Docker Compose & httpd – Architect-Level Interview Q&A;

L1 → L4 (Junior → Senior → Architect Level)

Based on real-world static website hosting using Docker Compose

■ L1 – Junior DevOps (Fundamentals)

1. What was the goal of this task?
Answer: To host static website content using an Apache httpd container deployed via Docker Compose, exposing it on host port 3001 and mounting content from the host filesystem.

2. Which Docker image did you use?
Answer: I used the official httpd:latest image.

3. Why use the httpd image?
Answer: It is the official Apache HTTP Server image, optimized for serving static content and widely used in production.

4. What port does Apache listen on inside the container?
Answer: Apache listens on port 80 by default.

5. How did you expose the application?
Answer: By mapping host port 3001 to container port 80 using Docker Compose.

6. What is Docker Compose?
Answer: Docker Compose is a tool used to define and manage multi-container Docker applications using a YAML configuration file.

7. Why was Docker Compose preferred over docker run?
Answer: Docker Compose provides declarative configuration, easier reusability, and consistent deployments.

■ L2 – Mid-Level DevOps (Operations & Troubleshooting)

8. Why was volume mounting required in this task?
Answer: To serve static website files directly from the host without modifying the container image.

9. Which host directory was mounted?
Answer: /opt/sysops was mounted to /usr/local/apache2/htdocs.

10. How did you verify volume mounting?
Answer: Using docker inspect httpd | grep -A5 Mounts.

11. How did you verify the container was running?
Answer: Using docker ps.

12. How did you confirm the website was accessible?
Answer: Using curl http://localhost:3001.

13. What happens if the host directory is empty?
Answer: Apache will start, but no content will be served.

14. What default network does Docker Compose create?
Answer: A bridge network named <project>_default.

15. What could cause a 403 Forbidden error?
Answer: Incorrect file permissions on the mounted host directory.

■ L3 – Senior DevOps (Production, Security & Scaling)

16. Why should static content not be baked into the image?
Answer: It improves maintainability, avoids image rebuilds, and separates application code from infrastructure.

17. What security improvements would you apply?
Answer: Use read-only volume mounts, restrict exposed ports, run containers as non-root, and apply firewall rules.

18. How would you enable automatic container restart?
Answer: By adding restart: unless-stopped to the Docker Compose file.

19. How would you scale this service?
Answer: Using Docker Compose scaling for small setups or Kubernetes for production-scale deployments.

20. How would you enable HTTPS?
Answer: By placing a reverse proxy (Nginx/Traefik) in front of Apache or configuring SSL inside Apache.

21. What metrics would you monitor?
Answer: HTTP response codes, CPU and memory usage, disk usage of mounted volumes, and container uptime.

■ L4 – Architect Level (Design, HA & Cloud-Native Thinking)

22. How would you design this setup for high availability?
Answer: Deploy multiple Apache containers behind a load balancer with shared storage or object storage for static content.

23. Would you use Docker Compose in production?
Answer: For small or internal workloads yes, but for enterprise-scale production, Kubernetes is preferred.

24. How would you redesign this using Kubernetes?
Answer: Use a Deployment for Apache pods, a Service for exposure, PersistentVolumes or object storage for content, and Ingress for routing.

25. How would you handle zero-downtime deployments?
Answer: By using rolling updates, multiple replicas, and load balancers.

26. How would you externalize configuration?
Answer: Using environment variables, ConfigMaps, or external configuration services.

27. How would you secure static content access?
Answer: Network-level restrictions, authentication at the proxy layer, and TLS encryption.

28. How would you log and audit access?
Answer: Centralized logging using ELK/EFK stack and shipping access logs from containers.

29. What cloud services could replace this architecture?
Answer: AWS S3 + CloudFront, Azure Blob Storage + CDN, or GCP Cloud Storage + CDN.

30. When would containers be preferred over managed services?
Answer: When custom configuration, legacy dependencies, or on-prem deployments are required.

Final Architect-Level Interview Summary

> *“I designed and deployed a containerized Apache httpd service using Docker Compose, exposed it via controlled port mapping, mounted static content from the host, and validated access. I can further evolve this architecture to a secure, scalable, and highly available design using load balancers, Kubernetes, and cloud-native services.”*