Ultimate Certified Kubernetes Administrator (CKA) Certification Guide

Become CKA Certified with Ease by Mastering Cluster Management and Orchestration with Kubernetes

Rajesh Vishnupant Gheware

www.orangeava.com

First published: July 2024

Published by: Orange Education Pvt Ltd, AVA™

Address: 9, Daryaganj, Delhi, 110002, India

275 New North Road Islington Suite 1314 London,
N1 7AA, United Kingdom

ISBN: 978-81-97651-16-8

www.orangeava.com

Dedicated To

Prof. L. S. Ganesh Sir (IIT Madras)

For Being my Unending Source of Inspiration,

Your Guidance and Wisdom have Shaped my Professional Journey

My Parents,

who I am Eternally Indebted for Shaping my Early Years of Life

My Bundles of Joy – Shreyansi and Naina,

Your Boundless Love Fuels my Ambitions,

May you Both Always Shine Brightly

My Best Critic Aka Wife – Deepti, for Your Unwavering Support

This Book Is Dedicated to You All,

With Heartfelt Gratitude and Love,

For Being the Pillars of my Life

About the Author

Rajesh Gheware is a seasoned professional in the IT industry, known for his expertise and contributions in the field of DevOps. With over two decades of experience, Rajesh has made a significant impact in the areas of cloud computing, containerization, and strategic IT architectures. His career has been marked by progressive roles, starting as a software engineer and evolving into a Chief Architect, a position he has held at several prestigious organizations.

Rajesh's journey in technology began with a strong foundation in engineering, having earned an M.Tech from IIT Madras. His passion for technology, combined with a keen understanding of business needs, has enabled him to lead complex IT projects and strategies successfully. Rajesh's technical skills are diverse, including expertise in Kubernetes, Docker, AWS services, and various software frameworks and protocols.

Rajesh boasts a diverse IT career with impactful roles at UniGPS Solutions, JP Morgan Chase, and Deutsche Bank. He spearheaded IT strategies and cutting-edge solutions, particularly in cloud computing and containerization, setting industry benchmarks. A thought leader in DevOps, Rajesh actively contributes to platforms like DZone, LinkedIn, and OpenSourceForU, offering valuable insights into DevOps tools and methodologies.

Rajesh's expertise extends to mentoring. His workshops and training sessions in Kubernetes, Docker, AWS, and DevOps practices have established him as a sought-after speaker and educator with an approachable and informative style. Known for his problem-solving skills, mentorship, and contributions to DevOps best practices, Rajesh is a respected figure in the community, driven by continuous learning and knowledge sharing.

Technical Review Partner

Thinknyx® Technologies is a team of professionals with years of experience in IT technology, ranging from Software Development to the Management of IT Infrastructure, Cloud, Automation, Container Management, Web and APP Development, Security, and Professional Services. Recognized as a reputable brand, Thinknyx® Technologies provides IT consulting services, offering comprehensive Information Technology and Soft Skills' Training. Additionally, they offer Talent Acquisition and Recruitment solutions to diverse organizations worldwide.

Mr. Yogesh Raheja, the Founder and CEO of Thinknyx® Technologies, is a certified expert in DevOps, SRE, Cloud, and Containerisation, with two decades of IT experience.

He has expertise in technologies such as Public/Private Cloud, Containers, Automation tools, Continuous Integration/Deployment/Delivery tools, Monitoring and Logging tools, and more. Mr. Raheja is passionate about sharing his technical expertise with a global audience through various forums, conferences, webinars, blogs, and LinkedIn. Moreover, he has authored multiple books, including *Effective DevOps with AWS*, *Automation with Puppet 5*, and *Automation with Ansible*, and has published his online courses on various platforms. Furthermore, he has reviewed multiple books, including *Implementing Splunk 7, Third Edition* and *Splunk Operational Intelligence Cookbook, Third Edition*, among others.

Acknowledgements

I am immensely grateful to everyone who supported me through the journey of writing this book. First and foremost, I extend my deepest appreciation to my youngest daughter, Naina, whose sharp insights and dedication to refining the content, greatly enhanced the clarity and precision of this guide. Her involvement has been invaluable, and I am profoundly thankful for her continuous help.

I also owe a huge debt of gratitude to all my family members for their unwavering patience and support. Their understanding and encouragement have been crucial as I navigate the various facets of life, always striving to learn more and share that knowledge through writing.

To all those who lent their expertise, provided feedback, and encouraged me along the way—thank you. Your collective wisdom and support have not only made this guide possible but have also made it better in every conceivable way.

Thank you all for being my pillars during this endeavor.

Preface

This book, *Ultimate Certified Kubernetes Administrator (CKA) Certification Guide*, is the product of years spent navigating the intricacies of IT architecture and witnessing the transformative impact of technology, particularly in cloud computing and containerization. The goal is to provide a comprehensive resource that empowers IT professionals. It aims to equip them for the CKA and CKS certifications while fostering a deeper understanding of Kubernetes as a vital tool for modern infrastructure management. By delving into the world of Kubernetes, readers will gain the knowledge and skills necessary to effectively manage and orchestrate containerized applications.

