: Tables are files that join similar actions. A table consists of several chains.

Chains: A chain is a string of rules. When a packet is received, iptables finds the appropriate table, then runs it through the chain of rules until it finds a match.

Rules: A rule is a statement that tells the system what to do with a packet. Rules can block one type of packet, or forward another type of packet. The outcome, where a packet is sent, is called a target.

Targets: A target is a decision of what to do with a packet. Typically, this is to accept it, drop it, or reject it (which sends an error back to the sender).

iptables


TABLES
FILTER
INPUT
OUTPUT
FORWARD
NAT
OUTPUT
PREROUTING
POSTROUTING
MANGLE
PREROUTING
POSTROUTING
OUTPUT
INPUT
FORWSRDING
RAW
OUTPUT
PREROUTING
CHAINS

iptables

Tables and Chains. Linux firewall iptables has four default tables.

1. **Filter**

The Filter table is the most frequently used one. It acts as a bouncer, deciding who gets in and out of your network. It has the following default chains:

Input – the rules in this chain control the packets received by the server.

Output – this chain controls the packets for outbound traffic.

Forward – this set of rules controls the packets that are routed through the server.

2. **Network Address Translation (NAT)**

This table contains NAT (Network Address Translation) rules for routing packets to networks that cannot be accessed directly. When the destination or source of the packet has to be altered, the NAT table is used. It includes the following chains:

Prerouting – this chain assigns packets as soon as the server receives them.

Output – works the same as the output chain we described in the filter table.

Postrouting – the rules in this chain allow making changes to packets after they leave the output chain.

iptables

3. **Mangle**

The Mangle table adjusts the IP header properties of packets. The table has all the following chains we described above:

Prerouting
Postrouting
Output
Input
Forward

4. **Raw**

The Raw table is used to exempt packets from connection tracking. The raw table has two of the chains we previously mentioned:

Prerouting
Output

iptables

Targets

A target is what happens after a packet matches a rule criteria. The targets in Linux iptables are:

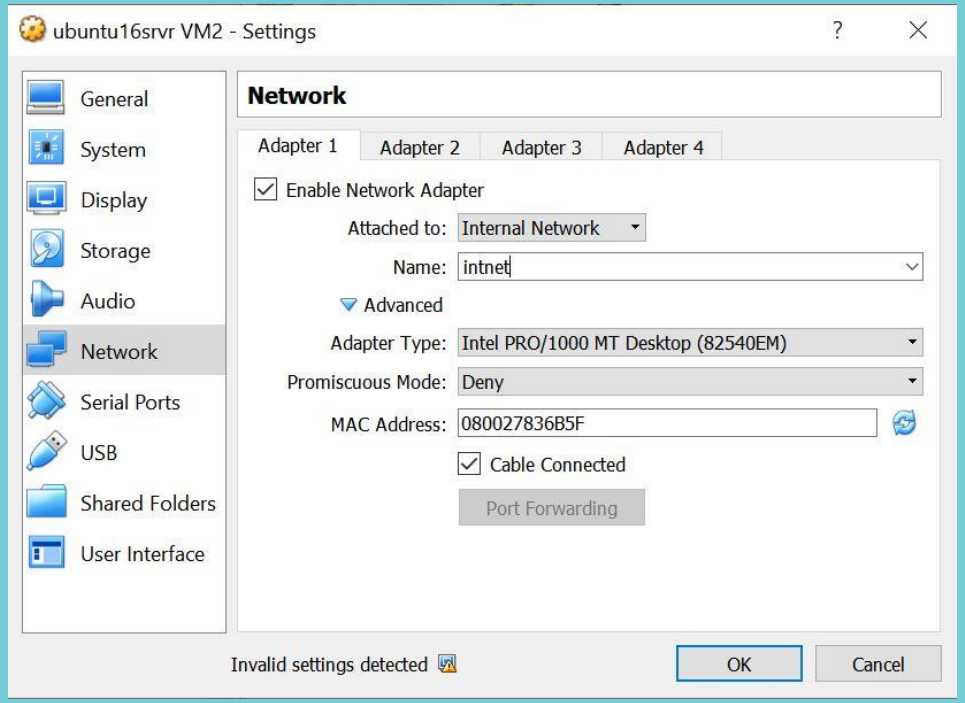
Accept – this rule accepts the packets to come through the iptables firewall.

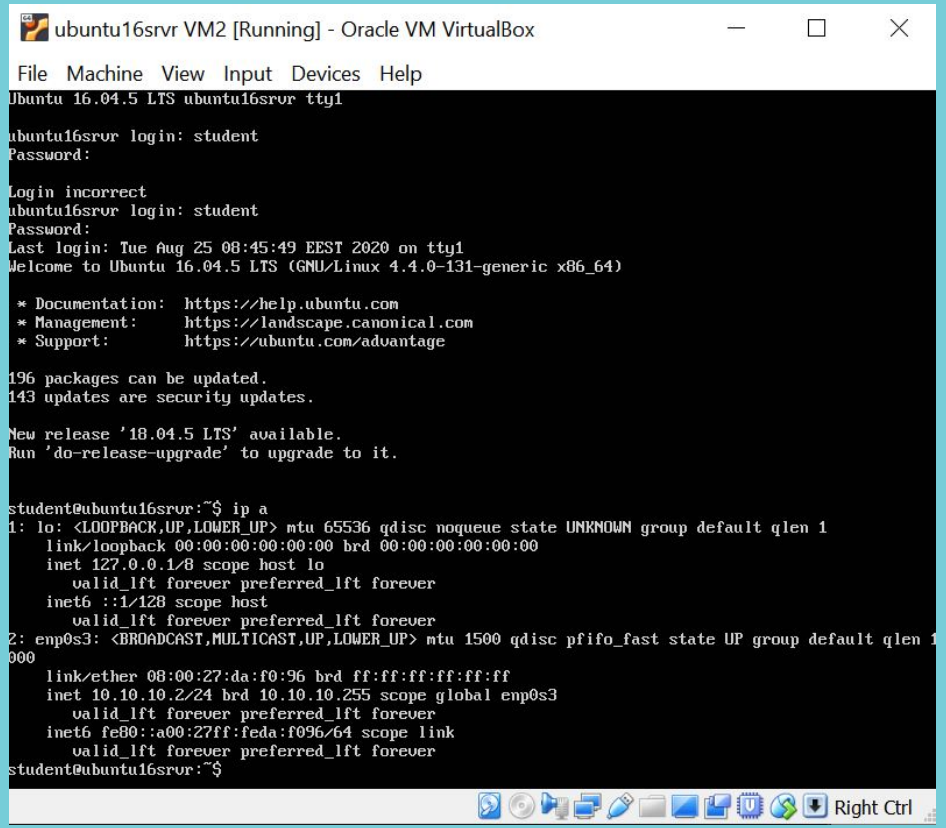
Drop – the dropped package is not matched against any further chain. When Linux iptables drop an incoming connection to your server, the person trying to connect does not receive an error. It appears as if they are trying to connect to a non-existing machine.

