# Free Questions for Professional-Cloud-DevOps-Engineer

## Shared by Mann on 12-12-2023

For More Free Questions and Preparation Resources

Check the Links on Last Page

# Question 1

Question Type: MultipleChoice

You are analyzing Java applications in production. All applications have Cloud Profiler and Cloud Trace installed and configured by default. You want to determine which applications need performance tuning. What should you do?

Choose 2 answers

## Options:

A- Examine the wall-clock time and the CPU time Of the application. If the difference is substantial, increase the CPU resource allocation.
B- Examine the wall-clock time and the CPU time of the application. If the difference is substantial, increase the memory resource allocation.
C- 17 Examine the wall-clock time and the CPU time of the application. If the difference is substantial, increase the local disk storage allocation.
D- O Examine the latency time, the wall-clock time, and the CPU time of the application. If the latency time is slowly burning down the error budget, and the difference between wall-clock time and CPU time is minimal, mark the application for optimization.
E- Examine the heap usage Of the application. If the usage is low, mark the application for optimization.

## Answer:

A, D

## Explanation:

The correct answers are A and D)

Examine the wall-clock time and the CPU time of the application. If the difference is substantial, increase the CPU resource allocation. This is a good way to determine if the application is CPU-bound, meaning that it spends more time waiting for the CPU than performing actual computation. Increasing the CPU resource allocation can improve the performance of CPU-bound applications1.

Examine the latency time, the wall-clock time, and the CPU time of the application. If the latency time is slowly burning down the error budget, and the difference between wall-clock time and CPU time is minimal, mark the application for optimization. This is a good way to determine if the application is I/O-bound, meaning that it spends more time waiting for input/output operations than performing actual computation. Increasing the CPU resource allocation will not help I/O-bound applications, and they may need optimization to reduce the number or duration of I/O

operations2.

Answer B is incorrect because increasing the memory resource allocation will not help if the application is CPU-bound or I/O-bound. Memory allocation affects how much data the application can store and access in memory, but it does not affect how fast the application can process that data.

Answer C is incorrect because increasing the local disk storage allocation will not help if the application is CPU-bound or I/O-bound. Disk storage affects how much data the application can store and access on disk, but it does not affect how fast the application can process that data.

Answer E is incorrect because examining the heap usage of the application will not help to determine if the application needs performance tuning. Heap usage affects how much memory the application allocates for dynamic objects, but it does not affect how fast the application can process those objects. Moreover, low heap usage does not necessarily mean that the application is inefficient or unoptimized.

# Question 2

Question Type: MultipleChoice

You need to build a CI/CD pipeline for a containerized application in Google Cloud Your development team uses a central Git repository for trunk-based development You want to run all your tests in the pipeline for any new versions of the application to improve the quality What should you do?

## Options:

A- 1. Install a Git hook to require developers to run unit tests before pushing the code to a central repository
2. Trigger Cloud Build to build the application container Deploy the application container to a testing environment, and run integration tests
3. If the integration tests are successful deploy the application container to your production environment. and run acceptance tests

B- 1. Install a Git hook to require developers to run unit tests before pushing the code to a central repository
If all tests are successful build a container
2. Trigger Cloud Build to deploy the application container to a testing environment, and run integration
tests and acceptance tests
3. If all tests are successful tag the code as production ready Trigger Cloud Build to build and deploy the application container to the production environment

C- 1. Trigger Cloud Build to build the application container and run unit tests with the container
2. If unit tests are successful, deploy the application container to a testing environment, and run integration tests
3. If the integration tests are successful the pipeline deploys the application container to the production environment After that, run acceptance tests
D- 1. Trigger Cloud Build to run unit tests when the code is pushed If all unit tests are successful, build and push the application container to a central registry.
2. Trigger Cloud Build to deploy the container to a testing environment, and run integration tests and acceptance tests
3. If all tests are successful the pipeline deploys the application to the production environment and runs smoke tests

## Answer:

D

## Explanation:

The best option for building a CI/CD pipeline for a containerized application in Google Cloud is to trigger Cloud Build to run unit tests when the code is pushed, if all unit tests are successful, build and push the application container to a central registry, trigger Cloud Build to deploy the container to a testing environment, and run integration tests and acceptance tests, and if all tests are successful, the pipeline deploys the application to the production environment and runs smoke tests. This option follows the best practices for CI/CD pipelines, such as running tests at different stages of the pipeline, using a central registry for storing and managing containers, deploying to different environments, and using Cloud Build as a unified tool for building, testing, and deploying.

