Linux troubleshooting

1. How do you troubleshoot a slow Linux system?

When a Linux system becomes slow, the issue could be related to CPU, memory, disk, or network bottlenecks. Here's a structured approach to troubleshooting and resolving performance issues.

1. Check System Load and Running Processes

Use **top** or **htop** to Identify High CPU/Memory Usage

- Look for processes consuming high **CPU** (e.g., 100% usage).
- Check **Memory Usage** to see if RAM is fully utilized.
- Identify any **Zombie Processes** (Z state).

✅ **Solution:**

- Kill unnecessary high-CPU processes using: kill -9 <PID>
- Restart the affected service: systemctl restart <service_name>

---

2. Check Disk Usage and I/O Bottlenecks

Check Free Disk Space (df): df -h

✅ **Solution:**

- Delete unnecessary files
- Identify large files: du -ah / | sort -rh | head -10

Check Disk I/O Performance (iostat, iotop)

iostat -x 1 5 # Check disk performance

iotop # Show real-time disk usage by processes

- If disk I/O is high, check for **heavy read/write processes**.

✅ **Solution:**

- Limit disk-heavy processes using ionice: ionice -c 3 -p <PID>
- Consider adding **more storage** if disk space is low.

---

3. Check Memory and Swap Usage

Check Memory Usage (**free -m**)

- If **Swap usage is high**, it indicates RAM shortage.

Check Processes Using Most Memory (ps): ps aux --sort=-%mem | head -10

✅ **Solution:**

Kill heavy processes: *kill -9 <PID>*

Increase swap space (if RAM is low):

sudo fallocate -l 2G /swapfile

sudo chmod 600 /swapfile

sudo mkswap /swapfile

sudo swapon /swapfile

---

4. Check CPU Bottlenecks

Check CPU Usage (**mpstat**)

mpstat -P ALL 1

- If **CPU usage is 100%**, find the high-usage processes using:

ps -eo pid,ppid,cmd,%cpu --sort=-%cpu | head -10

✅ **Solution:**

- Reduce CPU load by lowering priority:

renice 10 -p <PID>

- If running a high-load application, consider **upgrading CPU resources**.

---

5. Check Network Performance Issues

Identify High Network Usage (**iftop, netstat**)

iftop -i eth0 # Shows real-time network traffic

netstat -tulnp # Lists active network connections

- Look for processes consuming excessive **bandwidth**.

✅ **Solution:**

- Restart affected service or limit network-heavy applications.
- Use **traffic shaping** tools like tc to limit bandwidth usage.

---

6. Check System Logs for Errors

Check **dmesg** for Kernel Errors: *dmesg | tail -50*

- Look for **I/O errors, memory errors, or CPU throttling messages**.

Check System Logs (**journalctl**): journalctl -xe

- Investigate application failures or crashes.

✅ **Solution:**

- Fix errors based on log messages.
- Restart affected services using: *systemctl restart <service_name>*

---

7. Optimize Performance and Prevent Future Issues

Clear Cache and Buffers**:** sync && echo 3 > /proc/sys/vm/drop_caches

Disable Unnecessary Services

systemctl list-units --type=service

systemctl disable <service_name>

Automate Regular Cleanup

Add a cron job to clean logs:

crontab -e

0 2 * * * rm -rf /var/log/*

2. How do you check which process is using high CPU or memory in Linux?

To check which process is consuming high CPU or memory in Linux, you can use the following commands:

1. **Using top (Real-time Monitoring):** *top*

- CPU-consuming processes are sorted by %CPU.
- Memory-consuming processes are sorted by %MEM.
- Press Shift + M to sort by memory usage.
- Press Shift + P to sort by CPU usage.

2. **Using htop (Interactive & User-Friendly)**

If not installed, install it using:

- sudo apt install htop # Debian/Ubuntu
- sudo yum install htop # CentOS/RHEL

3. **Using ps (List Top Processes by CPU and Memory)**

ps -eo pid,ppid,cmd,%cpu --sort=-%cpu | head -10 #High cpu usage
ps -eo pid,ppid,cmd,%mem --sort=-%mem | head -10 #High memory usage

4. **Using pidstat (Per-Process CPU Usage Over Time)**

pidstat 1 #Monitors CPU usage per process dynamically.

