




# OpenStack vs Kubernetes on System Architecture: Benchmarking from the Perspective of DevOps and Cloud Networks Connections

I Putu Agus Eka Pratama<sup>1</sup>, I Nyoman Didik Ariadi<sup>2</sup>, and  
Mahmud Dwi Sulistiyo<sup>3</sup>

<sup>1</sup> Dept. of Information Technology, Faculty of Engineering, Udayana University, Indonesia  
eka.pratama@unud.ac.id

<sup>2</sup> Dept. of Information Technology, Faculty of Engineering, Udayana University, Indonesia  
komingdidik04@gmail.com

<sup>3</sup> Artificial Intelligence Laboratory, School of Computing, Telkom University, Indonesia  
mahmuddwis@telkomuniversity.ac.id

**Abstract.** In the field of industrial information technology, especially in system architecture, system management can be done based on DevOps and Cloud Computing. Cloud Computing offers three types of services and four deployment models, while DevOps helps produce stable software products while increasing their value. OpenStack and Kubernetes are two tools commonly used in DevOps and Cloud; OpenStack is specifically for building and managing cloud infrastructure, while Kubernetes is for managing applications in an organized and scalable manner through DevOps. Although OpenStack and Kubernetes can be used together in industry for system architecture, it is important to compare OpenStack and Kubernetes from the perspective of Cloud network connections and DevOps. Both tools were tested, and then the test results were recorded and made into a comparative chart. Based on the results of cloud network connection testing, Kubernetes has a higher number of connections because it has container orchestration, which functions to manage and run applications. In DevOps testing, OpenStack is easier and faster to use because data can be created directly from the Dashboard without having to go through the Terminal.

**Keywords:** Cloud Computing, DevOps, Kubernetes, OpenStack, system management.

## 1 Introduction

System is one of the important things in the world of Information Technology, so it is necessary to do system management. System management is a set of processes to change part or all of the system, rebuild a system, and manage the system using hardware and software[1]. General system management is based on DevOps and Cloud Computing, where both technologies are used to help system management from installation, and service, to automation.

© The Author(s) 2023

M. D. Sulistiyo and R. A. Nugraha (eds.), *Proceedings of the International Conference on Enterprise and Industrial Systems (ICOEINS 2023)*, Advances in Economics, Business and Management Research 270,

[https://doi.org/10.2991/978-94-6463-340-5\\_30](https://doi.org/10.2991/978-94-6463-340-5_30)

DevOps is a series of methods, techniques, concepts, and paradigms that combine software development and operations, to automate the software development process quickly and efficiently[2]. DevOps helps produce stable software products while increasing the value of the software. Cloud Computing is the development of parallel computing technology utilizing the concept of virtualization, where all computing resources can be distributed to users as a service according to their needs[3]. Cloud Computing provides three types of services to users:

- 1) As infrastructure and network computing resources in the form of Infrastructure as a Service (IaaS) Cloud;
- 2) As a software development platform in the form of Platform as a Service (PaaS) Cloud; and
- 3) As ready-to-use software in the form of Software as a Service (SaaS) Cloud[4].

The development of Cloud Computing technology was followed by an increase in user interest in Cloud Computing services, resulting in the emergence of Function as a Service (FaaS) Cloud, as a type of Cloud Computing service specifically for Cloud computing providers in managing all computing resources used for Cloud-based services (hardware, software, middleware, operating system)[5]. FaaS uses a number of open-source products such as OpenStack to control computing processes, network resources, administration control, and user access rights[6], OpenNebula for edge computing, data center, and virtualization[7], and OpenShift for container services, virtualization, and microservices[8]. In DevOps, Kubernetes is an open-source platform that is reliable for managing container clustering on server clusters as well as cluster orchestration for scheduling, scaling, recovery, and monitoring containers[9][10].

Several previous studies discuss OpenStack, Kubernetes, DevOps, and Cloud Computing, which are the state of the art for this research. The first research was regarding the implementation of Infrastructure as a Service (IaaS) Cloud based on OpenStack and Apache CloudStack, then a comparison was carried out in terms of performance and reliability[11]. The second research concerns the implementation of Private Cloud Computing in a case study of an OpenStack-based internal university network which is then analyzed in terms of network reliability in providing student services[12]. The third research describes the implementation of autoscaling in containers on a web server using Proxmox-based Kubernetes on an internal university network[13]. The fourth research concerns the implementation of DevOps to support the running of the COVID-19 e-screening system during the pandemic to increase system reliability[14]. The fifth research describes the implementation of Cloud Computing in a case study at one of the universities by focusing on the perspective of service reliability and the benefits obtained [15].

