

EXP.NO:1B**DATE:25.07.2024****BASIC NETWORKING COMMANDS IN LINUX OPERATING SYSTEMS****Linux Networking Commands**

Every computer is connected to some other computer through a network whether internally or externally to exchange some information. This network can be small as some computers connected in your home or office, or can be large or complicated as in large University or the entire Internet.

Maintaining a system's network is a task of System/Network administrator. Their task includes network configuration and troubleshooting.

Here is a list of Networking and Troubleshooting commands:

ifconfig	Display and manipulate route and network interfaces.
ip	It is a replacement of ifconfig command.
traceroute	Network troubleshooting utility.
tracepath	Similar to traceroute but doesn't require root privileges.
ping	To check connectivity between two nodes.
netstat	Display connection information.
ss	It is a replacement of netstat.
dig	Query DNS related information.
nslookup	Find DNS related query.
route	Shows and manipulate IP routing table.
host	Performs DNS lookups.

arp	View or add contents of the kernel's ARP table.
iwconfig	Used to configure wireless network interface.
hostname	To identify a network name.
curl or wget	To download a file from internet.
mtr	Combines ping and tracepath into a single command.
whois	Will tell you about the website's whois.
ifplugstatus	Tells whether a cable is plugged in or not.

Explanation of the above commands:

1.ifconfig: ifconfig is short for interface configurator. This command is utilized in network inspection, initializing the interface, enabling or disabling an IP address, and configuring an interface with an IP address. Also, it is used to show the network and route interface. The basic details shown with ifconfig are:

- MTU
- MAC address
- IP address

Syntax:

Ifconfig

```

root@ip-10-10-38-111:~# ifconfig
docker0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
    inet6 fe80::42:16ff:fec8:24d5 prefixlen 64 scopeid 0x20<link>
    ether 02:42:16:c8:24:d5 txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 35 bytes 4761 (4.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

ens5: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 9001
    inet 10.10.38.111 netmask 255.255.0.0 broadcast 10.10.255.255
    inet6 fe80::4a:89ff:fe31:79dd prefixlen 64 scopeid 0x20<link>
    ether 02:4a:89:31:79:dd txqueuelen 1000 (Ethernet)
    RX packets 8907 bytes 715564 (715.5 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 6758 bytes 4150018 (4.1 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 15766 bytes 4606708 (4.6 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 15766 bytes 4606708 (4.6 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

veth9ddb7c8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::540c:a4ff:fe12:a53b prefixlen 64 scopeid 0x20<link>
    ether 56:0c:a4:12:a5:3b txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 57 bytes 7476 (7.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

vethf098cf2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::10a6:daff:fe84:d9f2 prefixlen 64 scopeid 0x20<link>
    ether 12:a6:da:84:d9:f2 txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 58 bytes 7566 (7.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

2. **ip:** It is the updated and latest edition of ifconfig command. The command provides the information of every network, such as ifconfig. Also, it can be used to get information about a particular interface. **Syntax:**

1. ip a
2. ip addr

```

File Edit View Search Terminal Help
veth9ddb7c8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::540c:a4ff:fe12:a53b prefixlen 64 scopeid 0x20<link>
    ether 56:0c:a4:12:a5:3b txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 57 bytes 7476 (7.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

