Table of Contents

$ tcpdump

Advanced

Capture and analyze network traffic

tcpdump captures and displays network packets in real time. It is the most widely used command-line packet analyzer, essential for network troubleshooting, protocol analysis, and security monitoring.

tcpdump captures raw packets from network interfaces and can filter by protocol, port, host, and...

Options & Flags

`-i`	Capture on specific interface
`-n`	Do not resolve hostnames
`-c`	Capture N packets then stop
`-w`	Write to pcap file
`-r`	Read from pcap file
`-A`	Print packet content as ASCII
`port`	Filter by port
`host`	Filter by host

Practical Examples

Example: Capture HTTP traffic

```
$ sudo tcpdump -n port 80 -c 50
```

Captures 50 packets on port 80 without DNS resolution.

Example: Capture from specific host

```
$ sudo tcpdump -n host 192.168.1.100
```

Shows all traffic to/from a specific IP.

Example: Save to file

```
$ sudo tcpdump -w /tmp/capture.pcap -c 1000
```

Captures 1000 packets and saves for analysis in Wireshark.

Example: DNS queries

```
$ sudo tcpdump -n port 53
```

Captures DNS query and response packets.

Example: HTTP content

```
$ sudo tcpdump -A -n port 80 | grep -i "GET\|POST\|Host"
```

Shows HTTP request methods and hosts in ASCII.

Tips & Best Practices

Pro Tip: Capture and analyze later: sudo tcpdump -w file.pcap captures packets. Open file.pcap in Wireshark for detailed graphical analysis.

Warning: Can capture sensitive data: tcpdump can capture passwords, tokens, and other sensitive data in unencrypted traffic. Handle captures securely.

Note: BPF filter syntax: Combine filters: tcpdump 'host 10.0.0.1 and port 443'. Use and, or, not for complex expressions.

$ traceroute

Intermediate

Trace the route packets take to a network host

traceroute traces the network path from your computer to a destination, showing each router (hop) along the way. It reveals the network route, identifies where delays occur, and helps diagnose routing problems.

traceroute works by sending packets with incrementally increasing TTL (Time To Live) ...

Options & Flags

`-n`	Show IP addresses only (no DNS lookup)
`-m`	Maximum hops
`-w`	Wait time per probe
`-I`	Use ICMP instead of UDP
`-T`	Use TCP SYN
`-q`	Number of probes per hop

Practical Examples

Example: Trace route to host

```
$ traceroute example.com
1  gateway (192.168.1.1)  1.234 ms\n2  isp-router (10.0.0.1)  5.678 ms\n3  backbone (172.16.0.1)  15.234 ms
```

Shows every router hop between you and the destination.

Example: Numeric only

```
$ traceroute -n 8.8.8.8
```

Shows IP addresses without DNS resolution (faster).

Example: TCP traceroute

```
$ sudo traceroute -T -p 443 example.com
```

Uses TCP to trace - works when UDP is blocked by firewalls.

Example: Quick trace

```
$ traceroute -q 1 -n example.com
```

One probe per hop, no DNS - fastest traceroute.

Example: ICMP traceroute

```
$ sudo traceroute -I example.com
```

Uses ICMP echo (like ping) - same protocol as Windows tracert.

Tips & Best Practices

Pro Tip: Use mtr instead: mtr combines traceroute and ping, providing continuous monitoring. It is more useful for diagnosing intermittent issues.

Note: * * * means filtered: Asterisks mean the hop did not respond. Many routers block traceroute probes - this does not necessarily mean a problem.

Warning: UDP vs ICMP vs TCP: Default UDP may be blocked. Try -I (ICMP) or -T (TCP) if you see all asterisks. TCP on port 443 works through most firewalls.

$ wg

Intermediate

WireGuard VPN management tool

The wg command is the configuration utility for WireGuard, the modern high-performance VPN protocol built into the Linux kernel since version 5.6. It allows you to create and manage WireGuard interfaces, generate cryptographic keys, add and remove peers, and monitor tunnel status.

WireGuard is f...

Options & Flags

`show`	Show current WireGuard interface configuration and status
`show wg0`	Show status of specific interface
`showconf wg0`	Show running configuration in config file format
`genkey`	Generate a new private key
`pubkey`	Derive public key from private key
`genpsk`	Generate a preshared key for quantum resistance
`set wg0 peer KEY allowed-ips IPs`	Add or update a peer on a running interface
`set wg0 peer KEY remove`	Remove a peer from a running interface
`show wg0 transfer`	Show data transfer statistics per peer
`show wg0 latest-handshakes`	Show timestamp of last handshake per peer

Practical Examples

Example: Generate a key pair

```
$ wg genkey | tee /etc/wireguard/private.key | wg pubkey > /etc/wireguard/public.key && chmod 600 /etc/wireguard/private.key
```

Generate private and public keys in one command. Sets proper permissions on the private key.

Example: Show interface status

```
$ sudo wg show wg0
interface: wg0\n  public key: abc123...\n  private key: (hidden)\n  listening port: 51820\n\npeer: def456...\n  endpoint: 203.0.11
```

Display full status including peers, endpoints, allowed IPs, latest handshake, and transfer stats.

