RHCEv8 Online Class 30052021 10:00pm
RHCSA-sysadmin
- Managing Networking II

Managing Networking
3-Network
-output is packet
-as device router, l3switch
-as protocols ipv4, ipv6

what is binary, decimal and hexadecimal units

binary	**decimal**	**hexadecimal**
0/1	0	0
on/off	1	1
	2	2
	3	3
	4	4
	5	5
	6	6
	7	7
	8	8
	9	9
		A 10
		B 11
		C 12
		D 13
		E 14
		F 15

smallest unit computer science, its **bit** it's **on/off 0/1**

1bit	
4bits	**nibble**
8bits	**byte/octet**

7	**6**	**5**	**4**	**3**	**2**	**1**	**0**

2 power bit number

7	6	5	4	3	2	1	0
2	2	2	2	2	2	2	2
128	**64**	**32**	**16**	**8**	**4**	**2**	**1**

128+65+32+16+8+4+2+1=**255**

if whole bits be **on/1**->**255**

|	**|**	**|**	**|**	**|**	**|**	**|**	**|**

or
if whole bits be **off/0**->**0**

0	**0**	**0**	**0**	**0**	**0**	**0**	**0**

ex:
||00|000->200

what is ipv4

its 32bits in to 4byts/octet divided by **.** and each byte is between 0 to 255
||||||||.||||||||.||||||||.||||||||

ipv4 class
to better use ipv4 divided in to default 5classes

A0	to	126	
B128	to	191	
C192	to	223	
D224	to	239	multicasting
E240	to	255	R/D

NOTE:
-to detect ip's class check first octet from left with ip's classes
-127 uses for loopback interface, internal peruse
-0 uses to define network-id
-255 uses to define broadcast-ip

ipv4 comes in to 2types:
1-public, means has ping over internet network.
2-private, means doesn’t have ping over internet network.
A10.0.0.0 to 10.255.255.255
B172.16.0.0 to B172.31.255.255
C192.168.0.0 to 192.168.255.255

ipv4 by default has 2parts
network part + host part
fixed changeable

```
A ||||||||.||||||||.||||||||.||||||||     most ipv4’s
     N       H       H      H
B||||||||.||||||||.||||||||.||||||||
    N       N      H      H
C||||||||.||||||||.||||||||.||||||||      least ipv4’s
    N       N       N      H
```

ipv4 example
192.168.50.12
10.1.1.1
172.25.36.20
180.56.25.63

when2ipv4 can connect direct to each other without router
1-should be in same CLASS
2-should has same network part

ex:

```
10.1.1.1.1 and 11.1.1.2            ->no direct ping
      A                  A
172.25.10.1 and 172.25.12.25       ->yes, direct ping
        B                       B
```

how machine will detect ip’s class
humans:
compare with default classes
ex:
152.36.25.89 ->B class
machines:
through subnet mask.

what is subnet mask
defines ip’s class for machines
who calculate
to calculate subnet mask Network part should be 1/on and Host part should be 0/off
A 255.0.0.0
B 255.255.0.0
C 255.255.255.0

192.168.1.1 255.255.255.0
172.25.250.10 255.255.0.0

NOTE: subnet mask decides about ip’s behavior

ipv4 comes in to:
-classfull
standard ip and subnet mask
10.1.1.1 255.0.0.0
-classless
ip doesn’t match with subnet mask
10.1.1.1 255.255.255.0

what is cidr/prefix
instead of **subnet mask** use **prefix**
to calculate prefix, **count each class network bits**
A/8
B/16
C/24
ex:
classless
10.1.1.1/24
10.1.1.1/8
10.1.1.1/16
classfull
10.1.1.1/8
172.25.250.10/16
192.168.100.1/24

NOTE: ip without subnet mask or prefix, should take it as **classfull**
150.41.23.125 ->B class
ipv4 Analyze
CLASS
Network-ID
First IP
Last IP
Broadcast IP
number of hosts

ex:
192.168.10.1/24

CLASS >C

Network-ID ->Host part should be 0
192.168.10.0

First IP ->network-id +1
192.168.10.1

Last IP ->Broadcast IP - 1
192.168.10.254

Broadcast IP -> Host part should be 1
192.168.10.255

number of available hosts
2^h-2
2^8-2=256-2=254
192.168.10.1 to 192.168.10.254

NOTE: don’t set network-id and broadcast-ip over nic card.

what is ipv6

it comes to 128bits in binary language. 8parts*16bits=128bits

ipv6 example

||||||||||||||||.||||||||||||||||.||||||||||||||||.||||||||||||||||.||||||||||||||||.||||||||||||||||.||||||||||||||||.||||||||||||||||

binary should concert to hexadecimal

0010000000000001:0000110110111000:0000101000001011:0001001011110000:0000000000000000:0000000000000000:0000000000000000:0000000000000001

ex:
[0000][1101][1011][1000]

[0000]	[1101]	[1011]	[1000]	
3 2 1 0 ->bits number	3 2 1 0	3 2 1 0		
3 2 1 0	8+4+0+1	8+0+2+1		
2 2 2 2	**13**	**11**	**8**	->**0DB8**

