International Journal of Trend in Scientific Research and Development (IJTSRD)
Special Issue on Smart Innovations in Computer Science and Applications
Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

Deploying Application Using Kube Rnetes

Vivek Someshwar Ghosekar
Department of Science and Technology,
G. H. Raisoni Skill Tech University, Nagpur, India

Annotation

In modern software applications, it is necessary to ensure that the applications are deployed in scalable, reliable, and efficient environments. It has been identified that traditional application deployment techniques face difficulties in handling modern distributed applications. This has created problems in terms of scalability, reliability, and maintenance. However, an open-source system called Kubernetes has been proposed to resolve the problems associated with modern application deployment. This research paper aims to explore the architecture and working of the Kubernetes system in application deployment. This paper is based on the study of the application of Kubernetes in modern application deployment. It has been identified that the application of Kubernetes in modern application deployment is effective in terms of system reliability, scalability, and efficiency.

This research paper explores the architecture, components, and working principles of Kubernetes in application deployment. It also analyzes deployment workflows, strategies, and real-world implementation scenarios. The study concludes that Kubernetes significantly enhances system reliability, scalability, and operational efficiency, making it a cornerstone technology in modern DevOps and cloud-native environments.

Keywords: Kubernetes, Containerization, DevOps, Docker, Microservices, Cloud Computing, Application Deployment.

1. Introduction

1.1. Background

Due to the introduction of cloud-based computing, more businesses are creating services and products on distributed architecture—application development has therefore changed from an entirely independent application to a distributed application development ecosystem based on various services.

As these services grow in number, it becomes increasingly difficult to manage them. By using Docker to create containerized applications, businesses are able to create those applications in a consistent manner across different environments (e.g., development vs production). While Docker helps with deploying containerized applications into multiple environments, many businesses have created hundreds of containers that run on multiple servers; therefore, it has become even harder for IT teams to manage these containers across all of the servers.

Kubernetes is currently the best technology to manage containerized applications.

1.2. Motivation

Traditional application deployment faces several challenges:

1. Manual configuration and deployment errors
2. Difficulty scaling applications dynamically
3. Limited fault tolerance and recovery
4. Inefficient resource utilization

Kubernetes solves these problems by providing automated orchestration and infrastructure abstraction.

1.3. Objectives

The objectives of this research are:

1. To analyze Kubernetes architecture for application deployment
2. To study container orchestration mechanisms
3. To evaluate Kubernetes deployment workflow
4. To understand advantages of Kubernetes in DevOps environments

2. Related Work

2.1. Containerization Technologies

Applications can be consistently deployed in multiple environments by utilizing containerization. One of the most popular containerization platforms is Docker. An application code together with runtime, libraries, and dependencies are packaged into a container called a "Container" and executed as an isolated lightweight volume that may be placed in any environment.

2.2. Container Orchestration

Container orchestration platforms manage container deployment across clusters. Popular tools include:

- Kubernetes
- Docker Swarm
- Apache Mesos

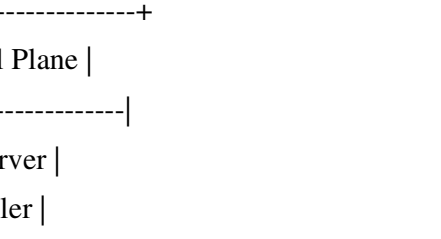
Among these, Kubernetes has become the industry standard due to its scalability and strong community support.

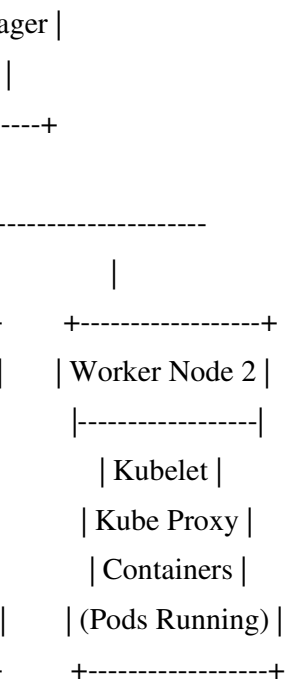
2.3. Kubernetes in Cloud Environments

Cloud providers such as AWS, Azure, and Google Cloud offer managed Kubernetes services like:

- Amazon EKS
- Azure AKS
- Google Kubernetes Engine

These services simplify cluster management and enable organizations to focus on application development rather than infrastructure maintenance.





