DUMPSHQ

Free Questions for 1Z0-1084-23 by dumpshq

Shared by Barnes on 15-04-2024

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

Your organization has mandated that all deployed container images used for microservices must be signed by a specified master encryption key (MEK). You have appropriately signed the container images as part of your build process, but must now ensure that they are automatically verified when they are deployed to Oracle Cloud Infrastructure (OCI) Container Engine for Kubemetes (OKE) clusters. Which option should be used to mandate image verification when deploying to OKE clusters, assuming that MEK is already stored in an available OCI Vault? (Choose the best answer.)

Options:

A- Enable image verification policies separately for each Kubemetes pod deployment because this is enforced at the pod level.

B- Enable image verification policies separately for each node pool within each OKE cluster because this is enforced at the node pool level.

C- Enable image verification policies separately for each OKE cluster because this is enforced at the cluster level.
(Correct)

D- Enable Image verification policies for your OKE service control plane which will enforce this for all OKE clusters.

Answer:

C

Explanation:

To mandate image verification when deploying container images to Oracle Cloud Infrastructure (OCI) Container Engine for Kubernetes (OKE) clusters, you should enable image verification policies separately for each OKE cluster. This is enforced at the cluster level. Enabling image verification policies at the cluster level ensures that all container images deployed to the OKE cluster are automatically verified against the specified master encryption key (MEK). This helps maintain the security and integrity of the deployed microservices by ensuring that only signed and trusted container images are used. Enabling image verification policies at the cluster level allows for consistent and centralized enforcement of the verification process across all nodes and node pools within the cluster. It provides a standardized approach to image verification for the entire cluster, simplifying management and ensuring compliance with the organization's mandate. Enabling image verification policies separately for each node pool or at the pod level would introduce complexity and potential inconsistencies in the verification process. Therefore, enforcing image verification at the cluster level is the recommended approach.

Question 2

Question Type: MultipleChoice

You are using Oracle Cloud Infrastructure (OCI) Resource Manager to manage your infrastructure lifecycle and wish to receive an email each time a Terraform action begins. How should you use the OCI Events service to do this without writing any code?

Options:

A- Create a rule in OCI Events service matching the 'Resource Manager Stack - Update' condition. Then select 'Action Type: Email' and provide the destination email address.

B- Create an OCI Notification topic and email subscription with the destination email address. Then create an OCI Events rule matching 'Resource Manager Job - Create' condition, and select the notification topic for the corresponding action.

C- Create an OCI Email Delivery configuration with the destination email address. Then create an OCI Events rule matching 'Resource Manager Job - Create' condition, and select the email configuration for the corresponding action.

D- Create an OCI Notifications topic and email subscription with the destination email address. Then create an OCI Events rule matching 'Resource Manager Stack - Update' condition, and select the notification topic for the corresponding action.

Answer:

B

Explanation:

The correct approach to receive an email each time a Terraform action begins in Oracle Cloud Infrastructure (OCI) Resource Manager without writing any code is as follows: Create an OCI Notification topic and email subscription with the destination email address. This will define the email delivery configuration. Create an OCI Events rule that matches the 'Resource Manager Job - Create' condition. This rule will be triggered when a Resource Manager job is created. In the OCI Events rule, select the notification topic that was created in step 1 as the action for the corresponding event. This will ensure that the notification is sent to the specified email address. By following these steps, you can configure the OCI Events service to send an email notification whenever a Resource Manager job is created in OCI Resource Manager.

Question 3

Question Type: MultipleChoice

You have a containerized application that requires access to an Autonomous Transaction Processing (ATP) Database. Which option is NOT valid when the container is deployed in an OKE cluster? (Choose the best answer.)

Options:

A- Use Kubernetes secrets to configure environment variables on the container with ATP instance OCID, and OCI API credentials. Then use the CreateConnection API endpoint from the service runtime.

B- Install the Oracle Cloud Infrastructure Service Broker on the Kubernetes cluster and deploy ServiceInstance and ServiceBinding resources for ATP. Then use the specified binding name as a volume in the application deployment manifest.

C- Create a Kubernetes secret with contents from the instance Wallet files. Use this secret to create a volume mounted to the appropriate path in the application deployment manifest.

D- Enable Oracle REST Data Services for the required schemas and connect via HTTPS.

