HUMBER
Faculty of Applied Sciences & Technology

OPERATING SYSTEMS
CCGC-5000

Module - 11(Linux Networking, Firewall)

WE ARE
HUMBER

Agenda

Authentic information is available from the given resources in course outline and URL's mentioned from this slides, and this presentation is only supportive document to be read with the given resources and corrected accordingly if required..

- Hostname, DHCP and DNS
- Network configuration files
- Network commands & tools
- Network routing
- Static IP setup
- Firewall, Linux firewall
- firewalld
- Config files and directories
- Firewalld Zones, trust levels
- nftables

Refer RHEL8 Course Book – Chapter 8,16,20, 21
Refer Michael Palmer - Chapter 8,9,10,11

hostname, DHCP, DNS

- Hostname is a unique alphanumeric label that is assigned to a system to identify it on the network, stored in /etc/hostname
- Hostname can be viewed as below :

hostname

hostnamectl |grep hostname

uname –n

cat /etc/hostname

nmcli general hostname

- Hostname can be changed using **hostnamectl** command and restart service systemd-hostnamed

hostnamectl set-hostname *hostname*

systemctl restart systemd-hostnamed

- Command **hostname –s** for short hostname, **hostname –d** for domain name, likewise refer to man pages of hostname for more options and details.

- DHCP – Dynamic Host Configuration Protocol
- Configures hosts for connection to network by assigning IP address to NIC of the host
- IP address are assigned as lease for a specific period
- DNS – Domain Name Server that resolves the domain names/hostnames to ip address and connect to network if required
- Package to install DNS in linux – **bind**
- DNS Configuration files **/etc/named.conf**
- Other files

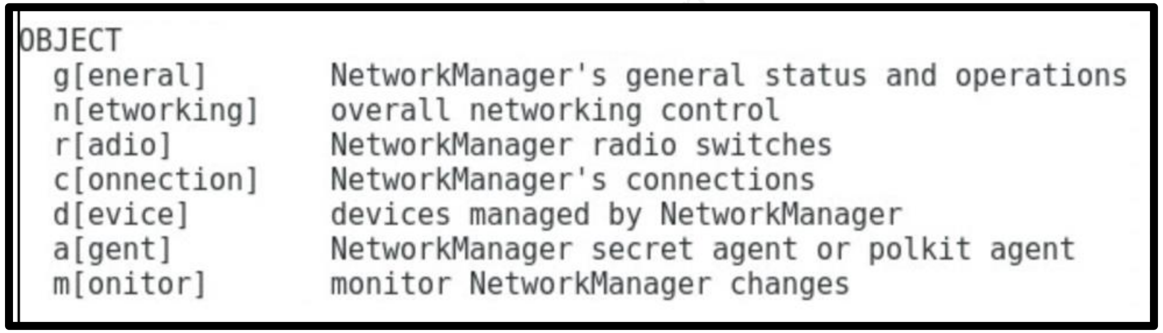
/etc/resolv.conf – DNS domain search settings