This guide is crafted to serve both newcomers and seasoned practitioners by breaking down complex Kubernetes concepts into understandable segments, supported by real-world applications and examples. Through this book, we share not just the technical know-how but also the strategic thinking required to harness the full potential of Kubernetes in any organizational context. Whether you're looking to certify your skills or simply aiming to enhance your technical arsenal, this guide stands as a beacon of knowledge and practical insight in the ever-evolving world of cloud-native technologies.

Chapter 1. **Introduction to Kubernetes:** This chapter introduces Kubernetes, detailing its evolution from Google's internal system to the leading open-source container orchestration platform. It covers the architecture, benefits, and fundamental concepts, providing a solid foundation for understanding how Kubernetes facilitates complex deployments and operations.

Chapter 2. **Installing Kubernetes:** This chapter delves into the practical setup of Kubernetes environments, covering essential tools, such as Oracle VirtualBox, Vagrant, and various Kubernetes installation methods, including Minikube, Microk8s, KIND, and Kubeadm. This chapter equips readers with the knowledge to successfully install and configure Kubernetes clusters on different platforms for various use cases.

Chapter 3. **Workload Objects - Pod Deploy StatefulSet**: This chapter delves into key Kubernetes concepts such as Pods, Deployments, and StatefulSets, detailing their configurations, uses, and management. It extensively covers container probes—startupProbe, readinessProbe, and livenessProbe—to monitor and improve

the health and responsiveness of applications. It is pivotal for understanding the lifecycle of Kubernetes workloads and how to ensure their robustness and reliability. It serves as a comprehensive guide to deploying and managing stable, scalable applications in a Kubernetes environment.

Chapter 4. Service and Ingress - Exposing Apps Outside the Cluster: This chapter explores the methods and strategies for exposing Kubernetes applications to internal and external network traffic through Services and Ingress Controllers. It covers different types of services including ClusterIP, NodePort, LoadBalancer, and ExternalName, detailing their use cases and configurations. The chapter also introduces Ingress as a powerful tool for managing external access to services within a cluster. This chapter is essential for mastering the networking aspects of Kubernetes, ensuring applications are accessible and secure.

Chapter 5. Deploy and Scale - Stateless Apps: This chapter delves into the deployment and scaling of stateless applications within Kubernetes. It begins by explaining how to set application-specific runtime specifications that aid Kubernetes in effectively placing Pods on Nodes, enhancing overall performance and enabling efficient autoscaling. It highlights the role of the metrics-server, an addon component that tracks CPU and memory usage, providing crucial data for managing resources. It covers the configuration and activation of the Horizontal Pod Autoscaler (HPA), the dynamic nature of application loads and discusses the HPA's strategy to maintain cluster stability.

Chapter 6. Deployment Strategies – RollingUpdate, Recreate: This chapter explores advanced deployment strategies in Kubernetes, focusing on RollingUpdate, Recreate, Blue-Green, and Canary deployments. It begins with the straightforward Recreate strategy, which involves full replacement of old instances, potentially causing downtime. It then covers the RollingUpdate method, which allows for seamless, zero-downtime updates by incrementally replacing pods. The chapter further delves into Blue-Green deployments that facilitate testing by running two identical environments before fully switching traffic to the new version. Finally, it examines Canary deployments, which roll out changes to a small subset of users to gauge impact before a broader rollout.

Chapter 7. Data Persistence - Local and Cloud: The chapter thoroughly addresses the topic of data persistence in Kubernetes, focusing on both local and cloud storage solutions. It explores the mechanisms through which Kubernetes manages persistent data across various environments, ensuring data durability

and accessibility. It details the use of Persistent Volumes (PVs) and Persistent Volume Claims (PVCs) for managing local storage resources, including their lifecycle, provisioning, and dynamic binding processes. The chapter also delves into cloud storage integrations, discussing how Kubernetes interfaces with cloud providers to leverage their storage services for stateful applications.

Chapter 8. **Deploy and Scale – StatefulSet:** This chapter is dedicated to the deployment and scaling of stateful applications using StatefulSets in Kubernetes. It provides a comprehensive understanding of StatefulSets, which are crucial for applications that require stable, unique network identifiers and persistent storage. It covers the creation, management, and scaling of StatefulSets, detailing how they maintain the order and uniqueness of pods during updates and scaling operations. The chapter also discusses strategies for effectively managing stateful resources, such as databases, within a Kubernetes environment, ensuring data consistency and high availability.

Chapter 9. **Configure Apps for Production Deployment:** The chapter focuses on configuring applications for production deployment in Kubernetes. This chapter provides detailed guidance on best practices for preparing applications for the rigors of a production environment. It covers essential topics such as setting up robust logging and monitoring, implementing efficient resource limits, and applying security configurations. It also discusses the importance of creating reliable liveness and readiness probes to ensure application health and resilience. By emphasizing automation and adherence to best practices, this chapter equips readers with the necessary tools and knowledge to configure their Kubernetes applications efficiently, ensuring smooth and stable production deployments.

Chapter 10. **Cluster Database - Backup and Restore:** The chapter is dedicated to the comprehensive management of the ETCD cluster database, which is critical for maintaining the state and configuration of Kubernetes clusters. It begins with an in-depth look at understanding ETCD and its role within Kubernetes, followed by detailed discussions on how to configure, manage, and secure ETCD clusters effectively. It covers key aspects of backup and disaster recovery strategies, ensuring data integrity and availability even in the face of system failures. The chapter also addresses scaling ETCD clusters to meet growing demands and optimizing performance for high availability.

