**You said:**

linux advance networking cmds please all commands that are used most by devops engineer - please include all the cmds first networking cmds with all the flags that are used in them - and please create examples from scaratch like first create file and then apply cmd - please all importnatn flags used in them - please do it -

**ChatGPT said:**

Here’s a comprehensive guide to the most commonly used Linux networking commands, complete with flags and examples to demonstrate their usage. These are essential for DevOps engineers working with networking tasks.

**1. ifconfig (Interface Configuration)**

The ifconfig command is used for configuring, managing, and querying network interfaces.

**Common flags:**

* -a – Display all interfaces, including those that are down.
* eth0 – Specifies the network interface (like eth0).
* up – Activate the interface.
* down – Deactivate the interface.

**Example:**

bash

CopyEdit

# Create a simple text file

echo "Testing networking commands" > /tmp/example.txt

# Display all network interfaces

ifconfig -a

# Bring the network interface up (replace eth0 with your interface name)

ifconfig eth0 up

# Bring the network interface down

ifconfig eth0 down

# Show the details of a specific interface (eth0)

ifconfig eth0

**2. ip (IP Configuration)**

The ip command is more modern and powerful compared to ifconfig. It is used to show/manipulate routing, network devices, interfaces, and tunnels.

**Common flags:**

* link – Shows or manipulates link layer attributes (MAC address).
* addr – Shows IP address info.
* route – Displays or manipulates IP routing.
* netns – Manage network namespaces.

**Example:**

bash

CopyEdit

# Create a file to store output

echo "Network IP Details" > /tmp/ip\_output.txt

# Show IP address of all interfaces

ip addr

# Show detailed routing table

ip route show

# Bring an interface up

ip link set eth0 up

# Add a new IP address to an interface

ip addr add 192.168.1.100/24 dev eth0

# Delete an IP address from an interface

ip addr del 192.168.1.100/24 dev eth0

**3. ping (Network Connectivity Check)**

ping is used to check the connectivity between the local machine and a remote host.

**Common flags:**

* -c – Specify the number of echo requests to send.
* -i – Specify the interval between sending each packet.

**Example:**

bash

CopyEdit

# Ping a remote host (Google DNS server)

ping -c 4 8.8.8.8

# Ping a domain name

ping -c 4 www.google.com

# Ping a host with a 1-second interval between packets

ping -i 1 8.8.8.8

**4. traceroute (Trace the Path to a Network Host)**

traceroute traces the route packets take to a remote host and can help identify network bottlenecks or issues.

**Common flags:**

* -m – Set the maximum number of hops.
* -w – Set the timeout in seconds for each probe.

**Example:**

bash

CopyEdit

# Tracing route to google.com

traceroute www.google.com

# Set max hops to 15

traceroute -m 15 www.google.com

# Set timeout to 1 second

traceroute -w 1 www.google.com

**5. netstat (Network Statistics)**

netstat provides network connections, routing tables, interface statistics, masquerade connections, and multicast memberships.

**Common flags:**

* -a – Show all connections and listening ports.
* -n – Show numerical addresses (skip DNS resolution).
* -t – Display TCP connections.
* -u – Display UDP connections.

**Example:**

bash

CopyEdit

# Display all connections and listening ports

netstat -an

# Show active TCP connections

netstat -at

# Show active UDP connections

netstat -au

# Show routing table

netstat -r

**6. ss (Socket Statistics)**

ss is a utility to investigate sockets and is used as a replacement for netstat on modern Linux distributions.

**Common flags:**

* -t – Show TCP sockets.
* -u – Show UDP sockets.
* -a – Show all sockets.
* -n – Show numerical addresses.

**Example:**

bash

CopyEdit

# Display all TCP sockets

ss -t

# Display all UDP sockets

ss -u

# Display all sockets (TCP/UDP)

ss -a

# Show detailed information with numerical addresses

ss -an

**7. nslookup (DNS Lookup)**

nslookup is used to query DNS servers for information about hostnames and IP addresses.

**Common flags:**

* -type – Specify the type of record to query (e.g., A, MX, CNAME).

**Example:**

bash

CopyEdit

# Perform a DNS lookup for google.com

nslookup google.com

# Query a specific DNS record (MX record for a domain)

nslookup -type=MX google.com

**8. curl (Transfer Data from or to a Server)**

curl is a powerful tool for transferring data over a network using various protocols (HTTP, FTP, etc.).

**Common flags:**

* -I – Fetch HTTP headers only.
* -X – Specify request method (e.g., GET, POST).
* -d – Send data in the request body (for POST).
* -u – Provide a username and password for authentication.