vethf098cf2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::10a6:daff:fe84:d9f2 prefixlen 64 scopeid 0x20<link>
    ether 12:a6:da:84:d9:f2 txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 58 bytes 7566 (7.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@ip-10-10-38-111:~# 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: ens5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc mq state UP group default qlen 1000
    link/ether 02:4a:89:31:79:dd brd ff:ff:ff:ff:ff:ff
    inet 10.10.38.111/16 brd 10.10.255.255 scope global dynamic ens5
        valid_lft 2768sec preferred_lft 2768sec
    inet6 fe80::4a:89ff:fe31:79dd/64 scope link
        valid_lft forever preferred_lft forever
3: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:16:c8:24:d5 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
    inet6 fe80::42:16ff:fec8:24d5/64 scope link
        valid_lft forever preferred_lft forever
5: vethf098cf2@if4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UP group default
    link/ether 12:a6:da:84:d9:f2 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet6 fe80::10a6:daff:fe84:d9f2/64 scope link
        valid_lft forever preferred_lft forever
7: veth9ddb7c8@if6: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UP group default
    link/ether 56:0c:a4:12:a5:3b brd ff:ff:ff:ff:ff:ff link-netnsid 1
    inet6 fe80::540c:a4ff:fe12:a53b/64 scope link
        valid_lft forever preferred_lft forever
root@ip-10-10-38-111:~#
  
```

3. **traceroute:** The traceroute command is one of the most helpful commands in the networking field. It's used to balance the network. It identifies the delay and decides the pathway to our target. Basically, it aids in the below ways:
 - It determines the location of the network latency and informs it.
 - It follows the path to the destination.
 - It gives the names and recognizes all devices on the path.

Syntax:

traceroute <destination>

```
root@ip-10-10-38-111:~# traceroute www.google.com
traceroute to www.google.com (209.85.202.104), 30 hops max, 60 byte packets
 1  * * *
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
```

4. **tracpath:** The tracpath command is the same as the traceroute command, and it is used to find network delays. Besides, it does not need root privileges. By default, it comes preinstalled in Ubuntu. It traces the path to the destination and recognizes all hops in it. It identifies the point at which the network is weak if our network is not strong enough.

Syntax: tracpath

<destination>

```
Applications Places System Fri 9 Aug, 07:10 AttackBox IP: 10.10.38.111
root@ip-10-10-38-111:~#
File Edit View Search Terminal Help
root@ip-10-10-38-111:~# tracpath www.google.com
17: [LOCALHOST] pmtu 1500
1: no reply
2: no reply
3: no reply
4: no reply
5: no reply
6: no reply
7: no reply
8: no reply
9: no reply
10: no reply
11: no reply
12: no reply
13: no reply
14: no reply
15: no reply
```

5. **ping:** It is short for Packet Internet Groper. The ping command is one of the widely used commands for network troubleshooting. Basically, it inspects the network connectivity between two different nodes.

Syntax: ping

<destination>