5. **Using vmstat (System-Wide Resource Monitoring)**
   vmstat 1 5 #Shows CPU, memory, and disk usage every second for 5 iterations.

6. **Using sar (Historical CPU Usage)**
   sar -u 5 10 #Displays CPU usage every 5 seconds, 10 times.

7. **Using ps aux with Sorting**
   ps aux --sort=-%cpu | head -10 #Top 10 CPU-consuming Processes
   ps aux --sort=-%mem | head -10 #Top 10 Memory-consuming Processes

8. **Using iotop (Disk I/O Usage by Process)**
   Iotop #Shows which process is consuming disk I/O.

9. **Using lsof (Find Processes Accessing a File)**
   lsof /path/to/file #Useful if a file is being heavily read/written.

10. **Killing a High CPU/Memory Process**
    Once you find the PID of the problematic process, you can kill it:
    kill -9 <PID>
    pkill -9 <process_name>

3. What steps do you take when a server is unreachable via SSH?

When a server is unreachable via SSH (Connection timed out or Connection refused), follow these steps to diagnose and resolve the issue:

1. **Check Network Connectivity:**

⏳ Ping the Server: **ping <server_ip>**

If **ping fails**, the server might be down, or there's a network issue.

If **ping works**, the network is fine, and SSH may be the issue.

⏳ Check Route and Firewall

- traceroute <server_ip> # For Linux/macOS
- tracert <server_ip> # For Windows

If packets stop at a certain point, there might be a **network routing issue**.

⏳ Verify Port Accessibility

- nc -zv <server_ip> 22
- telnet <server_ip> 22

If **port 22 is closed**, SSH might not be running or a firewall is blocking access.

2. Check Firewall Rules

- **Local Firewall**: Ensure the server's firewall (e.g., iptables, ufw) allows traffic on the SSH port.
- **Network Firewall**: Verify that any intermediate firewalls or security groups (e.g., AWS Security Groups) permit SSH traffic.

3. Check for Server Load or Resource Issues

If the server is overloaded or out of resources, SSH may fail.

Use console access to check CPU, memory, and disk usage: top, htop, df -h, or free -m.

4. Verify SSH Configuration

Check the SSH configuration file (/etc/ssh/sshd_config) for any misconfigurations:

Ensure **PermitRootLogin** and **PasswordAuthentication** are set correctly.

Confirm the **ListenAddress** and **Port** settings are correct.

5. Check for IP or DNS Issues

- Ensure you're using the correct IP address or hostname.
- If using a hostname, verify DNS resolution with nslookup <hostname> or dig <hostname>.

6. Investigate Logs: Check SSH logs for errors:

On the server: /var/log/auth.log or /var/log/secure.

On the client: Use ssh -vvv <server_ip> for verbose output to identify connection issues.

7. Check for Account Issues

- Ensure your user account is not locked or expired.
- Verify your SSH key or password is correct.

8. Reboot the Server

- If all else fails and you have console access, reboot the server to resolve potential software issues.

9. Contact Hosting Provider

- If the server is hosted by a cloud provider (e.g., AWS, DigitalOcean), check the provider's dashboard for outages or issues.
- Use the provider's console access to troubleshoot further.

Commands used for troubleshooting:

- ping <server_ip>
- traceroute <server_ip>

- nmap <server_ip>
- telnet <server_ip> 22
- systemctl status sshd
- ufw status
- ssh -vvv <server_ip>
- Check logs: /var/log/auth.log or /var/log/secure

4. How do you troubleshoot a disk space issue on a Linux server?

Troubleshooting a disk space issue on a Linux server involves identifying the cause of high disk usage and taking corrective actions.

Here's a step-by-step approach:

Check Disk Usage

Run the following command to check overall disk space usage: df -h

- -h (human-readable) displays disk space in GB/MB instead of blocks.
- Look at the % Used column to identify full partitions.

If / or any other partition is nearly full, proceed with further investigation.

Identify Large Files and Directories

Use the du command to find large directories: du -sh /path/to/directory/*

Check for Large Files

Use the find command to locate large files (e.g., files larger than 100MB):

find / -type f -size +100M -exec ls -lh {} \;

Check for Log File Growth

Log files can consume significant space. Check common log directories: du -sh /var/log

Inspect large log files: ls -lh /var/log/*.log

Clear or rotate logs if necessary: truncate -s 0 /var/log/large-logfile.log

Check for Deleted Files Still in Use

Deleted files that are still held open by processes can consume space. Use lsof to identify them:

lsof | grep deleted

Restart the associated processes to free up space.