/etc/host.conf – resolver configuration file

/etc/services – maps port number to services

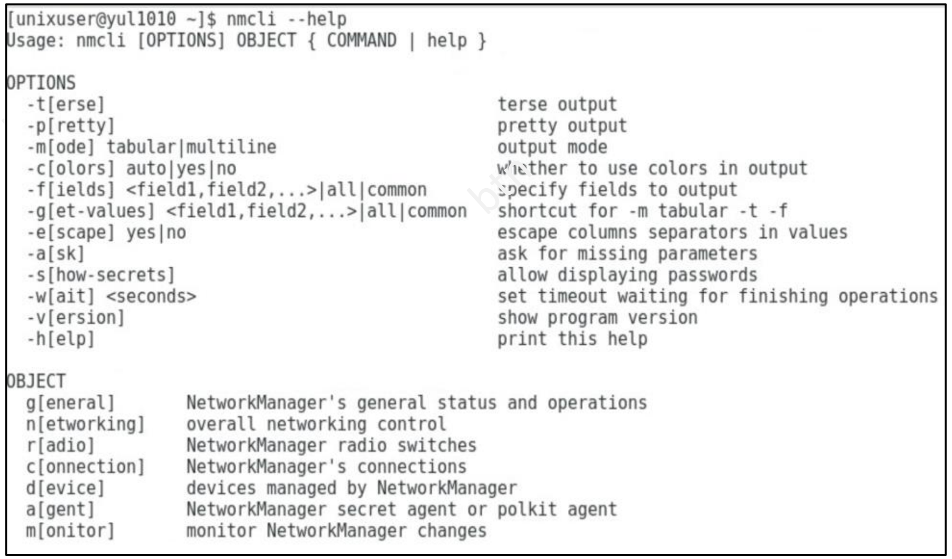
Interface (NIC) Administration

- Activating an interface: **nmcli n on** *interfacename* **OR**
  nmcli con up *interfacename*
- Command **ifup** *interfacename* will also activate an interface.
- Deactivating an interface: **nmcli n off** *interfacename* **OR**
  nmcli con down *interfacename*
- Command **ifdown** *interfacename* will also deactivate an interface.
- Displaying interfaces: **nmcli** d
- Displaying connections: **nmcli** c
- Monitoring: **nmcli** m

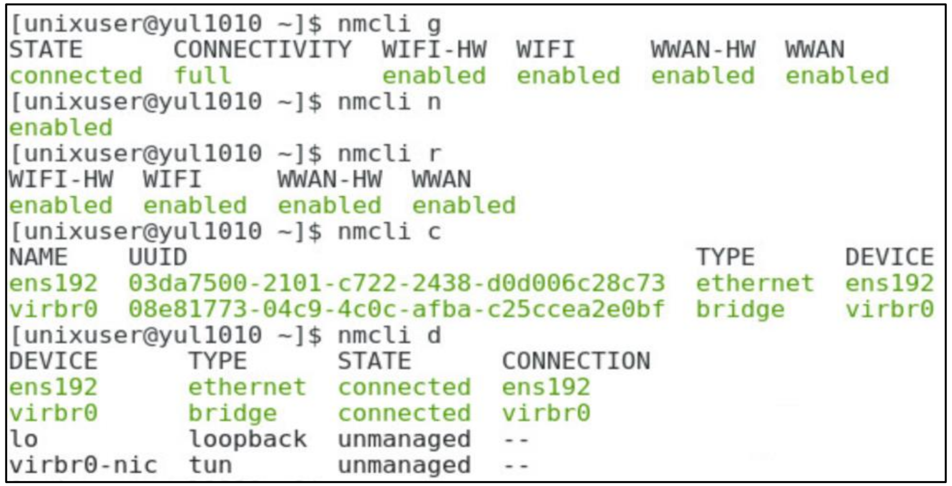


```
OBJECT
  g[eneral]       NetworkManager's general status and operations
  n[etworking]    overall networking control
  r[adio]         NetworkManager radio switches
  c[onnection]    NetworkManager's connections
  d[evice]        devices managed by NetworkManager
  a[gent]         NetworkManager secret agent or polkit agent
  m[onitor]       monitor NetworkManager changes
```