The sixth research discusses the provisioning of the Google Kubernetes Engine (GKE) in the development environment and production environment to help the DevOps team handle increasing service requests[16]. The seventh research describes the implementation of Cloud Computing in the form of Infrastructure as a Service (IaaS) Cloud services to support the running of web server services[17]. The eighth research describes the implementation of Load Balancing on OpenStack to balance net-

work load using the Round Robin method and algorithm[18]. The ninth research describes the implementation of OpenStack-based Infrastructure as Code (IaC) to assist Cloud Computing network management in government[19]. The tenth research discusses the implementation of the Private Cloud Computing deployment model in the form of Infrastructure as a Service (IaaS) Cloud services based on OpenStack[20].

The eleventh research describes the analysis and implementation of Private Cloud on computer networks using OpenStack on the Linux operating system[21]. The twelfth research explains the implementation of multi-tenant computing infrastructure systems and services using OpenStack based on Metal as a Service (MaaS)[22]. The thirteenth research discusses the design, implementation, and analysis of Cluster Containers using Kubernetes and Google Cloud Platform infrastructure[23]. The fourteenth study discusses the Analysis of Server Deployment Implementation Using Kubernetes to Avoid Single Failure[24]. The fifteenth research describes the implementation of the Infrastructure as a Service (IaaS) Cloud for web server computing needs using the Load Balancing method[25].

Based on these studies, in this research, a comparison was carried out between OpenStack and Kubernetes in terms of system management on the Cloud network connection and the ease of DevOps (creating data, using Pod and Podman), accompanied by analysis and comparison of the values and performance graphs of both. The formulation of the problem in this research is which is better, OpenStack and Kubernetes from the perspective of DevOps testing and Cloud network connection testing. The test results are expected to provide information about the best utilization of OpenStack and Kubernetes for DevOps and Cloud Computing.

## **2 Research Method**

### **2.1 Literature Review**

The data collection method used in this research is the literature review. A literature review is a method of collecting data in research with a series of activities related to collecting library literature data, reading, and taking notes, and managing the research materials[26]. In this research, a literature review was carried out by collecting and reading various literature from websites, books, and published papers in national and international journals and papers from proceedings at conferences and seminars related to the research topic.

### **2.2 Tool and Materials**

Tools and materials that support this research include computer hardware and computer software. The hardware used is a notebook with Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz 2.59 GHz processor specifications, 16 GB memory (RAM), and 512 GB SSD. The software used includes the Linux operating system Ubuntu 22.04, Kubernetes, and OpenStack. Apart from that, a modem and internet connection are also used.

### 2.3 The Flowchart

The flowchart is a diagram that displays the steps in carrying out a process of a program. The shape of the diagram, and lines or arrow directions are depicted and connected in each process, with the aim of simplifying a series of procedures to make it easier to understand the information. The flowchart of this research is shown in Fig. 1.

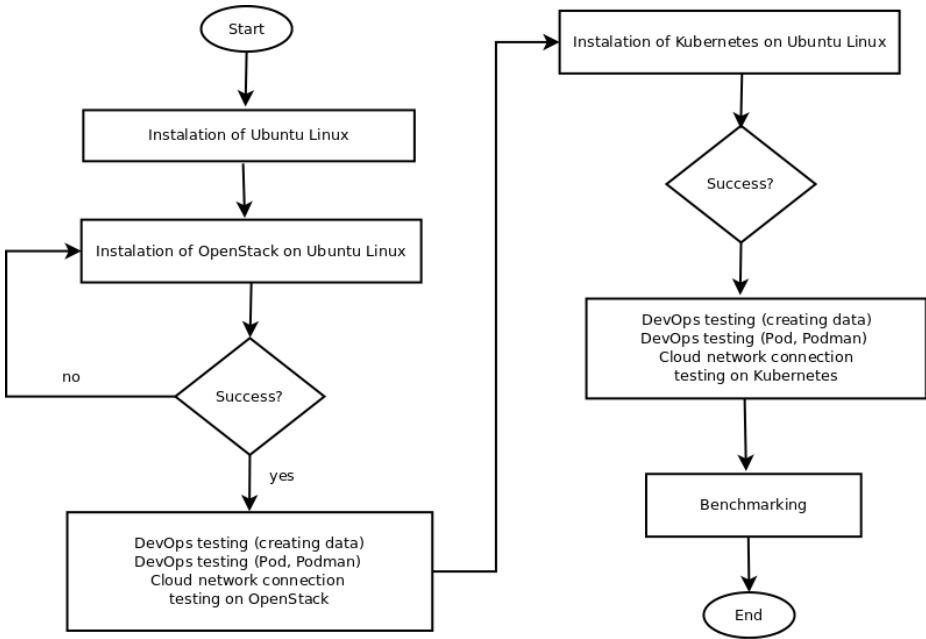


Fig. 1. Flowchart of the proposed method.