Check for Temporary Files

Temporary files in /tmp or /var/tmp can accumulate:

du -sh /tmp

du -sh /var/tmp

Other Activities:

- Remove unnecessary files: rm -rf /tmp/*
- Remove unused packages to free up space: sudo apt autoremove
- List LVM snapshots: lvs #get large snapshot file and if no needed, delete it.
- Clean up unused containers, images, and volumes: docker system prune -a
- If disk quotas are enabled, check user usage: repquota -a

5. How do you check which process is using a specific port?

To check which process is using a specific port on a Linux server, you can use several tools and commands.

1. **netstat** is a traditional tool for network statistics. Use it with the -tuln and -p options:

   sudo netstat -tulnp | grep :<port_number>

2. **ss** is a modern replacement for netstat. Use it with the -tuln and -p options:

   sudo ss -tulnp | grep :<port_number>

3. **lsof** lists open files and can be used to find processes using a specific port:

   sudo lsof -i :<port_number>

4. **fuser** identifies processes using a specific file or socket:

   sudo fuser <port_number>/tcp

5. If you don't have netstat, ss, or lsof, you can manually check **/proc**:
   grep -l <port_number> /proc/[0-9]*/net/tcp

6.What do you do if a service fails to start in Linux?

If a service fails to start in Linux, follow these troubleshooting steps:

1. Check Service Status: **systemctl status <service-name>**
   This will provide details on whether the service is running, stopped, or has errors.

2. Restart the Service: **sudo systemctl restart <service-name>**
   If it was a temporary issue, this might resolve it.
3. Check Logs for Errors: **journalctl -xe -u <service-name>**
4. Ensure required dependent-services are running
   systemctl list-dependencies <service-name>
   Start missing dependencies if needed:
   sudo systemctl start <dependency-service>
5. Check for Port Conflicts:
   Find if the required port is occupied: sudo netstat -tulnp | grep <port-number>
   If another process is using the port, kill it or change the service's port.
6. Ensure the system has enough space: df -h
7. Check service file permissions: ls -l /etc/systemd/system/<service-name>.service

7. How do you troubleshoot high disk I/O usage?

- **Check I/O Usage with iostat:** Look for high %util and long await times.
  iostat -dx 1 5
- **Identify Heavy I/O Processes:** Shows processes consuming the most disk I/O.
  sudo iotop
- **Analyze Disk Usage:** Check if the disk is full or nearing capacity.
  df -h
- **Find Large Files Consuming Space:** Identifies large files consuming storage.
  sudo du -ah / | sort -rh | head -20
- **Monitor System-wide I/O with vmstat:** Check **bi** (blocks in) and **bo** (blocks out) to detect high I/O.
  vmstat 1 5
- **Check Active Disk Reads/Writes:** Measures disk latency.
  sudo ioping -c 10 /
- **Verify Logs for Errors:** Look for disk-related errors.
  dmesg | grep -i error
  journalctl -xe
- **Check for Zombie or Stuck Processes:** Unresponsive processes in "D" state can cause high I/O.
  ps aux | awk '$8=="D" {print $0}'
- **Examine Swap Usage:** High swap usage may indicate excessive paging.
  free -m

8.How do you check and repair a corrupted file system?

- **Identify the Affected Partition:**
  List available disks and partitions: lsblk
  Check mounted filesystems: df -h
  Find filesystem type: sudo blkid /dev/sdX
- **Unmount the Affected Partition (If Mounted):** sudo umount /dev/sdX
- **Check for Errors Using fsck:** sudo umount /dev/sdX

- **Repair the File System:**
  sudo e2fsck -p /dev/sdX # For ext4, ext3, ext2 filesystems
  sudo xfs_repair /dev/sdX # For XFS filesystem
  sudo btrfs check --repair /dev/sdX # For Btrfs filesystem
- **Remount the Partition:** sudo mount /dev/sdX /mnt
- **Verify File System Health:** sudo fsck -n /dev/sdX
  -n runs a read-only check without making changes.
- **Backup Critical Data:** If errors persist, consider backing up your data:
  rsync -av /mnt /backup/
- **Consider Reformatting (If Repair Fails):**

