**Scenario 1: File Permissions Debugging**

**Scenario:**  
An application fails to start because it cannot access a configuration file.

**Solution Explanation:**

1. **List permissions:** Use ls -l /path/to/file to confirm the file’s ownership and permissions.
2. **Modify permissions:** If the file needs read/write access, use chmod (e.g., chmod 644 /path/to/file for read/write for the owner and read-only for others).
3. **Change ownership:** If the application user doesn’t own the file, assign ownership using chown (e.g., chown appuser:appgroup /path/to/file).

**Mock Interview Questions:**

1. What does the first column in the ls -l output represent?  
   *(Answer: File type and permissions in the format drwxr-xr-x.)*
2. How do you recursively set permissions for a directory?  
   *(Answer: chmod -R <permissions> <directory>.)*

**Scenario 2: Application Slowness on a Cloud Server**

**Scenario:**  
You’re tasked with identifying why an application on an AWS EC2 instance is running slowly.

**Solution Explanation:**

1. **Monitor resource usage:** Use top or htop to identify high CPU or memory usage.
2. **Inspect processes:** Use ps aux | grep app\_process to locate the specific process.
3. **Terminate runaway processes:** Use kill or kill -9 for immediate termination.

**Mock Interview Questions:**

1. What is the difference between kill and kill -9?  
   *(Answer: kill sends a default TERM signal, allowing the process to clean up. kill -9 sends a KILL signal to force termination.)*
2. How can you ensure an application’s resource usage remains within limits?  
   *(Answer: Use tools like nice, cgroups, or Kubernetes resource quotas.)*

**Scenario 3: Automating Backups**

**Scenario:**  
Automate the daily backup of application logs to an AWS S3 bucket.

**Solution Explanation:**

1. **Write the backup script:**

bash

Copy code

#!/bin/bash

tar -czf /backup/app\_logs\_$(date +%F).tar.gz /var/log/app/

aws s3 cp /backup/app\_logs\_$(date +%F).tar.gz s3://my-bucket/backups/

1. **Schedule it with cron:**

bash

Copy code

crontab -e

0 2 \* \* \* /path/to/backup\_script.sh

**Mock Interview Questions:**

1. How do you debug if a cron job does not run as expected?  
   *(Answer: Check the cron logs using journalctl -u cron or look for errors in /var/log/syslog.)*
2. How can you secure sensitive information like AWS credentials in scripts?  
   *(Answer: Use AWS IAM roles or environment variables.)*

**Scenario 4: Network Troubleshooting**

**Scenario:**  
A web server running on an AWS instance isn’t accessible from the internet.

**Solution Explanation:**

1. **Test connectivity:**
   * Use ping to check if the server is reachable.
   * Use curl or telnet to test if the web server port (e.g., 80) is open.
2. **Inspect firewall rules:**
   * On the server: Use iptables -L or ufw status.
   * On AWS: Verify Security Groups and Network ACLs.

**Mock Interview Questions:**

1. How would you check if port 80 is listening on a Linux server?  
   *(Answer: Use netstat -tuln | grep :80 or ss -tuln | grep :80.)*
2. How do you resolve DNS issues on a Linux server?  
   *(Answer: Use nslookup or dig to check DNS resolution.)*

**Scenario 5: Debugging a Dockerized Application**

**Scenario:**  
A Docker container fails to start or behaves unexpectedly after deployment.

**Solution Explanation:**

1. **Check container status:** Use docker ps -a to view stopped containers.
2. **Inspect logs:** Use docker logs <container\_id> to view errors.
3. **Access the container shell:** Use docker exec -it <container\_id> bash to debug issues interactively.

**Mock Interview Questions:**

1. What does the -it flag do in docker exec?  
   *(Answer: -i keeps the session open, and -t allocates a pseudo-TTY.)*
2. How would you persist container logs for analysis?  
   *(Answer: Use logging drivers like json-file or forward logs to a centralized service like AWS CloudWatch.)*

**Scenario 6: Kubernetes Debugging**

**Scenario:**  
An application deployed on Kubernetes isn’t accessible via the service.

**Solution Explanation:**

1. **Check pod status:** Use kubectl get pods to ensure the pod is running.
2. **View logs:** Use kubectl logs <pod\_name> for errors.
3. **Inspect the service:** Use kubectl get svc to confirm the service is exposing the application correctly.

**Mock Interview Questions:**

1. How would you expose a Kubernetes deployment to the internet?  
   *(Answer: Use a LoadBalancer service or an Ingress controller.)*
2. How do you debug when a pod enters a CrashLoopBackOff state?  
   *(Answer: Check pod logs and describe the pod using kubectl describe pod <pod\_name>.)*

**Scenario 7: Optimizing System Performance**

**Scenario:**  
A database server on Linux shows high disk I/O, leading to latency issues.

**Solution Explanation:**

1. **Monitor I/O:** Use iostat -x to identify disk bottlenecks.
2. **Check active processes:** Use lsof or iotop to identify processes causing high I/O.
3. **Optimize filesystem:** Use tune2fs to adjust ext4 parameters.

**Mock Interview Questions:**

1. How would you reduce I/O contention on a heavily used server?  
   *(Answer: Use caching, separate workloads across disks, or upgrade to faster storage like SSDs.)*
2. How do you measure the time it takes for a command to execute?  
   *(Answer: Use the time command.)*

**Scenario 8: Deployment Automation**

**Scenario:**  
Deploy a web application on an AWS EC2 instance using a script.

**Solution Explanation:**

1. **Create a deployment script:**

bash

Copy code

#!/bin/bash

sudo yum update -y

sudo yum install httpd -y

sudo systemctl start httpd

sudo systemctl enable httpd

echo "<h1>Deployed Successfully</h1>" > /var/www/html/index.html

1. **Run the script during instance launch:** Pass it as user data.

**Mock Interview Questions:**

1. What is the difference between systemctl enable and systemctl start?  
   *(Answer: enable sets the service to start on boot, while start starts it immediately.)*
2. How would you ensure idempotency in deployment scripts?  
   *(Answer: Use checks to verify the state before making changes, e.g., check if a package is already installed.)*

Bottom of Form