Based on Fig. 1, the research flowchart starts with preparing to install Ubuntu Linux, then installing OpenStack, and then testing DevOps and Cloud network connection on OpenStack. Next, install Kubernetes, followed by testing DevOps and Cloud network connection on Kubernetes. Test results are recorded, and comparisons are made along with graphic visualization.

## 3 Testing and Result

### 3.1 DevOps Testing (Creating Data) on OpenStack

The first DevOps testing on OpenStack is creating data via the OpenStack Dashboard. Fig. 2 shows the DevOps testing process (creating data) on OpenStack.

Based on Fig. 2, DevOps testing (creating data) on OpenStack can be done quickly and easily through the Dashboard, where a server group is first created by writing a name and policy and then submitting it. Then, the ID will automatically be created by OpenStack. Fig. 3 shows the results of DevOps testing on OpenStack.

Based on Fig. 3, DevOps testing results on OpenStack can be done easily and faster via the OpenStack Dashboard, using the create menu.

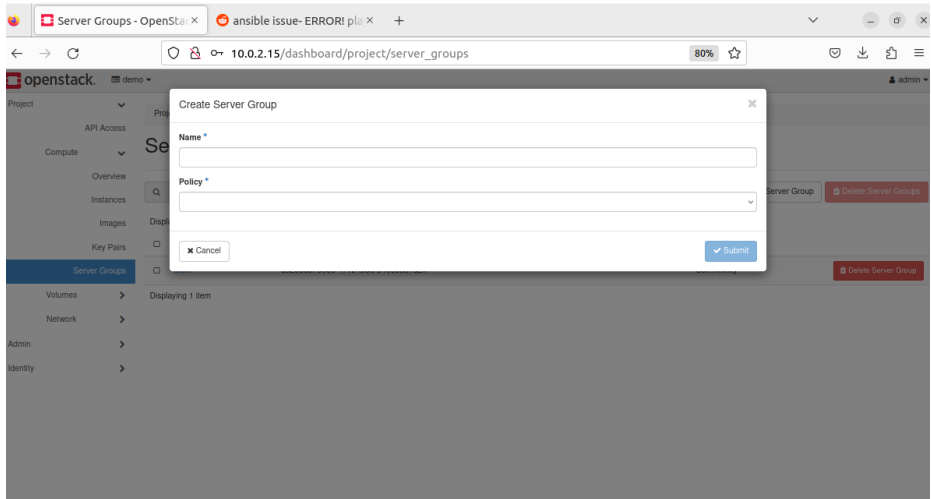


Fig. 2. DevOps Testing (Creating Data) Process in OpenStack.

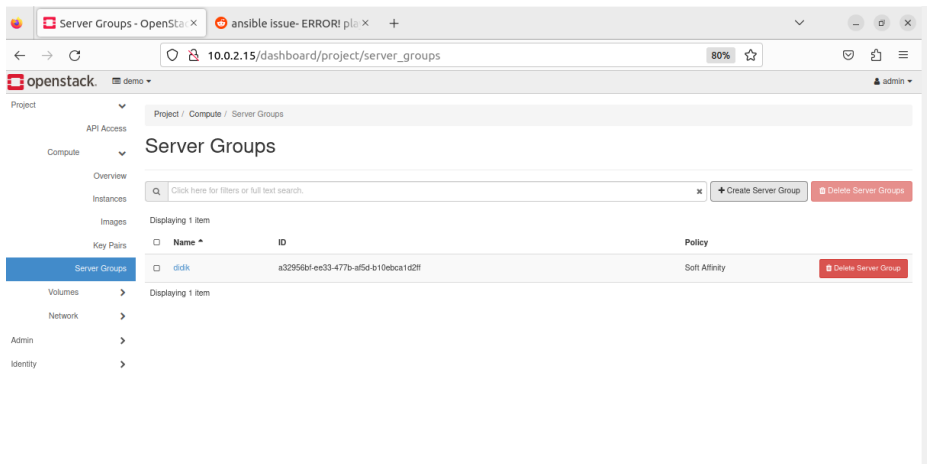


Fig. 3. DevOps Testing (Creating Data) Results on OpenStack.

