澳 門 理 工 學 院
Instituto Politécnico de Macau
Macao Polytechnic Institute

COMP 225 Network and System Administration

Notes #8: Linux Commands on Network Cards

K. L. Eddie Law, PhD

Macao Polytechnic Institute
School of Applied Sciences
Academic Year 2020-2021, 2nd Semester

Network Interface Cards

- Computers nowadays are set with at least one network card
- Easily to add a few more if deemed needed
- Run `ping` to test if network card working and connecting to the network

```
$ ping www.google.com
```

- `ping` belongs to ICMP messages

IP Commands

- Evolving commands to set IP addresses for network interfaces
- The old time favorites were
  - `ifconfig`
  - `netstat`
  - `route`
- These commands can still be re-installed on Ubuntu again using

```
$ sudo apt install net-tools
```


IP Assignment Tools

- The new "ip" tool is adopted by Ubuntu and Red Hat (in fact, Red Hat supports both tools)

Legacy Tools	New Tools
ifconfig	ip addr
netstat	ss, ip route, ip -s link, ip maddr
route	ip route
arp	ip neighbor
iptunnel	ip tunnel

To Check Network Interfaces

- For the Link layer hardware interfaces, usually
  - Wired – Ethernet card, its name usually starts with an "e..."
  - Wireless – wifi card, its name usually starts with an "w..."
  - These address are the MAC (Medium Access Control) addresses
- To check the names of all hardware interfaces, we can use either

```
$ ip link [show | list]
$ ip addr [show | list]
```

- Both commands show all found network interfaces, the word "show" and "`list`" are interchangeable; but in these cases, in fact, both are redundant

MAC Addresses

- Hardware interfaces have names, and MAC addresses
  - 6-byte in size, similar to IPv6, uses colon-hexadecimal notation
  - One byte then a colon, and so on
  - No shorten notations, as those available for IPv6
- Device name and the associated MAC address can be found with

```
$ ip link
```

  - For example

Name of the interface

```
1: lo: <LOOPBACK,UP,LOWER)UP> mtu 65535 qdisc noqueue state…
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00:00
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER,UP> mtu 1500 qdisc…
   link/ether 08:00:27:a7:6e:d2 brd ff:ff:ff:ff:ff:ff
```

MAC address

The IP Address

- Through the command
  ```
  $ ip addr
  ```
  - We may get

```
1: lo: <LOOPBACK,UP,LOWER)UP> mtu 65535 qdisc noqueue state…
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00:00:00
   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…
   link/ether 08:00:27:a7:6e:d2 brd ff:ff:ff:ff:ff:ff
   inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
      valid_lft 76196sec preferred_lft 76196sec
   inet6 fe80::a00:27ff:fea7:6ed2/64 scope link
      valid_lft forever preferred_lft forever
```

IPv4 address

IPv6 address

Find "up" Devices or Find One Device

- Sometimes, only listing those some interfaces that are up
  ```
  $ ip link list up
  $ ip addr show up
  ```
- Like to get information about one interface only, e.g.,
  ```
  $ ip link list enp0s3
  ```
  ← **My interface card only**
  ```
  $ ip -4 addr show enp0s3
  ```
  ← **IPv4 address only**
  ```
  $ ip -6 addr show enp0s3
  ```
  ← **IPv6 address only**
- For hostname, can use
  ```
  $ hostname
  $ hostnamectl
  ```

To Change the State of an Interface

- Change the setting of an interface
  ```
  $ sudo ip link set dev {device name} {up | down}
  ```
- E.g., turn off the enp0s3 interface
  ```
  $ sudo ip link set dev enp0s3 down
  $ ip link list
  $ ip link show up
  ```
- Ok, enough fun, put it back up
  ```
  $ sudo ip link set dev enp0s3 up
  ```


The Command "ip addr"

- $ sudo ip addr {add|change|replace|del} IPADDR dev {device name}
  - Where IPADDR is typical IP address with netmask, support CIDR notation
  - Broadcast address not set in this command! (different from the old "ifconfig")
- E.g., add an IPv4 and associated broadcast address to enp0s3
  ```
  $ sudo ip addr add 10.0.0.2/24 broadcast 10.0.0.255 dev enp0s3
  ```

```
elaw@s1:~$ sudo ip addr add 10.0.0.2/24 broadcast 10.0.0.255 dev enp0s3
elaw@s1:~$ ip addr list enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 100
0
    link/ether 08:00:27:a7:6e:d2 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
       valid_lft 85381sec preferred_lft 85381sec
    inet 10.0.0.2/24 brd 10.0.0.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fea7:6ed2/64 scope link
       valid_lft forever preferred_lft forever
```

Other Settings

- Remove the added-on IPv4 address

```
$ sudo ip addr del 10.0.0.2/24 broadcast 10.0.0.255 dev enp0s3
```

- The interface gets IPv4 dynamically; if not, can set it using dhcp
  - `$ sudo dhclient -4 enp0s3`
  - Use "-6" for IPv6
- However, all we have done on screen are only working for the current active session
- All added/changed/modified/ settings do **NOT** survive a system reboot!

Netplan and Renderers

- Netplan has been used since 2019 in Ubuntu 19.04
- The old "`ifupdown`" scripting system was removed
- It indicates rendering software for setting up network addresses
  - `systemd-networkd` ← **Ubuntu server**
  - `NetworkManager` ← **Ubuntu desktop**
- Check if one of them is running

```
$ systemctl status { systemd-networkd | NetworkManager }
```

Netplan: Setting Network Interfaces on Booting

- Go check the file `$ ls /etc/netplan`
  - Suppose it shows a file named `00-installer-config.yaml`
  - If the filename is different, it is ok if it is a "yaml" file
  - on screen could be:

```
network:
  ethernets:
    enp0s3:
      dhcp4: true
  version: 2
```

Quite easy to read, if modifying it, run command `$ sudo netplan apply` to facilitate the changes

Static IP with Netplan: Example

- Configure with the `.yaml` file

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3:
      dhcp4: false
      addresses: [10.0.2.4/24]
      gateway4: 10.0.2.1
      nameservers:
        search: [example.com, otherdomain]
        addresses: [10.0.2.1, 8.8.8.8]
```

← Or, **NetworkManager**

- Then run `$ sudo netplan apply`

Gateway and Friends

- To get to the Internet , we need a gateway or router
- To find the default gateway

  `$ ip route`

  - And it shows for a host using DHCP service, for example,

```
default via 10.0.2.1 dev enp0s3 proto dhcp src 10.0.2.6 metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.6
10.0.2.1 dev enp0s3 proto dhcp scope link src 10.0.2.6 metric 100
```

- Any neighboring computers around??

  `$ ip neigh show`

  - If any shown, the last field can be STALE, DELAY, or REACHABLE

Routing Table (Not in Exam)

- Following commands are for adding new routes or deleting routes
- For adding:

  "default" is optional, add it to change the default route

  $ sudo ip route add {default} [network/netmask] via [gatewayIP]

  $ sudo ip route add {default} [network/netmask] dev [deviceName]
- For deleting:

  $ sudo ip route del default ← Delete the default route

  $ sudo ip route del [network/netmask] dev [deviceName]


QA