Free Questions for Professional-Cloud-DevOps-Engineer by braindumpscollection

Shared by Mason on 24-05-2024

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

You are configuring Cloud Logging for a new application that runs on a Compute Engine instance with a public IP address. A user-managed service account is attached to the instance. You confirmed that the necessary agents are running on the instance but you cannot see any log entries from the instance in Cloud Logging. You want to resolve the issue by following Google-recommended practices. What should you do?

Add the Logs Writer role to the service account.

Enable Private Google Access on the subnet that the instance is in.

Update the instance to use the default Compute Engine service account.

Export the service account key and configure the agents to use the key.

Options:

A- Add the Logs Writer role to the service account.

To use Cloud Logging, the service account attached to the Compute Engine instance must have the necessary permissions to write log entries. The Logs Writer role (roles/logging.logWriter) provides this permission. You can grant this role to the user-managed service account at the project, folder, or organization level1.

Private Google Access is not required for Cloud Logging, as it allows instances without external IP addresses to access Google APIs

and services2. The default Compute Engine service account already has the Logs Writer role, but it is not a recommended practice to use it for user applications3. Exporting the service account key and configuring the agents to use the key is not a secure way of authenticating the service account, as it exposes the key to potential compromise4.

Answer:

A

Explanation:

The correct answer is

1: Access control with IAM | Cloud Logging | Google Cloud

2: Private Google Access overview | VPC | Google Cloud

3: Service accounts | Compute Engine Documentation | Google Cloud

4: Best practices for securing service accounts | IAM Documentation | Google Cloud

Question 2

Question Type: MultipleChoice

You want to share a Cloud Monitoring custom dashboard with a partner team What should you do?

Options:

A- Provide the partner team with the dashboard URL to enable the partner team to create a copy of the dashboard

B- Export the metrics to BigQuery Use Looker Studio to create a dashboard, and share the dashboard with the partner team

C- Copy the Monitoring Query Language (MQL) query from the dashboard; and send the MQL query to the partner team

D- Download the JSON definition of the dashboard, and send the JSON file to the partner team

Answer:

A

Explanation:

The best option for sharing a Cloud Monitoring custom dashboard with a partner team is to provide the partner team with the dashboard URL to enable the partner team to create a copy of the dashboard. A Cloud Monitoring custom dashboard is a dashboard that allows you to create and customize charts and widgets to display metrics, logs, and traces from your Google Cloud resources and applications. You can share a custom dashboard with a partner team by providing them with the dashboard URL, which is a link that allows them to view the dashboard in their browser. The partner team can then create a copy of the dashboard in their own project by using the Copy Dashboard option. This way, they can access and modify the dashboard without affecting the original one.

Question 3

Question Type: MultipleChoice

You have an application that runs in Google Kubernetes Engine (GKE). The application consists of several microservices that are deployed to GKE by using Deployments and Services One of the microservices is experiencing an issue where a Pod returns 403 errors after the Pod has been running for more than five hours Your development team is working on a solution but the issue will not be resolved for a month You need to ensure continued operations until the microservice is fixed You want to follow Google-recommended practices and use the fewest number of steps What should you do?

Options:

A- Create a cron job to terminate any Pods that have been running for more than five hours

B- Add a HTTP liveness probe to the microservice s deployment

C- Monitor the Pods and terminate any Pods that have been running for more than five hours

D- Configure an alert to notify you whenever a Pod returns 403 errors

Answer:

B

Explanation:

The best option for ensuring continued operations until the microservice is fixed is to add a HTTP liveness probe to the microservice's deployment. A HTTP liveness probe is a type of probe that checks if a Pod is alive by sending an HTTP request and expecting a success response code. If the probe fails, Kubernetes will restart the Pod. You can add a HTTP liveness probe to your microservice's deployment by using a livenessProbe field in your Pod spec. This way, you can ensure that any Pod that returns 403 errors after running for more than five hours will be restarted automatically and resume normal operations.

Question 4

Question Type: MultipleChoice

As part of your company's initiative to shift left on security, the infoSec team is asking all teams to implement guard rails on all the Google Kubernetes Engine (GKE) clusters to only allow the deployment of trusted and approved images You need to determine how to satisfy the InfoSec teams goal of shifting left on security. What should you do?

