Dev-Ops Comprehensive Study Outline

1. AWS Fundamentals

Identity and Access Management (IAM)

Overview of IAM concepts: users, groups, roles, policies, and permissions.

Key components and their interrelation in access control.

Guidelines for implementing IAM security best practices.

User operations: create, manage, and delete IAM users; configure MFA; rotate passwords; handle access keys securely.

Group management: create groups, attach policies for centralized control.

Roles: create and assign roles for service access and cross-account operations; understand trust and permission policies.

Craft custom IAM policies in JSON, using effect, action, resource, and condition.

Link IAM policies to AWS services like SNS, SQS for access control.

Amazon S3

Principles of S3 bucket creation: naming rules, region selection, and endpoints.

Storage class options (Standard, Intelligent-Tiering, Glacier, etc.) and when to use each.

Upload/download object handling.

Access control: bucket policies, IAM policies, object ACLs, and pre-signed URLs.

Enabling versioning, lifecycle management, and cross-region replication for redundancy.

Amazon EC2

Understanding EC2 instance types and virtualization models (HVM, PV).

Steps to launch and connect to EC2.

Instance storage types: ephemeral vs. EBS; performance differences between SSD and HDD volumes.

Assigning and configuring security groups for traffic control.

Working with Elastic IPs and subnets in VPC.

Accessing instance metadata and using UserData for automation.

AWS Networking

VPC concepts: subnets, route tables, NACLs, and security groups.

Networking components: Elastic IPs, ENIs, IGWs, NAT Gateways, VGWs.

Security measures: AWS Shield, Firewall Manager, WAF.

Route 53 basics: routing policies, DNS resolution.

Elastic Load Balancers: CLB, ALB, NLB configuration and health checks.

2. Linux Essentials

Core commands: mkdir, cd, pwd, ls, cat, touch, cp, mv, df, du, head, tail, diff, locate, find, chmod, whoami, chown, grep, zip, unzip, kill, wget, history, alias, hostname, useradd, userdel, uname, free, id, ping, operators, scp, telnet, ssh, curl, nslookup, vi/vim, dpkg, netstat, ps, top, tar, groupadd, rsync.

Linux directory layout and important system paths.

Identifying and checking logs for services or processes.

Pipes and redirection for output filtering.

Commands to inspect OS version, CPU/memory/disk usage.

Differences between Ubuntu and CentOS/RHEL distributions.

Listing installed packages and verifying specific package versions.

Setting up and managing firewalls; comparison of firewall tools on different distros.

Using iptables and checking port status.

Optional productivity setup: Zsh with OhMyZsh + Powerlevel10k.

3. Networking Principles

Network classifications and models.

TCP vs UDP characteristics and typical use cases.

Relationship between TCP and HTTP.

Common ports in TCP/UDP.

How DNS resolves a domain.

Subnetting and CIDR notation.

Detailed sequence of events when accessing a website (browser → DNS → server).

Public vs private keys; symmetric vs asymmetric cryptography.

HTTPS encryption mechanisms.

Security aspects of SSH and how public keys grant access to servers.

Managing SSH keys for user access.

Using tcpdump to check packet flow to specific ports.

4. Git & GitHub

Distinction between Git (VCS) and GitHub (remote hosting).

Create GitHub account using <name>-ksolves format.

Setting up Git for first use.

Staging and committing changes.

Creating and switching branches.

Pulling, pushing, and merging branches.

Git stash for temporary changes.

Resetting commits (soft/hard).

Merging local repositories.

5. Docker & Docker Compose

Purpose of Docker and the problem it solves in development and deployment.

Core commands for managing images and containers.

Docker client vs Docker server (daemon).

Networking in Docker: bridge, host, none, and custom networks.

Volumes: persistence and data sharing between containers.

Using docker save and docker load for image export/import.

Scenarios:

MySQL server and client running in separate containers, with client connecting to server to create a database and table.

Nginx container exposed to host system on a specific port.

Image registries and available image types.

6. Application Development with Containers

Create a basic to-do application using HTML/React for frontend, Node.js for backend, MySQL for DB.

Each service runs in its own container.

Configuration via environment variables (no hardcoding).

Inter-service communication using container service names instead of localhost.

Serve frontend via Nginx container.

7. Kubernetes

Core Concepts

Purpose, benefits (scaling, self-healing, automation).

Cluster components:

Control Plane: API Server, Scheduler, Controller Manager, etcd.

Worker Nodes: Kubelet, Kube-proxy, container runtime.

Terminology: clusters, nodes, pods, namespaces, services.

Setup & Configuration

Using Kind for local clusters.

Cluster setup on AWS with kubeadm.

kubeconfig usage and context switching.

Pods & Workloads

Pod lifecycle, multi-container pods, init containers.

Running pods with kubectl run.

Deployments: create, update, rollback, scale.

ReplicaSets and their link to deployments and pods.

DaemonSets, StatefulSets, Jobs, CronJobs.

Networking

ClusterIP, NodePort, LoadBalancer, ExternalName services.

Ingress controllers (e.g., nginx ingress).

How DNS works inside Kubernetes.

Storage

Volumes: emptyDir, hostPath, ConfigMap, Secret.

Persistent Volumes (PVs) and Persistent Volume Claims (PVCs).

Using ConfigMaps and Secrets for configs.

Observability

Logging from pods.

Liveness and readiness probes.

Debugging tools: kubectl logs, kubectl describe, kubectl exec.

Helm

Purpose of Helm and chart management.

Creating charts, using repos, upgrading and rolling back releases.

Project

Deploy the earlier to-do app in Kubernetes using Helm charts.

8. Ansible

Architecture: control node and managed nodes.

Inventory files and defining hosts/groups.

Ad-hoc commands (e.g., ansible all -m ping).

Playbooks: YAML-based task automation.

Common modules: copy, file, service, command.

Variables, facts, and templating with Jinja2.

Roles for structuring reusable automation.

Handlers for conditional actions.

Error handling and debugging methods.

9. CI/CD

Jenkins overview and role in automation.

Installation, setup, and navigating the dashboard.

Freestyle jobs, parameterized jobs, and pipelines.

Integrating Jenkins with Git/GitHub using webhooks.

Common plugins: Git, Pipeline, Credentials.

Incorporating testing and code quality checks with tools like SonarQube.

Securing Jenkins with RBAC and HTTPS.

Explore GitHub Actions and GitLab CI pipelines.

10. Terraform

Infrastructure as Code principles and how Terraform solves infra provisioning challenges.

Providers and provisioners.

State file purpose and usage.

Key commands.

Project: Provision AWS environment with VPC, subnets, internet/NAT gateways, security groups, EC2 with Nginx behind a load balancer. Output load balancer DNS on completion.

11. Monitoring with Prometheus & Grafana

Purpose of monitoring and alerting.

Overview of other monitoring tools and trade-offs.

Setting up Prometheus and Grafana to track EC2 instance metrics.

Alerts for high memory (>90%) or storage usage (>90%).

Custom dashboards for visualizing instance performance.