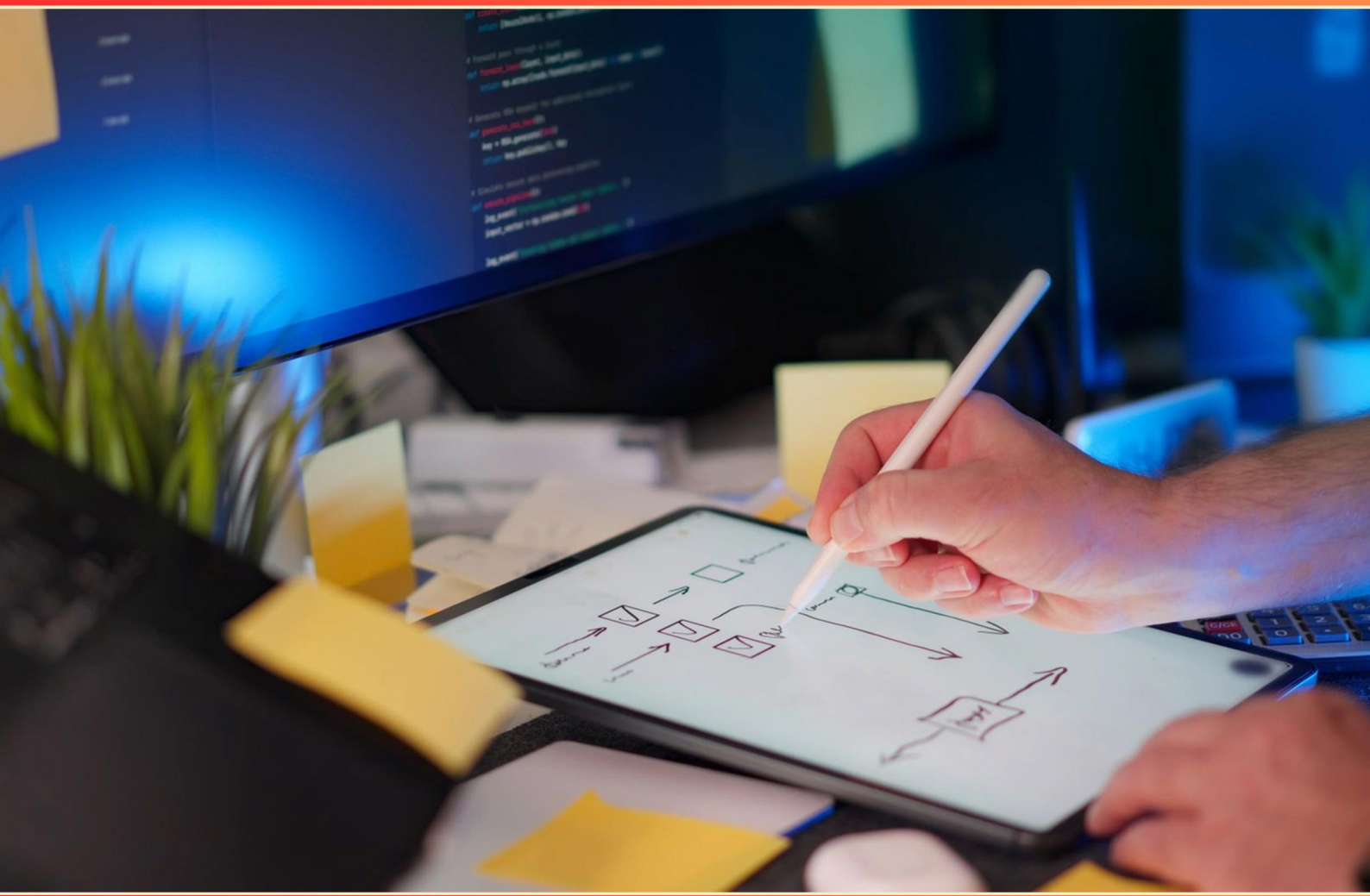
ITGSS CERTIFIED DEVOPS ENGINEER PRACTICE TEST (SAMPLE)

STUDY GUIDE



SAMPLE STUDY GUIDE WITH LIMITED QUESTIONS

VISIT US ONLINE FOR THE FULL VERSION STUDY GUIDE

https://itgss-certifieddevopsengineer.examzify.com

Copyright © 2026 by Examzify - A Kaluba Technologies Inc. product.