Chapter 11. **Cluster Upgrade – Kubeadm:** This chapter is centered on upgrading Kubernetes clusters using kubeadm. It starts with an overview of the Kubernetes

release cycle and versioning to set the stage for understanding the importance of timely and efficient upgrades. It then progresses into detailed preparation steps necessary for a successful upgrade, including backup strategies and the importance of reviewing release notes. The chapter concludes with insights into rollback strategies and troubleshooting common issues, along with best practices to ensure a smooth and effective upgrade process, making it an essential resource for anyone responsible for maintaining Kubernetes clusters.

Chapter 12. **CoreDNS:** This chapter thoroughly examines CoreDNS within Kubernetes, emphasizing its architectural components, deployment, and pivotal role in DNS resolution and service discovery. It details customization and configuration strategies, explores its integration with Kubernetes networking, and addresses performance, scalability, and security challenges. The chapter concludes by providing solutions for common operational issues, making it a vital resource for Kubernetes administrators.

Chapter 13. **Networking - Pod Service and Ingress:** This chapter dives into the Kubernetes networking model, explaining how containerized applications benefit from unique and efficient networking approaches. It covers intra-pod communication through shared network namespaces, inter-pod connectivity across the cluster via a flat network and CNI plugins, and the role of Kubernetes services in providing stable endpoints for load balancing and service discovery. Additionally, it details the mechanisms for external access to services, such as NodePort, LoadBalancer, and Ingress, highlighting their roles in external communication and access control.

Chapter 14. **Kubernetes CNI:** This chapter delves into the intricacies of Kubernetes Container Network Interface (CNI), crucial for the implementation and management of network plugins that facilitate secure and isolated container communications. It provides detailed insights into the functionality, selection, and configuration of popular CNI plugins, including Calico, Flannel, and Weave Net, illustrating how they integrate with kubelet to ensure efficient pod networking. This chapter is vital for administrators and developers seeking to understand and leverage the full capabilities of networking within Kubernetes environments.

Chapter 15. **Kubernetes Security:** This chapter offers an extensive examination of Kubernetes security, detailing its comprehensive security model which includes the API server as a central access point for authentication, authorization, and admission control. It explores the implementation of Pod Security Standards

and the transition towards more streamlined security practices that enhance flexibility in policy enforcement, auditing, and warning. It also covers Role-Based Access Control (RBAC), network policies, securing the control plane, and the importance of audit logging. By integrating these security measures with TLS for secure communications, the chapter outlines how to significantly strengthen the security posture of Kubernetes environments, ensuring resource protection and adherence to best security practices.

Chapter 16. **Troubleshooting:** This chapter is dedicated to troubleshooting within Kubernetes environments, providing essential techniques for diagnosing and resolving common issues that arise in cluster management and application deployment. The chapter thoroughly covers health checks of cluster components, application failure analysis, service connectivity, and network troubleshooting, including DNS resolution and load balancing. It places significant emphasis on practical skills, such as effective use of `kubectl` commands, understanding Kubernetes events, and verifying configurations. Additionally, the chapter explores complex scenarios involving resource quota management, pod security, persistent storage configurations, and advanced deployment strategies like blue-green deployments. By blending theory with hands-on practices, this chapter equips Kubernetes administrators with the tools necessary to maintain security, performance, and resource efficiency, preparing them for real-world challenges and enhancing their proficiency for the Certified Kubernetes Administrator (CKA) exam.

Chapter 17. **Kubernetes Production Essentials:** This chapter addresses the critical steps needed to enhance a basic Kubernetes cluster to a production-grade environment by integrating essential third-party tools. The chapter provides detailed guidance on deploying the NGINX Ingress Controller for traffic management, Cert-Manager for TLS certificate automation, and Metrics Server for monitoring resources. It also explores the implementation of the EFK stack (Elasticsearch, Fluent-bit, Kibana) for advanced logging and integrates Prometheus and Grafana for in-depth monitoring and analytics. The importance of Application Performance Monitoring tools is highlighted to gain real-time insights into application performance. This comprehensive approach not only enhances cluster performance, security, and observability but also ensures the cluster's readiness for production deployments, making it robust and resilient for handling real-world workloads effectively.

Chapter 18. Microservices Observability: This chapter provides a detailed exploration of observability within Kubernetes environments, focusing on microservices architecture. It underscores the importance of comprehensive monitoring and management practices, incorporating logging, distributed tracing, and metrics collection as foundational elements. The chapter guides Kubernetes practitioners through best practices and tools like Elasticsearch, Fluent-bit, Kibana, APM Server, Prometheus, and Grafana, which are essential for building a robust observability framework. Through practical examples, including a case study on optimizing a Weather Application, the chapter illustrates how logs, tracing, and metrics—collectively the three pillars of observability—play critical roles in enhancing the understanding, performance, and reliability of microservices. This knowledge empowers IT professionals to improve the resilience and efficiency of their systems, ensuring scalable and performance-optimized deployments.

Chapter 19. Scalable Jenkins on Kubernetes: This chapter focuses on optimizing Jenkins for scalability within Kubernetes environments. The chapter dives into effective strategies for a scalable Jenkins setup, particularly highlighting the use of external storage solutions and the dynamic creation of pods for job execution using the Jenkins Kubernetes plugin. Detailed, step-by-step guidance on configuring this setup is provided, followed by practical demonstrations through an inline pipeline example. This approach not only enhances Jenkins' performance but also its flexibility, making it well-suited for cloud-native ecosystems. The chapter serves as a comprehensive guide for professionals looking to leverage Kubernetes to improve the efficiency and scalability of their Jenkins pipelines.