**Example:**

bash

CopyEdit

# Fetch headers of a webpage

curl -I www.google.com

# Perform a GET request (default method)

curl www.google.com

# Perform a POST request with data

curl -X POST -d "name=DevOps&role=Engineer" www.example.com

# Provide authentication

curl -u user:password www.example.com

**9. iptables (IP Firewall Management)**

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

**Common flags:**

* -L – List all rules.
* -A – Append a new rule to the end of a chain.
* -D – Delete a rule.
* -I – Insert a rule at the beginning of a chain.

**Example:**

bash

CopyEdit

# List all rules

iptables -L

# Allow incoming traffic on port 80 (HTTP)

iptables -A INPUT -p tcp --dport 80 -j ACCEPT

# Block incoming traffic on port 80

iptables -D INPUT -p tcp --dport 80 -j ACCEPT

# Save the iptables rules (on some systems)

service iptables save

**10. hostname (Set or Display System’s Hostname)**

The hostname command is used to display or set the system's hostname.

**Common flags:**

* -I – Display the IP address of the host.
* -f – Show the fully qualified domain name.

**Example:**

bash

CopyEdit

# Display the current hostname

hostname

# Display the IP address of the host

hostname -I

# Set the hostname

hostname new-hostname

**11. nmcli (NetworkManager Command-Line Interface)**

nmcli is used for managing network connections on systems with NetworkManager.

**Common flags:**

* dev – Show network device status.
* con – Manage network connections.
* nmcli – Basic command to interact with NetworkManager.

**Example:**

bash

CopyEdit

# Show all network devices

nmcli dev

# Show all network connections

nmcli con show

# Connect to a WiFi network

nmcli dev wifi connect "SSID" password "password"

# Disconnect from a network

nmcli con down "ConnectionName"

**Conclusion**

These are some of the most important Linux networking commands, commonly used by DevOps engineers. They are crucial for troubleshooting, configuring, and managing network connections on Linux systems. Make sure you understand the flags and use cases for each command to optimize your workflows and enhance your network management efficiency.

If you have any further questions or need examples for specific cases, feel free to ask!

**You said:**

please continue with more networking advanced cmds that are used by devops engineer - with examp and flags please most used

**ChatGPT said:**

Continuing from the previous list, here are more advanced networking commands that DevOps engineers frequently use, along with examples and flags. These commands are vital for network diagnostics, security, and system performance monitoring.

**12. tcpdump (Network Packet Analyzer)**

tcpdump is a command-line tool for network packet capturing and analyzing network traffic.

**Common flags:**

* -i – Specify the network interface.
* -w – Write captured packets to a file.
* -n – Show numerical addresses (skip DNS resolution).
* -v – Provide more detailed information.
* -c – Capture a specified number of packets.

**Example:**

bash

CopyEdit

# Capture packets on the eth0 interface (live)

tcpdump -i eth0

# Capture packets and save to a file

tcpdump -i eth0 -w /tmp/capture.pcap

# Capture only 10 packets and display more details

tcpdump -i eth0 -c 10 -v

# Capture only TCP traffic on port 80 (HTTP)

tcpdump -i eth0 tcp port 80

# Capture packets on a specific IP

tcpdump -i eth0 host 192.168.1.100

**13. nmap (Network Exploration and Security Auditing)**

nmap is a network scanning tool used for discovering hosts and services on a computer network.

**Common flags:**

* -sP – Ping scan to determine which hosts are up.
* -p – Specify ports to scan.
* -v – Increase verbosity.
* -A – Enable OS detection, version detection, script scanning, and traceroute.

**Example:**

bash

CopyEdit

# Scan a specific host (host discovery)

nmap -sP 192.168.1.1

# Scan a range of IP addresses

nmap 192.168.1.1-10

# Scan a specific port (e.g., port 80) on a host

nmap -p 80 192.168.1.1

# Perform an aggressive scan (includes OS detection, version detection)

nmap -A 192.168.1.1

# Scan a subnet for open ports

nmap 192.168.1.0/24

**14. route (Show or Manipulate IP Routing Table)**

The route command is used to view and modify the routing table on Linux systems.

**Common flags:**

* -n – Display numerical addresses (skip DNS resolution).
* add – Add a new route.
* del – Delete a route.
* -net – Specify the destination network.

**Example:**

bash

CopyEdit

# Show the routing table

route -n

# Add a new route

route add -net 192.168.1.0/24 gw 192.168.0.1

# Delete a route

route del -net 192.168.1.0/24

# Set the default gateway

route add default gw 192.168.0.1

**15. dig (DNS Lookup)**

dig is used to perform DNS queries and provide detailed information about DNS records.

**Common flags:**

* @server – Specify a DNS server.
* -t – Query a specific type of DNS record (e.g., A, MX).
* +short – Display a short answer.