Return – this rule sends the packet back to the originating chain so you can match it against other rules.

Reject – the iptables firewall rejects a packet and sends an error to the connecting device

Iptables – Example. Before





Iptables – Example. Before

ubuntu16srvr VM1 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

```
Password:
Last login: Tue Aug 25 07:58:22 EEST 2020 on tty1
Welcome to Ubuntu 16.04.5 LTS (GNU/Linux 4.4.0-131-generic x86_64)

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

196 packages can be updated.
143 updates are security updates.

New release '18.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


student@ubuntu16srvr:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1
000
    link/ether 08:00:27:ac:1b:56 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:feac:1b56/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1
000
    link/ether 08:00:27:4c:53:00 brd ff:ff:ff:ff:ff:ff
    inet 10.10.10.1/24 brd 10.10.10.255 scope global enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe4c:5300/64 scope link
       valid_lft forever preferred_lft forever
student@ubuntu16srvr:~$
```

Right Ctrl

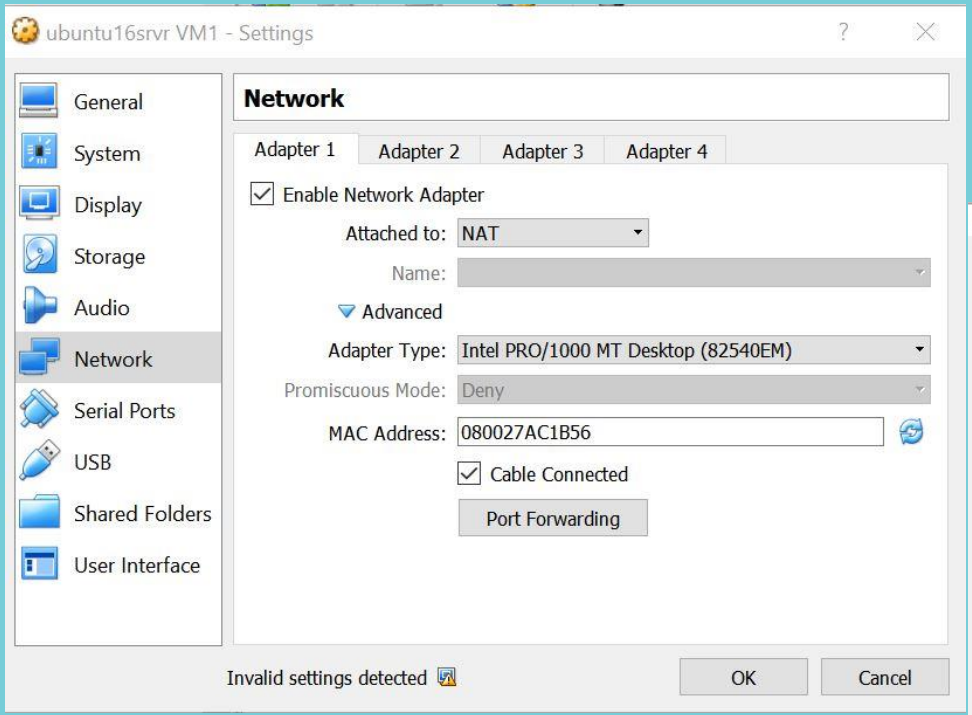
ubuntu16srvr VM1 [Running] - Oracle VM VirtualBox