Chapter 20. GitOps using ArgoCD and GitHub: This chapter delves into GitOps, a modern methodology that revolutionizes DevOps by integrating Git with operational workflows for enhanced efficiency and control. The chapter thoroughly explores the use of GitHub and ArgoCD, a Kubernetes-native continuous delivery tool, to implement GitOps practices effectively. It demonstrates how these technologies can streamline the deployment process within Kubernetes environments, making it more transparent and automated. Through practical implementation, the chapter outlines how to establish a robust CI/CD pipeline combining Jenkins and ArgoCD, ensuring consistent deployment and synchronization of Kubernetes applications with their desired states in GitHub. This setup not only enhances operational efficiency but also aligns with the best practices of cloud-native application management, providing clear insights into integrating GitOps within modern DevOps frameworks.

Chapter 21. CKA Exam Mastery: This chapter serves as a crucial guide for mastering the Certified Kubernetes Administrator (CKA) exam. The chapter provides effective strategies and insights, emphasizing the necessity of a deep, comprehensive understanding of Kubernetes coupled with extensive, hands-on experience. It details a targeted approach to excel in the exam, focusing on the critical areas such as cluster architecture, workloads, services, networking, and troubleshooting. Highlighting the practical, hands-on nature of the exam, this chapter equips candidates with the knowledge and skills required to succeed in achieving one of the most esteemed certifications in cloud-native technologies.

Chapters 18, 19 and 20, while not directly part of the Certified Kubernetes Administrator (CKA) syllabus, offer invaluable knowledge for professionals operating in modern cloud-native environments. Chapter 18 delves into microservices observability, equipping practitioners with the tools and strategies needed for effective monitoring and management, crucial for maintaining robust and scalable services. Chapter 19 explores the integration of Jenkins within Kubernetes, focusing on scalable CI/CD practices that enhance development pipelines and operational efficiency. Chapter 20 introduces GitOps using ArgoCD and GitHub, presenting advanced deployment techniques that streamline and secure application management processes. These chapters provide essential insights and practical skills that transcend certification requirements, preparing professionals to tackle real-world challenges in Kubernetes deployments and operations, thus adding significant value to their expertise in the evolving landscape of cloud technologies.

Downloading the code bundles and colored images

Please follow the links or scan the QR codes to download the ***Code Bundles and Images*** of the book:

https://github.com/OrangeAVA/Ultimate-Certified-Kubernetes-Administrator-CKA-Certification-Guide

The code bundles and images of the book are also hosted on ***https://rebrand.ly/0a6fd4***

In case there's an update to the code, it will be updated on the existing GitHub repository.

Errata

We take immense pride in our work at **Orange Education Pvt Ltd,** and follow best practices to ensure the accuracy of our content to provide an indulging reading experience to our subscribers. Our readers are our mirrors, and we use their inputs to reflect and improve upon human errors, if any, that may have occurred during the publishing processes involved. To let us maintain the quality and help us reach out to any readers who might be having difficulties due to any unforeseen errors, please write to us at :

errata@orangeava.com

Your support, suggestions, and feedback are highly appreciated.

DID YOU KNOW

Did you know that Orange Education Pvt Ltd offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at **www.orangeava.com** and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at: **info@orangeava.com** for more details.

At **www.orangeava.com**, you can also read a collection of free technical articles, sign up for a range of free newsletters, and receive exclusive discounts and offers on AVA™ Books and eBooks.

PIRACY

If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at **info@orangeava.com** with a link to the material.

ARE YOU INTERESTED IN AUTHORING WITH US?

If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, please write to us at **business@orangeava.com**. We are on a journey to help developers and tech professionals to gain insights on the present technological advancements and innovations happening across the globe and build a community that believes Knowledge is best acquired by sharing and learning with others. Please reach out to us to learn what our audience demands and how you can be part of this educational reform. We also welcome ideas from tech experts and help them build learning and development content for their domains.

REVIEWS

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions. We at Orange Education would love to know what you think about our products, and our authors can learn from your feedback. Thank you!

For more information about Orange Education, please visit **www.orangeava.com**.

Table of Contents

1. Introduction to Kubernetes 1
Introduction 1
Structure 1
Overview 1
Benefits of Kubernetes 2
Service Discovery and Load Balancing 2
Automated Rollouts and Rollbacks 2
Automatic Bin Packing 3
Self-Healing 3
Storage Orchestration 3
Secrets and Configuration Management 3
Kubernetes Architecture 4
Logical View 4
Control Plane 5
API Controller 5
ETCD Database 5
Scheduler 5
Controller (KCM - Kube Controller Manager) 5
Cloud Controller (CCM - Cloud Controller Manager) 6
Worker Plane 6
Kubelet 6
Kube-proxy 6
Dynamic View 6
Conclusion 7
Points to Remember 8
Multiple Choice Questions 8
Answers 8
Questions 9
Keywords 9