3. Kubernetes Architecture

3.1. Overview

Kubernetes follows a **master-worker architecture** where the control plane manages worker nodes responsible for running containerized applications.

Main Components

Control Plane Components:

- API Server
- Controller Manager
- Scheduler
- etcd (cluster data store)

Worker Node Components:

- Kubelet
- Container Runtime
- Kube Proxy

3.2. Kubernetes Objects

Pod

A Pod is the smallest deployable unit in Kubernetes and contains one or more containers.

Deployment

Deployment ensures that the desired number of Pods are running.

Service

A Service exposes Pods to network traffic and ensures stable connectivity.

ConfigMap and Secret

These store configuration data and sensitive information securely.

4. Application Deployment Workflow

Step 1: Create Docker Image

docker build -t myapp .

Step 2: Push Image to Registry

docker push mydockerhub/myapp

Step 3: Create Deployment YAML

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: myapp-deployment
spec:
 replicas: 3
 selector:
 matchLabels:
 app: myapp
 template:
 metadata:
 labels:
 app: myapp
 spec:
 containers:
 - name: myapp
 image: mydockerhub/myapp
 ports:
 - containerPort: 80
```

Step 4: Deploy Application

kubectl apply -f deployment.yaml

Step 5: Expose Service

kubectl expose deployment myapp-deployment --type=NodePort --port=80

Developer Code

↓

Docker Build Image

↓

Push to Docker Hub

↓

Create YAML File

↓

kubectl Apply

↓

Kubernetes Cluster

↓

Pods Created

↓

Service Exposed to Users

5. Kubernetes Deployment Strategies

5.1 Rolling Update

Gradually replaces old application versions with new ones without downtime.

5.2 Blue-Green Deployment

Two identical environments are used where traffic switches from the old version to the new version.

5.3 Canary Deployment

New application version is released to a small group of users before full deployment.

6. Benefits of Kubernetes

Scalability

Automatically scales applications based on workload.

High Availability

Ensures applications remain available by restarting failed containers.

Self-Healing

Kubernetes automatically replaces failed containers.

Load Balancing

Distributes network traffic across multiple Pods.

Resource Optimization

Efficiently uses system resources across cluster nodes.

7. Challenges and Limitations

Despite its advantages, Kubernetes also has some limitations:

1. Complex learning curve
2. Requires proper cluster configuration
3. Monitoring and debugging can be challenging
4. Security configuration requires expertise

Future improvements may include better automation tools and improved monitoring frameworks.

8. Case Study: Deploying a Web Application

A sample web application was deployed using Kubernetes with three replicas. The deployment demonstrated:

- Automatic scaling when traffic increased

- Self-healing when containers crashed
- Efficient resource utilization across nodes

The results show that Kubernetes significantly improves system reliability and reduces manual operational effort.

9. Conclusion

Kubernetes has changed the way companies deploy applications. Kubernetes enables companies to grow and manage their resources efficiently as well as make their applications reliable across many physical locations around the world. Companies adopting Kubernetes will be able to deploy applications more quickly; their applications will have greater levels of reliability; managing their infrastructure will become easier. Kubernetes is critical to developing the future of cloud-native applications.

References

1. Burns, B., Grant, B., Oppenheimer, D., Brewer, E., and Wilkes, J. In ACM Queue: "Bork, Omega and Kubernetes." (2016)
2. Hightower, K., Burns, B. and Beda, J. In O'Reilly Media: "Kubernetes: Up and Running." (2017)
3. Merkel, D., in Linux Journal: "Docker: Lightweight Linux Containers." (2014)
4. Pahl, C., in IEEE Cloud Computing: "Containerization and the PaaS Cloud." (2015)
5. Kubernetes Documentation is also available at https://kubernetes.io/doc