ALL RIGHTS RESERVED.

No part of this book may be reproduced or transferred in any form or by any means, graphic, electronic, or mechanical, including photocopying, recording, web distribution, taping, or by any information storage retrieval system, without the written permission of the author.

Notice: Examzify makes every reasonable effort to obtain accurate, complete, and timely information about this product from reliable sources.

SAMPLE

Table of Contents

Copyright 1

Table of Contents 2

Introduction 3

How to Use This Guide 4

Questions 5

Answers 8

Explanations 10

Next Steps 16

SAMPLE

Introduction

Preparing for a certification exam can feel overwhelming, but with the right tools, it becomes an opportunity to build confidence, sharpen your skills, and move one step closer to your goals. At Examzify, we believe that effective exam preparation isn't just about memorization, it's about understanding the material, identifying knowledge gaps, and building the test-taking strategies that lead to success.

This guide was designed to help you do exactly that.

Whether you're preparing for a licensing exam, professional certification, or entry-level qualification, this book offers structured practice to reinforce key concepts. You'll find a wide range of multiple-choice questions, each followed by clear explanations to help you understand not just the right answer, but why it's correct.

The content in this guide is based on real-world exam objectives and aligned with the types of questions and topics commonly found on official tests. It's ideal for learners who want to:

- Practice answering questions under realistic conditions,
- Improve accuracy and speed,
- Review explanations to strengthen weak areas, and
- Approach the exam with greater confidence.

We recommend using this book not as a stand-alone study tool, but alongside other resources like flashcards, textbooks, or hands-on training. For best results, we recommend working through each question, reflecting on the explanation provided, and revisiting the topics that challenge you most.

Remember: successful test preparation isn't about getting every question right the first time, it's about learning from your mistakes and improving over time. Stay focused, trust the process, and know that every page you turn brings you closer to success.

Let's begin.

SAMPLE

How to Use This Guide

This guide is designed to help you study more effectively and approach your exam with confidence. Whether you're reviewing for the first time or doing a final refresh, here's how to get the most out of your Examzify study guide:

1. Start with a Diagnostic Review

Skim through the questions to get a sense of what you know and what you need to focus on. Your goal is to identify knowledge gaps early.

2. Study in Short, Focused Sessions

Break your study time into manageable blocks (e.g. 30 – 45 minutes). Review a handful of questions, reflect on the explanations.

3. Learn from the Explanations

After answering a question, always read the explanation, even if you got it right. It reinforces key points, corrects misunderstandings, and teaches subtle distinctions between similar answers.

4. Track Your Progress

Use bookmarks or notes (if reading digitally) to mark difficult questions. Revisit these regularly and track improvements over time.

5. Simulate the Real Exam

Once you're comfortable, try taking a full set of questions without pausing. Set a timer and simulate test-day conditions to build confidence and time management skills.

6. Repeat and Review

Don't just study once, repetition builds retention. Re-attempt questions after a few days and revisit explanations to reinforce learning. Pair this guide with other Examzify tools like flashcards, and digital practice tests to strengthen your preparation across formats.

There's no single right way to study, but consistent, thoughtful effort always wins. Use this guide flexibly, adapt the tips above to fit your pace and learning style. You've got this!