Further Reading 9
2. Installing Kubernetes 10
Introduction 10
Structure 10
Basic Tools 11
Installing Oracle VirtualBox 11
Installing `Vagrant` 11
`Minikube` 12
Launching Ubuntu 12
Installing `Docker` 13
Installing `Minikube` 14
Launching Kubernetes Cluster 14
Testing the `Minikube` Kubernetes Cluster 14
Cleaning Up `Minikube` Cluster 15
`Microk8s` 15
Testing the `Microk8s` Kubernetes Cluster 16
Cleaning Up `Microk8s` Cluster 16
KIND - Kubernetes INside Docker 16
Testing the KIND Kubernetes Cluster 17
Multi-node KIND Cluster 18
Cleaning Up KIND Cluster 18
Kubeadm 19
Launching Cluster Nodes 19
Initializing the Master/Control Plane Node 21
Testing the `Kubeadm` Kubernetes Cluster 23
Cleaning Up the `Kubeadm` Cluster 24
Conclusion 24
Points to Remember 24
Multiple Choice Questions 25
Answers 26
Questions 27
Keywords 27

3. Workload Objects – Pod, Deploy, StatefulSet 28
Introduction 28
Structure 28
`Pod` 28
Running `Pods` 29
Restart Policy 30
`Pod` Lifecycle 31
Container Probes 31
`startupProbe` 31
`readinessProbe` 32
`livenessProbe` 33
`Deployment/Deploy` 33
Scalability 35
Rollout of a New Change 36
Rollback 36
StatefulSet 37
Limitations 37
Running StatefulSet - Postgre Database 37
Conclusion 39
Points to Remember 40
Multiple Choice Questions 40
Answers 42
Questions 42
Keywords 42
4. Service and Ingress - Exposing Apps Outside the Cluster 43
Introduction 43
Structure 43
Service 43
Creating a Service 44
ClusterIP 46
`NodePort` 47
Running a `NodePort` Service 47

Clean Up 48
`LoadBalancer` 48
Running a `LoadBalancer` Service 48
`ExternalName` 49
Ingress 50
Installing Ingress Controller 50
Running the Hello App 50
Setting Up Load Balancer - Nginx 51
Testing the App URL 53
Interaction Summary - Ingress 53
Conclusion 54
Points to Remember 54
Multiple Choice Questions 54
Answers 56
Questions 56
Keywords 56
5. Deploy and Scale - Stateless Apps 57
Introduction 57
Structure 57
Setup 58
Metrics Server 58
Deployment 60
Scalability 62
Auto-scaling/HPA - Horizontal Pod Autoscaler 64
Conclusion 68
Points to Remember 69
Multiple Choice Questions 69
Answers 71
Questions 71
Keywords 71

6. Deployment Strategies - RollingUpdate, Recreate 72
Introduction 72
Structure 72
Deployment Strategies 73
`RollingUpdate` 73
`Recreate` 75
`Blue Green` (aka Red Black) 75
`Canary` Deployment 79
Conclusion 80
Points to Remember 81
Multiple Choice Questions 81
Answers 83
Questions 83
Keywords 83
7. Data Persistence - Local and Cloud 84
Introduction 84
Structure 84
Pod Storage (`emptyDir`) 85
Node Storage (`hostPath`) 86
Persistent Volume 87
`PersistentVolumeClaim` 87
`PersistentVolume` Types 89
Access Modes 89
Storage Class 89
Cloud Storage - AWS EBS 90
Conclusion 91
Points to Remember 91
Multiple Choice Questions 92
Answers 93
Questions 94
Keywords 94

8. Deploy and Scale - StatefulSet 95
Introduction 95
Structure 96
Overview 96
Key Features 97
Use Case - Redis Database 97
Scaling 100
Pod Management Policies 101
Rolling Updates 101
Partitioned Rolling Updates 103
Limitations 103
Conclusion 103
Points to Remember 104
Multiple Choice Questions 105
Answers 106
Questions 107
Keywords 107
9. Configure Apps for Production Deployment 108
Introduction 108
Structure 108
Overview 109
ConfigMap 109
Container `env` 110
`Volume mount` 112
Command Line Arguments 114
Secret 114
Probes 117
`startupProbe` 117
`readinessProbe` 119
`livenessProbe` 121
Probe Mechanisms 122
Conclusion 124

Points to Remember 124
Multiple Choice Questions....125
Answers....127
Questions....127
Keywords 128
10. Cluster Database - Backup and Restore....129
Introduction.... 129
Structure.... 129
Understanding ETCD....130
Core Features of ETCD.... 130
ETCD's Role in Kubernetes Cluster Management.... 130
ETCD's Consensus Algorithm (Raft).... 130
How Kubernetes Uses ETCD.... 131
Understanding Data Stored in ETCD 131
Data Model and Key Schemas....132
ETCD Cluster Configuration and Management....132
Setting Up an ETCD Cluster 132
Configuring ETCD for Kubernetes 132
Backup and Disaster Recovery....134
Importance of ETCD Backup....134
ETCD Data Manipulation and Access....134
Accessing ETCD with `etcdctl`....134
Basic Commands 135
Common `etcdctl` Commands and Operations....136
Step-by-Step Guide to Backing Up and Restoring ETCD 136
Backing Up ETCD....136
ETCD Data Restoration.... 137
Scaling ETCD Clusters.... 139
Security....139
Conclusion.... 139
Points to Remember 140
Answers....142