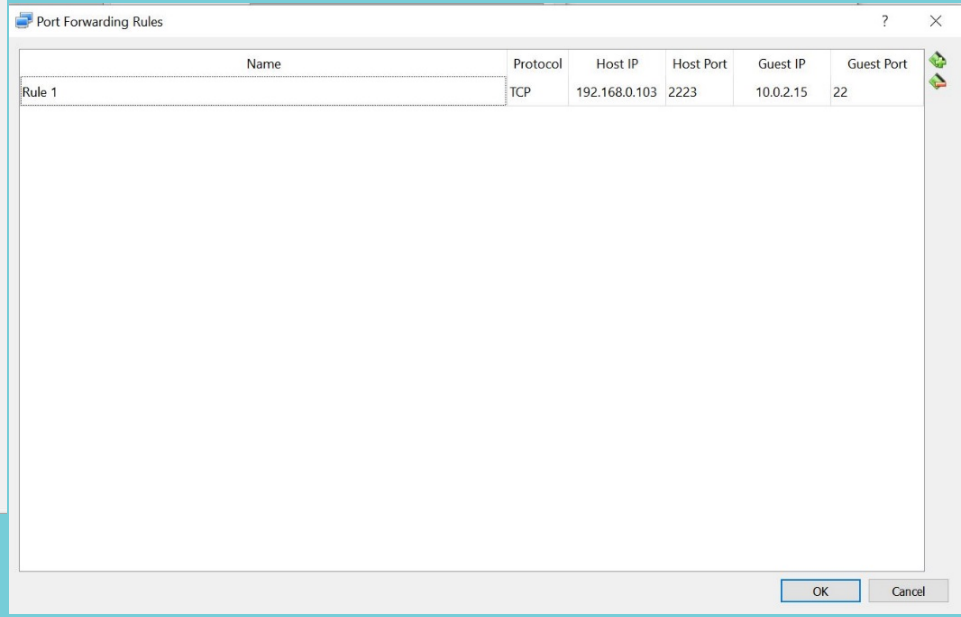
File Machine View Input Devices Help

```
 GNU nano 2.5.3                File: /etc/network/interfaces

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# NAT
auto enp0s3
iface enp0s3 inet dhcp

#internal
auto enp0s8
iface enp0s8 inet static
address 10.10.10.1
netmask 255.255.255.0
broadcast 10.10.10.255

                                [ Read 20 lines ]
^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos  ^Y Prev Page
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text ^T To Spell ^_ Go To Line ^V Next Page
```