Example: Add a peer dynamically

```
$ sudo wg set wg0 peer "PEER_PUBLIC_KEY" allowed-ips 10.0.0.10/32 endpoint "vpn.example.com:51820"
```

Add a new peer to a running WireGuard interface without restarting the tunnel.

Example: Remove a peer

```
$ sudo wg set wg0 peer "PEER_PUBLIC_KEY" remove
```

Remove a peer from the running interface. Does not modify the config file.

Example: Start WireGuard with wg-quick

```
$ sudo wg-quick up wg0
```

Bring up wg0 interface using /etc/wireguard/wg0.conf. Configures routing, DNS, and firewall rules automatically.

Tips & Best Practices

Warning: Protect private keys: Private keys should be chmod 600, owned by root. Never share private keys. Only exchange public keys between peers.

Pro Tip: Use wg-quick for daily operation: Use wg-quick up/down for starting/stopping tunnels. Use raw wg commands for dynamic peer management and monitoring.

Note: Handshake timeout: WireGuard performs a new handshake every 2 minutes. If latest handshake is older than 5 minutes, the peer is likely unreachable.

Pro Tip: PersistentKeepalive for NAT: If a peer is behind NAT, set PersistentKeepalive = 25 to keep the NAT mapping alive with 25-second keepalive packets.

$ wget

Beginner

Download files from the web (HTTP, HTTPS, FTP)

wget is a non-interactive file downloader that supports HTTP, HTTPS, and FTP protocols. It is designed for reliable downloads even on unstable connections, with automatic retry and resume capabilities.

wget excels at downloading files, mirroring websites, and recursive downloads. Unlike curl whi...

Options & Flags

`-O`	Save to specific filename
`-c`	Resume interrupted download
`-r`	Recursive download
`-q`	Quiet mode
`-b`	Download in background
`--mirror`	Mirror a website (recursive + timestamps)
`-P`	Download to specific directory
`--limit-rate`	Limit download speed
`-i`	Download URLs listed in a file

Practical Examples

Example: Download a file

```
$ wget https://example.com/archive.tar.gz
```

Downloads the file and saves it with its original name.

Example: Resume interrupted download

```
$ wget -c https://example.com/large-file.iso
```

Continues a previously interrupted download from where it stopped.

Example: Download with custom filename

```
$ wget -O backup.sql.gz https://db.example.com/dump/latest
```

Saves the download with a specified filename.

Example: Mirror a website

```
$ wget --mirror --convert-links --page-requisites https://docs.example.com
```

Creates an offline copy of the website with working links.

Example: Download multiple files

```
$ wget -i download-list.txt
```

Downloads all URLs listed in the file, one per line.

Tips & Best Practices

Pro Tip: Resume large downloads: Always use wget -c for large files. If the connection drops, re-running the same command resumes from where it stopped.

Note: wget vs curl: wget is better for file downloads, recursive crawling, and background operations. curl is better for API testing, custom headers, and supporting many protocols.

Warning: Recursive download caution: wget -r without -l (depth limit) can download entire websites and consume massive disk space. Always set a depth limit.

$ whois

Beginner

Query domain and IP registration information

The whois command queries WHOIS databases to retrieve registration information about domain names and IP addresses. It shows who owns a domain, when it was registered and expires, the registrar, name servers, and contact information (when available).

For IP addresses, whois returns the organizat...

Options & Flags

`DOMAIN`	Query domain registration info
`IP`	Query IP address ownership
`-h SERVER`	Query a specific WHOIS server
`-p PORT`	Connect to WHOIS server on specified port
`AS NUMBER`	Query an Autonomous System number

Practical Examples

Example: Check domain registration

```
$ whois example.com | grep -E "Registrar|Creation|Expiry|Name Server"
```

Get key domain info: registrar, creation date, expiry date, and name servers.

Example: Check IP ownership

```
$ whois 8.8.8.8 | grep -E "OrgName|NetRange|CIDR|Country"
OrgName:  Google LLC\nNetRange: 8.8.8.0 - 8.8.8.255\nCIDR:     8.8.8.0/24\nCountry:  US
```

Find who owns an IP address - useful for identifying scanners, attackers, or hosting providers.

Example: Check domain availability

```
$ whois newdomain.com | grep -i "no match\|not found\|available"
```

Check if a domain is registered. "No match" or "Not found" typically means it is available.

Example: Find domain expiry date

```
$ whois example.com | grep -i expir
Registry Expiry Date: 2025-08-13T04:00:00Z
```

Find when a domain expires - important for renewal planning and competitor monitoring.

Example: Look up ASN

```
$ whois AS15169
```

Query Autonomous System information - shows the organization and their IP allocations.

Tips & Best Practices

Note: GDPR and privacy: Many domains now show redacted WHOIS info due to GDPR. Registrant details may be hidden behind privacy services. IP WHOIS is not affected.

Pro Tip: Use specific WHOIS servers: For more reliable results, query the registry directly: whois -h whois.verisign-grs.com example.com (.com/.net), whois -h whois.ripe.net IP (European IPs).

Note: Install whois: Install with: apt install whois (Debian/Ubuntu), dnf install whois (Fedora/RHEL), brew install whois (macOS).

Pro Tip: Rate limiting: WHOIS servers rate-limit queries. Avoid scripting bulk lookups without delays, or your IP may be temporarily blocked.