# AWS, Terraform, Kubernetes, and Observability Q&A Summary

## 1. What is the difference between Internet Gateway and NAT Gateway?

- Internet Gateway (IGW): Enables resources in a public subnet to access the internet directly.  
- NAT Gateway: Allows instances in a private subnet to access the internet without exposing them to inbound traffic.  
Use IGW for public-facing services; use NAT Gateway for backend services needing outbound internet.

## 2. How do you decide between VPC Peering and Transit Gateway?

- VPC Peering: Direct connection between two VPCs, suitable for small-scale communication.  
- Transit Gateway: Central hub for connecting multiple VPCs and on-prem networks.  
Choose Transit Gateway for scalability and centralized routing.

## 3. Why is Transit Gateway more costly than VPC Peering?

- Transit Gateway offers managed routing, high throughput, and centralized control.  
- It incurs per-hour and per-GB charges.  
VPC Peering is cheaper but lacks scalability and routing simplicity.

## 4. How do you use API Gateway and when would you choose public vs private APIs?

- Public API: Accessible over the internet.  
- Private API: Accessible only within your VPC via VPC endpoints.  
Use Public for external clients; Private for internal microservices.

## 5. Explain the difference between S3, DynamoDB, and RDS.

- S3: Object storage for files, backups, logs.  
- DynamoDB: NoSQL database, highly scalable and low latency.  
- RDS: Managed relational database (e.g., MySQL, PostgreSQL).

## 6. What are the key benefits of cloud migration?

- Scalability  
- Cost optimization  
- High availability  
- Security  
- Faster innovation

## 7. How do you ensure security in AWS?

- Use IAM roles and policies  
- Enable MFA  
- Encrypt data (S3, EBS, RDS)  
- Use Security Groups and NACLs  
- Monitor with CloudTrail, GuardDuty, Inspector

## 8. How do you manage large migrations in Terraform?

- Use modular structure  
- Break into workspaces or environments  
- Use remote state (e.g., S3 + DynamoDB)  
- Apply state locking  
- Use CI/CD pipelines

## 9. What are the limitations of auto-generated Terraform code?

- Lacks modularity  
- May miss best practices (e.g., tagging, versioning)  
- Hard to maintain  
- Doesn’t handle complex dependencies well

## 10. How do you secure secrets in Terraform?

- Use SSM Parameter Store with encryption  
- Use AWS Secrets Manager  
- Use Vault by HashiCorp  
- Use Lowkey for local secret management  
Avoid hardcoding secrets in .tf files or state files.

## 11. What are the best practices missing in a simple Terraform S3 bucket config?

- Versioning  
- Encryption (SSE-S3 or SSE-KMS)  
- Bucket policies  
- Access logging  
- Lifecycle rules  
- Tags

## 12. What are the common pod errors in Kubernetes?

- CrashLoopBackOff: App crashes repeatedly  
- OOMKilled: Pod exceeds memory limit  
- ImagePullBackOff: Image not found or auth issue  
- ErrImagePull: Image pull failed  
- CreateContainerConfigError: Misconfigured container spec

## 13. How do you size an EKS cluster for a project?

- Estimate CPU/memory per pod  
- Estimate pod count  
- Consider HA requirements  
- Use Cluster Autoscaler  
- Consider node types (e.g., spot vs on-demand)

## 14. What happens if a microservice pod fails due to port already in use?

- Container fails to start  
- Fix by changing port, ensuring no other process binds to it  
- Use readiness/liveness probes to detect issues

## 15. How do you enable autoscaling in Kubernetes?

- Horizontal Pod Autoscaler (HPA): Scales pods based on CPU/memory  
- Vertical Pod Autoscaler (VPA): Adjusts pod resource requests  
- Cluster Autoscaler: Adds/removes nodes

## 16. How do you create Grafana queries for metrics like CPU, memory, error rate?

- CPU: rate(container\_cpu\_usage\_seconds\_total[5m])  
- Memory: container\_memory\_usage\_bytes  
- Error Rate: rate(http\_requests\_total{status=~"5.."}[5m])  
Use PromQL for Prometheus data sources.

## 17. What steps do you take when Grafana dashboards load slowly?

- Reduce panel queries  
- Use time range filters  
- Optimize Prometheus retention  
- Use recording rules  
- Cache frequently used queries

## 18. What are Beats in the ELK stack and their use cases?

- Filebeat: Logs  
- Metricbeat: Metrics  
- Packetbeat: Network traffic  
- Auditbeat: Security events  
They ship data to Logstash or Elasticsearch.