```

Applications Places System Fri 9 Aug, 07:12 AttackBox IP: 10.10.38.111
root@ip-10-10-38-111: ~
File Edit View Search Terminal Help
Usage: ping -6 [-aAbBdDrHhLnOqrRUVv] [-c count] [-i interval] [-I interface]
[-l preload] [-m mark] [-M pmtudisc_option]
[-N nodeinfo_option] [-p pattern] [-Q tclass] [-s packetsize]
[-S sndbuf] [-t ttl] [-T timestamp_option] [-w deadline]
[-W timeout] destination
root@ip-10-10-38-111:~# ping 10.10.38.111
ping: 10.10.38.111: Name or service not known
root@ip-10-10-38-111:~# ping 10.10.38.111
PING 10.10.38.111 (10.10.38.111) 56(84) bytes of data.
64 bytes from 10.10.38.111: icmp_seq=1 ttl=64 time=0.042 ms
64 bytes from 10.10.38.111: icmp_seq=2 ttl=64 time=0.037 ms
64 bytes from 10.10.38.111: icmp_seq=3 ttl=64 time=0.027 ms
64 bytes from 10.10.38.111: icmp_seq=4 ttl=64 time=0.029 ms
64 bytes from 10.10.38.111: icmp_seq=5 ttl=64 time=0.023 ms
64 bytes from 10.10.38.111: icmp_seq=6 ttl=64 time=0.044 ms
64 bytes from 10.10.38.111: icmp_seq=7 ttl=64 time=0.025 ms
64 bytes from 10.10.38.111: icmp_seq=8 ttl=64 time=0.027 ms
64 bytes from 10.10.38.111: icmp_seq=9 ttl=64 time=0.032 ms
64 bytes from 10.10.38.111: icmp_seq=10 ttl=64 time=0.025 ms
64 bytes from 10.10.38.111: icmp_seq=11 ttl=64 time=0.035 ms
64 bytes from 10.10.38.111: icmp_seq=12 ttl=64 time=0.029 ms
64 bytes from 10.10.38.111: icmp_seq=13 ttl=64 time=0.041 ms
64 bytes from 10.10.38.111: icmp_seq=14 ttl=64 time=0.044 ms
64 bytes from 10.10.38.111: icmp_seq=15 ttl=64 time=0.039 ms
64 bytes from 10.10.38.111: icmp_seq=16 ttl=64 time=0.030 ms
64 bytes from 10.10.38.111: icmp_seq=17 ttl=64 time=0.040 ms
64 bytes from 10.10.38.111: icmp_seq=18 ttl=64 time=0.027 ms
64 bytes from 10.10.38.111: icmp_seq=19 ttl=64 time=0.042 ms
64 bytes from 10.10.38.111: icmp_seq=20 ttl=64 time=0.042 ms
64 bytes from 10.10.38.111: icmp_seq=21 ttl=64 time=0.027 ms
64 bytes from 10.10.38.111: icmp_seq=22 ttl=64 time=0.025 ms
64 bytes from 10.10.38.111: icmp_seq=23 ttl=64 time=0.037 ms
64 bytes from 10.10.38.111: icmp_seq=24 ttl=64 time=0.042 ms
64 bytes from 10.10.38.111: icmp_seq=25 ttl=64 time=0.181 ms
64 bytes from 10.10.38.111: icmp_seq=26 ttl=64 time=0.028 ms
64 bytes from 10.10.38.111: icmp_seq=27 ttl=64 time=0.043 ms
64 bytes from 10.10.38.111: icmp_seq=28 ttl=64 time=0.025 ms
64 bytes from 10.10.38.111: icmp_seq=29 ttl=64 time=0.032 ms
64 bytes from 10.10.38.111: icmp_seq=30 ttl=64 time=0.031 ms
64 bytes from 10.10.38.111: icmp_seq=31 ttl=64 time=0.047 ms
^C
--- 10.10.38.111 ping statistics ---
31 packets transmitted, 31 received, 0% packet loss, time 30723ms
rtt min/avg/max/mdev = 0.023/0.038/0.181/0.027 ms
root@ip-10-10-38-111:~#

```

6. **netstat:** It is short for network statistics. It gives statistical figures of many interfaces, which contain open sockets, connection information, and routing tables.

Syntax:

Netstat