Questions....142
Key Terms....143
11. Cluster Upgrade – kubeadm....144
Introduction....144
Structure....144
Understanding the Release Cycle....145
Preparing for Cluster Upgrade....145
Executing the Cluster Upgrade....145
Installing KIND Utility....145
Launching the Cluster....146
Upgrading Worker Nodes....149
Version Skew and Compatibility Considerations....152
Rollback and Troubleshooting....152
Conclusion....153
Points to Remember....153
Multiple Choice Questions....153
Answers....155
Questions....155
Keywords....155
12. CoreDNS....156
Introduction....156
Structure....156
Overview of CoreDNS....157
Comparison with Kube-DNS....157
CoreDNS Architecture and Components....158
Key Components of CoreDNS....158
CoreDNS Integration with Kubernetes....158
CoreDNS Deployment and Configuration....159
Configuration....160
Understanding `Corefile` File....160
Customizing CoreDNS....161
DNS in Kubernetes Networking....162

CoreDNS in Pods 162
Interaction with Other Components 163
Performance and Scalability 163
Performance Considerations 163
Scaling CoreDNS 164
Security: Safeguarding Kubernetes DNS Services 164
Securing CoreDNS Deployment 164
Operational challenges - CoreDNS 165
Conclusion 166
Points to Remember 166
Multiple Choice Questions 166
Answers 168
Questions 168
Keywords 168
13. Networking - Pod Service and Ingress 169
Introduction 169
Structure 169
Kubernetes Networking Model 169
Understanding Within Pod communication (Multi Containers) 171
Understanding Pod-to-Pod Communication - Same Node 173
Understanding Pod-to-Pod Communication - Different Nodes 175
Understanding Pod-to-Service Communication 176
Understanding External-to-Service Communication 177
Via `NodePort` Service 177
Via `LoadBalancer` 178
Ingress-to-Service Communication 178
Conclusion 180
Points to Remember 180
Multiple Choice Questions 181
Answers 182
Questions 183
Key Terms 183

14. Kubernetes CNI 184
Introduction 184
Structure 184
Overview: Kubernetes CNI Model 185
CNI Plugins 185
Selection Criteria 186
Installing CNI Plugin - Calico 186
Kubelet and CNI Plugin Discovery 188
Pod Networking Setup 188
Calico CNI Plugin 189
Network Policy 190
Network Policy Enforcement by Calico 191
Flannel CNI Plugin 192
Weave Net CNI Plugin 193
Conclusion 194
Points to Remember 194
Multiple Choice Questions 195
Answers 196
Questions 196
Key Terms 197
15. Kubernetes Security 198
Introduction 198
Structure 198
Overview - Kubernetes Security Model 199
API Server - Central Access Point 200
Authentication and Authorization 200
Admission Control 201
Pod Security Standards 201
RBAC - Role Based Access Control 202
Example - Access to Dev Teams 203
Example - Jenkins CICD 205
Network Policy 205

Example - HR and Payroll apps 206
Creating Network Policies 206
Testing Network Policies 208
Securing Control Plane 209
Audit Logging 210
Example - Audit Policy 211
Conclusion 211
Multiple Choice Questions 212
Answers 213
Questions 214
Key Terms 214
16. Troubleshooting 215
Introduction 215
Structure 215
Overview 215
Troubleshooting Scenarios 216
Scenario 1: Network Policy Verification 216
Scenario 2: Resource Quota Management 217
Scenario 3: Investigating Node Issues 218
Scenario 4: Service and Ingress Troubleshooting 218
Scenario 5: Persistent Volume Troubleshooting 219
Scenario 6: `ConfigMap` and Secret Management 221
Scenario 7: Role-Based Access Control (RBAC) Configuration 222
Scenario 8: Service and Network Policy Configuration 224
Scenario 9: Securing Sensitive Data with Secrets 225
Scenario 10: Kubernetes Cluster Upgrade 227
Scenario 11: Setting Up and Troubleshooting Cluster Monitoring 228
Scenario 12: Optimizing Application Resource Usage 229
Scenario 13: Implementing and Testing Network Policies 230
Scenario 14: Cluster Backup and Restore 233
Scenario 15: Auto-Scaling Based on Custom Metrics 234
Scenario 16: Securing Sensitive Data in Transit and at Rest 235

Scenario 17: Implementing Advanced Scheduling Techniques 237
Scenario 18: Setting Up Cluster-Wide Logging 240
Scenario 20: Configuring Ingress for Service Routing and SSL Termination 241
Scenario 21: Debugging Cluster Network Issues 244
Scenario 22: Configuring Persistent Storage for Stateful Applications 245
Scenario 23: Managing Resource Quotas and Namespace Configurations 247
Scenario 24: Implementing a Blue-Green Deployment Strategy 249
Conclusion 252
Points to Remember 253
Multiple Choice Questions 253
Answers 255
Questions 255
Key Terms 256
17. Kubernetes Production Essentials 257
Introduction 257
Structure 257
Overview 258
Production Essentials - Extensions 258
CNI Plugin 258
Metrics Server 259
Installation 259
Verification 259
Ingress Controller 259
Installation 260
Verification 260
Certificate Manager 260
Installation 260
Configuration 260
Verification 261