0				

after convert to Hexadecimal

2001:**0db8**:**0a0b**:12f0:0000:0000:0000:**0001**

3rules to make ipv6 better

1-dicard leading 0
2001:**0db8**:**0a0b**:12f0:0000:0000:0000:**1**
to
2001:**db8**:**a0b**:12f0:0000:0000:0000:1

2-if 2 or more 0block coming after each, remove 0blocks and put ::
2001:db8:a0b:12f0:**0000**:**0000**:**0000**:1
to
2001:db8:a0b:12f0**::**1

3-replace 0000 to 0
2001:db8:a0b:12f0**::0000**
to
2001:db8:a0b:12f0**::0**

ipv6 type

1-global-unicast	->ipv6 public, it has ping on internet	ipv4public == ipv6 public
2-link-local	->ipv6 private, doesn’t ping on internet	ipv4private == ipv6 private
3-unique-local	->in public network acts as ipv6 public and in private network acts as private ipv6	

	ipv4	ipv6
local host	127.0.0.0	::1/128
default route	0.0.0./0	::
global-unicast		2000::/3
link-local		fe80::/64
unique-local		fd00::/8

Network services

1-DNS-Domain Name Service

maps name to ip and ip to name

```
# cat /etc/resolv.conf
search lab.example.com
nameserver 8.8.8.8
bindserver 8.8.8.8
```

2-DHCP-Dynamic Host Configuration Protocol

method to set ip
1-manual/static
set ip manual on nic
2-dynamic
DHCP Service take responsibility through:
DORA process

- Discover
- Offer
- Request
- Ack

3-GATEWAY

its exit interface
mostly to access from lan to Internet

How to set ip on NIC on linux

1-nmcli
2-nmtui
3-change through NIC configuration file

1-nmcli-Network Manager Command Line

```
# ip link
# ip addr show
or
# ip a s
# ifconfig
# ifconfig -a
# ifconfig ens160
```

till rhel6 nic cards came by eth0, eth1, …
rhel7 onwards linux shows us nic card firmware name
ens192

en	->Ethernet
ww	->wireless wan
wl	->wireless
s	->hostplug
o	->onboard
p	->pci

Format	Description
o<index>	on-board device index number
s<slot>[f<function>][d<dev_id>]	hotplug slot index number
x<MAC>	MAC address
p<bus>s<slot>[f<function>][d<dev_id>]	PCI geographical location
p<bus>s<slot>[f<function>][u<port>][..][c<config>][i<interface>]	USB port number chain

```
# ifconfig
ens160: flags=4163 ->interface is up
mtu 1500        ->maximum transfer unit 1500bytes
inet 192.168.216.3  netmask 255.255.255.0  broadcast 192.168.216.255        ->ipv4
inet6 fe80::2398:bdf9:43b4:51fa                                             ->ipv6
ether 00:0c:29:92:a4:30                                                     ->mac address
```

what is connection/profile

create connection for each time as u want change configuration on nic
and next time just call it
ex:

coss	->dynamic/dhcp
ibm	->static
dell	->static

create connection
-coss
dhcp

-ibm
ip: 10.1.1.50/8

-dell
ip:150.16.35.188/24
dns: 150.16.35.254
gw: 150.16.35.253
dns-search: dell.in

-show number of connections

```
# nmcli connection show
NAME                          UUID                                  TYPE      DEVICE
ens160  27b51475-262c-456c-a3ca-a5c2aad262ff  ethernet  ens160
# nmcli connection show --active
```

-create new connection

```
# nmcli connection add con-name "coss" type ethernet autoconnect yes ifname ens192 ipv4.method auto
# nmcli connection add con-name "ibm" ifname ens192 autoconnect yes type ethernet ipv4.addresses "10.1.1.150/24" ipv4.method manual
# nmcli connection add con-name "dell" autoconnect yes type ethernet ifname ens192 ipv4.addresses "150.16.35.188/24"
ipv4.dns "150.16.35.254" ipv4.gateway "150.16.35.253" ipv4.dns-search "dell.in" ipv4.method manual
```

-reload nmcli after interaction

```
# nmcli connection reload
```

-connection details

```
# nmcli connection show "dell"
# nmcli connection show "dell" | grep -i "ipv4"
# nmcli connection show "dell" | grep -i "ipv4.address"
```

-active connections

```
# nmcli connection up "dell"
```

-modify connection
-ibm
ip: 10.1.1.201/8

```
# nmcli connection modify "ibm" ipv4.addresses "10.1.1.201/8" ipv4.method manual
# nmcli connection reload
# nmcli connection show "ibm" | grep -i "ipv4"
```

-delete connection

```
# nmcli connection delete "coss"
# nmcli connection reload
```

How to set hostname

```
# hostnamectl set-hostname <new name>
# bash                                                                    ->when type bash history will clean
# hostnamectl
```