## 19. How do you diagnose findings in AWS Inspector?

- Review severity levels  
- Check affected resources  
- Use recommendations for remediation  
- Integrate with Security Hub for centralized view

# Kubernetes, Linux, and Networking Interview Questions

## 4. Describe your experience with CI/CD pipelines.

CI/CD (Continuous Integration/Continuous Deployment) pipelines automate the software delivery process.  
- Tools used: Jenkins, GitHub Actions, GitLab CI, CircleCI, ArgoCD.  
- Stages:  
 - CI: Code is pushed → automated tests run → build artifacts are created.  
 - CD: Artifacts are deployed to staging/production environments.  
- Practices:  
 - Automated testing (unit, integration).  
 - Docker image builds and vulnerability scanning.  
 - Kubernetes manifests or Helm charts deployment.  
 - Rollbacks and blue-green/canary deployments.

## 5. How do you expose a Kubernetes application to external traffic?

You can expose it using:  
- Service of type LoadBalancer: Creates an external IP via cloud provider.  
- Ingress Controller: Manages HTTP/S routing with domain names and TLS.  
- NodePort: Opens a port on each node (less common in production).  
- ExternalDNS: Automates DNS record creation for services/ingress.

## 6. What is the difference between Deployment and StatefulSet in Kubernetes?

Feature | Deployment | StatefulSet  
--------------------|--------------------------------------|--------------------------------------  
Pod Identity | Stateless, interchangeable pods | Each pod has a stable identity  
Storage | Shared or ephemeral | PersistentVolume per pod  
Use Case | Web apps, APIs | Databases, Kafka, Zookeeper  
Pod Naming | Random | Predictable (e.g., pod-0, pod-1)

## 7. What is a ConfigMap, and how is it different from a Secret?

- ConfigMap: Stores non-sensitive config data (e.g., app settings).  
- Secret: Stores sensitive data (e.g., passwords, tokens) in base64.  
- Difference:  
 - Secrets are encrypted at rest (depending on setup).  
 - ConfigMaps are plain text and not meant for sensitive info.

## 8. What is the purpose of a NAT Gateway?

A NAT Gateway allows instances in a private subnet to access the internet (e.g., for updates or API calls) without exposing them to incoming traffic from the internet.

## 9. How do you check network connectivity between two servers?

- Ping: ping <IP or hostname>  
- Traceroute: traceroute <destination> (Linux) or tracert (Windows)  
- Telnet/Netcat: telnet <host> <port> or nc -zv <host> <port>  
- Curl: curl <URL> for HTTP connectivity  
- SSH: ssh <user>@<host> if applicable

## 10. How do you check running processes in Linux?

- ps aux – Lists all running processes.  
- top or htop – Interactive view of processes.  
- pgrep <process\_name> – Finds processes by name.

## 11. What command would you use to find files larger than 100MB?

find /path/to/search -type f -size +100M  
You can add -exec ls -lh {} \; to show file sizes.

DevOps & Kubernetes Interview Questions and Answers

# 1. You have an application in Account A that needs to access an S3 bucket in Account B. How would you configure this?

Steps:  
In Account B (S3 Bucket Owner):  
- Add a bucket policy to allow access from Account A’s IAM role:  
{  
 "Version": "2012-10-17",  
 "Statement": [  
 {  
 "Effect": "Allow",  
 "Principal": {  
 "AWS": "arn:aws:iam::<AccountA-ID>:role/<RoleName>"  
 },  
 "Action": "s3:GetObject",  
 "Resource": "arn:aws:s3:::<bucket-name>/\*"  
 }  
 ]  
}  
  
In Account A:  
- Attach an IAM role to the application (EC2, Lambda, etc.) with permissions to access the S3 bucket in Account B.

# 2. Your EC2 instance in a private subnet needs to download packages without a NAT Gateway. What alternatives exist?

Alternatives:  
- VPC Endpoints (Interface or Gateway): Use Gateway VPC Endpoint for S3 and DynamoDB; Interface Endpoints for services like ECR, SSM.  
- S3 Proxy or Mirror: Host a proxy in a public subnet that fetches packages.  
- Use AWS Systems Manager (SSM): If SSM Agent is installed and IAM permissions are set, use Session Manager to run commands or transfer files.

# 3. How would you set up geolocation-based routing using AWS services?

Use Route 53:  
- Create a Geolocation Routing Policy in Route 53.  
- Define DNS records based on user location (continent, country, or state).  
- Example: Users from India → in.example.com, US → us.example.com.