9. What steps do you follow to troubleshoot network connectivity issues?

Troubleshooting network connectivity issues on a Linux server involves a systematic approach to identifying and resolving the problem. Below are the key steps:

---

1. **Check Network Interface Status:** Start by verifying that the network interface is up and running:
   ip a **OR** ifconfig -a # Older distributions
   Look for UP status. If the interface is down, bring it up using: ***sudo ip link set eth0 up***
2. **Verify IP Address Assignment:** Ensure the server has a valid IP address:
   ip addr show **OR** ip route show
3. **Test Local Network Connectivity:** Ping the gateway to check if the server can communicate within the network. **ping -c 4 <gateway_IP>**

   If there's no response:
   Check if another device on the network can reach the gateway.
   Restart the router or network switch.

4. **Check Routing Table:** Ensure the default gateway is set correctly:
   ip route show OR If missing, add a default route: **sudo ip route add default via <gateway_IP>**
5. **Check DNS Resolution:** Try resolving a domain: **nslookup google.com**
6. Check if firewall rules are blocking traffic**: sudo iptables -L -v -n**
7. Check for Network Interface Error: Check for dropped packets or errors:
   ip -s link show eth0
8. Sometimes, restarting network services resolves issues:
   sudo systemctl restart networking
   sudo systemctl restart NetworkManager

10.How do you recover a lost root password in Linux?

Recovering a lost root password in Linux requires booting into recovery mode or single-user mode. Here's a step-by-step guide:

Using GRUB (Most Common)

This method works for most Linux distributions, including Ubuntu, CentOS, RHEL, and Debian.

1. Reboot and Access GRUB

- Restart the system.
- Press **Esc**, **Shift**, or **F8** during boot to enter the **GRUB menu**.
- If you don't see the GRUB menu, hold **Shift** continuously while booting.

2. Edit the GRUB Boot Entry

- Highlight the **default boot option** (usually the first one).
- Press **e** to edit the entry.

3. Modify the Kernel Parameters

- Find the line starting with linux or linux16 (depending on your system).
- Locate the section containing ro quiet splash or similar.
- Replace ro (read-only) with rw (read-write) and add init=/bin/bash.

The modified line should look like this:

linux /boot/vmlinuz-... rw init=/bin/bash

Press **Ctrl + X** or **F10** to boot with these settings.

4. Reset the Root Password

Once the system boots into a bash shell, reset the root password: ***passwd root***

Enter the new root password twice.

5. Reboot the System

Remount the filesystem to ensure changes persist: **mount -o remount,rw /**

Then, restart: **exec /sbin/init**

11.How do you troubleshoot a server rebooting unexpectedly?

1. **Check System Logs:** Review /var/log/messages, /var/log/syslog, or Event Viewer (Windows) for crash reports or error messages.
2. **Analyze Crash Dumps:** Check for kernel panic logs (/var/crash/ on Linux) or memory dump files on Windows (C:\Windows\Minidump\).
3. **Monitor Resource Usage:**

- Check CPU, memory, and disk usage (top, htop, free -m, df -h).
- Identify resource exhaustion or OOM killer events (dmesg | grep -i oom).

4. **Hardware Issues:** Look for power failures, overheating (check dmesg | grep -i temperature), or faulty RAM (memtest).
5. **Check Recent Changes:** Review recent OS patches, application updates, or config changes.
6. **Verify Auto-Reboot Settings:** Ensure no scheduled reboots (crontab -l, shutdown -r commands, Task Scheduler in Windows).
7. **Check Security & Intrusions:** Look for unauthorized access (last -x reboot, auth.log, security logs).
8. **Inspect Services & Applications:** Check if any process is triggering the reboot (journalctl -xe, systemctl status).
9. **Test Power Supply & UPS:** Ensure stable power input and check UPS logs.
10. **Kernel & Driver Issues:** Check for driver failures (lsmod, dmesg | grep -i error).

12.How do you investigate high load average on a Linux system?

