Linux Networking Commands Cheat Sheet — Practical Guide with Real-World Scenarios

LinkedIn: Amit Singh

Medium: Amit Singh – Medium

The only Linux Networking Cheat Sheet you need! This cheat sheet compiles the most important Linux networking commands, grouped by real-world scenarios and organized into step-by-step workflows. Each command is accompanied by practical usage examples to help you quickly diagnose, monitor, or configure your network environment with confidence.

1. Interface & IP Management

Command	Description
ip link	View and manage network interfaces
ip addr	Show assigned IPs (same as ip a)
ip route	Show or manage routing table
ip neigh	View ARP table
nmcli	NetworkManager CLI tool (used on modern distros)
ethtool eth0	View and configure interface hardware settings
hostname -I	Show IP addresses
macchanger	Change MAC address (manual install needed)

2. DNS & Name Resolution

Command	Description
dig <domain>	DNS lookup (modern replacement for nslookup)
nslookup <domain>	Legacy DNS lookup tool
host <domain>	Simple DNS query
resolvectl / systemd-resolve	DNS resolver info (systemd-based distros)

3. Port Scanning & Network Discovery

Command	Description
nmap <host>	Powerful network scanner
arp-scan <network>	ARP scan local subnet (manual install)
netdiscover	Find live hosts on network (ARP-based)

4. Packet Capture & Analysis

Command	Description
tcpdump	Capture network packets
wireshark	GUI for deep packet analysis
ngrep	Packet sniffing using regex
tshark	Terminal version of Wireshark

5. Connection Testing & Performance

Command	Description
telnet <host> <port>	Test TCP connection to a port
nc or netcat	Network Swiss-army knife (listen, connect, scan)
curl <url>	Test HTTP/HTTPS, download web content
wget <url>	Another tool to test HTTP/FTP downloads
iperf3 -c <host>	Network speed test (requires iperf3 server)

6. Firewall & Traffic Rules

Command	Description
iptables	Packet filtering and NAT rules
nft	Modern replacement for iptables
ufw	User-friendly firewall manager (Ubuntu/Debian)
firewalld / firewall-cmd	Firewall controller used on RHEL/Fedora

7. Network Services & Status

Command	Description
systemctl status NetworkManager	Check if NetworkManager is running
systemctl restart network	Restart network services (Red Hat/CentOS)
service networking restart	Debian-based service restart

8. Misc Tools & Utilities

Command	Description
whois <domain>	Get domain info and registrar details
route -n	Show kernel routing table
arp -a	Show ARP table
watch -n 1 netstat -tuln	Continuously monitor open ports

Summary

Command	Function
ifconfig / ip a	Show IP addresses & interfaces
ip -s link	Show network interface statistics
ss -tuln	Show open/listening ports
netstat -tuln	Legacy alternative for ss
ping	Check network connectivity
traceroute	Path tracing to destination
mtr	Live route + ping combo
iftop	Real-time per-connection bandwidth
nload	Graphical bandwidth usage tool

Tip: Install Advanced Tools

Some tools like iftop, nload, mtr, tcpdump, and nmap are **not pre-installed** on most systems. Install them using:

sudo apt install iftop nload mtr nmap tcpdump -y # Debian/Ubuntu

sudo yum install iftop nload mtr nmap tcpdump -y # CentOS/RHEL

Linux Networking Commands Cheat Sheet with Scenarios

1. Network Interface & IP Management

➜ ip a / ifconfig

Purpose: View network interfaces and their IP addresses.

Scenario: You want to find the IP address of your Linux server.

Command:

- ip a or ifconfig

➜ ip link

Purpose: Display or manage network interfaces.

Scenario: You're checking if a physical network adapter is up.

Command:

- ip link show

➜ ip -s link

- **Purpose**: View stats like dropped packets, errors, etc.
- **Scenario**: You suspect interface errors or packet loss.
- **Command**:
- bash
- CopyEdit
- ip -s link

➜ ethtool eth0

Purpose: Check NIC settings like speed, duplex.

Scenario: You suspect slow performance due to wrong NIC settings.

Command:

- sudo ethtool eth0

2. Connectivity Testing

→ ping <host>

Purpose: Test basic network connectivity.

Scenario: You're checking if your server can reach Google.

Command:

- ping 8.8.8.8

→ traceroute <host>

Purpose: Shows the path packets take to reach a host.

Scenario: A website is slow to reach, and you want to see where.

Command:

- traceroute example.com

→ mtr <host>

Purpose: Combines traceroute and ping for live path analysis.

Scenario: Continuous monitoring of route latency/loss.

Command:

- mtr google.com

→ curl or wget

Purpose: Test HTTP/HTTPS reachability or download files.

Scenario: You want to test if your server can reach a REST API.

Command:

- curl -I https://example.com

3. Port & Service Monitoring

→ ss -tuln

Purpose: Show listening ports and associated services (TCP/UDP).

Scenario: You want to know what services are running on your server.

