Here are **real-world use cases and interview scenarios** that involve Linux commands relevant to cloud and DevOps roles:

**1. File Management and Permissions**

**Scenario:**

* You're troubleshooting a file access issue for an application running on a cloud server.

**Problem:**

* An application fails to read a configuration file. Investigate and resolve it.

**Solution Commands:**

bash

Copy code

ls -l /path/to/file # Check file permissions.

chmod 644 /path/to/file # Set correct permissions.

chown appuser:appgroup /path/to/file # Set correct ownership.

**Follow-up Question:**

* What do 644 permissions mean?

**2. Process and Resource Monitoring**

**Scenario:**

* The application on an AWS EC2 instance is slow, and you suspect high CPU or memory usage.

**Problem:**

* Identify the resource bottleneck and resolve the issue.

**Solution Commands:**

bash

Copy code

top # Check live CPU and memory usage.

ps aux | grep app\_process # Locate the application process.

kill -9 <PID> # Force-stop the problematic process.

**Follow-up Question:**

* How would you prevent a process from consuming too much CPU in the future?  
  (Answer: Use nice or cpulimit.)

**3. Disk Usage and Cleanup**

**Scenario:**

* A server has run out of disk space, impacting application performance.

**Problem:**

* Free up disk space while retaining critical files.

**Solution Commands:**

bash

Copy code

df -h # Check disk space usage.

du -sh /var/log/\* # Find large files in logs.

gzip /var/log/access.log # Compress large logs.

find /tmp -type f -mtime +7 -delete # Delete files older than 7 days.

**Follow-up Question:**

* How would you automate this cleanup process? (Answer: Use a cron job.)

**4. Network Troubleshooting**

**Scenario:**

* An application cannot connect to a remote database hosted on another AWS instance.

**Problem:**

* Diagnose and fix the connectivity issue.

**Solution Commands:**

bash

Copy code

ping <database-host> # Check basic connectivity.

telnet <database-host> 3306 # Test database port connectivity.

curl -v http://<database-host>:3306 # Check if the database service responds.

iptables -L # Verify firewall rules.

**Follow-up Question:**

* How would you allow database connections through the firewall? (Answer: Use iptables or AWS Security Groups.)

**5. Logs and Debugging**

**Scenario:**

* An application fails after deployment, and you need to identify the issue.

**Problem:**

* Analyze logs and find the error.

**Solution Commands:**

bash

Copy code

journalctl -u app.service # Check systemd logs for the application.

tail -f /var/log/app.log # Monitor application logs in real-time.

grep -i "error" /var/log/app.log # Search for error messages in the log.

strace -p <PID> # Trace system calls to debug the process.

**Follow-up Question:**

* How would you make logs easier to search in the future? (Answer: Use centralized log management tools like ELK Stack.)

**6. Automation Using Scripts**

**Scenario:**

* You need to create a script to back up important files daily to an S3 bucket.

**Problem:**

* Write and schedule the script.

**Solution Commands:**

bash

Copy code

#!/bin/bash

tar -czf /backup/files.tar.gz /data\_to\_backup

aws s3 cp /backup/files.tar.gz s3://my-bucket/backups/

**Schedule with Cron:**

bash

Copy code

crontab -e

0 2 \* \* \* /path/to/backup\_script.sh # Run the script daily at 2 AM.

**Follow-up Question:**

* How would you ensure the backup script does not run if the previous job is still running? (Answer: Use lock files or flock.)

**7. Container Management**

**Scenario:**

* You need to debug a containerized application that is not working.

**Problem:**

* Investigate and fix the issue.

**Solution Commands:**

bash

Copy code

docker ps # List running containers.

docker logs <container\_id> # Check container logs.

docker exec -it <container\_id> bash # Access container shell.

docker inspect <container\_id> # Get detailed container configuration.

**Follow-up Question:**

* How would you optimize container logs for production use? (Answer: Use logging drivers or redirect logs to cloud storage like S3.)

**8. Kubernetes Deployment**

**Scenario:**

* You deployed a Kubernetes application, but it is not accessible.

**Problem:**

* Debug and resolve the issue.

**Solution Commands:**

bash

Copy code

kubectl get pods # Check pod status.

kubectl logs <pod\_name> # View logs of the pod.

kubectl describe pod <pod\_name> # Get detailed information about the pod.

kubectl get svc # Check if the service is exposing the application.

**Follow-up Question:**

* How would you monitor Kubernetes cluster health? (Answer: Use tools like Prometheus and Grafana.)

**9. Cloud-Specific Tasks**

**Scenario:**

* Launch and configure an AWS EC2 instance via CLI.

**Problem:**

* Automate instance setup using AWS CLI.

**Solution Commands:**

bash

Copy code

aws ec2 run-instances \

--image-id ami-12345678 \

--count 1 \

--instance-type t2.micro \

--key-name MyKeyPair \

--security-group-ids sg-12345678 \

--subnet-id subnet-12345678

**Follow-up Question:**

* How would you ensure the instance is configured with the required software? (Answer: Use a user-data script or configuration management tools like Ansible.)

**10. Performance Tuning**

**Scenario:**

* Optimize an application server running on Linux.

**Problem:**

* Identify and fix performance bottlenecks.

**Solution Commands:**

bash

Copy code

iostat -x # Check disk I/O statistics.

vmstat # Analyze system performance metrics.

sar # Monitor CPU, memory, and disk usage.

sysctl -a # View kernel parameters.

**Follow-up Question:**

* How would you make kernel parameter changes persistent across reboots? (Answer: Update /etc/sysctl.conf.)