Exam : **1z0-1084-20**

Title : Oracle Cloud Infrastructure Developer 2020 Associate

Vendor : Oracle

Version : DEMO

NO.1 You are developing a serverless application with Oracle Functions. You have created a function in compartment named prod. When you try to invoke your function you get the following error.
Error invoking function. status: dhcpoptions.oc1.phx.aaaaaaaac... does not exist or Oracle Functions is How can you resolve this error?
A. Create a policy:
Allow function-family to use virtual-network-family in compartment prod
B. Create a policy:
Allow any-user to manage function-family and virtual-network-family in compartment prod
C. Create a policy:
Allow service FaaS to use virtual-network-family in compartment prod
D. Deleting the function and redeploying it will fix the problem
Answer: C

NO.2 You are developing a distributed application and you need a call to a path to always return a specific JSON content deploy an Oracle Cloud Infrastructure API Gateway with the below API deployment specification.

```
{
  "routes": [{
    "path": "/hello",
    "methods": ["GET"],
    "backend": {
      "type": "______________________",
      "status": 200,
      "headers": [{
        "name": "Content-Type",
        "value": "application/json"
      }],
      "body" : "{\"myjson\": \"consistent response\"}"
    }
  }]
}
```

What is the correct value for type?
A. STOCK_RESPONSE_BACKEND
B. CONSTANT_BACKEND
C. JSON_BACKEND
D. HTTP_BACKEND
Answer: A
Explanation
Adding Stock Responses as an API Gateway Back End:
You'll often want to verify that an API has been successfully deployed on an API gateway without having to set up an actual back-end service. One approach is to define a route in the API deployment specification that has a path to a 'dummy' back end. On receiving a request to that path, the API gateway itself acts as the back end and returns a stock response you've specified.
Equally, there are some situations in a production deployment where you'll want a particular path for a route to consistently return the same stock response without sending a request to a back end. For example, when you want a call to a path to always return a specific HTTP status code in the response.

Using the API Gateway service, you can define a path to a stock response back end that always returns the same:
HTTP status code
HTTP header fields (name-value pairs)
content in the body of the response
"type": "STOCK_RESPONSE_BACKEND" indicates that the API gateway itself will act as the back end and return the stock response you define (the status code, the header fields and the body content).
References:
https://docs.cloud.oracle.com/en-us/iaas/Content/APIGateway/Tasks/apigatewayaddingstockresponses.htm

NO.3 In the sample Kubernetes manifest file below, what annotations should you add to create a private load balancer In oracle Cloud infrastructure Container Engine for Kubermetes?

```
apiVersion: v1
kind: Service
metadata:
  name: my-nginx-svc
  labels:
    app: nginx
  annotations:
    <Fill in>
spec:
  type: LoadBalancer
  ports:
  - port: 80
  selector:
    app: nginx

apiVersion: v1
kind: Service
metadata:
  name: my-nginx-svc
  labels:
   app: nginx
  annotations:
   <Fill in>
spec:
  type: LoadBalancer
  ports:
  - port: 80
  selector:
   app: nginx
```

A)

```
service.beta.kubernetes.io/oci-load-balancer-private: "true"
```

B)

```
service.beta.kubernetes.io/oci-load-balancer-private: "true"
service.beta.kubernetes.io/oci-load-balancer-subnet1: "ocid1.subnet.oc1..aaaaaa....vdfw"
```

C)

```
service.beta.kubernetes.io/oci-load-balancer-internal: "true"
```

D)

```
service.beta.kubernetes.io/oci-load-balancer-internal: "true"
service.beta.kubernetes.io/oci-load-balancer-subnet1: "ocid1.subnet.oc1..aaaaaa....vdfw"
```