Command:

- ss -tuln

→ netstat -tuln

Purpose: Older alternative to ss, still widely used.

Scenario: Same as above; useful on older systems.

Command:

- netstat -tuln

→ lsof -i :<port>

Purpose: List processes using a specific port.

Scenario: Find which process is using port 8080.

Command:

- sudo lsof -i :8080

→ nmap <host>

Purpose: Scan for open ports/services on a host.

Scenario: You want to audit a server for exposed ports.

Command:

- nmap 192.168.1.10

→ nc -zv <host> <port>

Purpose: Check if a TCP port is open.

Scenario: You're testing if port 22 (SSH) is accessible.

Command:

- nc -zv 192.168.1.10 22

4. Bandwidth & Traffic Monitoring

→ iftop

Purpose: Show real-time bandwidth usage per connection.

Scenario: You want to see which remote IP is using the most data.

Command:

- sudo iftop

→ nload

Purpose: Live graph of upload/download usage.

Scenario: You want a visual of real-time traffic on eth0.

Command:

- sudo nload eth0

→ bmon

Purpose: Visual bandwidth monitor with interface stats.

Scenario: Monitor multiple interfaces with bar graphs.

Command:

- sudo bmon

→ vnstat

Purpose: Track historical bandwidth usage.

Scenario: You're auditing data transfer from the past week.

Command:

- vnstat -w

→ iperf3 -c <server>

Purpose: Measure network speed between two machines.

Scenario: You're testing performance between two endpoints.

Command:

- iperf3 -c 192.168.1.20

5. DNS & Name Resolution

→ dig <domain>

- **Purpose**: Get detailed DNS info (A, MX, NS, etc.).
- **Scenario**: You're troubleshooting DNS resolution for a domain.
- **Command**:
- bash
- CopyEdit
- dig example.com

→ nslookup <domain>

Purpose: Legacy DNS lookup tool.

Scenario: Simple test to resolve a domain name.

Command:

- nslookup example.com

→ host <domain>

Purpose: Lightweight DNS lookup tool.

Scenario: Quickly get the IP of a domain.

Command:

- host example.com

6. Packet Capture & Deep Analysis

→ tcpdump

Purpose: Capture and inspect network packets.

Scenario: You're debugging why HTTP requests aren't reaching your app.

Command:

- sudo tcpdump -i eth0 port 80

→ wireshark

- **Purpose**: GUI tool for full packet inspection.
- **Scenario**: You want to visually analyze packets at a protocol level.
- **Note**: Use on local systems or over SSH X11 forwarding.

→ ngrep

Purpose: Grep network traffic using regex.

Scenario: Monitor for specific strings (e.g., login attempts).

Command:

- sudo ngrep -d eth0 "password"

7. Firewall & Security

→ iptables -L

Purpose: List all current firewall rules.

Scenario: You're debugging why port 80 is blocked.

Command:

- sudo iptables -L -n -v

→ nft list ruleset

Purpose: View rules with nftables (modern firewall).

Scenario: Your system uses nft instead of iptables.

Command:

- sudo nft list ruleset

→ ufw status

Purpose: Check firewall status on systems using UFW (Ubuntu).

Scenario: See if port 22 is allowed.

Command:

- sudo ufw status

Interrelated Linux Networking Commands with Scenarios

Scenario 1: Find Out What's Wrong With the Network Connection

Step-by-Step Flow:

Check if your network interface is up

➤ ip a / ifconfig

Shows IP addresses, interface status.

⬅ If no IP? Then check: ip link

View interface-level stats (errors, dropped packets)
➤ ip -s link
Helps identify NIC or cable issues.

Try basic connectivity to gateway or internet
➤ ping 8.8.8.8 or ping <default-gateway>
Tests if IP routing and gateway work.

Suspect DNS? Test domain resolution
➤ dig example.com or nslookup example.com
Determines if DNS resolution is failing.

Still can't reach domains? Check full path
➤ mtr example.com or traceroute example.com
Visualizes route to target and shows where it's failing.

Scenario 2: See Which Ports/Services Are Active or Listening

Step-by-Step Flow:

Check active/open ports
➤ ss -tuln or netstat -tuln
Lists services listening for connections.

Want to know which process is bound to a port?
➤ lsof -i :22 or ss -tulpn
Shows PID + process name.

Verify service accessibility from outside
➤ nc -zv <IP> <port> or telnet <IP> <port>
Confirms if a service is reachable remotely.

Not sure what ports are exposed? Scan it!
➤ nmap <host>
Finds open ports, detects services, version info.

Scenario 3: Monitor Bandwidth and Network Usage

Step-by-Step Flow:

Real-time bandwidth per connection
➤ iftop
Great for seeing which IPs are communicating heavily.

Graphical interface stats
➤ nload, bmon
Simple RX/TX line graphs, helpful for visual monitoring.

Historical traffic logs
➤ vnstat
View usage per day, week, month — useful for audits.

Test bandwidth between two systems
➤ iperf3 -s (server) and iperf3 -c <server> (client)
Benchmark your LAN or WAN link speed.