Prometheus and Grafana 262
Installation 262
Verification 263
EFK and Elastic APM 263
Installation 263
Verification 264
Conclusion 264
Multiple Choice Questions 265
Answers 266
Questions 267
Key Terms 267
18. Microservices Observability 268
Introduction 268
Structure 268
Overview 269
Observability 269
Tools Setup 269
Event Logs - EFK 270
Distributed Tracing - APM Server 272
Installation - APM Server 272
Weather Application - Architecture 273
Installation - Weather Application 273
Metrics - Prometheus and Grafana 276
Installation - Prometheus and Grafana 276
Metrics - Application 277
Metrics - Cluster 277
Conclusion 278
Points to Remember 278
Multiple Choice Questions 279
Answers 280
Questions 280
Key Terms 281

19. Scalable Jenkins on Kubernetes 282
Introduction 282
Structure 282
Overview 283
Scalable Jenkins 283
Installation - Jenkins 283
Access Control 284
Cloud Kubernetes Plugin setup 285
Conclusion 287
Points to Remember 288
Multiple Choice Questions 288
Answers 289
Questions 289
Key Terms 290
20. GitOps using ArgoCD and GitHub 291
Introduction 291
Structure 291
Overview 291
GitOps - Setup 292
Conclusion 297
Points to Remember 298
Multiple Choice Questions 298
Answers 299
Questions 299
Key Terms 299
21. CKA Exam Mastery 300
Introduction 300
Structure 300
Overview 300
Exam Areas 301
Format 302
Scenario-Based Tasks 302

Tips for Success 303
Conclusion 304
Points to Remember 304
Multiple Choice Questions 304
Answers 310
Questions 311
Key Terms 311
Index 313

Chapter 1
Introduction to Kubernetes

Introduction

In the pre-Kubernetes era, the cost of deploying and managing software was considerably high. Moreover, the operations team faced formidable challenges with regard to replicating software within and across data centers, geographical regions, and for increased load. These were just a few of the numerous complex requirements, making deploying and managing a highly available distributed system a nightmare.

Google knew that these challenges would become bigger with time and thus developed a cluster management software called 'Borg'. With this software, Google was running billions of containers every week on machines spread across multiple geographic locations. Nearly a decade later, Google open-sourced this software under the name of Kubernetes, and a year later, in 2015, handed it over to the Linux Foundation.

Structure

In this chapter, we will cover the following topics:

- Overview
- Benefits of Kubernetes
- Kubernetes Architecture
    - Logical View
    - Dynamic View

Overview

Kubernetes is an open-source software used for the management of containerized applications. This includes managing the scalability of applications, facilitating

automated deployment, and so on. In a nutshell, Kubernetes is a cluster management software that lets you create and deploy highly available distributed applications.

Benefits of Kubernetes

Following is the list of a few benefits that Kubernetes offers:

- Service discovery and load balancing
- Automated rollouts and rollbacks
- Automated bin packing
- Self-healing
- Storage orchestration
- Secrets and configuration management

You will learn and experience these benefits/advantages in greater detail in subsequent chapters. We will provide a brief explanation of the benefits mentioned here.

Service Discovery and Load Balancing

Kubernetes provides a service-level abstraction to expose your application. This service abstraction enables decoupling between callers of your application and the application itself. For example, a frontend application can call the backend application via the Service URL - a stable network URL that can be accessed within the cluster from anywhere. While the backend application replicas may be running on different nodes or replaced by new ones, the frontend application does not need to worry about which backend application replica responds. This is similar to applications accessing a database using the database URL without worrying about the underlying database instances.

Besides providing decoupling between the caller and the callee, Kubernetes Service also acts as a load balancer. For example, if there is more than one replica of a backend application, then Kubernetes will route the request to one of the available backend replicas. This routing is mostly round-robin in nature; however, it can be customized. This, in turn, helps cater to varying levels of load on the backend application.

Automated Rollouts and Rollbacks

Often in the enterprise world, many replicas of applications are deployed on dozens if not hundreds of servers. Now rolling out any new change would not only be cumbersome but also impractical in most cases. Using Kubernetes, you can roll out new changes or hotfixes with just a single line of command, and it will take care of automatically rolling out this change in the cluster, no matter how big it is.

In case of any issue during or post-rollout, you can roll back too! Again, using a single line of command, you can instruct Kubernetes, and it will roll back automatically.

This automated rollout and rollback are generally pretty quick, meaning that you can see the changes happening within a few minutes.

Automatic Bin Packing

In the pre-Kubernetes era, deployment procedures invariably contained details of servers on which the application was to be deployed. The Operations team would then deploy the application, ensuring adequate capacity was available on those hosts where the application was to be deployed.

In the Kubernetes world, you just need to specify the application's runtime requirements, be it in RAM, CPU, or even parameters like disk type, GPU availability, and more. Kubernetes will then find the appropriate server(s) in the cluster to deploy the application onto. This is called Automatic Bin Packing.

Self-Healing

Some of the application pods may crash or hang, or even the node on which they are running may go off the network or keep crashing. This presents a challenging scenario to the Operations teams, who often end up spending a significant amount of effort, sometimes even sleepless nights, recovering from such failures to ensure the high availability of applications.

Kubernetes runs continuous checks through its controller component, identifies failures, and launches application pods on the available computer capacity (servers) to ensure the desired number of application replicas are running.