```

Applications Places System Fri 9 Aug, 07:14 AttackBox IP: 10.10.38.111
root@ip-10-10-38-111: ~
File Edit View Search Terminal Help
unlX 3 [ ] STREAM CONNECTED 30757 @/tmp/dbus-syGt6LJFW9
unlX 3 [ ] STREAM CONNECTED 29383 /run/systemd/journal/stdout
unlX 3 [ ] STREAM CONNECTED 28959 /run/systemd/journal/stdout
unlX 3 [ ] STREAM CONNECTED 63562
unlX 3 [ ] STREAM CONNECTED 30129
unlX 3 [ ] STREAM CONNECTED 25464 /var/run/dbus/system_bus_socket
unlX 3 [ ] STREAM CONNECTED 27535
unlX 3 [ ] STREAM CONNECTED 29397 /run/systemd/journal/stdout
unlX 2 [ ] DGRAM 32416
unlX 3 [ ] STREAM CONNECTED 29811
unlX 3 [ ] STREAM CONNECTED 29148
unlX 3 [ ] STREAM CONNECTED 24921
unlX 3 [ ] STREAM CONNECTED 25382
unlX 3 [ ] STREAM CONNECTED 27880
unlX 3 [ ] STREAM CONNECTED 27351
unlX 2 [ ] DGRAM 22033
unlX 3 [ ] STREAM CONNECTED 30767 @/tmp/dbus-syGt6LJFW9
unlX 3 [ ] STREAM CONNECTED 29605
unlX 3 [ ] STREAM CONNECTED 27130
unlX 2 [ ] DGRAM 25476
unlX 3 [ ] STREAM CONNECTED 19423 /var/run/dbus/system_bus_socket
unlX 3 [ ] STREAM CONNECTED 32978
unlX 3 [ ] STREAM CONNECTED 29381
unlX 3 [ ] STREAM CONNECTED 34353
unlX 3 [ ] STREAM CONNECTED 30112 /run/systemd/journal/stdout
unlX 3 [ ] STREAM CONNECTED 29382 /run/systemd/journal/stdout
unlX 3 [ ] STREAM CONNECTED 27022 /run/systemd/journal/stdout
unlX 3 [ ] STREAM CONNECTED 18037 /var/run/dbus/system_bus_socket
unlX 3 [ ] STREAM CONNECTED 29776 @/tmp/dbus-syGt6LJFW9
unlX 3 [ ] STREAM CONNECTED 27866 @/tmp/dbus-syGt6LJFW9
unlX 3 [ ] DGRAM 17056
unlX 3 [ ] SEQPACKET CONNECTED 63556
unlX 3 [ ] STREAM CONNECTED 30734
unlX 3 [ ] STREAM CONNECTED 29785
unlX 3 [ ] STREAM CONNECTED 29159 @/tmp/dbus-syGt6LJFW9
unlX 3 [ ] STREAM CONNECTED 27005
unlX 3 [ ] STREAM CONNECTED 17677
unlX 3 [ ] STREAM CONNECTED 33137
unlX 3 [ ] STREAM CONNECTED 28112 @/tmp/dbus-SetFr4GY3I
unlX 3 [ ] STREAM CONNECTED 30012 @/tmp/.X11-unix/X1
unlX 3 [ ] STREAM CONNECTED 26693 @/tmp/.X11-unix/X1
unlX 2 [ ] DGRAM 881
unlX 3 [ ] STREAM CONNECTED 45152
unlX 3 [ ] STREAM CONNECTED 31428
unlX 3 [ ] STREAM CONNECTED 28557 @/tmp/dbus-SetFr4GY3I
unlX 3 [ ] STREAM CONNECTED 27871

```


7. **ss:** This command is the substitution for the netstat command. The ss command is more informative and much faster than netstat. The ss command's faster response is possible because it fetches every information from inside the kernel userspace.

Syntax:

Ss

```

root@ip-10-10-38-111: ~
File Edit View Search Terminal Help
68180      * 68181
u_str      0
ESTAB      0
63443      * 63444
u_str      0
ESTAB      0
44635      * 44635
u_str      0
ESTAB      0
29772      * 29378
u_str      0
ESTAB      0
62417      * 62416
u_str      0
ESTAB      0
24854      * 26289
u_str      0
ESTAB      0
62776      * 63530
u_str      0
ESTAB      0
29679      * 30095
u_str      0
ESTAB      0
29656      * 30068
u_str      0
ESTAB      0
19962      * 18852
u_str      0
ESTAB      0
27917      * 27916
u_str      64
ESTAB      0
26859      * 25446
u_str      0
ESTAB      0
28978      * 28369
u_str      0
ESTAB      0
28317      * 28906
u_str      0
ESTAB      0
63890      * 63006
u_str      0
ESTAB      0
32451      * 33491

```

8. **nslookup:** The nslookup command is an older edition of the dig command. Also, it is utilized for DNS related problems.

Syntax:

nslookup <domainname>

```

Applications Places System Fri 9 Aug, 07:16 AttackBox IP: 10.10.38.111
root@ip-10-10-38-111: ~
File Edit View Search Terminal Help
tcp SYN-SENT 0 1 10.10.38.111:
44124 tcp SYN-SENT 0 34.117.188.166:https 1 10.10.38.111:
59546 tcp ESTAB 0 34.120.208.123:https 0 127.0.0.1:
5901 tcp SYN-SENT 0 127.0.0.1:54532 1 10.10.38.111:
56484 tcp ESTAB 0 34.107.221.82:http 0 10.10.38.111:
http tcp ESTAB 0 10.100.2.28:52654 0 127.0.0.1:
54532
root@ip-10-10-38-111:~# nslookup
* www.google.com
Server: 127.0.0.53
Address: 127.0.0.53#53