SAMPLE

Questions

SAMPLE

1. Kubernetes clusters consist of which primary component types?

A. Pods and Services

B. Nodes and Containers

C. Controllers and Schedulers

D. Services and Deployments

2. What is a rollback in DevOps?

A. Reverting to a previous version of software

B. Deploying the latest build to production

C. Testing a new feature in isolation

D. Upgrading software to a new version

3. What is the role of network segmentation in a Kubernetes environment?

A. To enhance container performance

B. To allow for better resource allocation

C. To improve security by limiting communications

D. To manage container versions effectively

4. Which component manages controller processes such as DaemonSet in Kubernetes?

A. Scheduler

B. ControllerManager

C. API Server

D. Worker Node

5. Kubernetes uses which of the following to run logging agents?

A. Google Cloud Functions

B. Fluentd agents

C. Custom logging frameworks

D. AWS Lambda

6. The term 'pod' in Kubernetes refers to what?

A. A group of nodes

B. A single container

C. A group of one or more containers

D. A collection of services

SAMPLE

7. What are the key practices of Continuous Integration (CI)?

A. Automated testing, frequent code commits, and build automation

B. Manual testing and prolonged releases

C. Single code commits and project management tools

D. Bug tracking and user feedback collection

8. What is the purpose of container orchestration?

A. To increase bandwidth efficiency

B. To automate deployment and operation of application containers

C. To establish firewall protections

D. To manage user permissions

9. When including security within the software development lifecycle, what is commonly referred to as DevSecOps?

A. Figuring out post-deployment vulnerabilities

B. Integrating security practices into DevOps processes

C. Creating a dedicated security team

D. Focusing solely on compliance checks

10. When a Service is created in Kubernetes, what entity is generated alongside it?

A. DNS entry

B. IP address

C. Container

D. Pod

SAMPLE

Answers

SAMPLE

1. B
2. A
3. C
4. B
5. B
6. C
7. A
8. B
9. B
10. A

SAMPLE

Explanations

SAMPLE

1. Kubernetes clusters consist of which primary component types?

A. Pods and Services

B. Nodes and Containers

C. Controllers and Schedulers

D. Services and Deployments

The correct answer identifies nodes and containers as the primary component types of Kubernetes clusters. Nodes serve as the physical or virtual machines that provide the infrastructure required to run applications and workloads in a Kubernetes environment. Each node can contain multiple pods, which encapsulate one or more containers. Containers are the runtime instances of applications, packaged with their configuration and dependencies. Understanding the distinction between these components is crucial in a Kubernetes architecture. Nodes are responsible for hosting the pods, while containers run the actual applications. This relationship between nodes and containers highlights the fundamental structure of a Kubernetes cluster, emphasizing how resources are allocated and applications are managed across the distributed environment. Other options mention components that are indeed important to Kubernetes, such as services, deployments, and pods, but they do not encompass the primary components in the same foundational manner as nodes and containers. Pods and services relate more to how applications are accessed and managed on the cluster, while controllers and schedulers focus on workload management and orchestration, rather than the core structural elements of the clusters themselves.

2. What is a rollback in DevOps?

A. Reverting to a previous version of software

B. Deploying the latest build to production

C. Testing a new feature in isolation

D. Upgrading software to a new version

A rollback in DevOps refers to the process of reverting to a previous version of software that was previously known to work correctly. This is a critical procedure used to maintain software stability and performance in the event that a new deployment introduces bugs or undesired behaviors. Rollbacks are essential for establishing a reliable deployment process, as they allow teams to quickly revert to a stable state without significant downtime. In practice, if a recent deployment introduces issues that could affect end users negatively, rather than spending time diagnosing and addressing the problems right away, a rollback can be initiated to return to the last stable version. This helps ensure that users have continuous access to functional software while developers fix the issues in the problematic release. This process is a vital part of deployment strategies in DevOps, where rapid iterations and continuous delivery are prevalent. It underscores the importance of versioning and maintaining a history of builds to facilitate such reversions when necessary.

SAMPLE

3. What is the role of network segmentation in a Kubernetes environment?

A. To enhance container performance

B. To allow for better resource allocation

C. To improve security by limiting communications

D. To manage container versions effectively

In a Kubernetes environment, network segmentation plays a crucial role in improving security by effectively managing and restricting communications between different components and services. By segmenting the network, you can create isolated environments for various applications or workloads. This isolation reduces the risk of unauthorized access and potential attacks, as it limits communication pathways and minimizes the attack surface. For example, certain services may only need to communicate with specific components within the cluster. By implementing network segmentation, these services can be grouped and controlled using network policies, ensuring that only intended communications are allowed while blocking all others. This approach not only protects sensitive data but also helps to contain breaches in the event they occur, as attackers would find it more challenging to traverse the segmented network. The other options do not directly pertain to the primary role of network segmentation. Enhancing container performance relates more to resource management and container orchestration rather than network isolation. Similarly, while effective resource allocation is vital in Kubernetes, it doesn't specifically link to the concept of network segmentation. Lastly, managing container versions focuses on deployment and version control practices rather than the communication paths within the network. Thus, the emphasis on improving security through limiting communications is the most accurate and relevant role of network segmentation in a Kubernetes context.