Network Administration



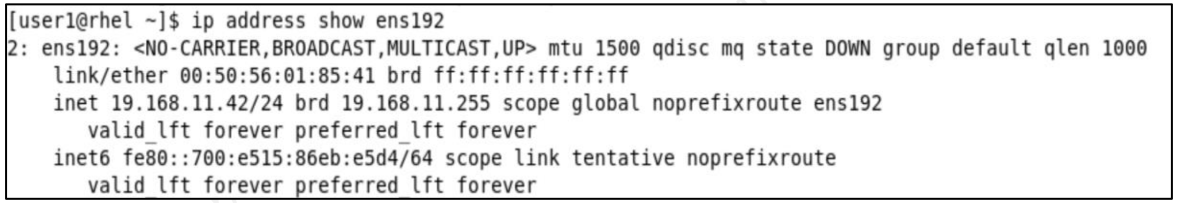
```
[unixuser@yul1010 ~]$ nmcli --help
Usage: nmcli [OPTIONS] OBJECT { COMMAND | help }

OPTIONS
  -t[erse]                                   terse output
  -p[retty]                                  pretty output
  -m[ode] tabular|multiline                  output mode
  -c[olors] auto|yes|no                      whether to use colors in output
  -f[ields] <field1,field2,...>|all|common   specify fields to output
  -g[et-values] <field1,field2,...>|all|common  shortcut for -m tabular -t -f
  -e[scape] yes|no                           escape columns separators in values
  -a[sk]                                     ask for missing parameters
  -s[how-secrets]                            allow displaying passwords
  -w[ait] <seconds>                          set timeout waiting for finishing operations
  -v[ersion]                                 show program version
  -h[elp]                                    print this help

OBJECT
  g[eneral]       NetworkManager's general status and operations
  n[etworking]    overall networking control
  r[adio]         NetworkManager radio switches
  c[onnection]    NetworkManager's connections
  d[evice]        devices managed by NetworkManager
  a[gent]         NetworkManager secret agent or polkit agent
  m[onitor]       monitor NetworkManager changes
```



```
[unixuser@yul1010 ~]$ nmcli g
STATE      CONNECTIVITY  WIFI-HW  WIFI     WWAN-HW  WWAN
connected  full          enabled  enabled  enabled  enabled
[unixuser@yul1010 ~]$ nmcli n
enabled
[unixuser@yul1010 ~]$ nmcli r
WIFI-HW  WIFI     WWAN-HW  WWAN
enabled  enabled  enabled  enabled
[unixuser@yul1010 ~]$ nmcli c
NAME     UUID                                  TYPE      DEVICE
ens192   03da7500-2101-c722-2438-d0d006c28c73  ethernet  ens192
virbr0   08e81773-04c9-4c0c-afba-c25ccea2e0bf  bridge    virbr0
[unixuser@yul1010 ~]$ nmcli d
DEVICE      TYPE      STATE      CONNECTION
ens192      ethernet  connected  ens192
virbr0      bridge    connected  virbr0
lo          loopback  unmanaged  --
virbr0-nic  tun       unmanaged  --
```

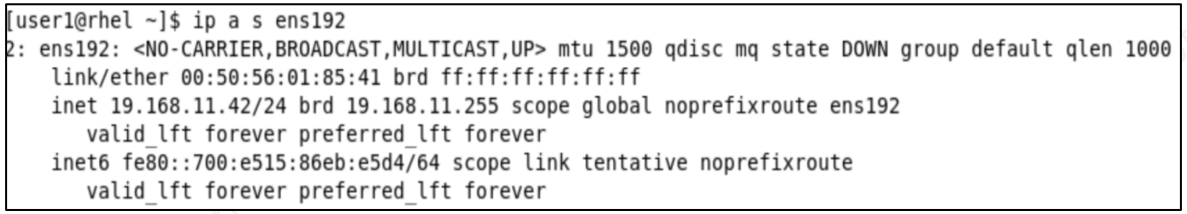
ip command

ip address can also be used to get ip address



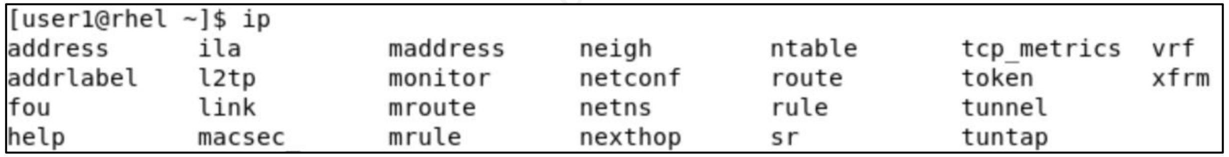
```
[user1@rhel ~]$ ip address show ens192
2: ens192: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN group default qlen 1000
    link/ether 00:50:56:01:85:41 brd ff:ff:ff:ff:ff:ff
    inet 19.168.11.42/24 brd 19.168.11.255 scope global noprefixroute ens192
       valid_lft forever preferred_lft forever
    inet6 fe80::700:e515:86eb:e5d4/64 scope link tentative noprefixroute
       valid_lft forever preferred_lft forever
```

Alternatively, shorter form of options



```
[user1@rhel ~]$ ip a s ens192
2: ens192: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN group default qlen 1000
    link/ether 00:50:56:01:85:41 brd ff:ff:ff:ff:ff:ff
    inet 19.168.11.42/24 brd 19.168.11.255 scope global noprefixroute ens192
       valid_lft forever preferred_lft forever
    inet6 fe80::700:e515:86eb:e5d4/64 scope link tentative noprefixroute
       valid_lft forever preferred_lft forever
```