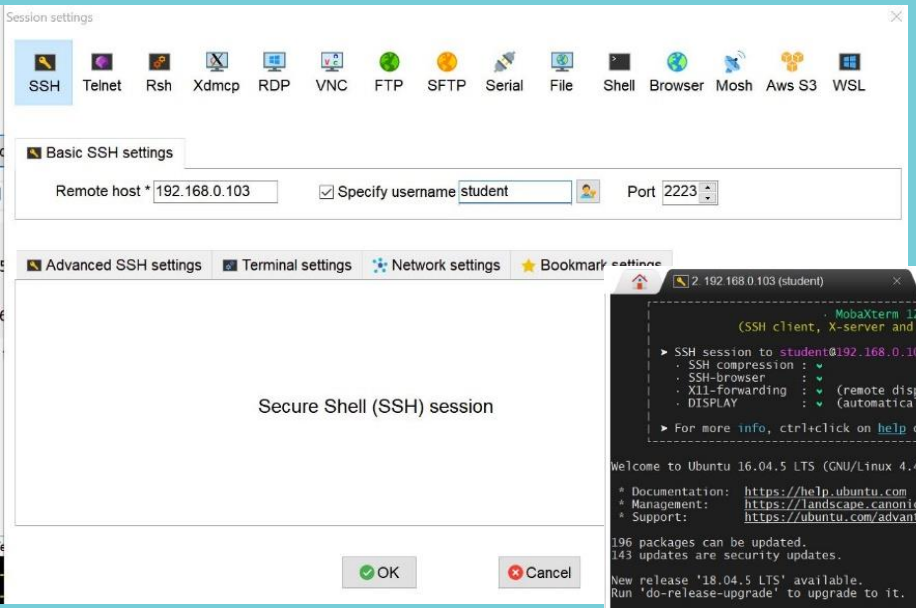
Right Ctrl

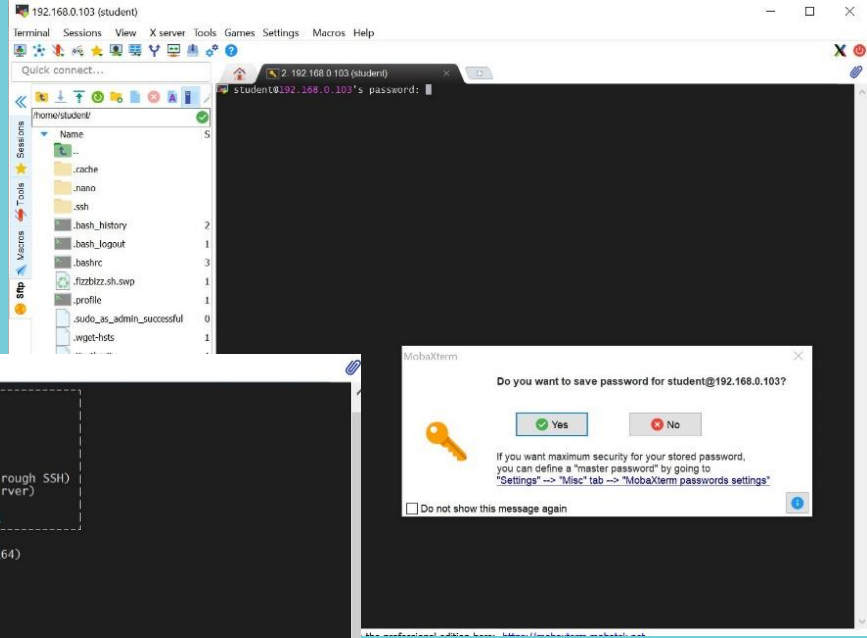
Iptables – Example. Before

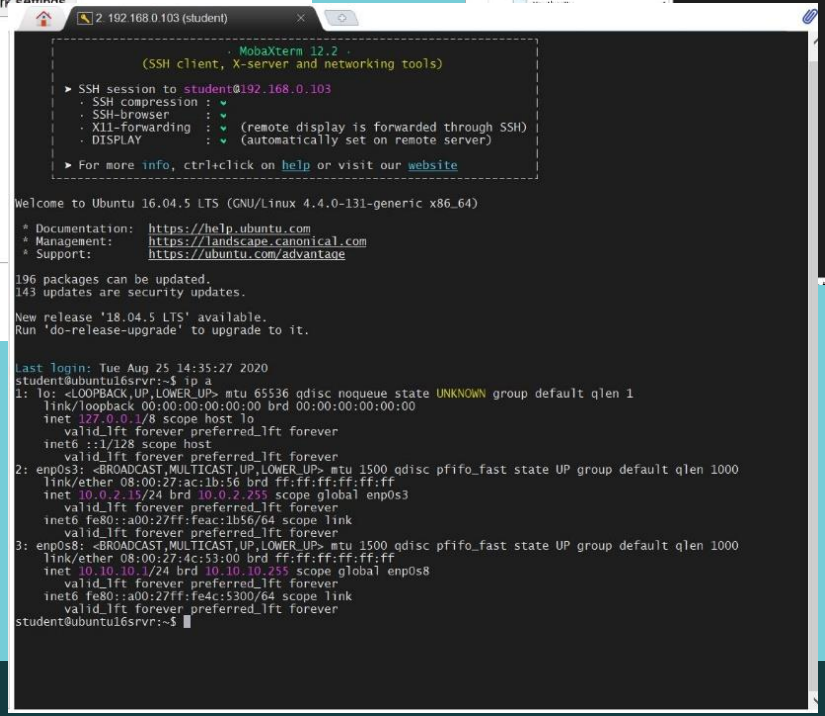




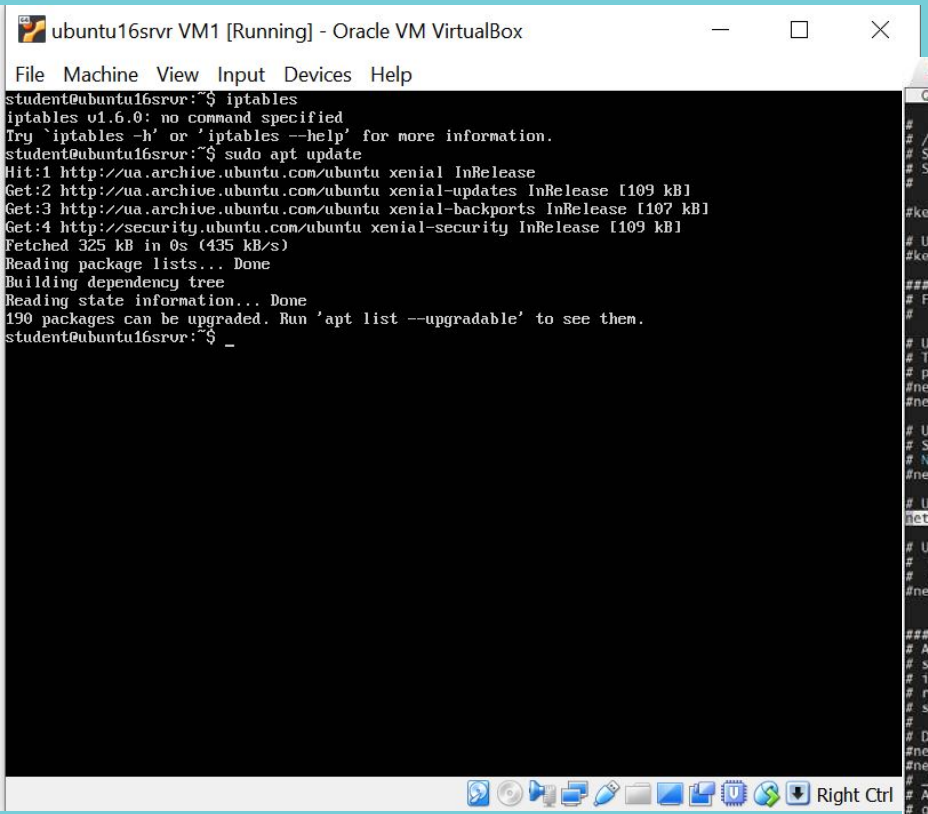
Iptables – Example. Before

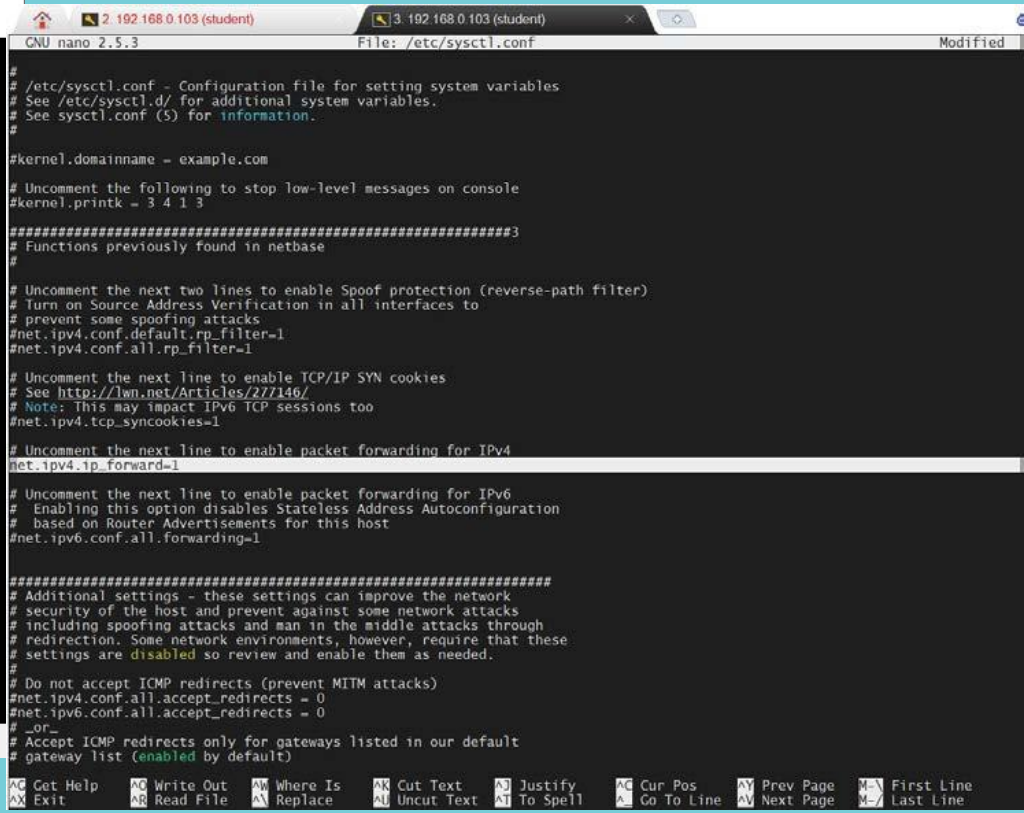




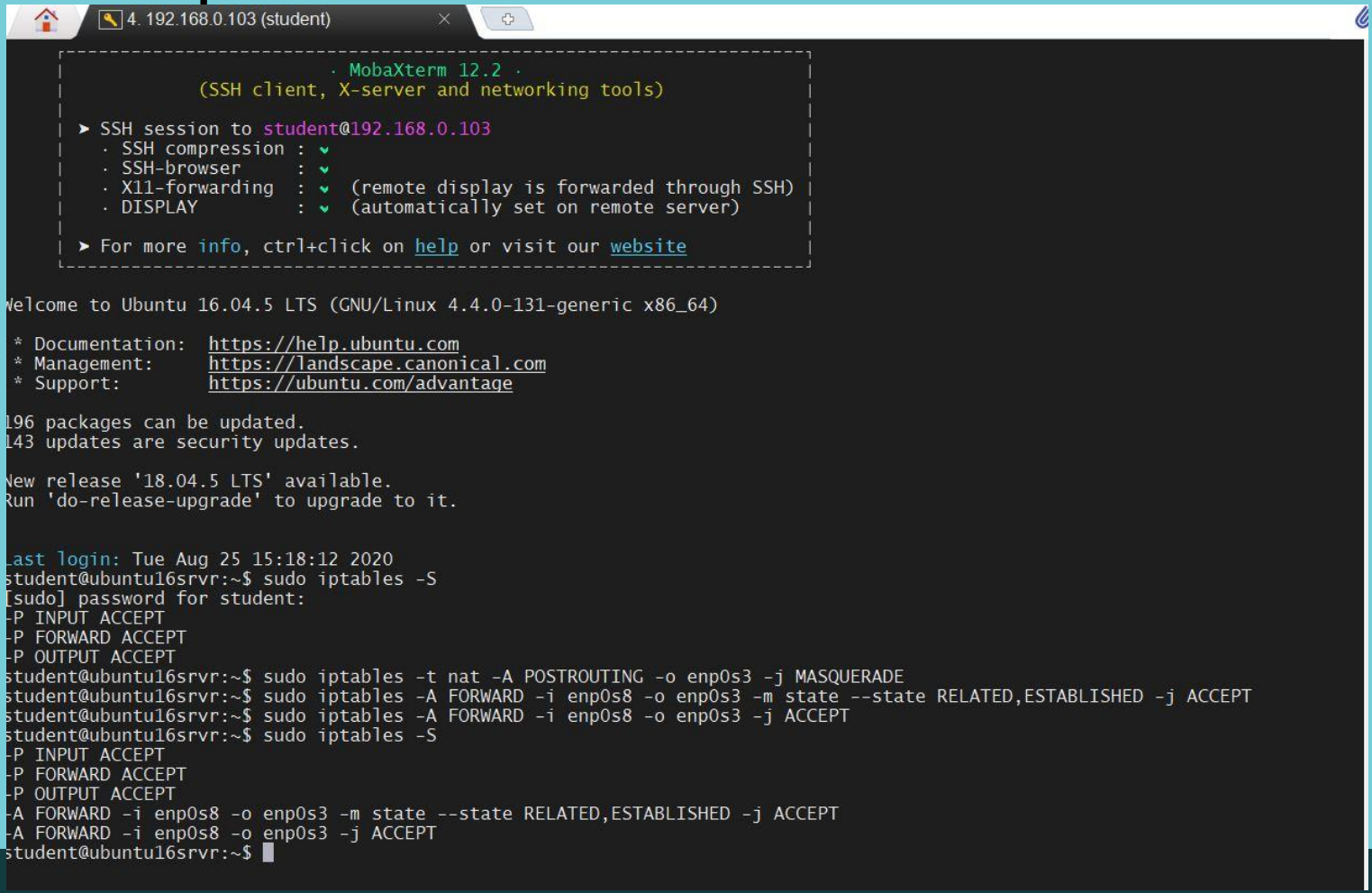


Iptables – Example. Now

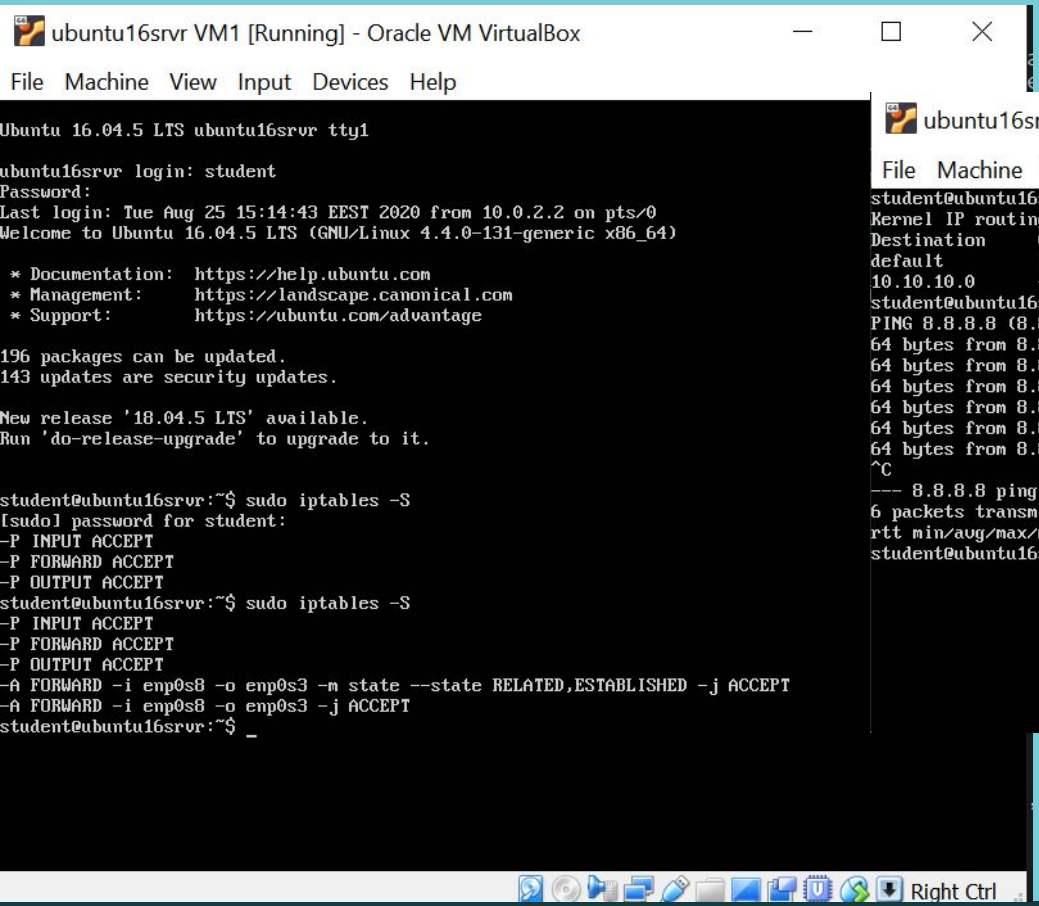


