LINUX + NETWORKING · FREE COMMAND CHEAT SHEET

Find the fault. Follow the packets.

A practical guide for when an application cannot connect, a name will not resolve, or a service is not listening. Start with the question you need to answer, then run the smallest useful check.

SOFTWARE ENGINEER EDITION

Trace one request, layer by layer

A success at one layer only verifies that layer. Follow the request path; stop when a check fails and investigate that boundary.

01	02	03	04	05
NAME	**ROUTE**	**PORT**	**HTTP**	**APP**
Does DNS return an address?	**Can packets leave this host?**	**Is a TCP listener reachable?**	**What does the server return?**	**Does the service behave?**
`dig`	`ip route`	`ss / nc`	`curl`	`logs`

A QUICK ORDER

`dig` → `ip route` → `ss / nc` → `curl` → `application logs`

Use the order as a map, not a ritual. If DNS is healthy but the TCP connection is refused, move to the listener and service. If HTTP responds with an error, focus on the request, gateway, and application.

HOW TO USE THIS GUIDE

1. Copy one command and substitute your own host, service, or path. 2. Read the explanation before interpreting the output. 3. Compare a working and failing environment. 4. Keep timestamps and redact secrets before sharing logs.

Commands are examples for authorized troubleshooting. Options and output can vary by Linux distribution.

Linux + networking command deck

These checks help answer specific questions. Use the command that matches the symptom; you rarely need to run the whole list.

HOST · files and processes

`pwd`	Print the current working directory.
`ls -lah`	List visible and hidden files with readable sizes.
`find . -name '*.log' -type f`	Find matching log files under the current directory.
`ps aux`	List processes and the user that runs each one.
`top`	Watch CPU and memory use update live; press q to quit.

HOST · capacity and service

`df -h`	Check free space on mounted filesystems.
`free -h`	Review memory totals and available memory.
`systemctl status <service>`	Check a systemd service state and recent status.
`journalctl -u <service> -n 100`	Read the latest 100 service log lines.
`tail -n 50 -f app.log`	Follow new log lines; stop with Ctrl+C.

NETWORK · interface, route, DNS

`ip addr`	List interfaces and assigned addresses.
`ip route`	Show the route table and default gateway.
`ip route get 1.1.1.1`	Ask which route the host would use for a destination.
`dig example.com`	Query DNS and inspect the answer and resolver.
`getent hosts example.com`	Check lookup through the host's configured name service.

NETWORK · sockets and HTTP

`ss -lntp`	List listening TCP sockets; process detail may need sudo.
`ss -s`	Summarize socket counts and connection states.
`nc -vz host.example 443`	Check whether a TCP connection can reach a host and port.
`curl -I https://example.com`	Inspect response headers without downloading the body.
`curl -v --max-time 5 URL`	Show connection details with a bounded timeout.

ON macOS

If you are on a Mac, use ifconfig for interface details, route -n get default for the default route, and lsof -nP -iTCP -sTCP:LISTEN to see listening TCP ports. dig and curl are also available in the standard Terminal.

Privacy tip: logs and command output can contain tokens, email addresses, internal hostnames, and user data. Redact before sharing.

Read the symptom. Pick the next check.

Start with the exact error and timestamp. The same message can have different causes, so use each command to narrow the boundary.

NAME ERROR

Could not resolve host

The client could not turn the hostname into an address.

CHECK NEXT

dig <host>; getent hosts <host>. Compare the answer with the expected record and resolver.

CONNECTION REFUSED

Connection refused

The host replied, but that port is not accepting a connection.

CHECK NEXT

Check ss -lntp, service state, bind address, and host firewall.

TIMEOUT

Operation timed out

No response arrived within the allowed time.

CHECK NEXT

Check ip route, firewall/security groups, network ACLs, destination health, and reachability from this network.

HTTP ERROR

401 / 403 / 404 / 5xx

DNS and TCP probably worked; an HTTP layer returned an error.

CHECK NEXT

Use curl -i. Check path, auth, gateway/proxy response, and application logs.

LOCAL ONLY

Works on my machine

The environments differ somewhere in the path or configuration.

CHECK NEXT

Compare DNS, proxy, environment variables, container port mapping, and service bind address.

INTERMITTENT

It works sometimes

Some requests succeed and others fail or slow down.

CHECK NEXT

Repeat a bounded check with timestamps. Inspect DNS rotation, load balancer health, pools, loss, and deploys.

A 60-SECOND TRIAGE

1 Capture the exact error

2 Check DNS + route

3 Test port + HTTP

4 Check service logs

TROUBLESHOOT SAFELY

Prefer read-only checks first. Avoid unfamiliar scripts with sudo. Do not paste secrets into shell history or shared logs. Probe only systems you own or are authorized to test. Some commands require elevated privileges for complete process details.