Various options can be used with ip command



```
[user1@rhel ~]$ ip
address    ila      maddress  neigh    ntable  tcp_metrics  vrf
addrlabel  l2tp     monitor   netconf  route   token        xfrm
fou        link     mroute    netns    rule    tunnel
help       macsec   mrule     nexthop  sr      tuntap
```

Refer Chapter 16 of Course Book- Required reading

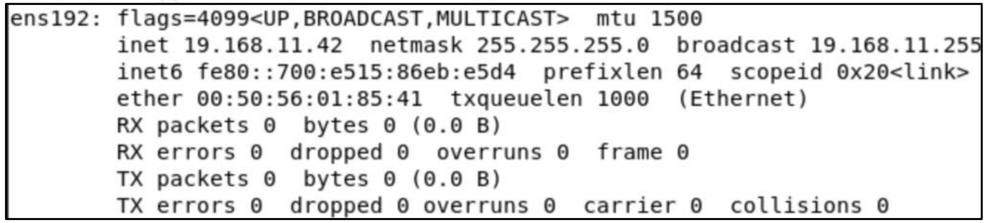
Configuring IP networking with nmcli

- To find detailed information about network interface :
  nmcli -p con show *networkinterface*
- To create a static Ethernet connection with IPv4 address and gateway: **nmcli con add type ethernet con-name** *connectionname* **ifname** *nwinterfacename* **ip4** *ipv4address/cidr* **gw4** *gatewayaddress*
- IPv6 can be added with ipv6 address and gateway information by adding ip6 and gw6 options
- To set IPv4 DNS server address: **nmcli con mod** connectionname **ipv4.dns** "*dnsserver addresses with space*"
- To set IPv6 DNS server addresses replace it with ipv6.dns

Recommended readings: Configuring IP Networking with nmcli

Network Administration

- **ifconfig**
  - Required net-tools package installed
  - Lists all networking interfaces available, including loopback interface



```
ens192: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 19.168.11.42  netmask 255.255.255.0  broadcast 19.168.11.255
        inet6 fe80::700:e515:86eb:e5d4  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:01:85:41  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

- To disable the network interface

  ifconfig *nw_interface* **down**
- To enable the network interface

  ifconfig *nw_interface* **up**

  ifconfig *nw_interface IP_addr* **netmask** *netmask_addr* **up**

- This **ifconfig** command is replaced by **ip** command
  https://www.redhat.com/sysadmin/ifconfig-vs-ip

arp replaced by **ip neigh**

- Address resolution protocol used for resolution of ip address to physical address (MAC Address). A table, usually called the ARP cache, is used to maintain a correlation between each MAC address and its corresponding IP address. Files : /proc/net/arp

Refer Chapter 16 of Course Book- Required reading

Network tools

lshw is a small tool to extract detailed information on the hardware configuration of the machine.

- It can report exact memory configuration, firmware version, mainboard configuration, CPU version and speed, cache configuration, bus speed, etc.
- To extract specific information about network or memory or other hardware the options -class can be used.
- To find the available class, -short or -businfo option can be used
- The output can be saved to html or other available options.
- Refer **man lshw** for more information

ethtool - query or control network driver and hardware settings

- It can help to view supported features and configured settings of an Ethernet interface: **ethtool** *nw_interface_name*

Network commands/tools

ping

- To check if an interface is responding

ping6

- Check if an ipv6 interface is responding

traceroute

- Tracks the route that packets take on an IP network from local computer to the network host specified.

route

- Is used to build the routing tables (in memory) implemented for routing packets and to display the routing information
- replaced with **ip route**

netstat

- Prints routing tables, network statistics, network connections
- replaced with **ss** command

nslookup

- Nslookup is a program to query Internet domain name servers.
- replaced with **dig** command

Routing

- To add routing entry in the routing table,

  sudo ip route add *network* **via** *gateway* **dev** *interface_name*

- The route entry is not permanent (static/persistent), it is only temporary and will be rolled back on next reboot
- To make this static route permanent (persistent/static), need to add this route in **/etc/sysconfig/network-scripts/route-*interface_name*** file and an example is given below.
- For example if need to route to network 10.1.1.0/24 using gateway 10.10.10.1, on interface ens192, use **ANY ONE** of the following format in the file /etc/sysconfig/network-scripts/route-interface_name.
- **Static routing – IP command arguments format**

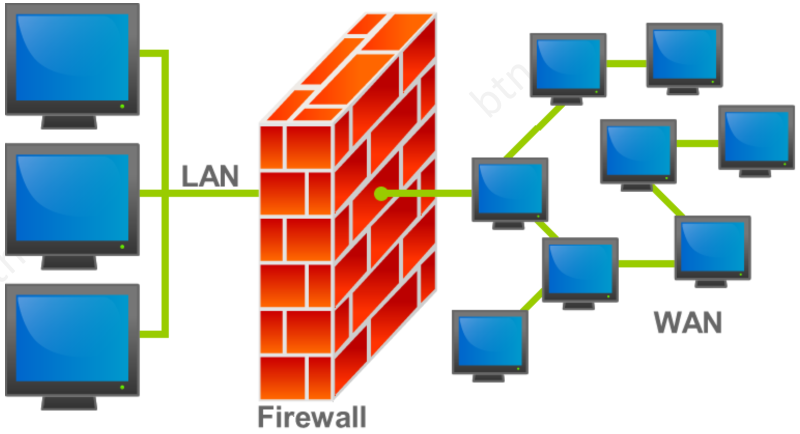
```
10.1.1.0/24 via 10.10.10.1 dev ens192
```

- **Static routing – Network/Netmask Directives format**

