Linux Troubleshooting & Networking Commands for DevOps Interview

---

Section 1: Basic Linux Commands for Troubleshooting (with Explanation)

1. `top` / `htop` – Displays real-time information about CPU, memory, processes, and load average. `htop` is more user-friendly and interactive.
2. `ps aux` – Lists all processes currently running, including the user, CPU/memory usage, and command used.
3. `df -h` – Shows disk space usage across all mounted filesystems in a human-readable format.
4. `du -sh *` – Summarizes disk usage of each file/folder in the current directory.
5. `free -m` – Provides memory usage in MBs, showing total, used, free, and available memory.
6. `uptime` – Shows how long the system has been running and system load averages.
7. `dmesg` – Prints kernel-related messages, useful for diagnosing hardware or boot issues.
8. `journalctl -xe` – Shows recent system logs, useful for checking service errors and systemd failures.
9. `tail -f /var/log/messages` or `/var/log/syslog` – Live monitoring of log files for debugging runtime issues.
10. `kill -9 <PID>` – Sends SIGKILL to forcefully terminate a process.
11. `netstat -tuln` – Lists all open TCP/UDP ports with no DNS resolution for speed.
12. `lsof -i :<port>` – Lists the process using a specific port.
13. `ping <host>` – Tests connectivity and round-trip time to another host.
14. `traceroute <host>` – Shows each hop in the network path to a destination.
15. `telnet <host> <port>` / `nc -zv <host> <port>` – Checks connectivity to a remote host on a specific port.
16. `ss -tuln` – Displays socket statistics, faster and more modern than `netstat`.
17. `ip a` / `ifconfig` – Lists network interfaces and assigned IPs.
18. `ip route` / `route -n` – Shows routing tables.
19. `iptables -L -n -v` – Displays current firewall rules and packet counters.
20. `systemctl status <service>` – Shows current status, logs, and recent failure info for systemd services.

Section 2: Advanced Linux/Networking Commands (with Explanation)

1. `strace -p <PID>` – Traces system calls and signals of a process, useful to debug hang or crash.
2. `tcpdump -i eth0 port 80` – Captures network traffic on interface `eth0` for HTTP traffic.
3. `nmap <host>` – Port scanning tool to identify open ports and services on a host.
4. `ip netns` / `ip link` – Manage network namespaces and virtual interfaces.
5. `ss -plnt` – Shows listening ports and the processes using them.
6. `systemctl list-units --failed` – Lists all failed systemd services and units.
7. `journalctl -u <service>` – Fetch logs specific to a particular service.
8. `vmstat` / `iostat` / `sar` – Collects and reports CPU, I/O, and memory statistics.
9. `iotop` – Displays real-time disk I/O usage by processes.
10. `tshark` – CLI-based packet capture tool (like Wireshark).

11. `hostnamectl` – Displays or sets hostname and related settings.
12. `ethtool <interface>` – Displays settings like speed, duplex, and allows hardware offload tuning.
13. `nmcli` – Command-line interface for NetworkManager to manage network connections.
14. `firewalld-cmd --list-all` – View runtime firewall rules managed by firewalld.
15. `sshd -T` – Validates the SSH daemon configuration.

Section 3: Linux Troubleshooting Interview Questions (Basic to Advanced)

1. How do you check CPU and memory usage in Linux?
2. How would you troubleshoot a server that is not responding?
3. How do you find which process is using a specific port?
4. What does the `top` command show, and how can you identify memory-hogging processes?
5. How can you monitor a log file in real time?
6. A service is not starting – how would you debug it?
7. How can you check if a port is open on a remote host?
8. What is the difference between `ss` and `netstat`?
9. How do you trace the execution of a process?
10. How can you identify if disk I/O is a bottleneck?
11. How do you check firewall rules on a Linux machine?
12. What is `journalctl` and how is it different from `/var/log/messages`?
13. How do you debug a DNS issue?
14. How do you troubleshoot a network interface that's down?
15. How can you list all failed services on a system?

Section 4: Related Topics & Concepts (with Explanations)

Systemd vs SysVinit

- **SysVinit**: Legacy init system using `/etc/init.d` scripts.
- **Systemd**: Modern init system using unit files. Faster boot, dependency management, logging via `journald`.
- Systemd commands: `systemctl`, `journalctl`, `loginctl`, `timedatectl`, etc.

Inodes and Disk Space

- **Inode**: Metadata for files – permissions, ownership, timestamp.
- Running out of inodes = can't create new files, even if space is available.
- Check with `df -i` and `stat <file>`.

SELinux/AppArmor Basics

- **SELinux** (RedHat/CentOS): Mandatory Access Control (MAC) system.
- **AppArmor** (Ubuntu): Profile-based access control.
- Modes: Enforcing, Permissive, Disabled.
- Tools: `getenforce`, `sestatus`, `audit2why`, `audit2allow`.

Cron Troubleshooting

- Check cron logs: `/var/log/cron`, `/var/log/syslog`.
- Ensure correct PATH is set in scripts.
- Use absolute paths in cron jobs.
- Test script manually outside of cron first.

SSH Key Authentication vs Password-Based Login

- **Password-based**: Less secure, vulnerable to brute force.
- **Key-based**: Uses public/private key pair. More secure.
- Public key added to `~/.ssh/authorized_keys`, client uses private key.

Ulimit and Open Files

- `ulimit` sets limits on user processes/resources (open files, memory, etc).
- `ulimit -n`: max number of open files.
- Increase via `/etc/security/limits.conf` or `/etc/systemd/system.conf`.

Load Average Understanding

- Displayed in `uptime`, `top`, etc.
- Format: `1min avg`, `5min avg`, `15min avg`.
- A load average of 1.0 means one process per core is queued/running.
- High load with low CPU usage may indicate I/O bottlenecks.

DNS Resolution Flow

1. Application calls `gethostbyname()`.
2. System checks `/etc/hosts`.
3. Then queries DNS servers in `/etc/resolv.conf`.
4. Resolver library sends query to the configured DNS.
5. DNS server replies with an IP or NXDOMAIN.

End of Document

end