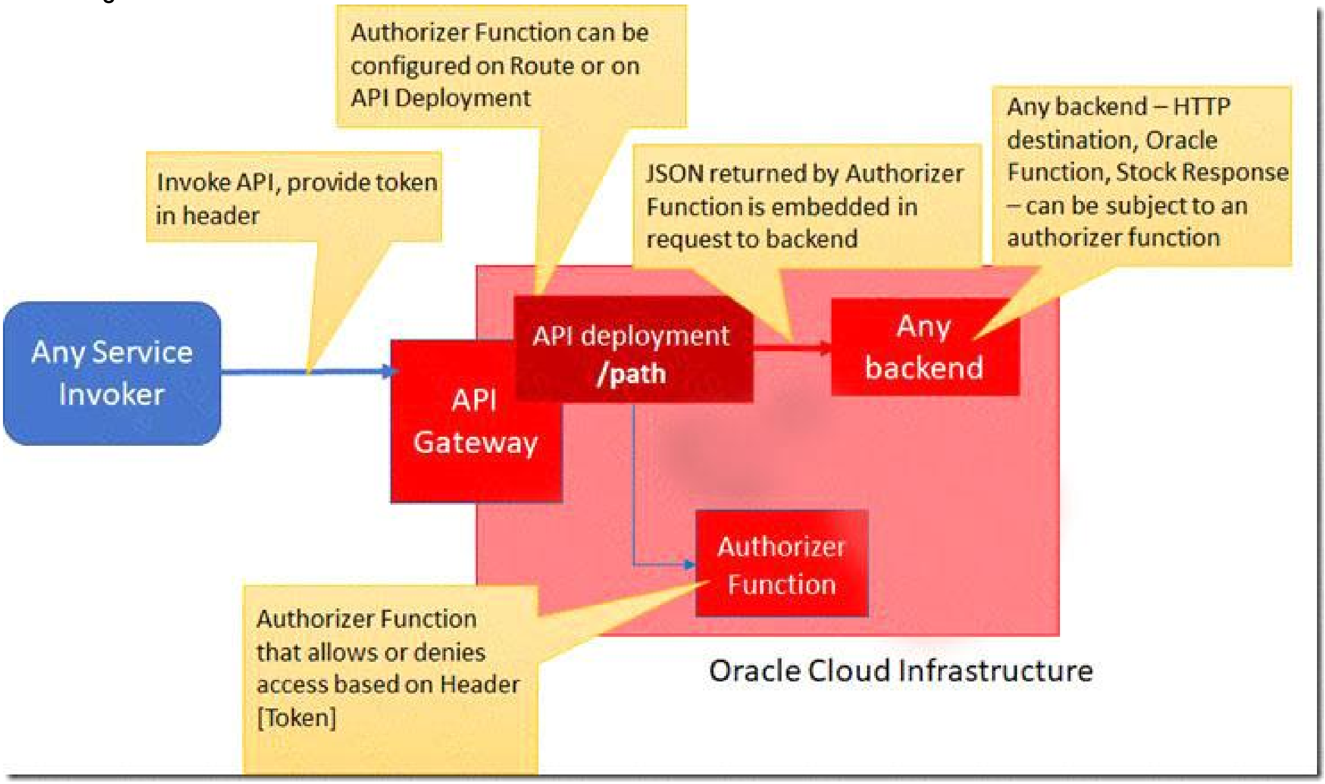
A. Option A
B. Option B
C. Option C
D. Option D
Answer: D
Explanation
https://docs.cloud.oracle.com/en-us/iaas/Content/ContEng/Tasks/contengcreatingloadbalancer.htm?TocPath=Ser Creating Internal Load Balancers in Public and Private Subnets You can create Oracle Cloud Infrastructure load balancers to control access to services running on a cluster:
When you create a 'custom' cluster, you select an existing VCN that contains the network resources to be used by the new cluster. If you want to use load balancers to control traffic into the VCN, you select existing public or private subnets in that VCN to host the load balancers.
When you create a 'quick cluster', the VCN that's automatically created contains a public regional subnet to host a load balancer. If you want to host load balancers in private subnets, you can add private subnets to the VCN later.
Alternatively, you can create an internal load balancer service in a cluster to enable other programs running in the same VCN as the cluster to access services in the cluster. You can host internal load balancers in public subnets and private subnets.
To create an internal load balancer hosted on a public subnet, add the following annotation in the metadata section of the manifest file:
service.beta.kubernetes.io/oci-load-balancer-internal: "true"
To create an internal load balancer hosted on a private subnet, add both following annotations in the metadata section of the manifest file:
service.beta.kubernetes.io/oci-load-balancer-internal: "true"
service.beta.kubernetes.io/oci-load-balancer-subnet1: "ocid1.subnet.oc1..aaaaaa....vdfw" where ocid1.subnet.oc1..aaaaaa....vdfw is the OCID of the private subnet.