1. **Check Load Average:** Run uptime or w to check the load average values.
2. **Analyze CPU Usage:** Use top or htop to check CPU utilization and identify high-consuming processes.
3. **Check Running Processes:** Run ps -eo pid,ppid,cmd,%cpu,%mem --sort=-%cpu | head to identify top processes.
4. **Investigate I/O Bottlenecks:** Use iostat -x 1 (from sysstat package) or iotop to check disk activity.
5. **Check Memory Usage & Swapping:**

- Run free -m to check RAM usage.
- Use vmstat 1 to check swap activity and paging issues.

6. **Analyze Disk Usage & Latency:**

- Use df -h to check disk space.
- Run dstat -d or iostat to check disk I/O latency.

7. **Inspect System Load with SAR:** Use sar -q (from sysstat package) to analyze historical load trends.
8. **Check Kernel Logs for Errors:** Run dmesg | tail -20 to look for hardware or kernel issues.
9. **Verify Network Load:** Use ss -tupn or iftop to check for high network activity.
10. **Investigate Zombie or Stuck Processes:**

- Run ps aux | grep Z to identify zombie processes.
- Check systemctl status for stuck services.

13. What do you do if a user cannot log in to a Linux server?

1. **Check User Account Status:**

- Run id username to verify the account exists.
- Check if the account is locked: passwd -S username or faillock --user username.

2. **Verify SSH Connectivity:**

- Check if the SSH service is running: systemctl status sshd.
- Verify SSH logs: journalctl -xeu sshd or /var/log/auth.log (Debian-based) / /var/log/secure (RHEL-based).

3. **Check User Permissions & Shell:**

- Ensure the correct shell is assigned: cat /etc/passwd | grep username.
- Verify the home directory permissions: ls -ld /home/username.

4. **Inspect Password & Authentication:**

- Check password expiration: chage -l username.
- Unlock account if locked: passwd -u username.

5. **Analyze PAM & SSH Configuration:**

- Check /etc/ssh/sshd_config for restrictions (AllowUsers, DenyUsers).
- Review /etc/pam.d/sshd for authentication rules.

6. **Check System Resource Limits:**

- Verify ulimit restrictions: ulimit -a.
- Inspect /etc/security/limits.conf for login restrictions.

7. **Test with a Different User:** Try logging in with another user to check if it's a system-wide issue.

8. **Check Network & Firewall Settings:**

- Verify connectivity: ping server-ip, telnet server-ip 22.
- Check firewall rules: iptables -L -n or firewalld --list-all.

9. **Investigate SELinux/AppArmor Restrictions:** Check SELinux logs: grep "denied" /var/log/audit/audit.log.
10. **Check for System-Wide Issues:** Review system logs: dmesg | tail -20 or journalctl -xe.

14. How do you check for failed login attempts on a Linux system?

Check Authentication Logs:

- **Debian/Ubuntu:** grep "Failed password" /var/log/auth.log
- **RHEL/CentOS:** grep "Failed password" /var/log/secure

View Last Failed Logins:

- Run lastb (reads from /var/log/btmp).
- Use who -a to check current login attempts.

Check Failures with faillock: faillock --user username (for RHEL-based systems).

Analyze Journal Logs: journalctl -xe | grep "Failed password" (for systemd-based logs).

Monitor SSH Login Failures:

- grep "authentication failure" /var/log/auth.log (or /var/log/secure).
- Use ausearch -m USER_LOGIN -sv no (for audit logs).

Check Brute-Force Attacks: Use grep "Failed password" /var/log/auth.log | awk '{print $11}' | sort | uniq -c | sort -nr to find repeated attempts from the same IP.

15. How do you troubleshoot a DNS resolution issue?

1. **Verify Network Connectivity:**

- Run ping 8.8.8.8 to check internet access.
- Use ifconfig or ip a to verify network interfaces.

2. **Check DNS Server Configuration:**

- View configured DNS servers: cat /etc/resolv.conf.
- Ensure correct nameservers are set (e.g., 8.8.8.8, 1.1.1.1).

3. **Test DNS Resolution with nslookup or dig:** nslookup example.com or dig example.com to check DNS response.
4. **Use host Command to Verify Resolution:** host example.com to check if the domain resolves.