# Question 3

Question Type: MultipleChoice

You are leading a DevOps project for your organization. The DevOps team is responsible for managing the service infrastructure and being on-call for incidents. The Software Development team is responsible for writing, submitting, and reviewing code. Neither team has any published SLOs. You want to design a new joint-ownership model for a service between the DevOps team and the Software Development team. Which responsibilities should be assigned to each team in the new joint-ownership model?

A.

| DevOps team responsibilities | Software Development team responsibilities |
|---|---|
| • Manage the service infrastructure<br>• Be on-call for incidents<br>• Perform code reviews | • Submit code to be reviewed by the DevOps team<br>• Publish the SLOs that the DevOps team must meet |

B.

| DevOps team responsibilities | Software Development team responsibilities |
|---|---|
| • Manage the service infrastructure<br>• Perform code reviews | • Submit code to be reviewed by the DevOps team<br>• Be on-call for incidents<br>• Publish the SLOs that the DevOps team must meet |

C.

| DevOps team responsibilities | Shared responsibilities | Software Development team responsibilities |
|---|---|---|
| • Manage the service infrastructure | • Perform code reviews<br>• Be on-call for incidents on a rotation basis<br>• Adopt and publish SLOs for the service | • Submit code to be reviewed |

D.

| DevOps team responsibilities | Shared responsibilities | Software Development team responsibilities |
|---|---|---|
| • Manage the service infrastructure<br>• Be on-call for incidents | • Adopt and publish SLOs for the service | • Submit code to be reviewed<br>• Perform code reviews |

## Options:

A- Option A
B- Option B
C- Option C
D- Option D

## Answer:

D

## Explanation:

The correct answer is D. Option D)

According to the DevOps best practices, a joint-ownership model for a service between the DevOps team and the Software Development team should follow these principles12:

The DevOps team and the Software Development team should share the responsibility and collaboration for managing the service infrastructure, performing code reviews, and adopting and sharing SLOs for the service.

The DevOps team and the Software Development team should have end-to-end ownership of the service, from design to development to deployment to operation to maintenance.

The DevOps team and the Software Development team should use common tools and processes to facilitate communication, coordination, and feedback.

The DevOps team and the Software Development team should align their goals and incentives with the business outcomes and customer satisfaction.

Option D is the only option that reflects these principles. Option D assigns both teams the responsibilities of managing the service infrastructure, performing code reviews, and adopting and sharing SLOs for the service. Option D also implies that both teams have end-to-end ownership of the service, as they are involved in every stage of the service lifecycle. Option D also encourages both teams to use common tools and processes, such as GitLab3, to collaborate and communicate effectively. Option D also aligns both teams with the business outcomes and customer satisfaction, as they use SLOs to measure and improve the service quality.

The other options are incorrect because they do not follow the DevOps best practices. Option A is incorrect because it assigns only the DevOps team the responsibility of managing the service infrastructure, which creates a silo between the two teams and reduces their collaboration. Option A also does not assign any responsibility for adopting and sharing SLOs for the service, which means that both teams lack a common metric for measuring and improving the service quality. Option B is incorrect because it assigns only the Software Development team the responsibility of performing code reviews, which creates a gap between the two teams and reduces their feedback. Option B also does not assign any responsibility for adopting and sharing SLOs for the service, which means that both teams lack a common metric for measuring and improving the service quality. Option C is incorrect because it assigns both teams the same responsibilities as option A and option B, which combines their drawbacks.

5 key organizational models for DevOps teams | GitLab, 5 key organizational models for DevOps teams | GitLab. Building a Culture of Full-Service Ownership - DevOps.com, Building a Culture of Full-Service Ownership - DevOps.com. GitLab, GitLab.

# Question 4

Question Type: MultipleChoice

You are the on-call Site Reliability Engineer for a microservice that is deployed to a Google Kubernetes Engine (GKE) Autopilot cluster. Your company runs an online store that publishes order messages to Pub/Sub and a microservice receives these messages and updates stock information in the warehousing system. A sales event caused an increase in orders, and the stock

information is not being updated quickly enough. This is causing a large number of orders to be accepted for products that are out of stock You check the metrics for the microservice and compare them to typical levels.

| Microservice metrics | Typical state | Current state |
|---|---|---|
| Average CPU across all Pods | 20% of Pod limit | 30% of Pod limit |
| Average memory across all Pods | 10% of Pod limit | 10% of Pod limit |
| Pub/Sub subscription: Average oldest unacknowledged message age | 347 milliseconds | 8074 milliseconds |
| Pub/Sub subscription: Average undelivered messages | 5 messages | 14705 messages |
| Pub/Sub subscription: Average acknowledgment latency | 312 milliseconds | 354 milliseconds |

You need to ensure that the warehouse system accurately reflects product inventory at the time orders are placed and minimize the impact on customers What should you do?

