

DevOps Interview Handbook – 200+ Practical Q&A

This handbook is written with a **real production + interview mindset**.

✓ It helps you answer **clearly** (what + why + how) ✓ Builds **hands-on troubleshooting thinking** ✓
Includes **follow-up questions** that interviewers commonly ask

How to use this:

- Try answering each question in your own words first.
 - Then validate with the expected answer + commands.
 - Practice explaining your steps like you are in a real incident call.
-

Section A — Linux Fundamentals (1–45)

1) Check disk usage quickly

Q: How do you check disk usage and identify the largest directories?

A (steps + commands):

```
df -h
sudo du -sh /* 2>/dev/null | sort -hr | head -10
sudo du -sh /var/* 2>/dev/null | sort -hr | head
```

Follow-up: How do you find large files over 1GB?

```
sudo find / -type f -size +1G 2>/dev/null -exec ls -lh {} \; | sort -k5 -hr | head
```

2) Difference between load average and CPU usage

Q: What does Linux load average mean?

A: Load average represents processes **running or waiting (especially I/O wait)**. High load doesn't always mean CPU is high.

Commands:

```
uptime  
top  
vmstat 1 5
```

Follow-up: How do you detect I/O wait bottleneck?

```
iostat -xz 1 5
```

3) Find top CPU-consuming processes

Q: How do you find which processes consume CPU?

A (commands):

```
top  
ps -eo pid,comm,%cpu,%mem --sort=-%cpu | head
```

Follow-up: How do you stop it gracefully?

```
kill -TERM <PID>  
kill -KILL <PID>
```

4) Find top memory-consuming processes

Q: How do you find high memory usage processes?

A (commands):

```
free -h  
ps -eo pid,comm,%mem,rss --sort=-%mem | head
```

5) What is swap? When is swap bad?

Q: What is swap and how do you check swap usage?

A: Swap is disk-based memory. Some swap is okay, but heavy swapping causes slowness.

Commands:

```
swapon --show  
free -h  
vmstat 1 5
```

6) Find port listening processes

Q: How do you find which process is listening on a port?

A (commands):

```
sudo ss -lntp | grep :8080  
sudo lsof -i :8080
```

7) Check open connections

Q: How do you check established connections to your server?

A (commands):

```
sudo ss -tan state established  
sudo ss -tan state established | wc -l
```

8) Check system uptime + reboot history

Q: How do you check uptime and last reboot events?

A (commands):

```
uptime  
who -b  
last reboot | head
```

9) Analyze system logs during issues

Q: How do you check logs when a service fails?

A (commands):

```
sudo journalctl -u <service> -n 200 --no-pager  
sudo systemctl status <service>
```

10) Identify OS and kernel version

Q: How do you identify Linux distribution and kernel?

A (commands):

```
cat /etc/os-release  
uname -a  
uname -r
```

11) Find a file quickly

Q: How do you find a file by name?

A (commands):

```
find / -name "nginx.conf" 2>/dev/null  
locate nginx.conf
```

12) Find recently modified files

Q: How to find files modified in last 24 hours?

A (commands):

```
find /var/log -type f -mtime -1
```

13) Permission denied troubleshooting

Q: How do you debug "Permission denied"?

A: Check file ownership, permissions, and SELinux.

Commands:

```
ls -l /path/to/file  
namei -l /path/to/file  
getenforce
```

14) Check user and group permissions

Q: How to check user groups and IDs?

A (commands):

```
id <user>
groups <user>
```

15) Understand chmod values

Q: What does chmod 755 mean?

A: Owner: rwx (7), Group: r-x (5), Others: r-x (5)

Section B — Git & Version Control (46–75)

46) Git reset vs revert

Q: What's the difference between `git reset` and `git revert`?

A:

- `reset` rewrites history (danger in shared branches)
- `revert` makes a new commit to undo changes (safe)

Commands:

```
git reset --hard <commit>
git revert <commit>
```

47) Resolve merge conflicts

Q: How do you handle merge conflicts?

A: Pull latest, resolve conflict markers, then commit.

```
git pull
# edit files

git add .
git commit -m "Resolve conflicts"
```

48) Cherry-pick a commit

Q: How do you apply one commit from another branch?

```
git cherry-pick <commit_id>
```

49) Rollback a pushed commit

Q: How do you undo a commit already pushed to main?

A (safe):

```
git revert <commit_id>
```

50) Stash changes

Q: How do you save work temporarily?

```
git stash  
git stash list  
git stash pop
```

51) Tagging releases

Q: How do you tag a release?

```
git tag -a v1.0 -m "Release v1.0"  
git push origin v1.0
```

52) Rebase vs merge

Q: When to use rebase?

A: Rebase keeps clean linear history but should be avoided on public/shared branches.

```
git rebase main
```

53) How to check who changed a line

Q: How do you identify who modified a line in code?

```
git blame <file>
```

Section C — CI/CD with Jenkins (76–110)

76) Jenkins pipeline stages

Q: What are typical Jenkins pipeline stages?

A: Checkout → Build → Test → Scan → Package → Deploy

77) Declarative vs Scripted pipeline

Q: Difference between declarative and scripted pipelines?