# 4. Write a Dockerfile for a Node.js application with multi-stage builds.

Dockerfile:  
# Stage 1: Build  
FROM node:18-alpine AS builder  
WORKDIR /app  
COPY package\*.json ./  
RUN npm install  
COPY . .  
RUN npm run build  
  
# Stage 2: Production  
FROM node:18-alpine  
WORKDIR /app  
COPY --from=builder /app/dist ./dist  
COPY --from=builder /app/package\*.json ./  
RUN npm install --only=production  
CMD ["node", "dist/index.js"]

# 5. What's the difference between COPY and ADD commands in Dockerfile?

COPY vs ADD:  
- COPY: Basic file copy, preferred for clarity.  
- ADD: Supports remote URLs and auto-extracts tar.gz files.  
Use COPY unless you need ADD features.

# 6. How do you debug a container that has exited?

Steps:  
- Check logs: docker logs <container-id>  
- Start with interactive shell: docker run -it --entrypoint /bin/sh <image>  
- Inspect container: docker inspect <container-id>

# 7. How would you handle secrets in a Docker container for a PHP application connecting to MySQL?

Best Practices:  
- Use Docker Secrets (Swarm) or Kubernetes Secrets.  
- Avoid hardcoding in Dockerfile or ENV.  
- Use AWS Secrets Manager or HashiCorp Vault.  
- Inject secrets at runtime using entrypoint scripts or volume mounts.

# 8. How would you implement blue-green deployment in Kubernetes?

Steps:  
1. Deploy two versions: app-blue and app-green.  
2. Use a Service to point to one version (e.g., app-blue).  
3. Deploy app-green and test it.  
4. Switch the Service selector to app-green.  
5. Optionally delete or keep app-blue for rollback.  
Tools: Argo Rollouts, Flagger, or manual with kubectl.

# 9. How do you implement network policies to restrict pod-to-pod communication in Kubernetes?

Example:  
apiVersion: networking.k8s.io/v1  
kind: NetworkPolicy  
metadata:  
 name: allow-from-frontend  
spec:  
 podSelector: {}  
 ingress:  
 - from:  
 - namespaceSelector:  
 matchLabels:  
 name: frontend  
Requires a CNI plugin that supports network policies (e.g., Calico, Cilium).

# 10. Critical production Kubernetes cluster is experiencing multiple issues:

a. Pods stuck in ImagePullBackOff:  
- Check image name and tag.  
- Check imagePullSecrets.  
- Check if DockerHub rate limits are hit.  
- Use kubectl describe pod <pod> for detailed error.  
  
b. Pods being evicted:  
- Likely due to resource pressure (CPU, memory, disk).  
- Check node status: kubectl describe node <node>  
- Use kubectl top nodes to monitor usage.  
- Consider adding nodes, tuning resource requests/limits, using taints/tolerations.  
  
c. Users reporting 503 errors:  
- Check if service endpoints are healthy: kubectl get endpoints <service>  
- Check readiness/liveness probes.  
- Check ingress controller logs (e.g., NGINX, ALB).  
- Check for DNS resolution issues.

Terraform and DevOps Interview Questions - 3 Years Experience

# 11. What troubleshooting steps will you follow, and how to avoid this in the future?

Troubleshooting Steps:

- Identify the symptoms using logs and monitoring tools.

- Isolate the issue to determine the root cause.

- Reproduce the issue in a lower environment.

- Review recent changes in code or infrastructure.

- Use observability tools like Prometheus, Grafana, or CloudWatch.

- Apply the fix, validate, and monitor post-resolution.

Prevention:

- Implement CI/CD with automated testing.

- Use Infrastructure as Code (IaC) with version control.

- Set up alerts and anomaly detection.

- Conduct postmortems and document learnings.

# 12. How do you handle Terraform state file corruption?

- Identify the corruption through error messages.

- Restore from backup if using remote backend like S3 with versioning.

- Attempt manual recovery only if the corruption is minor.

- Use terraform state commands to clean up invalid resources.

Prevention:

- Use remote backends with locking mechanisms.

- Enable versioning on the backend.

- Avoid manual edits to the state file.

- Use terraform plan and apply cautiously.

# 13. You need to import an existing AWS VPC into Terraform. What are the steps?

- Write a Terraform resource block for the VPC.

- Use terraform import command: terraform import aws\_vpc.my\_vpc vpc-xxxxxxxx

- Run terraform plan to compare state and configuration.

- Update the resource block to match actual configuration.

- Run terraform apply to sync the state.

# 14. How do you manage secrets in Terraform without hardcoding them?

- Use Terraform variables and pass secrets via environment variables or tfvars files.

- Use HashiCorp Vault or AWS Secrets Manager with Terraform providers.

- Use SOPS with Terraform for encrypted secrets in Git.

- Use backend-specific features like encrypted SSM parameters.

Best Practices:

- Never hardcode secrets.

- Use IAM roles and policies for secure access.

- Rotate secrets regularly.