Options:

A- Deploy Falco or Twistlock on GKE to monitor for vulnerabilities on your running Pods

B- Configure Identity and Access Management (1AM) policies to create a least privilege model on your GKE clusters

C- Use Binary Authorization to attest images during your CI CD pipeline

D- Enable Container Analysis in Artifact Registry, and check for common vulnerabilities and exposures (CVEs) in your container images

Answer:

C

Explanation:

The best option for implementing guard rails on all GKE clusters to only allow the deployment of trusted and approved images is to use Binary Authorization to attest images during your CI/CD pipeline. Binary Authorization is a feature that allows you to enforce signature-based validation when deploying container images. You can use Binary Authorization to create policies that specify which images are allowed or denied in your GKE clusters. You can also use Binary Authorization to attest images during your CI/CD pipeline by using tools such as Container Analysis or third-party integrations. An attestation is a digital signature that certifies that an image meets certain criteria, such as passing vulnerability scans or code reviews. By using Binary Authorization to attest images during your CI/CD pipeline, you can ensure that only trusted and approved images are deployed to your GKE clusters.

Question 5

Question Type: MultipleChoice

You are building and running client applications in Cloud Run and Cloud Functions Your client requires that all logs must be available for one year so that the client can import the logs into their logging service You must minimize required code changes What should you do?

Options:

A- Update all images in Cloud Run and all functions in Cloud Functions to send logs to both Cloud Logging and the client's logging service Ensure that all the ports required to send logs are open in the VPC firewall

B- Create a Pub/Sub topic subscription and logging sink Configure the logging sink to send all logs into the topic Give your client access to the topic to retrieve the logs

C- Create a storage bucket and appropriate VPC firewall rules Update all images in Cloud Run and all functions in Cloud Functions to send logs to a file within the storage bucket

D- Create a logs bucket and logging sink. Set the retention on the logs bucket to 365 days Configure the logging sink to send logs to the bucket Give your client access to the bucket to retrieve the logs

Answer:

D

Explanation:

The best option for storing all logs for one year and minimizing required code changes is to create a logs bucket and logging sink, set the retention on the logs bucket to 365 days, configure the logging sink to send logs to the bucket, and give your client access to the bucket

to retrieve the logs. A logs bucket is a Cloud Storage bucket that is used to store logs from Cloud Logging. A logging sink is a resource that defines where log entries are sent, such as a logs bucket, BigQuery dataset, or Pub/Sub topic. You can create a logs bucket and logging sink in Cloud Logging and set the retention on the logs bucket to 365 days. This way, you can ensure that all logs are stored for one year and protected from deletion. You can also configure the logging sink to send logs from Cloud Run and Cloud Functions to the logs bucket without any code changes. You can then give your client access to the logs bucket by using IAM policies or signed URLs.

Question 6

Question Type: MultipleChoice

You are building the Cl/CD pipeline for an application deployed to Google Kubernetes Engine (GKE) The application is deployed by using a Kubernetes Deployment, Service, and Ingress The application team asked you to deploy the application by using the blue'green deployment methodology You need to implement the rollback actions What should you do?

Options:

A- Run the kubectl rollout undo command

B- Delete the new container image, and delete the running Pods

C- Update the Kubernetes Service to point to the previous Kubernetes Deployment

D- Scale the new Kubernetes Deployment to zero

Answer:

C

Explanation:

The best option for implementing the rollback actions is to update the Kubernetes Service to point to the previous Kubernetes Deployment. A Kubernetes Service is a resource that defines how to access a set of Pods. A Kubernetes Deployment is a resource that manages the creation and update of Pods. By using the blue/green deployment methodology, you can create two Deployments, one for the current version (blue) and one for the new version (green), and use a Service to switch traffic between them. If you need to rollback, you can update the Service to point to the previous Deployment (blue) and stop sending traffic to the new Deployment (green).

Question 7

Question Type: MultipleChoice

Your company operates in a highly regulated domain that requires you to store all organization logs for seven years You want to minimize logging infrastructure complexity by using managed services You need to avoid any future loss of log capture or stored logs due to misconfiguration or human error What should you do?