```
ADDRESS0=10.1.1.0
NETMASK0=255.255.255.0
GATEWAY0=10.10.10.1
```

Firewall

- Firewall manages security by allowing or denying traffic inbound and outbound as configured for the private network to protect it from public network.
- There are several types of firewalls,
  - Proxy FW
  - Stateful Inspection FW
  - Unified Threat Management FW
  - Next-generation FW
  - Threat-focused Next-generation FW



- One of firewall types, performs **data packet filtering**.
- Based on pre-defined *rules*, a firewall intercepts each inbound and outbound data packet, inspects its header, and decides whether to allow the packet to pass through.
- A port is defined in the ***/etc/services*** file for each network service available on the system, and is typically standardized across all network operating systems, including RHEL

Linux firewall

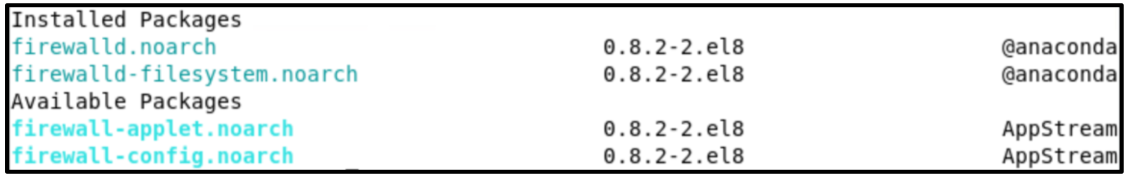
- firewalld: Use the firewalld utility to configure a firewall on workstations. The utility is easy to use and covers the typical use cases for this scenario.
- Since RHEL7, the iptables service and firewall rulesets may be configured and managed through a new dynamic firewall service daemon called **firewalld**.
- The major advantage is the daemon’s ability to immediately apply the updates without causing a disruption to current network connections, and this can be done anytime
- nftables: Use the nftables utility to set up complex firewalls, such as for a whole network.
- The **nftables** framework provides packet classification facilities and it is the designated successor to the iptables, ip6tables, arptables, and ebtables tools. It offers numerous improvements in convenience, features, and performance over previous packet-filtering tools
- **nftables** predecessor **iptables** uses host-based packet-filtering that communicates with the **netfilter** module in the kernel for policing the flow of data packets.
- iptables: The iptables utility on Red Hat Enterprise Linux 8 uses the nf_tables kernel API instead of the legacy back end. The nf_tables API provides backward compatibility so that scripts that use iptables commands still work on Red Hat Enterprise Linux 8. For new firewall scripts, Red Hat recommends to use nftables.

To avoid that the different firewall services influence each other, run only one of them on a RHEL host, and disable the other services.

Reference RedHat Documentation

firewalld

- Firewalld packages:



```
Installed Packages
firewalld.noarch                              0.8.2-2.el8                    @anaconda
firewalld-filesystem.noarch                   0.8.2-2.el8                    @anaconda
Available Packages
firewall-applet.noarch                        0.8.2-2.el8                    AppStream
firewall-config.noarch                        0.8.2-2.el8                    AppStream
```

- Service of firewalld: **firewalld**
- firewalld service daemon performs management operations at the
  - command line using **firewall-cmd**
  - graphically using the **firewall-config**
- Network ports in firewalld may also be defined directly using the cmdline or gui.

Required reading:
man firewalld
man firewall-cmd
firewall-cmd --help
https://firewalld.org/

- firewalld configuration directory: /etc/firewalld
- firewalld configuration file is /etc/firewalld/firewalld.conf
- Default fallback configuration provided by firewalld for icmptypes, services and zones are available in /usr/lib/firewalld *(also referred as firewalld templates location)*
- A service typically contains a port number, protocol, and an IP address.
- System defined rules are stored as xml files in /usr/lib/firewalld/services
- User-defined rules are stored as xml files in /etc/firewalld/services

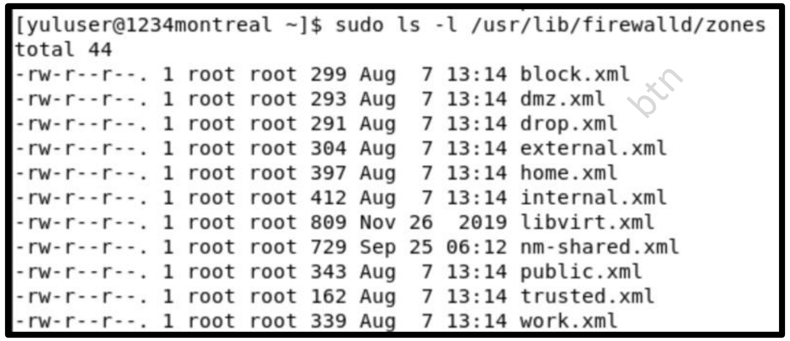
firewall-cmd is administrator command and requires **sudo** always.

Refer Chapter 20

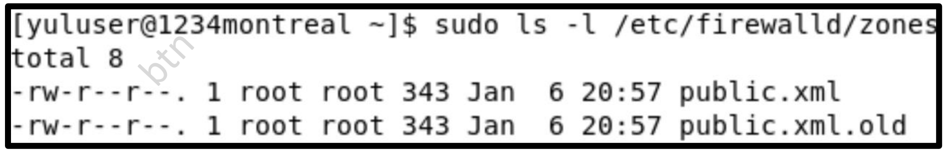
firewalld zones

- Zones define the level of trust for network connections based on principles such as a source IP or network interface for incoming network traffic
- firewalld presents the concept of zones that allow us to define policies based on the trust level for
  - network connections,
  - interfaces, and
  - source IP addresses that are bound to the zone.
- A zone may include configuration items comprising
  - services, ports,
  - protocols,
  - masquerading,
  - port forwarding,
  - ICMP filters, and
  - rich language rules.

- The firewalld software package provides several pre-defined zone files in the XML format in the /usr/lib/firewalld/zones/ directory.
- Of these, the **public zone** is the default and it is activated by default when the firewalld service is started.
- We may create custom zones to meet specific requirements
- Refer to Table 20-1 of course book for zone description