### 3.2 DevOps Testing (Pod, Podman) on OpenStack

The second DevOps testing on OpenStack using Pod and Podman, was carried out with root access and the command: `sudo podman ps -a --pod` to display pods and podman. Root access is required to make it easier to distinguish between Pod and Podman results from OpenStack and those from Kubernetes (in future experiments). Fig. 4 shows the results of testing Pod and Podman on OpenStack.

```

root@didik-VirtualBox:~/home/didik# sudo podman ps -a --pod
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS      PORTS          NAMES          POD ID          PODNAME
E              3cb6f693ddd4  k8s.gcr.io/pause:3.5    About a minute ago Created
ly_neumann

```

Fig. 4. DevOps Testing (Pod, Podman) on OpenStack.

### 3.3 Cloud Network Connection Testing on OpenStack

The test results on the Cloud on OpenStack are a test that is used to see the results of a Cloud that is owned by OpenStack with the `./benchmark.sh` command. Fig. 5 shows the testing results of the Cloud network connection on OpenStack.

Fig. 5 is the result of Cloud network connection testing on OpenStack. The results tested on the OpenStack cloud are the total connection owned, such as

- a) Min = 0 ms,
- b) Mean = 2 ms,
- c) Standard Deviation (SD) = 1.7 ms,
- d) Med = 1 ms, and
- e) Max = 12 ms.

```

Server Software:      Apache/2.4.52
Server Hostname:     10.0.2.15
Server Port:         80

Document Path:       /
Document Length:     286 bytes

Concurrency Level:   10
Time taken for tests: 0.186 seconds
Complete requests:   1000
Failed requests:     0
Non-2xx responses:  1000
Total transferred:   501000 bytes
HTML transferred:   286000 bytes
Requests per second: 5387.26 [#/sec] (mean)
Time per request:    1.856 [ms] (mean)
Time per request:    0.186 [ms] (mean, across all concurrent requests)
Transfer rate:       2635.76 [Kbytes/sec] received

Connection Times (ms)
      min  mean[+/-sd] median  max
Connect:    0     0  0.7   0     8
Processing:  0     1  1.4   1    11
Waiting:    0     1  1.2   1     8
Total:      0     2  1.7   1    12

Percentage of the requests served within a certain time (ms)
 50%    1
 66%    2
 75%    2
 80%    2
 90%    4
 95%    6
 98%    8
 99%   10
100%   12 (longest request)

```

Fig. 5. Cloud Network Connection Testing on OpenStack.

### 3.4 DevOps Testing (Creating Data) on Kubernetes

DevOps testing on Kubernetes is a test by creating data on Kubernetes, where data is created directly on the terminal. This is different from DevOps testing on OpenStack via the Dashboard. Fig. 6 shows Kubernetes’s DevOps testing (creating data) process.

Based on Fig. 6, the DevOps testing in Kubernetes in the form of adding data, use the terminal with the create deployment `hello-minikube` command along with the port number. Then use the `minikube service hello-minikube` command to display the data. Fig. 7 shows the results of DevOps testing (creating data) on Kubernetes.

Based on Fig. 7, the results of DevOps testing (creating data) in Kubernetes are not as easier and faster as in OpenStack, where data is created first through the terminal, and then displayed through the Dashboard.

```

didtk@didtk-VirtualBox:~$ kubectl create deployment hello-minikube --image=kicbase/echo-server:1.0
deployment.apps/hello-minikube created
didtk@didtk-VirtualBox:~$ kubectl expose deployment hello-minikube --type=NodePort --port=80
service/hello-minikube exposed
didtk@didtk-VirtualBox:~$ kubectl get service hello-minikube
NAME                TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
hello-minikube     NodePort    10.104.205.153  <none>           80:32635/TCP    7s
didtk@didtk-VirtualBox:~$ minikube service hello-minikube
-----
| NAMESPACE | NAME          | TARGET PORT | URL                |
|-----|-----|-----|-----|
| default   | hello-minikube | 80          | http://192.168.59.107:32635 |
|-----|-----|-----|-----|
🔥 Opening service default/hello-minikube in default browser...
didtk@didtk-VirtualBox:~$ update.go:85: cannot change mount namespace according to change mount (/var/lib/snappd/hostfs/usr/share/gimp/p/2.0/help /usr/share/gimp/2.0/help none bind,ro 0 0): cannot open directory "/var/lib": permission denied
update.go:85: cannot change mount namespace according to change mount (/var/lib/snappd/hostfs/usr/share/gtk-doc /usr/share/gtk-doc none bind,ro 0 0): cannot open directory "/var/lib": permission denied
update.go:85: cannot change mount namespace according to change mount (/var/lib/snappd/hostfs/usr/share/lbreeoffice/help /usr/share/lbreeoffice/help none bind,ro 0 0): cannot open directory "/var/lib": permission denied
update.go:85: cannot change mount namespace according to change mount (/var/lib/snappd/hostfs/usr/share/xubuntu-docs /usr/share/xubuntu-docs none bind,ro 0 0): cannot open directory "/var/lib": permission denied
Gtk-Message: 08:05:31.286: Not loading module "atk-bridge": The functionality is provided by GTK natively. Please try to not load it
ALSA lib conf.c:4120:(snd_config_update_r) Cannot access file /usr/share/alsa/alsa.conf
ALSA lib seq.c:935:(snd_seq_open_noupdate) Unknown SEQ default