Non-authoritative answer:
Name: www.google.com
Address: 209.85.202.104
Name: www.google.com
Address: 209.85.202.103
Name: www.google.com
Address: 209.85.202.106
Name: www.google.com
Address: 209.85.202.99
Name: www.google.com
Address: 209.85.202.147
Name: www.google.com
Address: 209.85.202.105
Name: www.google.com
Address: 2a00:1450:400b:c00::68
Name: www.google.com
Address: 2a00:1450:400b:c00::6a
Name: www.google.com
Address: 2a00:1450:400b:c00::63
Name: www.google.com
Address: 2a00:1450:400b:c00::67
>

```

9. **dig:** dig is short for Domain Information Groper. The dig command is an improvised edition of the nslookup command. It is utilized in DNS lookup to reserve the DNS name server. Also, it is used to balance DNS related problems. Mainly, it is used to authorize DNS mappings, host addresses, MX records, and every other DNS record for the best DNS topology understanding.

Syntax: dig

<domainname>

```

root@ip-10-10-38-111:~# dig www.google.com
; <<>> DiG 9.11.3-1ubuntu1.18-Ubuntu <<>> www.google.com
;; global options: +cmd
;; Got answer:
;;->HEADER<- opcode: QUERY, status: NOERROR, id: 41715
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;www.google.com.                IN      A
;; ANSWER SECTION:
www.google.com.                180     IN      A      209.85.202.105
www.google.com.                180     IN      A      209.85.202.147
www.google.com.                180     IN      A      209.85.202.99
www.google.com.                180     IN      A      209.85.202.106
www.google.com.                180     IN      A      209.85.202.103
www.google.com.                180     IN      A      209.85.202.104
;; Query time: 0 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Fri Aug 09 07:17:37 BST 2024
;; MSG SIZE rcvd: 139
root@ip-10-10-38-111:~#

```

10. **route:** The route command shows and employs the routing table available for our system. Basically, a router is used to detect a better way to transfer the packets around a destination.

Syntax: Route

11. **host:** The host command shows the IP address for a hostname and the domain name for an IP address. Also, it is used to get DNS lookup for DNS related issues.

Syntax:

host -t **<resourceName>**

12. **arp:** The arp command is short for Address Resolution Protocol. This command is used to see and include content in the ARP table of the kernel.

Arp

13. **iwconfig:** It is a simple command which is used to see and set the system's hostname.

Syntax:

Hostname

14. **curl and wget:** These commands are used to download files from CLI from the internet. curl must be specified with the "O" option to get the file, while wget is directly used.

curl Syntax:

curl -O **<fileLink>**

15. **wget**

Syntax:

wget **<fileLink>**

16. **mtr:** The mtr command is a mix of the traceroute and ping commands. It regularly shows information related to the packets transferred using the ping time of all hops. Also, it is used to see network problems.

Syntax:

mtr <path>

17. **whois:** The whois command fetches every website related information. We can get every information of a website, such as an owner and the registration information.

Syntax:

mtr <websiteName>

18. **ifplugstatus:** The ifplugstatus command checks whether a cable is currently plugged into a network interface. It is not available in Ubuntu directly. We can install it with the help of the below command:

sudo apt-get install ifplugd **Syntax:**

Ifplugstatus iftop: The iftop command is utilized in traffic monitoring. **tcpdump:** The tcpdump command is widely used in network analysis with other commands of the Linux network. It analyses the traffic passing from the network interface and shows it.

When balancing the network, this type of packet access will be crucial. **Syntax:**

\$ tcpdump -i <network_device>

RESULT:

Hence, Linux commands are executed successfully.