Options:

A- Use Cloud Logging to configure an aggregated sink at the organization level to export all logs into a BigQuery dataset

B- Use Cloud Logging to configure an aggregated sink at the organization level to export all logs into Cloud Storage with a seven-year retention policy and Bucket Lock

C- Use Cloud Logging to configure an export sink at each project level to export all logs into a BigQuery dataset

D- Use Cloud Logging to configure an export sink at each project level to export all logs into Cloud Storage with a seven-year retention policy and Bucket Lock

Answer:

B

Explanation:

The best option for storing all organization logs for seven years and avoiding any future loss of log capture or stored logs due to misconfiguration or human error is to use Cloud Logging to configure an aggregated sink at the organization level to export all logs into Cloud Storage with a seven-year retention policy and Bucket Lock. Cloud Logging is a service that allows you to collect and manage logs from your Google Cloud resources and applications. An aggregated sink is a sink that collects logs from multiple sources, such as projects, folders, or organizations. You can use Cloud Logging to configure an aggregated sink at the organization level to export all logs into Cloud Storage, which is a service that allows you to store and access data in Google Cloud. A retention policy is a policy that specifies how long objects in a bucket are retained before they are deleted. Bucket Lock is a feature that allows you to lock a retention policy on a bucket and prevent it from being reduced or removed. You can use Cloud Storage with a seven-year retention policy and Bucket Lock to ensure that your logs are stored for seven years and protected from accidental or malicious deletion.

Question 8

Question Type: MultipleChoice

You are performing a semi-annual capacity planning exercise for your flagship service You expect a service user growth rate of 10% month-over-month for the next six months Your service is fully containerized and runs on a Google Kubemetes Engine (GKE) standard cluster across three zones with cluster autoscaling enabled You currently consume about 30% of your total deployed CPU capacity and you require resilience against the failure of a zone. You want to ensure that your users experience minimal negative impact as a result of this growth o' as a result of zone failure while you avoid unnecessary costs How should you prepare to handle the predicted growth?

Options:

A- Verify the maximum node pool size enable a Horizontal Pod Autoscaler and then perform a load lest to verify your expected resource needs

B- Because you deployed the service on GKE and are using a cluster autoscaler your GKE cluster will scale automatically regardless of growth rate

C- Because you are only using 30% of deployed CPU capacity there is significant headroom and you do not need to add any additional capacity for this rate of growth

D- Proactively add 80% more node capacity to account for six months of 10% growth rate and then perform a load test to ensure that

you have enough capacity

Answer:

A

Explanation:

The best option for preparing to handle the predicted growth is to verify the maximum node pool size, enable a Horizontal Pod Autoscaler, and then perform a load test to verify your expected resource needs. The maximum node pool size is a parameter that specifies the maximum number of nodes that can be added to a node pool by the cluster autoscaler. You should verify that the maximum node pool size is sufficient to accommodate your expected growth rate and avoid hitting any quota limits. The Horizontal Pod Autoscaler is a feature that automatically adjusts the number of Pods in a deployment or replica set based on observed CPU utilization or custom metrics. You should enable a Horizontal Pod Autoscaler for your application to ensure that it runs enough Pods to handle the load. A load test is a test that simulates high user traffic and measures the performance and reliability of your application. You should perform a load test to verify your expected resource needs and identify any bottlenecks or issues.

Question 9

Question Type: **MultipleChoice**

You use Terraform to manage an application deployed to a Google Cloud environment The application runs on instances deployed by a managed instance group The Terraform code is deployed by using a CI/CD pipeline When you change the machine type on the instance template used by the managed instance group, the pipeline fails at the terraform apply stage with the following error message

```
Error waiting for Deleting Instance Template: The instance_template resource
'projects/my-project/global/instanceTemplates/my-it-20220101010101000000000001' is
already being used by 'projects/my-project/regions/us-central1/instanceGroupManagers/my
mig'
```

You need to update the instance template and minimize disruption to the application and the number of pipeline runs What should you do?

Options:

A- Delete the managed instance group and recreate it after updating the instance template

B- Add a new instance template update the managed instance group to use the new instance template and delete the old instance template

C- Remove the managed instance group from the Terraform state file update the instance template and reimport the managed instance group.

D- Set the create_bef ore_destroy meta-argument to true in the lifecycle block on the instance template

Answer:

D

Explanation:

The best option for updating the instance template and minimizing disruption to the application and the number of pipeline runs is to set the create_before_destroy meta-argument to true in the lifecycle block on the instance template. The create_before_destroy meta-argument is a Terraform feature that specifies that a new resource should be created before destroying an existing one during an update. This way, you can avoid downtime and errors when updating a resource that is in use by another resource, such as an instance template that is used by a managed instance group. By setting the create_before_destroy meta-argument to true in the lifecycle block on the instance template, you can ensure that Terraform creates a new instance template with the updated machine type, updates the managed instance group to use the new instance template, and then deletes the old instance template.