```
[yuluser@1234montreal ~]$ sudo ls -l /usr/lib/firewalld/zones
total 44
-rw-r--r--. 1 root root 299 Aug  7 13:14 block.xml
-rw-r--r--. 1 root root 293 Aug  7 13:14 dmz.xml
-rw-r--r--. 1 root root 291 Aug  7 13:14 drop.xml
-rw-r--r--. 1 root root 304 Aug  7 13:14 external.xml
-rw-r--r--. 1 root root 397 Aug  7 13:14 home.xml
-rw-r--r--. 1 root root 412 Aug  7 13:14 internal.xml
-rw-r--r--. 1 root root 809 Nov 26  2019 libvirt.xml
-rw-r--r--. 1 root root 729 Sep 25 06:12 nm-shared.xml
-rw-r--r--. 1 root root 343 Aug  7 13:14 public.xml
-rw-r--r--. 1 root root 162 Aug  7 13:14 trusted.xml
-rw-r--r--. 1 root root 339 Aug  7 13:14 work.xml
```



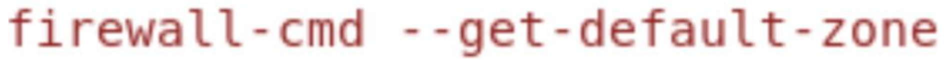
```
[yuluser@1234montreal ~]$ sudo ls -l /etc/firewalld/zones
total 8
-rw-r--r--. 1 root root 343 Jan  6 20:57 public.xml
-rw-r--r--. 1 root root 343 Jan  6 20:57 public.xml.old
```

Refer Chapter 20

firewalld

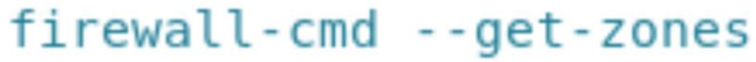
- To list the firewall services: firewall-cmd --list-services
- To display state: firewall-cmd --state
- To add firewall service:
  firewall-cmd --permanent --add-service *servicename*
  firewall-cmd --reload
- To add TCP port:
  firewall-cmd --permanent --add-port *port/*tcp
  firewall-cmd --reload
- To add host IP or network
  firewall-cmd --permanent --add-source *hostIP*
  firewall-cmd --reload
  (use networkaddress/CIDR inplace of hostIP for Network source)
- To remove firewall-service
  firewall-cmd --permanent --remove-service *servicename*
  firewall-cmd --reload
- To remove TCP port
  firewall-cmd --permanent --remove-port *port*/tcp
  firewall-cmd --reload
- For more options refer firewall-cmd --help

To find default zone :



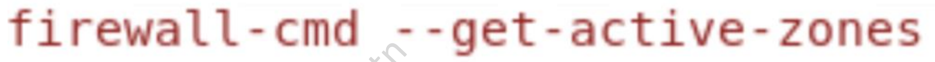
```
firewall-cmd --get-default-zone
```

To list all available zones:



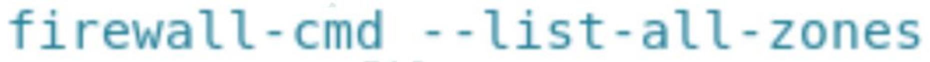
```
firewall-cmd --get-zones
```

To list active zone:



```
firewall-cmd --get-active-zones
```

To list all zone info:



```
firewall-cmd --list-all-zones
```

To list specific zone info:

firewall-cmd --info-zone *zonename*

To set default zone

firewall-cmd --set-default-zone *zonename*

To block a host IP or Network

firewall-cmd --permanent --add-rich-rule 'rule family=ipv4
source address=*networkaddr/CIDR* reject'

(user hostIP in place of networkaddr/CIDR for host IP)

For more options, use firewall-cmd --help

More reference rich-rule , add-rich-rule

Refer Chapter 20 – Course book

Exercise 20-1, 20-2, 20-3 of course book

nftables

- To find module information of nftables: modinfo nf_tables
- To list nftables modules: lsmod |grep nf
- systemd nftables service: **nftables**
- nftables configuration file: **/etc/sysconfig/nftables.conf**
- nftables scripts are stored in: **/etc/nftables**
- userspace command for nftables is **nft**
- In nftables a **table** is simply a namespace and collection of chains, rules, and sets, and other objects.
- table is top most in the hierarchy of nftables configuration followed by chains and rules
- **Chains** are the objects that will contain our firewall rules

https://linux-audit.com/nftables-beginners-guide-to-traffic-filtering/

getting-started-with-nftables_configuring-and-managing-networking

nftables

- nftables, table need to qualify address family - ip, ip6, inet, arp, bridge or netdev
- Address family **ip** for ip4, ip6 for ip6, inet for both ip4 and ip6, netdev for ingress filtering or traffic coming into system
- To create a **table**: **nft add table** *family tablename*
- To create **chain**: **nft add chain** *family tablename chainname*
- To create **rule**:

  nft add rule *family tablename chainname* **tcp dport** *protocol* **accept/reject/drop**
  nft add rule *family tablename chainname* **ip saddr** *ipaddress* **accept/reject/drop**

- To list the **rules** : **nft list ruleset**
- To delete **ruleset**, find the handle number of rule and then delete
- To find **handle: nft --handle list ruleset**
- To delete **rule: nft delete rule** *family tablename chainname* **handle** *handlenumber*
- To delete **chain:** nft delete chain *family tablename chainname*
- To delete **table:** nft delete table *family tablename*

*(**saddr** for source IP and **daddr** for destination IP)*

https://linux-audit. com/nftables-beginners-guide-to-traffic-filtering/
getting-started-with-nftables_configuring-and-managing-networking