A: Declarative is structured and easier to maintain; scripted is flexible (Groovy-heavy).

78) Jenkins agent not connecting

Q: Jenkins agent is offline. How to troubleshoot?

A (commands on agent):

```
systemctl status jenkins-agent  
ping <jenkins-master>  
ss -lntp
```

79) Jenkins workspace cleanup

Q: Builds fail due to workspace full. Fix?

```
sudo du -sh /var/lib/jenkins/workspace/* | sort -hr | head  
sudo rm -rf /var/lib/jenkins/workspace/<job>
```

80) Jenkins credentials best practices

Q: How do you store secrets in Jenkins?

A: Use Jenkins Credentials Store + bind via environment variables.

81) Pipeline fails in shell step

Q: Script works locally but fails in Jenkins.

A: Common issues: different PATH, missing permissions, missing env vars.

```
echo $PATH  
whoami  
pwd
```

82) Jenkins webhook not triggering

Q: Git push doesn't trigger job.

A: Check webhook delivery logs, Jenkins endpoint, network/firewall.

Section D — Docker (111-145)

111) Dockerfile best practices

Q: What are common Dockerfile best practices?

A: Small base image, multi-stage builds, pin versions, avoid root user.

112) Container exits immediately

Q: Container keeps stopping. How to debug?

```
docker ps -a  
docker logs <container>  
docker inspect <container>
```

113) Clean unused Docker resources

Q: How do you free Docker disk space safely?

```
docker system df  
docker system prune -a --volumes
```

114) Docker networking debugging

Q: Container cannot reach other container.

```
docker network ls  
docker network inspect <network>
```

115) Docker vs Containerd

Q: What is the difference between Docker and containerd?

A: Docker is a platform with CLI + build tools; containerd is the runtime used underneath.

Section E — Kubernetes (146–185)

146) Pod stuck in Pending

Q: Pod stuck in `Pending`. What do you check?

A: Scheduling issues: insufficient CPU/memory, taints, node selector.

```
kubectl describe pod <pod>
kubectl get nodes
```

147) Pod stuck in CrashLoopBackOff

Q: Pod restarts repeatedly. How to debug?

```
kubectl logs <pod>
kubectl logs <pod> --previous
kubectl describe pod <pod>
```

148) Service not reachable

Q: Service exists but app not reachable.

```
kubectl get svc
kubectl describe svc <svc>
kubectl get endpoints <svc>
```

149) ConfigMap vs Secret

Q: Difference between ConfigMap and Secret?

A: Both store config, but Secret is base64 encoded and intended for sensitive values.

150) Rolling update strategy

Q: What happens during deployment rolling update?

```
kubectl rollout status deploy/<name>
kubectl rollout history deploy/<name>
```

Section F — Terraform (186–205)

186) Terraform init/plan/apply

Q: Explain Terraform workflow.

A:

- `init` downloads providers
- `plan` shows changes
- `apply` executes changes

```
terraform init
terraform plan
terraform apply
```

187) Terraform state file importance

Q: What is Terraform state and why important?

A: Tracks real infra resources vs code. Losing state breaks drift management.

188) Remote state benefits

Q: Why use remote backend (S3 + DynamoDB)?

A: Team collaboration, locking, disaster recovery.

189) Terraform drift detection

Q: How do you detect drift?

```
terraform plan
terraform refresh
```

Section G — AWS + Cloud Basics (206–240)

206) EC2 high CPU troubleshooting

Q: EC2 instance is slow due to CPU high.

A (commands):

```
top  
ps -eo pid,comm,%cpu --sort=-%cpu | head
```

207) Security Group vs NACL

Q: Difference between SG and NACL?

A:

- SG: stateful, instance-level
- NACL: stateless, subnet-level

208) IAM Role vs IAM User

Q: Why use IAM role for EC2 instead of user keys?

A: Roles provide temporary credentials and reduce secret exposure.

209) S3 versioning use case

Q: Why enable versioning in S3?

A: Protect from accidental deletion and rollback objects.

Section H — Real-World Troubleshooting (241–220+)

241) Website down but server reachable

Q: Website is down, server SSH works. What steps do you follow?

A (triage checklist):

```
systemctl status nginx  
ss -lntp | grep -E ':80|:443'
```

```
curl -I http://localhost  
journalctl -u nginx -n 100 --no-pager
```

242) DNS resolves but connection fails

Q: DNS resolves but connection refused.

```
nslookup example.com  
curl -v https://example.com  
ss -tulpn
```

243) CI deploy succeeded but app not updated

Q: Deploy job success but old version running.

A: Possible caching, wrong tag, rollout not triggered.

```
kubectl rollout status deploy/<name>  
kubectl describe deploy/<name>
```

244) Disk space full in production

Q: Disk 100% full. What do you do first?

```
df -h  
sudo du -sh /* 2>/dev/null | sort -hr | head  
sudo journalctl --vacuum-time=7d
```

Quick Interview Answer Framework (Use This!)

When answering any DevOps question, structure it like this:

✓ 1) What is it? (definition) **✓ 2) Why do we use it?** (purpose) **✓ 3) How does it work?** (flow) **✓ 4) How do you troubleshoot?** (commands + steps) **✓**5) Production best practices**