4. Which component manages controller processes such as DaemonSet in Kubernetes?

A. Scheduler

B. ControllerManager

C. API Server

D. Worker Node

The component responsible for managing controller processes such as DaemonSet in Kubernetes is the ControllerManager. In Kubernetes, the Controller Manager runs controllers that regulate the state of the system, ensuring that your desired configurations are maintained. Controllers are control loops that watch the state of your cluster, typically through the API server, and make or request changes where needed. For instance, a DaemonSet controller ensures that a copy of a pod runs on all or specific nodes in the cluster, managing the deployment and scaling of these pods. This involves monitoring the current state of the pods and taking action when the actual state doesn't match the desired state defined by the DaemonSet. While the Scheduler is responsible for placing pods onto the appropriate nodes, and the API Server serves as the front end for the Kubernetes control plane handling requests, neither of these components directly manages DaemonSets. The Worker Node is where the containers are executed and does not handle the management of controller processes. Thus, the ControllerManager is specifically designed to manage and regulate such tasks, making it the correct choice.

SAMPLE

5. Kubernetes uses which of the following to run logging agents?

A. Google Cloud Functions

B. Fluentd agents

C. Custom logging frameworks

D. AWS Lambda

Kubernetes commonly employs Fluentd agents for logging purposes due to their designed capability to collect, filter, and aggregate logs efficiently from various sources within the Kubernetes environment. Fluentd integrates smoothly with Kubernetes, enabling it to gather logs from containers, nodes, and the system as a whole, summarizing this information into a format that can be easily processed or forwarded to various log storage and analysis solutions. Moreover, Fluentd is highly configurable, allowing users to specify log sources, apply custom parsing and filtering, and define output destinations, making it an ideal choice for managing logging in dynamic environments like Kubernetes. It supports a wide range of plugins, further enhancing its interoperability and versatility in handling logs from multiple systems and applications. The other options do not participate in running logging agents within Kubernetes. Google Cloud Functions and AWS Lambda are serverless computing platforms, not designed for log management or aggregation in container orchestration environments. Custom logging frameworks, while useful, typically require greater effort to integrate and manage compared to the standardized approach provided by Fluentd, making them less favorable for Kubernetes logging tasks.

6. The term 'pod' in Kubernetes refers to what?

A. A group of nodes

B. A single container

C. A group of one or more containers

D. A collection of services

In Kubernetes, a 'pod' is the smallest deployable unit and represents a group of one or more closely related containers that share the same network namespace and storage volumes. This grouping allows the containers to work together and communicate efficiently as they share an IP address and port space, making inter-container communication straightforward. Pods are designed to host applications that require multiple components to function cohesively, like a web server and a database that need to interact with one another. By allowing multiple containers to run in a single pod, Kubernetes optimizes resource usage and simplifies orchestration by managing these containers as a single unit. This design supports scenarios where containers have related functionalities, enhancing the application's performance and management.

SAMPLE

7. What are the key practices of Continuous Integration (CI)?

A. Automated testing, frequent code commits, and build automation

B. Manual testing and prolonged releases

C. Single code commits and project management tools

D. Bug tracking and user feedback collection

The key practices of Continuous Integration (CI) focus on improving software development processes through automation and teamwork, facilitating more efficient workflows and higher quality software. The correct option highlights three essential components: automated testing, frequent code commits, and build automation. Automated testing is crucial in CI because it allows developers to validate their code changes against a set of predefined tests automatically. This helps in identifying bugs early in the development cycle, ultimately saving time and costs associated with fixing issues later on. Frequent code commits are also a fundamental practice in CI. By committing code regularly, developers ensure that their changes are integrated into the main codebase frequently. This reduces the likelihood of integration conflicts and helps maintain a coherent project history. It encourages collaboration among team members, as everyone works with the most up-to-date code. Build automation plays a vital role as well, as it automates the process of compiling the code and running tests after each commit. This ensures that any integration issues or bugs are detected immediately, allowing developers to address them before they propagate further into the development process. In contrast, the other options do not encapsulate the essence of CI practices. For instance, manual testing and prolonged releases fall short of the automation and efficiency that CI aims to achieve. Similarly