Scenario 4: Analyze or Capture Traffic (Advanced Debugging)

Step-by-Step Flow:

Quick packet capture on interface
➤ tcpdump -i eth0 port 80
See real-time HTTP traffic on eth0.

Need deep packet inspection?
➤ wireshark (GUI) or tshark (CLI)
Decode protocols like TLS, SIP, etc.

Filter network traffic for strings (e.g., passwords)
➤ ngrep -d eth0 "login"
Pattern-match packet contents.

Scenario 5: Investigate Firewall or Routing Issues

Step-by-Step Flow:

Check if Linux firewall is blocking traffic
➤ iptables -L -n -v or ufw status
View firewall rules in action.

Modern systems? Use nftables
➤ nft list ruleset
Replaces iptables on newer distros.

Check routing table
➤ ip route or route -n
Ensure there's a correct route to the destination.

See ARP (MAC-IP) mappings
➤ ip neigh or arp -a
Useful for diagnosing LAN issues.

Scenario 6: Diagnose and Manage Local DNS & Host Configuration

Step-by-Step Flow:

Check current DNS servers
➤ cat /etc/resolv.conf or resolvectl status
Shows active name servers.

Force DNS query with specific server
➤ dig @8.8.8.8 example.com
Bypass local settings and test with Google DNS.

Update hostnames or temporary name resolution
➤ Edit /etc/hosts
Manual override for DNS.

🔄 All-In-One Relationship Summary

Goal	Commands Involved	Flow
Check network status	ip a → ip link → ip -s link	Interface → Link up? → Errors?
Test connectivity	ping → traceroute/mtr	ICMP ping → Route → Live path
DNS resolution	dig → nslookup → host	DNS test with multiple tools
Find open ports	ss → netstat → nmap → nc	Local listener → External reachability
Analyze bandwidth	iftop → nload → vnstat → iperf3	Real-time → Visual → Logs → Benchmark
Capture packets	tcpdump → ngrep → wireshark	CLI capture → Pattern match → GUI analysis
Investigate firewall	iptables/nft → ufw → ip route	Rule check → Routing table
Discover services/devices	nmap → arp-scan → netdiscover	Find live hosts, open services

Scenario-Based Structure for Troubleshooting & Monitoring

Scenario 1: Diagnose Network Connectivity Issues

```bash
ip a                    # Check if interface has IP
ip link                 # Check if interface is up
ip -s link              # Check for dropped/error packets

ping 8.8.8.8            # Test internet connectivity
dig example.com         # Check DNS resolution
mtr example.com         # Continuous traceroute with packet loss info
```

Scenario 2: Check Listening Ports and Services

Commands Used:

- ss -tuln, netstat -tuln, lsof, nc, nmap

```
ss -tuln                # View listening TCP/UDP ports
lsof -i :80             # See which process is using port 80
nc -zv localhost 22     # Check if SSH port is accessible
nmap 192.168.1.10       # Scan ports on a remote system
```

Scenario 3: Monitor Bandwidth and Network Usage

Commands Used:

- iftop, nload, bmon, vnstat, iperf3

```
sudo iftop              # Real-time connection-level usage
sudo nload              # Live traffic graphs (per interface)
vnstat -w               # View weekly historical bandwidth usage
iperf3 -s               # Run as server
iperf3 -c <server-ip>   # Run client test to server
```

Scenario 4: Capture and Analyze Network Packets

Commands Used:

- tcpdump, ngrep, wireshark

```
sudo tcpdump -i eth0 port 80      # Capture HTTP packets
sudo ngrep -d eth0 "password"     # Monitor sensitive data in transit
# Use Wireshark GUI for deep protocol analysis
```

Scenario 5: Investigate Firewall or Routing Problems

Commands Used:

- iptables, nft, ufw, ip route, arp, ip neigh

```
sudo iptables -L -n -v            # View firewall rules
sudo nft list ruleset             # View nftables firewall (modern)
sudo ufw status                   # View UFW firewall status (Ubuntu)
ip route                          # View system routing table
ip neigh                          # View ARP table
```

Scenario 6: DNS and Name Resolution

Commands Used:

- dig, host, nslookup, resolvectl, /etc/hosts

```
dig example.com                   # DNS record query
host example.com                  # Resolve domain to IP
cat /etc/resolv.conf              # View configured DNS servers
sudo nano /etc/hosts              # Add manual DNS entries
```

Command Relationship Table

Goal	Commands Involved
Interface diagnostics	ip a, ip link, ip -s link
Basic connectivity	ping, traceroute, mtr
DNS resolution	dig, host, nslookup, resolvectl
Service monitoring	ss, netstat, lsof, nmap, nc
Bandwidth monitoring	iftop, nload, vnstat, bmon, iperf3
Packet analysis	tcpdump, wireshark, ngrep
Firewall debugging	iptables, nft, ufw, ip route, ip neigh

Thanks Everyone!

Connect with me: Amit Singh