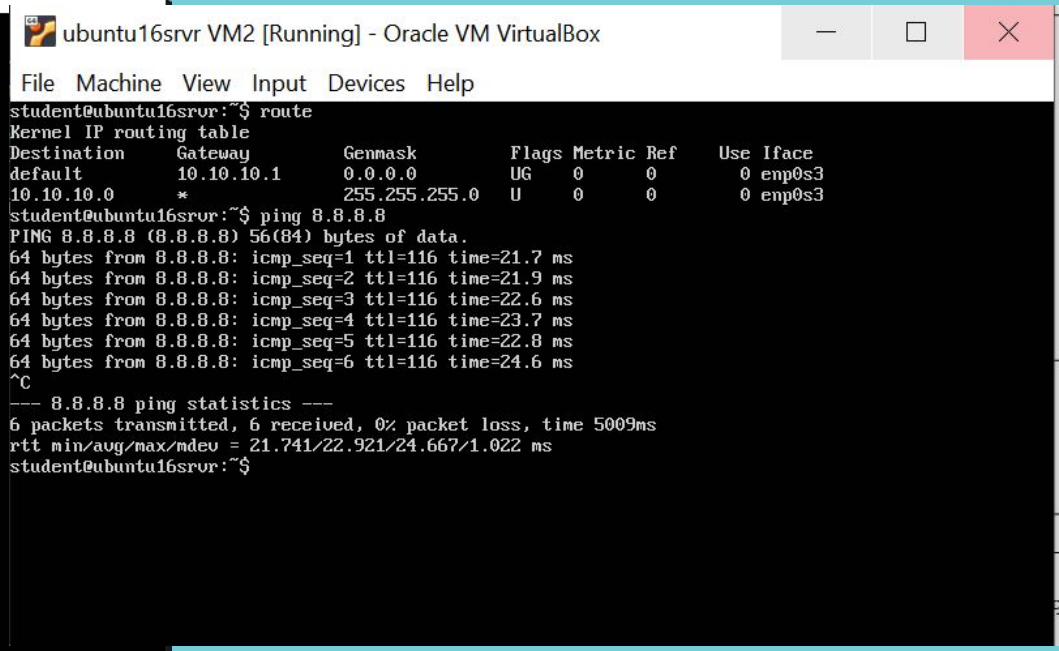


Iptables – Example. Now



Iptables – Example. Now



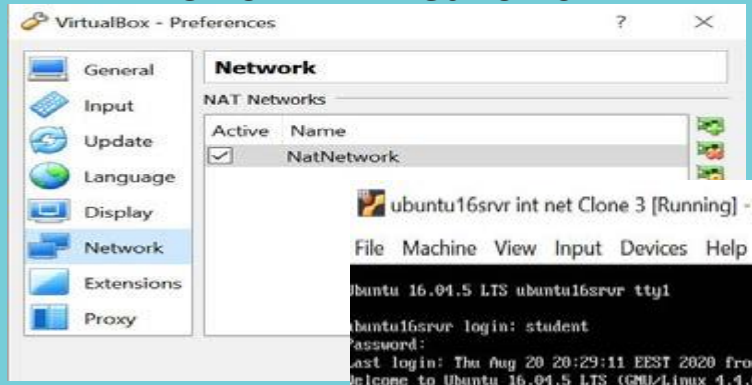


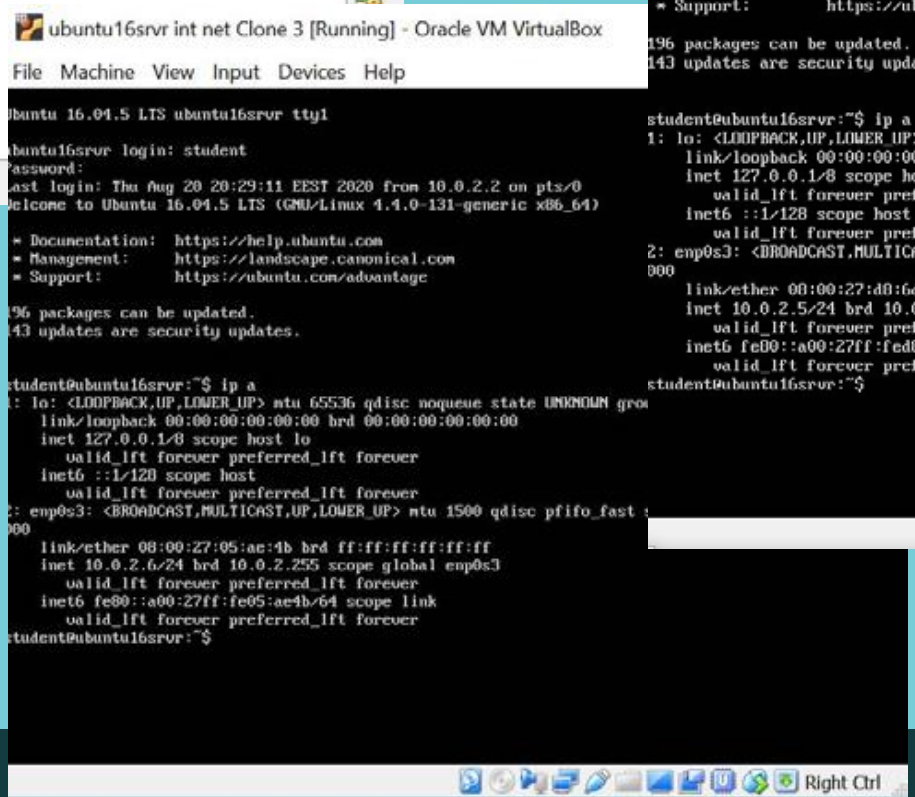
DHCP

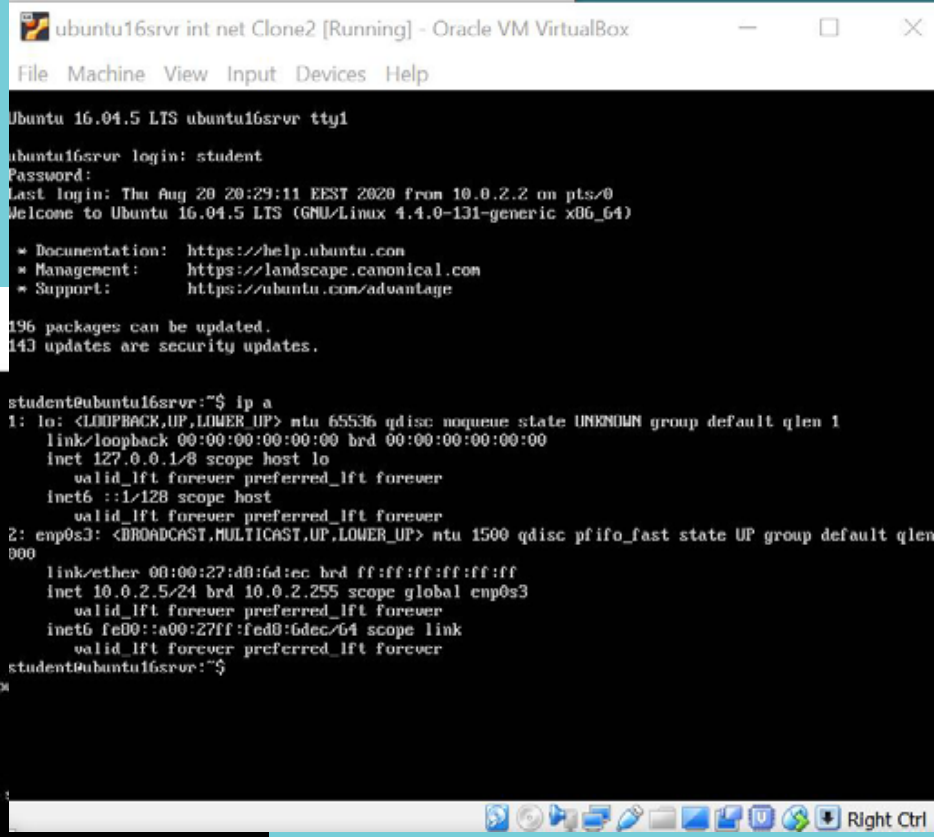
In computer science, the Dynamic Host Configuration Protocol (DHCP) is a network management protocol used on Internet Protocol (IP) networks, whereby a DHCP server dynamically assigns an IP address and other network configuration parameters to each device on the network, so they can communicate with other IP networks. A DHCP server enables computers to request IP addresses and networking parameters automatically from the Internet service provider (ISP), reducing the need for a network administrator or a user to manually assign IP addresses to all network devices.

DHCP

VB DHCP on NAT Networks.







DHCP

DHCP installation and configuring

1. The Internet Systems Consortium (ISC) Dynamic Host Configuration Protocol (DHCP) server is free, open-source, and easy to install. Both enterprises and small networks have used ISC DHCP in production for many years.