## Options:

A- Decrease the acknowledgment deadline on the subscription
B- Add a virtual queue to the online store that allows typical traffic levels
C- Increase the number of Pod replicas
D- Increase the Pod CPU and memory limits

## Answer:

C

## Explanation:

The best option for ensuring that the warehouse system accurately reflects product inventory at the time orders are placed and minimizing the impact on customers is to increase the number of Pod replicas. Increasing the number of Pod replicas will increase the scalability and availability of your microservice, which will allow it to handle more Pub/Sub messages and update stock information faster. This way, you can reduce the backlog of undelivered messages and oldest unacknowledged message age, which are causing delays in updating product inventory. You can use Horizontal Pod Autoscaler or Cloud Monitoring metrics-based autoscaling to automatically adjust the number of Pod replicas based on load or custom metrics.

# Question 5

Question Type: MultipleChoice

You are running an application on Compute Engine and collecting logs through Stackdriver. You

discover that some personally identifiable information (PII) is leaking into certain log entry fields. You want to prevent these fields from being written in new log entries as quickly as possible. What should you do?

## Options:

A- Use the filter-record-transformer Fluentd filter plugin to remove the fields from the log entries in flight.
B- Use the fluent-plugin-record-reformer Fluentd output plugin to remove the fields from the log entries in flight.
C- Wait for the application developers to patch the application, and then verify that the log entries are no longer exposing PII.
D- Stage log entries to Cloud Storage, and then trigger a Cloud Function to remove the fields and write the entries to Stackdriver via the Stackdriver Logging API.

## Answer:

A

# Question 6

Question Type: MultipleChoice

You are configuring your CI/CD pipeline natively on Google Cloud. You want builds in a pre-production Google Kubernetes Engine (GKE) environment to be automatically load-tested before being promoted to the production GKE environment. You need to ensure that only builds that have passed this test are deployed to production. You want to follow Google-recommended practices. How should you configure this pipeline with Binary Authorization?

## Options:

A- Create an attestation for the builds that pass the load test by requiring the lead quality assurance engineer to sign the attestation by using a key stored in Cloud Key Management Service (Cloud KMS).
B- Create an attestation for the builds that pass the load test by using a private key stored in Cloud Key Management Service (Cloud KMS) authenticated through Workload Identity.
C- Create an attestation for the builds that pass the load test by using a private key stored in Cloud Key Management Service (Cloud KMS) with a service account JSON key stored as a Kubernetes Secret.
D- Create an attestation for the builds that pass the load test by requiring the lead quality assurance engineer to sign the attestation by using their personal private key.

## Answer:

B

## Explanation:

The correct answer is B, Create an attestation for the builds that pass the load test by using a private key stored in Cloud Key Management Service (Cloud KMS) authenticated through Workload Identity.

According to the Google Cloud documentation, Binary Authorization is a deploy-time security control that ensures only trusted container images are deployed on Google Kubernetes Engine (GKE) or Cloud Run1. Binary Authorization uses attestations to certify that a specific image has completed a previous stage in the CI/CD pipeline, such as passing a load test2. Attestations are signed by private keys that are associated with attestors, which are entities that verify the attestations3. To follow Google-recommended practices, you should store your private keys in Cloud Key Management Service (Cloud KMS), which is a secure and scalable service for managing cryptographic keys4. You should also use Workload Identity, which is a feature that allows Kubernetes service accounts to act as Google service accounts, to authenticate to Cloud KMS and sign attestations without having to manage or expose service account keys5.

The other options are incorrect because they do not follow Google-recommended practices. Option A and option D require human intervention to sign the attestations, which is not scalable or automated. Option C exposes the service account JSON key as a Kubernetes Secret, which is less secure than using Workload Identity.

Creating an attestor, Creating an attestor. Cloud Key Management Service Documentation, Overview. Attestations overview, Attestations overview. Using Workload Identity with Binary Authorization, Using Workload Identity with Binary Authorization. Binary Authorization, Binary Authorization.

P2P exams

To Get Premium Files for Professional-Cloud-DevOps-Engineer Visit
https://www.p2pexams.com/products/professional-cloud-devops-engineer

For More Free Questions Visit
https://www.p2pexams.com/google/pdf/professional-cloud-devops-engineer