NO.4 As a cloud-native developer, you are designing an application that depends on Oracle Cloud Infrastructure (OCI) Object Storage wherever the application is running. Therefore, provisioning of storage buckets should be part of your Kubernetes deployment process for the application. Which should you leverage to meet this requirement?
A. OCI Service Broker for Kubernetes
B. OCI Container Engine for Kubernetes

C. Open Service Broker API
D. Oracle Functions
Answer: A
Explanation
Adding OCI Service Broker for Kubernetes to Clusters:
Service brokers offer a catalog of backing services to workloads running on cloud native platforms. The Open Service Broker API is a commonly-used standard for interactions between service brokers and platforms. The Open Service Broker API specification describes a simple set of API endpoints that platforms use to provision, gain access to, and manage service offerings. For more information about the Open Service Broker API, see resources available online including those at openservicebrokerapi.org.
OCI Service Broker for Kubernetes is an implementation of the Open Service Broker API. OCI Service Broker for Kubernetes is specifically for interacting with Oracle Cloud Infrastructure services from Kubernetes clusters. It includes three service broker adapters to bind to the following Oracle Cloud Infrastructure services:
-Object Storage
-Autonomous Transaction Processing
-Autonomous Data Warehouse
References:
https://docs.cloud.oracle.com/en-us/iaas/Content/ContEng/Tasks/contengaddingservicebrokers.htm

NO.5 A developer using Oracle Cloud Infrastructure (OCI) API Gateway must authenticate the API requests to their web application. The authentication process must be implemented using a custom scheme which accepts string parameters from the API caller. Which method can the developer use In this scenario?
A. Create an authorizer function using request header authorization.
B. Create an authorizer function using token-based authorization.
C. Create a cross account functions authorizer.
D. Create an authorizer function using OCI Identity and Access Management based authentication
Answer: A
Explanation
Using Authorizer Functions to Add Authentication and Authorization to API Deployments:
You can control access to APIs you deploy to API gateways using an 'authorizer function' (as described in this topic), or using JWTs (as described in Using JSON Web Tokens (JWTs) to Add Authentication and Authorization to API Deployments).
You can add authentication and authorization functionality to API gateways by writing an 'authorizer function' that:
1. Processes request attributes to verify the identity of a caller with an identity provider.
2.Determines the operations that the caller is allowed to perform.
3.Returns the operations the caller is allowed to perform as a list of 'access scopes' (an 'access scope' is an arbitrary string used to determine access).
Optionally returns a key-value pair for use by the API deployment. For example, as a context variable for use in an HTTP back end definition (see Adding Context Variables to Policies and HTTP Back End Definitions).
Create an authorizer function using request header authorization implemented using a custom