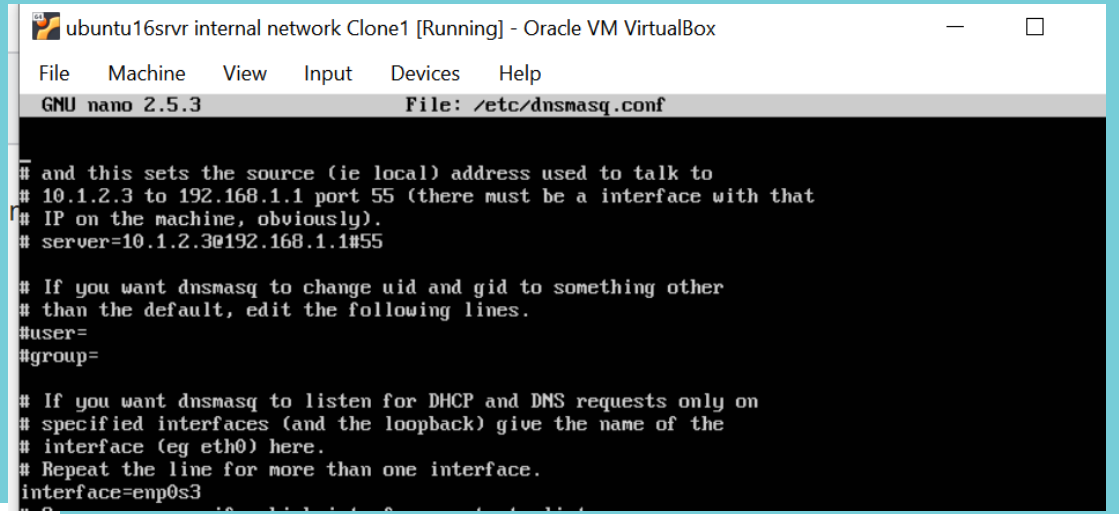
2. Dnsmasq is a lightweight, easy to configure, DNS forwarder and DHCP server. It is designed to provide DNS and optionally, DHCP, to a small network. It can serve the names of local machines which are not in the global DNS. The DHCP server integrates with the DNS server and allows machines with DHCPallocated addresses to appear in the DNS with names configured either in each host or in a central configuration file. Dnsmasq supports static and dynamic DHCP leases and BOOTP/TFTP for network booting of diskless machines

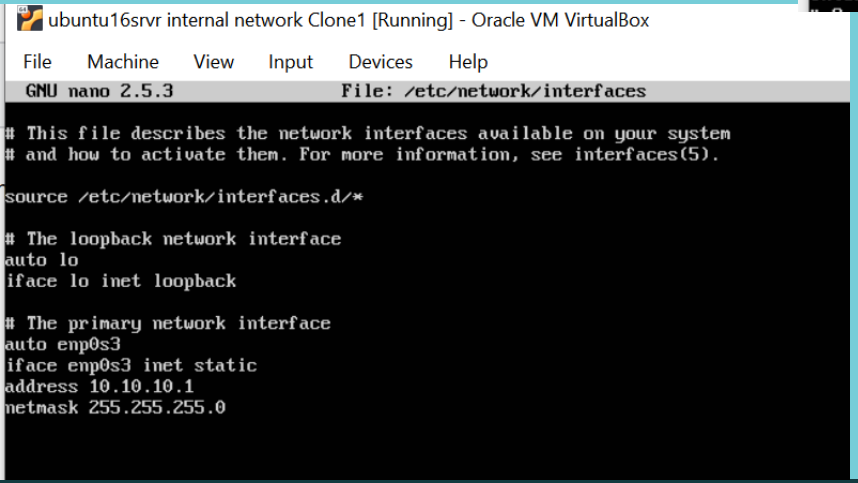
DHCP

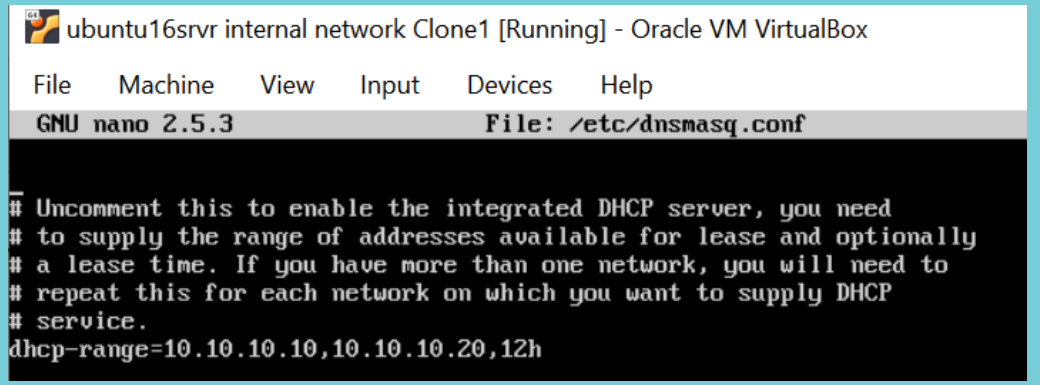
Dnsmasq installation and configuring:

> apt-get update

> apt-get install dnsmasq







DHCP

Dnsmasq installation and configuring:

ubuntu16srvr int net Clone 3 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

```
 GNU nano 2.5.3                File: /etc/network/interfaces

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto enp0s3
iface enp0s3 inet dhcp
```

ubuntu16srvr int net Clone2 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

```
 GNU nano 2.5.3                File: /etc/network/interfaces

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto enp0s3
iface enp0s3 inet dhcp
```

ubuntu16srvr int net Clone 3 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

```
student@ubuntu16srvr:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1
000
    link/ether 08:00:27:05:ae:4b brd ff:ff:ff:ff:ff:ff
    inet 10.10.10.10/24 brd 10.10.10.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe05:ae4b/64 scope link
       valid_lft forever preferred_lft forever
student@ubuntu16srvr:~$ _
```

ubuntu16srvr int net Clone2 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

```
student@ubuntu16srvr:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1
000
    link/ether 08:00:27:d8:6d:ec brd ff:ff:ff:ff:ff:ff
    inet 10.10.10.15/24 brd 10.10.10.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fed8:6dec/64 scope link
       valid_lft forever preferred_lft forever
student@ubuntu16srvr:~$
```

QUESTIONS & ANSWERS


THANK YOU!