8. What is the purpose of container orchestration?

A. To increase bandwidth efficiency

B. To automate deployment and operation of application containers

C. To establish firewall protections

D. To manage user permissions

Container orchestration refers to the automated management of containerized applications, particularly their deployment, scaling, networking, and availability across different environments. The main purpose of container orchestration is to streamline the operational tasks that would otherwise be complex and time-consuming to manage manually, allowing developers and operations teams to focus on building applications rather than on the intricacies of deployment. With orchestration, you can quickly deploy multiple containers as a single unit, manage their lifecycle, and ensure that the application runs reliably, even in the face of failures or changes in the underlying infrastructure. This includes functions such as scaling containers up or down based on demand, automatically replacing failed containers, and handling networking between containers. Other options mentioned do not align with the core purpose of container orchestration. For instance, increasing bandwidth efficiency is more related to network optimization techniques, establishing firewall protections pertains to network security measures, and managing user permissions is focused on access control within systems rather than the deployment and management of applications.

SAMPLE

9. When including security within the software development lifecycle, what is commonly referred to as DevSecOps?

A. Figuring out post-deployment vulnerabilities

B. Integrating security practices into DevOps processes

C. Creating a dedicated security team

D. Focusing solely on compliance checks

Integrating security practices into DevOps processes is a critical aspect of modern software development known as DevSecOps. This approach aims to embed security considerations at every stage of the software development lifecycle, rather than treating it as an afterthought or a separate phase. By doing this, organizations can identify and address vulnerabilities early in the development process, reducing the risk of security issues being discovered only after deployment. DevSecOps emphasizes collaboration between development, operations, and security teams to ensure that security is a shared responsibility. This integration helps in automating security checks, promoting secure coding practices, and maintaining compliance throughout the deployment pipeline. Thus, the result is a more robust and resilient software product that minimizes the risk of security breaches. A focus solely on compliance checks, the creation of a dedicated security team, or addressing vulnerabilities only after deployment does not embody the proactive and integrated approach that DevSecOps represents. Instead, DevSecOps strives to make security an integral part of the development process, ensuring that it is considered from the very beginning to the end of the project lifecycle.

10. When a Service is created in Kubernetes, what entity is generated alongside it?

A. DNS entry

B. IP address

C. Container

D. Pod

When a Service is created in Kubernetes, it automatically generates a DNS entry. This entry allows for easier access to the Service via a stable hostname rather than relying on the dynamic Pod IP addresses, which may change as Pods are started, stopped, or rescheduled. The DNS entry is part of Kubernetes' internal mechanisms, where it sets up an associated DNS record that maps the Service name to the appropriate IP address. This is crucial for microservices architecture where various services need to communicate with each other using their names, thus enhancing service discovery and orchestration. While the other entities, like an IP address, are also relevant components of a Service, the automatic generation of the DNS entry is specifically highlighted because it provides a more user-friendly way for different components in the cluster to connect. Moreover, while Services do route traffic to Pods and may have associated IP addresses, the primary role of DNS within this context is to facilitate connectivity in a more manageable manner, addressing the challenges posed by dynamic IPs in containerized environments.

SAMPLE

Next Steps

Congratulations on reaching the final section of this guide. You've taken a meaningful step toward passing your certification exam and advancing your career.

As you continue preparing, remember that consistent practice, review, and self-reflection are key to success. Make time to revisit difficult topics, simulate exam conditions, and track your progress along the way.

If you need help, have suggestions, or want to share feedback, we'd love to hear from you. Reach out to our team at hello@examzify.com.

Or visit your dedicated course page for more study tools and resources:

https://itgss-certifieddevopsengineer.examzify.com

We wish you the very best on your exam journey. You've got this!

SAMPLE