```

Fig. 6. DevOps Testing (Creating Data) Process in Kubernetes.

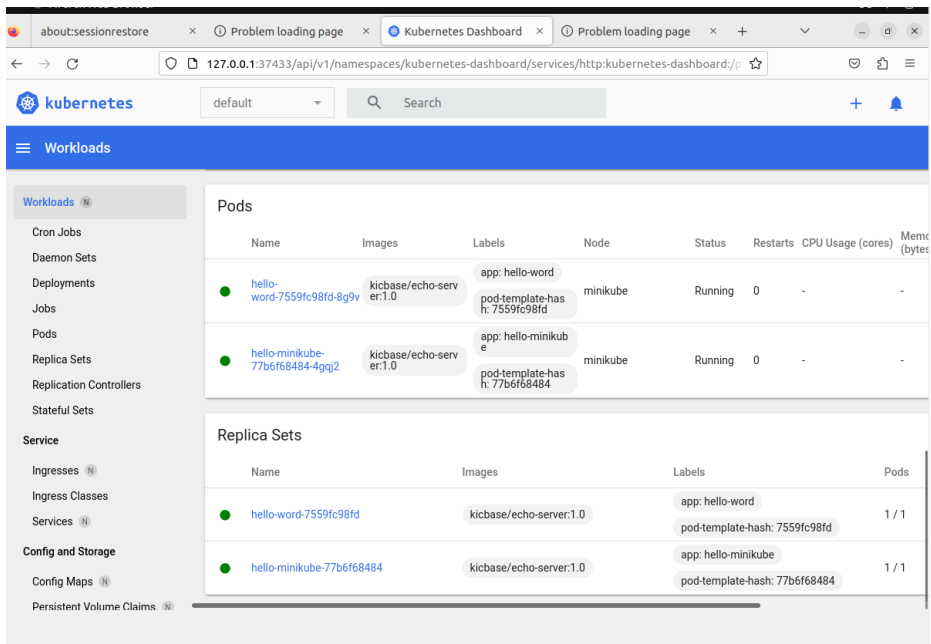


Fig. 7. DevOps Testing Results on Kubernetes.

### 3.5 DevOps Testing (Pod and Podman) on Kubernetes

The second DevOps testing on Kubernetes using Pods and Podman, was carried out by creating pods and Podman in Kubernetes. Fig. 8 shows the results of DevOps testing on Kubernetes (Pod and Podman).

Based on Fig. 8, the Pod and Podman test results in Kubernetes display Container ID, Image, Creates, Status, Names, Pod ID, and Pod Name.

```
didik@didik-VirtualBox: $ sudo podman ps -a --pod
CONTAINER ID   IMAGE                                COMMAND                  CREATED          STATUS          PORTS          NAMES                POD ID          PODNAME
429230575b69  k8s.gcr.io/pause:3.5                /pause                  2 days ago      Created         gate              80805eb59269-lnfra  80805eb59269  cranky_nighttn
```

Fig. 8. Pod and Podman Test Results in Kubernetes.

### 3.6 Cloud Network Connection Testing on Kubernetes

The results of testing the Cloud network connection in Kubernetes are the results of the `./benchmark.sh` command in Kubernetes, with the output of the test results as shown in Fig. 9.