5. **Check Local DNS Cache:** Flush cache: *systemctl restart nscd* **or** *systemctl restart systemd-resolved.*

6. **Test with Alternative DNS Servers***: nslookup example.com 8.8.8.8* to check resolution via Google DNS.

7. **Verify Firewall & IPTables Rules:**

- *iptables -L -n* to check for DNS-blocking rules.
- *systemctl status firewalld* to check firewall status.

8. **Check /etc/hosts File:** Ensure there are no incorrect entries in /etc/hosts.

9. **Restart the DNS Resolver:** *systemctl restart systemd-resolved* (for systemd-based systems).

10. **Check Logs for Errors***: journalctl -xe | grep dns or cat /var/log/syslog | grep named*.

16. How do you identify zombie processes and handle them?

Identifying Zombie Processes:

- **Using ps Command:** Run *ps aux | grep Z or ps -eo pid,ppid,stat,cmd | grep Z* to list zombie processes.
- **Using top or htop:** Check for processes with *Z* (zombie) in the status column.
- **Using proc Filesystem:** Run *grep -w "Z" /proc/*/status* to find zombie processes.

Handling Zombie Processes:

1. **Find Parent Process (PPID):** Use *ps -o ppid= -p <Zombie_PID>* to identify the parent process.
2. **Restart the Parent Process:** If the parent is still running, restart it: *kill -HUP <Parent_PID>.*
3. **Kill the Parent Process (if unresponsive):** Use *kill <Parent_PID>* or *kill -9 <Parent_PID>* to terminate it.
4. **Verify Process Cleanup:** Check again with *ps aux | grep Z* to confirm the zombie is gone.
5. **Reboot as a Last Resort:** If zombies persist, consider a system reboot.

17. What steps do you take when a system runs out of inodes?

An **inode (index node)** is a data structure used by Linux filesystems (like ext4, XFS) to store metadata about files and directories.

Identify Inode Usage:

1. **Check Inode Usage on Filesystem:** Run df -i to check available and used inodes.

2. **Find the Filesystem Consuming Inodes:** Use find / -xdev -type d -exec df -i {} + to locate the affected filesystem.

3. **Identify Small Files Consuming Excessive Inodes:**

- Run find /path -xdev -type f | wc -l to count files in a directory.

- Use find /path -xdev -type f -printf "%h\n" | sort | uniq -c | sort -nr | head to locate directories with excessive files.

Free Up Inodes:

Delete Unnecessary Small Files: Remove logs, cache, or temp files: rm -rf /path/to/unwanted/files/*.

Clear Package Manager Cache: apt-get clean (Debian-based) or yum clean all (RHEL-based).

Remove Old Logs and Temporary Files:

- journalctl --vacuum-time=7d (for system logs).
- find /var/log -type f -name "*.log" -delete.

Check for Stale Inode Usage:

- Run lsof | grep deleted to identify processes holding deleted files.
- Restart the affected service or process.

Prevent Future Inode Exhaustion:

Monitor Inode Usage Regularly: Set up alerts using df -i in monitoring tools.

Reformat Filesystem with More Inodes (Last Resort): If inode exhaustion is frequent, reformat with a higher inode count: *mkfs.ext4 -i 4096 /dev/sdX*

18. How do you troubleshoot a Linux system hanging or freezing?

1. Check System Logs for Clues

- Run journalctl -xe or dmesg -T for kernel errors.
- Check /var/log/syslog or /var/log/messages for relevant logs.

2. Monitor System Resource Usage

- top or htop → Check CPU, memory, and load average.
- free -m → Check available RAM and swap usage.
- vmstat 1 5 → Identify CPU, memory, and I/O bottlenecks.

3. Identify High I/O Wait or Disk Issues

- iostat -x 1 5 (from sysstat package) → Check disk usage.
- iotop → Identify processes causing high disk I/O.

4. Check for Out-of-Memory (OOM) Issues

- dmesg | grep -i oom → Look for OOM Killer logs.
- cat /proc/meminfo | grep -i swap → Check swap usage.

5. Look for High CPU Usage or Kernel Locks

- ps aux --sort=-%cpu → Identify CPU-intensive processes.

- cat /proc/loadavg → Check system load.

6. Check Disk Space and Inodes

- df -h → Ensure sufficient disk space.
- df -i → Check inode availability.