**Example:**

bash

CopyEdit

# Get the A record (IPv4 address) for a domain

dig google.com A

# Query MX (Mail Exchange) records for a domain

dig google.com MX

# Use a specific DNS server to resolve a domain

dig @8.8.8.8 google.com

# Show a short answer without extra details

dig google.com +short

**16. ethtool (Show/Change Ethernet Device Settings)**

ethtool is used to query and control the network interface parameters, especially for Ethernet devices.

**Common flags:**

* -i – Display driver information for the interface.
* -s – Change the device settings (e.g., speed, duplex).
* -p – Blink the LED on the interface to locate it.

**Example:**

bash

CopyEdit

# Display driver information for eth0

ethtool -i eth0

# Change the speed and duplex settings of eth0

ethtool -s eth0 speed 1000 duplex full

# Blink the LED of the eth0 interface (useful for locating the physical interface)

ethtool -p eth0

**17. mtr (Network Diagnostic Tool)**

mtr is a combination of traceroute and ping, providing real-time network diagnostic information.

**Common flags:**

* -r – Report mode (provides a summary of results).
* -c – Set the number of cycles (pings).
* -i – Set the interval between pings.

**Example:**

bash

CopyEdit

# Run mtr in report mode to google.com (with 10 cycles)

mtr -r -c 10 google.com

# Run mtr with an interval of 1 second

mtr -i 1 google.com

**18. host (DNS Lookup Utility)**

The host command is used to perform DNS lookups and retrieve information about domain names.

**Common flags:**

* -t – Specify the type of DNS record to query (e.g., A, MX).
* -v – Display verbose output.

**Example:**

bash

CopyEdit

# Query the A record (IP address) for google.com

host google.com

# Query the MX (Mail Exchange) records for a domain

host -t MX google.com

# Display detailed output for a domain lookup

host -v google.com

**19. iwconfig (Wireless Interface Configuration)**

iwconfig is used to configure wireless network interfaces.

**Common flags:**

* essid – Set the SSID (network name).
* key – Set the encryption key for a wireless network.
* mode – Set the mode (e.g., Managed, Ad-Hoc).

**Example:**

bash

CopyEdit

# Show the current wireless interface configuration

iwconfig

# Connect to a Wi-Fi network with SSID "MyWiFi" and a WPA key

iwconfig wlan0 essid "MyWiFi" key s:password

# Set the wireless interface to Managed mode

iwconfig wlan0 mode Managed

**20. iptables-save (Save IP Tables)**

iptables-save is used to save the current iptables rules to a file for persistence across reboots.

**Example:**

bash

CopyEdit

# Save current iptables rules to a file

iptables-save > /etc/iptables/rules.v4

# Restore iptables rules from a file

iptables-restore < /etc/iptables/rules.v4

**21. arp (Address Resolution Protocol)**

arp is used to manipulate the system’s ARP cache and can be useful for resolving IP addresses to MAC addresses.

**Common flags:**

* -a – Display the current ARP table.
* -d – Delete an entry from the ARP table.
* -s – Add a static entry to the ARP table.

**Example:**

bash

CopyEdit

# Display the current ARP table

arp -a

# Add a static ARP entry

arp -s 192.168.1.100 00:11:22:33:44:55

# Delete an entry from the ARP table

arp -d 192.168.1.100

**22. nc (Netcat)**

nc is a versatile tool for network diagnostics and can be used as a port scanner, banner grabber, or simple chat server.

**Common flags:**

* -l – Listen for incoming connections.
* -v – Verbose mode.
* -z – Scan for open ports.
* -u – Use UDP instead of TCP.

**Example:**

bash

CopyEdit

# Listen on port 12345 for incoming connections

nc -l -p 12345

# Scan for open ports on a remote host

nc -zv 192.168.1.1 80-90

# Send data to a remote server (e.g., HTTP request)

echo "GET / HTTP/1.1" | nc -v www.example.com 80

**23. systemctl (System and Service Manager)**

systemctl is used for managing services and the system itself, which includes starting, stopping, and viewing the status of networking services.

**Common flags:**

* start – Start a service.
* stop – Stop a service.
* status – View the status of a service.
* restart – Restart a service.

**Example:**

bash

CopyEdit

# Start the networking service

systemctl start networking

# Stop the networking service

systemctl stop networking

# View the status of the network service

systemctl status networking

# Restart the network service

systemctl restart networking

**Conclusion**

These are more advanced Linux networking commands frequently used by DevOps engineers. Each of these tools provides valuable insight into network diagnostics, configurations, and security management. They are crucial for maintaining the health and performance of a networked system. Understanding and applying these commands will make you proficient in managing complex networks and troubleshooting issues in real-time.

Feel free to explore these commands further to master them for your DevOps workflows!