2-nmtui

```
# nmtui
```

its semi gui mode

3-change through NIC configuration file

```
# ll /etc/sysconfig/network-scripts/ifcfg-
ifcfg-dell    ifcfg-ens160  ifcfg-ens192  ifcfg-ibm
# cat /etc/sysconfig/network-scripts/ifcfg-ens192
DEVICE=ens192
ONBOOT=yes   ==  autoconnect
NAME=ens192  == con-name
BOOTPROTO=none
none/static        ->static
dhcp               ->dynamic
```

check connectivity

```
# ping 8.8.8.8
# ping -i 3 8.8.8.8                 delay
# ping -c 3 8.8.8.8                 limit request
# ping -I ens160 8.8.8.8            select exit Interface
```

Managing Network Security
firewall helps us and control incoming and outgoing traffic in to network, host, device

on linux
FIREWALL ARCHITECTURE CONCEPTS
The Linux kernel includes **netfilter**, a framework for network traffic operations such as packet filtering, network address translation and port translation.

Nftables enhances netfilter
The Linux kernel also includes **nftables**, a new filter and packet classification subsystem that has enhanced portions of netfilter's code, but retaining the netfilter architecture such as networking stack hooks, connection tracking system, and the logging facility.

Introducing firewalld
Firewalld is a dynamic firewall manager, a front end to the nftables framework using the nft command.
https://firewalld.org/

work with firewalld
1-cli
2-gui
3-web access(cockpit)
4-edit firewalld configuration files

firewalld info
package:
    cli->firewalld.noarch
    gui->firewall-config.noarch
daemon: firewalld.service
config file:
    running->/etc/firewalld/
    main->/usr/lib/firewalld/
log: /var/log/firewalld

Implement firewalld on linux

```
# yum list firewall*
# yum install firewall* -y            ->install/update
# systemctl enable firewalld.service
# systemctl start firewalld.service
# systemctl status firewalld.service
# systemctl restart firewalld.service
```

-verify

```
# firewall-cmd -- press tab tab
# firewall-config
# firewall-cmd --version
0.8.2
# firewall-cmd --state
running
```

firewalld in cli

```
# firewall-cmd --
```

firewalld in gui

```
# firewall-config
```

firewalld in direct config files

```
# firewall-config cat /etc/firewalld/zones/public.xml
```

firewalld concepts

to operates firewalld need to know about:
1-zone
2-service-name

1-zone

zone is level of trust

```
# firewall-cmd --get-zones
block dmz drop external home internal public trusted work
# firewall-cmd --list-all-zones
# firewall-cmd --get-default-zone
public
```

2-service-name

instead of port can select service-name too

```
# cat /etc/services
```

how many ports are available?
65535 in to

0	to	1023	->well-known ports
1024	to	49151	->registered ports
49152	to	65535	->dynamic/private ports

current opened service in firewalld

```
# firewall-cmd --list-services
cockpit dhcpv6-client ssh
```

list of total service-name

```
# firewall-cmd --get-services
```

firewalld-cli

ex:
open port 23/tcp

current zone

```
# firewall-cmd --list-all
```

remote zone

```
# firewall-cmd --list-all --zone=drop
# firewall-cmd --permanent --add-port=23/tcp
or
# firewall-cmd --permanent --add-port=23/udp
```

after interaction with firewall, reload is mandatory

```
# firewall-cmd --reload
# firewall-cmd --list-all
```

ex:
remove **dhcpv6-client** from public zone

```
# firewall-cmd --list-all
# firewall-cmd --permanent --remove-service=dhcpv6-client
# firewall-cmd --reload
# firewall-cmd --list-all
```

ex:
add dns service-name

```
# firewall-cmd --permanent --add-service=dns
# firewall-cmd --reload
# firewall-cmd --list-all
```

ex:
add multiple service-name tftp, http, https

```
# firewall-cmd --permanent --add-service={tftp,http,https}
# firewall-cmd --reload
# firewall-cmd --list-all
```

ex:
add telnet to internal zone

```
# firewall-cmd --permanent --add-service=telnet --zone=internal
# firewall-cmd --reload
# firewall-cmd --list-all --zone=internal
```

firewalld in direct config files

```
# firewall-cmd --get-default-zone
public
# vim /etc/firewalld/zones/public.xml
<zone>
<service name="dhcp"/>
<port port="53" protocol="tcp"/>
</zone>
:wq!
```

Analyzing Servers and Getting Support
DESCRIBING THE WEB CONSOLE
Web Console is a web-based management interface for Red Hat Enterprise Linux 8 designed for managing and monitoring your servers.

web console(cockpit) info
package: cockpit.x86_64
daemon: cockpit.socket
port: 9090/tcp

implement web console on rhel

```
# yum install cockpit.x86_64 -y
# systemctl enable cockpit.socket
# systemctl start cockpit.socket
# systemctl status cockpit.socket
# firewall-cmd --permanent --add-port=9090/tcp
or
# firewall-cmd --permanent --add-service=cockpit
# firewall-cmd --reload
```

open browser and type:
https://172.25.250.10:9090/
u: root
p: redhat