# 15. How would you implement cross-account resource provisioning using Terraform?

- Set up AWS provider aliases for each account.

- Use assume role for cross-account access.

- Reference resources using the correct provider alias.

- Use remote state data sources to share outputs between accounts.

# 16. An S3 bucket was created via Terraform, but someone manually added a policy. How do you handle this drift?

- Run terraform plan to detect the drift.

- Decide whether to accept or override the manual change.

- Update Terraform code if the manual change is valid.

- Let Terraform override if the change is not valid.

Prevention:

- Use resource lifecycle rules like prevent\_destroy.

- Implement policy-as-code and restrict manual changes via IAM.

- Use drift detection tools like Terraform Cloud or Atlantis.

# 17. Write a Python script to backup all files older than 30 days from a directory.

Python Script:

import os  
import shutil  
import time  
  
SOURCE\_DIR = "/path/to/source"  
BACKUP\_DIR = "/path/to/backup"  
DAYS = 30  
now = time.time()  
  
for filename in os.listdir(SOURCE\_DIR):  
 filepath = os.path.join(SOURCE\_DIR, filename)  
 if os.path.isfile(filepath):  
 if os.stat(filepath).st\_mtime < now - DAYS \* 86400:  
 shutil.copy2(filepath, BACKUP\_DIR)  
 print(f"Backed up: {filename}")

# 18. Your company's cloud costs are increasing rapidly. How would you approach cost optimization without impacting performance?

- Analyze usage with AWS Cost Explorer, Azure Advisor, or GCP Billing.

- Identify idle resources and remove them.

- Right-size instances and use autoscaling and spot instances.

- Use savings plans or reserved instances for predictable workloads.

- Optimize storage with lifecycle policies and archive infrequent data.

- Review data transfer costs and minimize cross-region traffic.

- Implement tagging for cost attribution and accountability.

- Use monitoring tools to correlate cost with performance.

1. What will you do if a Jenkins pipeline fails?  
Answer: Check Jenkins logs → Identify stage of failure → Fix configuration/code issue → Re-run the pipeline. If infra-related, verify Terraform or Kubernetes changes before redeploying.  
2. How do you handle a failed deployment in Kubernetes?  
Answer: Use kubectl describe pod and kubectl logs to check errors → If critical, rollback with kubectl rollout undo deployment <name> → Fix and redeploy.  
3. How do you manage Terraform state in a team?  
Answer: Store state in remote backend (Azure Blob, GCS bucket, Terraform Cloud) → Enable state locking to prevent conflicts → Use versioning for recovery.  
4. You changed a Terraform variable and want to see the impact before applying. What do you do?  
Answer: Run terraform plan to preview changes before applying.  
5. How do you ensure zero downtime deployment in Kubernetes?  
Answer: Use RollingUpdate strategy in deployments, configure readiness probes, and keep replicas running until new pods are healthy.  
6. How do you roll back in Jenkins if a deployment causes issues?  
Answer: Keep artifact versioning → Redeploy the last stable build from Jenkins → Or trigger rollback pipeline.  
7. What if Terraform state file gets corrupted or deleted?  
Answer: Restore from remote backend version history (e.g., GCS versioning, Azure Blob snapshots) → If not available, use terraform import to rebuild state.  
8. How do you secure secrets in pipelines?  
Answer: Use Jenkins credentials manager, Vault, or cloud secret managers (GCP Secret Manager, Azure Key Vault) instead of storing secrets in code.  
9. How do you monitor Kubernetes clusters?  
Answer: Use Prometheus + Grafana for metrics, ELK/EFK stack for logs, and Kubernetes liveness/readiness probes for pod health.  
10. What will you do if a pod is stuck in CrashLoopBackOff?  
Answer: Run kubectl describe pod and kubectl logs → Check startup script, image, or config issue → Fix error → Redeploy.  
11. How do you manage infrastructure across multiple environments (Dev/QA/Prod) using Terraform?  
Answer: Use workspaces or separate state files with environment-specific variables.  
12. How do you optimize CI/CD pipelines in Jenkins?  
Answer: Use parallel stages, caching (e.g., Docker layers, Maven cache), and parameterized builds to save time.  
13. How do you perform blue-green deployment in Kubernetes?  
Answer: Run two environments (Blue = current, Green = new) → Route traffic to Green only after successful validation → Rollback to Blue if issues occur.  
14. How do you troubleshoot high CPU usage on a Linux server?  
Answer: Use top, htop, vmstat, and iostat → Identify process → Kill/fix process → Scale infra if required.  
15. How do you automate infrastructure scaling in cloud?  
Answer: Configure Auto Scaling Groups in GCP (Instance Groups) or Azure (VM Scale Sets) → Integrate with Terraform for automation.