Answer:

B

Explanation:

The option that is not valid for connecting to an Autonomous Transaction Processing (ATP) Database from a container in Kubernetes is: Install the Oracle Cloud Infrastructure Service Broker on the Kubernetes cluster and deploy ServiceInstance and ServiceBinding resources for ATP. Then use the specified binding name as a volume in the application deployment manifest. The Oracle Cloud Infrastructure Service Broker is not used for connecting to an ATP Database from a container in Kubernetes. The Service Broker is used for provisioning and managing cloud services directly from Kubernetes. It allows you to create and manage instances of OCI services using Kubernetes resources like ServiceInstance and ServiceBinding. To connect to an ATP Database from a container in Kubernetes, you can use one of the following valid options: Enable Oracle REST Data Services for the required schemas and connect via HTTPS. This involves enabling and configuring Oracle REST Data Services (ORDS) for the schemas in the ATP Database. You can then connect to the ATP Database using RESTful endpoints provided by ORDS. Use Kubernetes secrets to configure environment variables on the container with ATP instance OCID and OCI API credentials. Then use the CreateConnection API endpoint from the service runtime. This approach involves configuring the necessary environment variables on the container to provide the ATP instance OCID and OCI API credentials. The application can then use the OCI SDK or REST API (such as the CreateConnection endpoint) to establish a connection to the ATP Database. Create a Kubernetes secret with contents from the instance Wallet files. Use this secret to create a volume mounted to the appropriate path in the application deployment manifest. This method involves creating a Kubernetes secret that contains the necessary credentials from the ATP Database's instance wallet files. The secret can then be mounted as a volume in the application deployment, allowing the application to access the required credentials for connecting to the ATP Database. Both options 1 and 3 provide valid approaches for connecting to an ATP Database from a container in Kubernetes, depending on the specific requirements and preferences of the application.

Question 4

Question Type: **MultipleChoice**

Which term describes a group formed by a master machine and a worker machine in a Kubernetes architecture?

Options:

A- Cluster

B- Node

C- Deployment

D- Container

E- Pod

Answer:

A

Explanation:

The term that describes a group formed by a master machine and a worker machine in a Kubernetes architecture is 'Cluster'. A cluster in Kubernetes consists of one or more master machines and multiple worker machines (also known as nodes). The master machine manages the overall control plane and orchestrates the deployment and management of containers on the worker nodes. The worker nodes are responsible for running the containers and executing the workloads. The cluster is the fundamental unit of organization and management in Kubernetes, providing the infrastructure and resources to run and manage containerized applications. It ensures high availability, scalability, and fault tolerance for the applications deployed within it.

Question 5

Question Type: MultipleChoice

A developer has created another version of a microservice and wants 10% of the traffic to flow towards it for testing purposes. The application is already configured using OCI (Oracle Cloud Infrastructure) Service Mesh. Which of the following steps is the right approach to achieve this goal?

Options:

A- Create a new Kubernetes deployment for the new version of the microservice and set the traffic splitting percentage to 10% in the Kubernetes service manifest.

B- Use Kubernetes HPA (Horizontal Pod Autoscaler) to scale the new version of the microservice to handle 10% of the traffic automatically.

C- Create a new entry in the routeRules field of the ingress gateway route table manifest to configure traffic splitting between the old and new versions of the microservice and set the percentage to 10%.

D- Create a new entry in the routeRules field of the virtual service route table manifest to configure traffic splitting between the old and new versions of the microservice and set the percentage to 10%.

Answer:

D

Question 6

Question Type: **MultipleChoice**

When developing microservices, each one can be developed in the language of choice. Which term describes this type of development? (Choose the best answer.)

Options:

A- Agile

B- DevOps

C- Distributed

D- Polyglot

Answer:

C

Explanation:

The term that describes developing microservices in different languages of choice is 'Polyglot.' In a polyglot architecture, each microservice is developed using the most appropriate programming language or technology stack for its specific requirements. This approach allows developers to leverage the strengths of different languages and frameworks, enabling them to use the most suitable tool for each microservice while still maintaining interoperability between services.

Question 7

Question Type: **MultipleChoice**

Which TWO are characteristics of microservices? (Choose two.)

Options:

A- Microservices communicate over lightweight APIs.

B- Microservices can be implemented in limited number of programming languages.

C- All microservices share a data store.

D- Microservices are hard to test in isolation.

E- Microservices can be independently deployed.

Answer:

A, E

Explanation:

The two characteristics of microservices are: Microservices can be independently deployed: One of the key principles of microservices architecture is the ability to independently deploy each microservice. This means that changes or updates to one microservice can be made and deployed without affecting other microservices. It allows for faster and more frequent deployments, enabling agile development and scalability. Microservices communicate over lightweight APIs: Microservices communicate with each other through lightweight APIs (Application Programming Interfaces). This enables loose coupling between microservices, as they can interact with each other using standard protocols like HTTP/REST or messaging systems like RabbitMQ or Kafka. Lightweight APIs facilitate flexibility and interoperability between microservices, making it easier to develop and maintain complex systems. The remaining statement, 'All microservices share a data store,' is not a characteristic of microservices. Microservices are designed to be autonomous and have their own data storage or database. Each microservice has its own data store, which promotes the principle of bounded contexts and avoids tight coupling between services. This allows for better scalability and independence of data management within each microservice.

20%
DISCOUNT