RealTime Handson

10 DevSecOps AZURE CLOUD INTERVIEW QUESTIONS

Shruthi Chikkela
learnwithshruthi

1. How can I deploy multiple applications at the same time in Azure?

Answer: In Azure, you can deploy multiple applications simultaneously using tools like **Azure DevOps** pipelines or **Azure Resource Manager (ARM) templates**. You can:

- Create a **multi-stage pipeline** in Azure DevOps where each application is deployed to different environments concurrently or sequentially.
- Leverage **Azure Kubernetes Service (AKS)** to deploy containerised applications in parallel using Helm charts.
- Use **Terraform** to provision and deploy infrastructure for all applications simultaneously.

2. What exactly are Continuous Integration, Continuous Delivery, and Continuous Deployment in Azure DevOps?

Answer:

- **Continuous Integration (CI):** In Azure DevOps, CI involves automatically building and testing code each time changes are committed to a repository. It ensures that code is frequently integrated and issues are detected early.
- **Continuous Delivery (CD):** This process in Azure DevOps ensures that the application is always ready to be deployed to production but may require manual approval for deployment to live environments.
- **Continuous Deployment:** Extending Continuous Delivery, Continuous Deployment in Azure automatically deploys every change that passes tests to production, ensuring zero manual intervention after successful tests.

3. What is the difference between Azure Load Balancer (ALB) and Azure Network Load Balancer (NLB)?

Answer:

- **Azure Load Balancer (ALB):** This is a Layer 7 load balancer in Azure that can route HTTP/HTTPS traffic based on URL paths, making it suitable for web applications. It supports features like SSL termination and application fire-walling.

- **Azure Network Load Balancer (NLB):** This is a Layer 4 load balancer that distributes TCP/UDP traffic across multiple servers. It offers high throughput and low latency and is best suited for non-HTTP traffic, such as game servers or VoIP applications.

4. How can you integrate Azure Blob Storage to an Azure Virtual Machine (VM)?

Answer: To integrate **Azure Blob Storage** with a VM:

- Create an **Azure Blob Storage account** in the Azure portal.

- Use **Azure CLI** or **PowerShell** to mount the Blob storage to the VM as a network drive.

- Alternatively, you can use **Azure SDKs** (like Python, .NET) or **Azure Storage Explorer** to interact with the Blob storage from within the VM.

- For enhanced security, use **Azure Managed Identity** to grant the VM access to Blob Storage.

5. Can you explain the process of how an end-user request reaches the application in Azure? What are the stages involved?

Answer: In Azure, the stages include:

1. **DNS Resolution:** The user's request is routed through **Azure DNS** to resolve the domain name to an IP address.
2. **Traffic Routing:** The traffic is directed through an **Azure Load Balancer** (either ALB or NLB) to ensure high availability and fault tolerance.
3. **Web Application Firewall (WAF):** If configured, **Azure WAF** inspects the incoming traffic for malicious requests.
4. **Web Apps / AKS:** The request reaches the application hosted on **Azure App Service** or **Azure Kubernetes Service (AKS)**.
5. **Backend Services:** The application connects to backend services, such as databases hosted on **Azure SQL Database** or **Azure Cosmos DB**, and fetches the required data.

6. What is the difference between Observability and Monitoring in Azure?

Answer:

- **Monitoring:** In Azure, monitoring involves collecting metrics and logs to ensure that systems and applications are performing well. Services like **Azure Monitor** help you track performance and detect issues in real-time.
- **Observability:** Observability goes beyond monitoring by allowing you to understand the internal state of a system through logs, metrics, and traces. It involves deeper insights into the "why" behind problems, often using **Azure Application Insights** to track performance and detect anomalies.

7. How can you secure a CI/CD pipeline in Azure DevOps?

Answer: To secure a CI/CD pipeline in Azure DevOps:

- Use **Azure Key Vault** to securely manage secrets and certificates.
- Implement **Role-Based Access Control (RBAC)** in Azure DevOps to restrict who can modify or trigger pipelines.
- Enable **multi-factor authentication (MFA)** to add an additional layer of security.
- Use **Managed Identity** for secure access to Azure resources during pipeline execution.
- Set up **branch protection policies** to enforce code review and prevent unauthorised changes.
- Add **security tests** (e.g., SAST, DAST) in the pipeline to detect vulnerabilities early.

8. Do you have hands-on experience with Terraform workspaces, and what does it mean in Azure?

Answer: Terraform Workspaces in Azure are used to manage multiple environments (e.g., dev, prod) using the same configuration code.

Each workspace has its own state, making it easier to maintain different environments without interference.

- For example, you can have a workspace for **development**, another for **production**, and each will have its own state file.
- This helps you keep environment-specific configurations separate while reusing the same infrastructure code, improving workflow efficiency.

9. How do you configure Azure Blob Storage as a backend for Terraform and implement state locking?

Answer: In **Azure**, you can configure **Azure Blob Storage** as a backend for Terraform state management and implement state locking as follows:

- Create an **Azure Storage Account** and a **Blob container** to store the Terraform state file.
- Enable **state locking** using **Azure Blob Lease** to prevent multiple people from modifying the state concurrently.
- In your `terraform` configuration, configure the backend as:

```
terraform {
  backend "azurerm" {
    resource_group_name   = "myResourceGroup"
    storage_account_name  = "myStorageAccount"
    container_name        = "myContainer"
    key                   = "terraform.tfstate"
  }
}
```

10. What is a Helm Chart in Azure Kubernetes Service (AKS), and can you explain your experience with Helm charts?

Answer: A **Helm Chart** is a package that contains pre-configured Kubernetes resources like deployments, services, and ingress controllers, which simplifies deploying applications to a Kubernetes environment.

- In **Azure Kubernetes Service (AKS)**, Helm charts allow you to deploy, manage, and scale applications efficiently.
- I have experience using **Helm** to deploy containerized applications in AKS, where I utilized charts to manage versions, handle configurations, and ensure smooth rollbacks. Helm enables the reuse of templates, making deployment easier and faster across different environments.

For any RealTime Handson Projects

And for more tips like this

Follow

Shruthi Chikkela

Like & ReShare

learnwithshruthi