Question 10

Question Type: MultipleChoice

Your organization is using Helm to package containerized applications Your applications reference both public and private charts Your security team flagged that using a public Helm repository as a dependency is a risk You want to manage all charts uniformly, with native access control and VPC Service Controls What should you do?

Options:

A- Store public and private charts in OCI format by using Artifact Registry

B- Store public and private charts by using GitHub Enterprise with Google Workspace as the identity provider

C- Store public and private charts by using Git repository Configure Cloud Build to synchronize contents of the repository into a Cloud Storage bucket Connect Helm to the bucket by using https: // [bucket] .srorage.googleapis.com/ [holnchart] as the Helm repository

D- Configure a Helm chart repository server to run in Google Kubernetes Engine (GKE) with Cloud Storage bucket as the storage backend

Answer:

A

Explanation:

The best option for managing all charts uniformly, with native access control and VPC Service Controls is to store public and private charts in OCI format by using Artifact Registry. Artifact Registry is a service that allows you to store and manage container images and other artifacts in Google Cloud. Artifact Registry supports OCI format, which is an open standard for storing container images and other artifacts such as Helm charts. You can use Artifact Registry to store public and private charts in OCI format and manage them uniformly. You can also use Artifact Registry's native access control features, such as IAM policies and VPC Service Controls, to secure your charts and control who can access them.

Question 11

Question Type: MultipleChoice

You are running a web application deployed to a Compute Engine managed instance group Ops Agent is installed on all instances You recently noticed suspicious activity from a specific IP address You need to configure Cloud Monitoring to view the number of requests from that specific IP address with minimal operational overhead. What should you do?

Options:

A- Configure the Ops Agent with a logging receiver Create a logs-based metric

B- Create a script to scrape the web server log Export the IP address request metrics to the Cloud Monitoring API

C- Update the application to export the IP address request metrics to the Cloud Monitoring API

D- Configure the Ops Agent with a metrics receiver

Answer:

A

Explanation:

The best option for configuring Cloud Monitoring to view the number of requests from a specific IP address with minimal operational overhead is to configure the Ops Agent with a logging receiver and create a logs-based metric. The Ops Agent is an agent that collects system metrics and logs from your VM instances and sends them to Cloud Monitoring and Cloud Logging. A logging receiver is a configuration that specifies which logs are collected by the Ops Agent and how they are processed. You can use a logging receiver to collect web server logs from your VM instances and send them to Cloud Logging. A logs-based metric is a metric that is extracted from log entries in Cloud Logging. You can use a logs-based metric to count the number of requests from a specific IP address by using a filter expression. You can then use Cloud Monitoring to view and analyze the logs-based metric.

Question 12

Question Type: **MultipleChoice**

You manage an application that runs in Google Kubernetes Engine (GKE) and uses the blue/green deployment methodology Extracts of the Kubernetes manifests are shown below

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: app-green
  labels:
    app: my-app
    version: green
<other fields snipped>
---
apiVersion: v1
kind: Service
metadata:
  name: app-svc
spec:
  selector:
    app: my-app
    version: green
<other fields snipped>
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: app-ingress
```

The Deployment app-green was updated to use the new version of the application During post-deployment monitoring you notice that the majority of user requests are failing You did not observe this behavior in the testing environment You need to mitigate the incident impact on users and enable the developers to troubleshoot the issue What should you do?

Options:

A- Update the Deployment app-blue to use the new version of the application

B- Update the Deployment ape-green to use the previous version of the application

C- Change the selector on the Service app-2vc to app: my-app.

D- Change the selector on the Service app-svc to app: my-app, version: blue

Answer:

D

Explanation:

The best option for mitigating the incident impact on users and enabling the developers to troubleshoot the issue is to change the selector on the Service app-svc to app: my-app, version: blue. A Service is a resource that defines how to access a set of Pods. A selector is a field that specifies which Pods are selected by the Service. By changing the selector on the Service app-svc to app: my-app, version: blue, you can ensure that the Service only routes traffic to the Pods that have both labels app: my-app and version: blue. These Pods belong to the Deployment app-blue, which uses the previous version of the application. This way, you can mitigate the incident impact on users by switching back to the working version of the application. You can also enable the developers to troubleshoot

the issue with the new version of the application in the Deployment app-green without affecting users.