Storage Orchestration

Most enterprise applications need either temporary or permanent storage to work with. Kubernetes is designed to provide access to various kinds of storage through Volume APIs, regardless of whether storage is needed for the application's current runtime, across multiple runtimes, or post-restart to maintain the application state. Kubernetes is also designed to work with many external storage providers such as AWS EBS, Azure, Google, Ondat, CephFS, GlusterFS, PortWorx, Cinder, and so on.

Secrets and Configuration Management

Sensitive information required by the applications, such as the API key, can be stored in the **`Secret`** object provided by Kubernetes. All the information kept in the **`Secret`**

object is base64 encoded by Kubernetes. Information stored in these **`Secret`** objects can be referenced in the application via environment variables or container file systems.

Kubernetes provides a **`ConfigMap`** object to store runtime inputs, say environment variables. This gives the flexibility to deploy the application in various environments like test, pre-prod, prod, and more. Information stored in the Config object is made available to the application at runtime via environment variables or container file system. Kubernetes also ensures that if there is any change in the Config object, then the corresponding application is relaunched automatically to reflect the change in the Config for that particular application.

Kubernetes Architecture

Architecture can be best understood by looking at the system from various perspectives/views. We will first examine Kubernetes from a logical view. Then, to understand how different components interact with each other, we will use a dynamic view in the form of a sequence diagram.

Logical View

Now, let us understand the different components that make up the architecture of Kubernetes and how these components interact with each other.

Figure 1.1 shows the logical overview of the Kubernetes cluster:

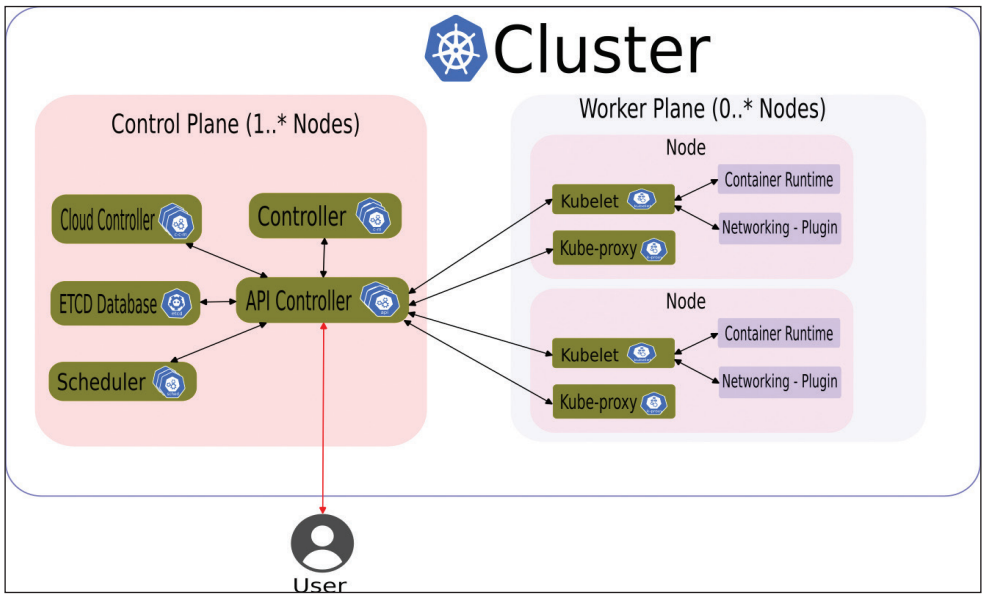

***Figure 1.1**: Kubernetes Architecture - Logical View*

Control Plane

The control plane consists of one or more servers, also called controller nodes, where the key components that are responsible for managing the Kubernetes cluster are deployed. All control plane components are designed for distributed scaling, and the number of nodes in the control plane is typically an odd number like 1, 3, 5, 7, and so on. Generally, it is 3 or more in the enterprise setups.

API Controller

This is the central component in Kubernetes architecture as all other parts/components interact with the API controller. This is the only component that interacts with the ETCD database to store and retrieve cluster information. To manage or operate on the cluster, the User (the external entity to the cluster) can send execution commands using Command Line Interface (CLI) known as '**`Kubectl`**'. The API Controller exposes itself over the network, allowing utilities like **`kubectl`** to send commands (JSON messages over HTTPS) to control the Kubernetes cluster.

ETCD Database

This component holds the state of the cluster, and the only component allowed to manage the state of the cluster is the API Controller. Cluster state information is stored in the ETCD database in the form of key and value pairs. Like other control plane components, the ETCD database is designed to be a highly available and scalable software component. It exposes itself over the network and on port 2379.

Scheduler

The main responsibility of the scheduler is to schedule pods on appropriate nodes. In Kubernetes, pods are the smallest deployable units. A pod may contain one or more containers. The scheduler watches for all newly created pods and assigns nodes to them so that Kubelet can launch those on the assigned nodes. While assigning nodes to the pod, the scheduler takes into account various computing needs required by the pod such as CPU, memory, disk type, GPU, affinity/anti-affinity preferences, and so on.

Controller (KCM - Kube Controller Manager)

Controller, also known as Kube Controller Manager (KCM), is a group of controller processes packaged together to reduce deployment complexity. Following are some of the controller processes:

- **Node controller**: Monitors and responds whenever the node goes down.
- **Job controller**: Watches for tasks represented as Jobs and launches them.