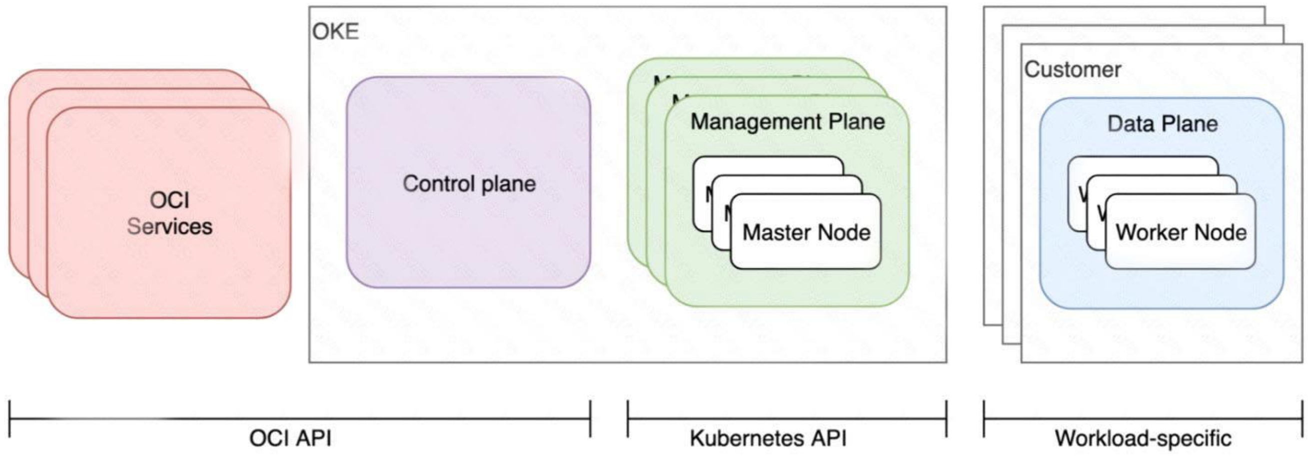
scheme which accepts string parameters from the API caller.
Managing Input Parameters
In our case we will need to manage quite a few static parameters in our code. For example the URLs of the secrets service endpoints, the username and other constant parameterised data. We can manage these either at Application or Function level (an OCI Function is packaged in an Application which can contain multiple Functions). In this case I will create function level parameters. You can use the following command to create the parameters:
fn config function test idcs-assert idcsClientId aedc15531bc8xxxxxxxxxxbd8a193



References:
https://technology.amis.nl/2020/01/03/oracle-cloud-api-gateway-using-an-authorizer-function-for-client-secret-a
https://docs.cloud.oracle.com/en-us/iaas/Content/APIGateway/Tasks/apigatewayusingauthorizerfunction.htm
https://www.ateam-oracle.com/how-to-implement-an-oci-api-gateway-authorization-fn-in-nodejs-that-accesses-o

NO.6 You are implementing logging in your services that will be running in Oracle Cloud Infrastructure Container Engine for Kubernetes. Which statement describes the appropriate logging approach?
A. Each service logs to its own log file.
B. All services log to an external logging system.
C. All services log to standard output only.
D. All services log to a shared log file.
Answer: C
Explanation
Application and systems logs can help you understand what is happening inside your cluster. The logs are particularly useful for debugging problems and monitoring cluster activity. Most modern

applications have some kind of logging mechanism; as such, most container engines are likewise designed to support some kind of logging. The easiest and most embraced logging method for containerized applications is to write to the standard output and standard error streams.
Kubernetes also provides cluster-based logging to record container activity into a central logging subsystem. The standard output and standard error output of each container in a Kubernetes cluster can be ingested using an agent like Fluentd running on each node into tools like Elasticsearch and viewed with Kibana. And finally, monitor containers, pods, applications, services, and other components of your cluster.
One can use tools such as Prometheus, Grafana, Jaeger for monitoring, visibility, and tracing the cluster.



References:
https://dzone.com/articles/5-best-security-practices-for-kubernetes-and-oracle-kubernetes-engine
https://kubernetes.io/docs/concepts/cluster-administration/logging/
https://blogs.oracle.com/developers/5-best-practices-for-kubernetes-security

NO.7 What can you use to dynamically make Kubernetes resources discoverable to public DNS servers?

A. ExternalDNS
B. CoreDNS
C. DynDNS
D. kubeDNS
Answer: A