7. Inspect Running and Stuck Processes

- ps aux | grep D → Find uninterruptible sleep (D-state) processes.
- kill -9 <PID> → Force kill stuck processes.

8. Identify Network Issues (if applicable)

- ping -c 4 google.com → Check connectivity.
- netstat -tulnp or ss -tulnp → Look for blocked ports.

9. Investigate Hardware Issues

- lspci -vvv → Check for hardware errors.
- smartctl -a /dev/sdX → Check disk health (install smartmontools).

10. Use Magic SysRq Keys (If System Is Unresponsive)

- Alt + SysRq + REISUB (Reboot safely without data loss).

19. How do you check logs for troubleshooting issues in Linux?

1. System Logs (General Issues & Boot Logs)

- journalctl -xe → View recent system logs.
- journalctl -b -1 → Check logs from the previous boot.
- dmesg -T → View kernel logs (hardware and boot issues).

2. Authentication & Security Logs

- cat /var/log/auth.log (Debian/Ubuntu) → Check login attempts, sudo activity.
- cat /var/log/secure (RHEL/CentOS) → Security-related logs.

3. Application & Service Logs

- systemctl status <service> → Check service status.
- journalctl -u <service> → View logs of a specific service.
- cat /var/log/syslog (Debian/Ubuntu) or cat /var/log/messages (RHEL/CentOS) → General system logs.

4. Web Server Logs (Apache/Nginx)

- Apache: /var/log/apache2/access.log and /var/log/apache2/error.log
- Nginx: /var/log/nginx/access.log and /var/log/nginx/error.log

5. SSH Logs

- cat /var/log/auth.log | grep ssh → Check SSH login attempts.

6. Cron Job Logs

- cat /var/log/syslog | grep CRON (Debian/Ubuntu)
- cat /var/log/cron (RHEL/CentOS)

7. Disk and Filesystem Issues

- cat /var/log/kern.log → Check kernel-related disk issues.
- journalctl -f → View real-time logs.

8. Network Issues Logs

- cat /var/log/syslog | grep -i network
- journalctl -u NetworkManager → Check network manager logs.

9. Container & Kubernetes Logs

- Docker: docker logs <container_id>
- Kubernetes: kubectl logs <pod_name> -n <namespace>

10. Real-time Log Monitoring

- tail -f /var/log/syslog → Monitor logs in real time.
- tail -f /var/log/messages (RHEL-based systems).

20. How do you investigate and fix SELinux-related permission issues?

SELinux (Security-Enhanced Linux) is a security module integrated into the Linux kernel that provides an additional layer of security by enforcing mandatory access control (MAC) policies. It was developed by the United States National Security Agency (NSA) to enhance the security of Linux systems.

1. Check SELinux Status

- Run sestatus to check if SELinux is enabled and its mode (Enforcing, Permissive, Disabled).

2. Review SELinux Logs

- Check /var/log/audit/audit.log for SELinux-related denials.
- Use ausearch -m avc to filter SELinux audit logs.

3. Identify Denied Operations

- Look for "AVC denials" in the logs to find the exact operation being blocked.
- Use grep to search for SELinux denials: ***grep "denied" /var/log/audit/audit.log***

4. Use sealert for Detailed Analysis

- Run sealert -a /var/log/audit/audit.log for detailed explanations and suggestions.
- Check the generated report for specific SELinux denial messages.

5. Modify SELinux Policy (if required)

- If a denial is caused by incorrect context, fix it using ***semanage fcontext*** and then ***restorecon***:

  semanage fcontext -a -t <context_type> <path>

  restorecon -v <path>

6. Temporarily Set SELinux to Permissive Mode

- For testing, switch SELinux to Permissive mode: ***setenforce 0***
- In Permissive mode, SELinux logs denials but doesn't block operations. Test the process to verify if SELinux is the cause.

7. Apply Custom SELinux Module (if required)

- If a permanent policy change is needed, create a custom module with audit2allow: *audit2allow -M mymodule < /var/log/audit/audit.log*

  semodule -i mymodule.pp

8. Restore Default Contexts (if required)

- If file or directory contexts are incorrect, restore them using: *restorecon -R -v /path/to/directory*

9. Re-enable SELinux (if in Permissive Mode)

- Once testing is complete, set SELinux back to Enforcing mode: *setenforce 1*