Based on Fig. 9, the results tested on cloud network connections in Kubernetes are the total connections owned, such as

- a) Min = 67 ms,
- b) Mean = 198 ms,
- c) Standard Deviation (SD) = 168.0 ms,
- d) Med = 276 ms, and
- e) Max = 801 ms.

```
Server Software:
Server Hostname: 127.0.0.1
Server Port: 37433

Document Path: /apl/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/#/workloads?namespace=default
Document Length: 19 bytes

Concurrency Level: 10
Time taken for tests: 3.114 seconds
Complete requests: 100
Failed requests: 0
Non-2xx responses: 100
Total transferred: 23800 bytes
HTML transferred: 1900 bytes
Requests per second: 32.12 [#/sec] (mean)
Time per request: 311.371 [ms] (mean)
Time per request: 31.137 [ms] (mean, across all concurrent requests)
Transfer rate: 7.46 [Kbytes/sec] received

Connection Times (ms)
  min  mean[+/-sd] median  max
Connect:  0    0  0.1    0    0
Processing: 67  298 168.0  276  801
Waiting: 66  288 165.0  273  799
Total: 67  298 168.0  276  801

Percentage of the requests served within a certain time (ms)
 50%  276
 60%  382
 75%  428
 80%  459
 90%  533
 95%  621
 98%  697
 99%  801
100% 801 (longest request)
```

Fig. 9. Testing Cloud Network Connection in Kubernetes.



## 4 Discussion

### 4.1 Comparison Results of DevOps Testing (Creating Data) on OpenStack and Kubernetes

The results of the comparison after creating data are made by making a table. Table 1 shows the comparison of DevOps Testing (creating data) in OpenStack and Kubernetes.

Based on Table 1, the results of the test show that the process of creating data at DevOps process in OpenStack is easier and faster than Kubernetes via Dashboard without Terminal.

**Table 1.** Comparison Result of DevOps Testing (Creating Data) on OpenStack and Kubernetes.

DevOps	Creating Data
OpenStack	Via the OpenStack Dashboard, using the create menu.
Kubernetes	Data is created first through the terminal and then displayed through the Dashboard.

### 4.2 Comparison Results of DevOps Testing (Creating Data) on OpenStack and Kubernetes

The results of the comparison after conducting the Pod and Podman experiments are made by making a table. Table 2 shows the comparison of Pod and Podman in OpenStack and Kubernetes.

Based on Table 2, the results of the test show that the Container IDs are different for OpenStack and Kubernetes. This is because Podman, to run each container, is run as a separate container, so it has a unique Container ID, which serves as an identity for each running container.

**Table 2.** Comparison Result of DevOps Testing (Pod and Podman) on OpenStack and Kubernetes.

DevOps	OpenStack	Kubernetes
Container ID	3cb6f603ddd4	429230575b69
Image	k8s.gcr.io/pause:3.5	k8s.gcr.io/pause:3.5
Creation time	about a minute	2 days
Status	created	created
Names	9a2523880af7.infra	80805eb59269-infra
Pod ID	9a2523880af7	80805eb59269
Pod Name	friendly_neumann	cranky_nightingale

### 4.3 Comparison Results of DevOps Testing (Creating Data) on OpenStack and Kubernetes

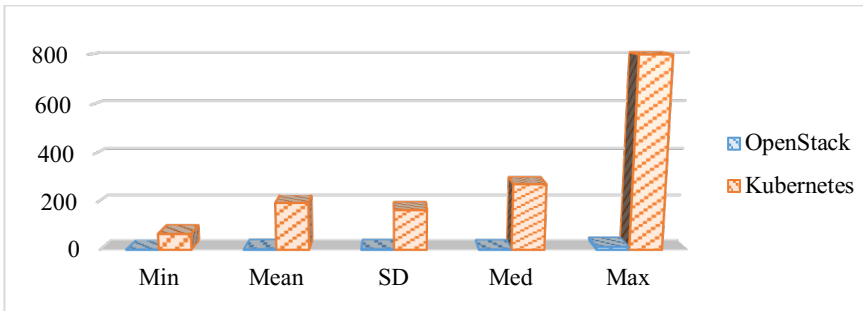
The results of the comparison of Cloud Network Connection on OpenStack and Kubernetes are made by making a table. Table 3 shows the comparison of Pod and Podman.

Based on Table 3, a comparative graph of Cloud Network Connection on OpenStack and Kubernetes can be qualitatively seen in Fig. 10.

Based on Table 3 and the graph in Fig. 10, it can be seen that Kubernetes has a significantly higher total connection than OpenStack. This is because Kubernetes uses container orchestration to manage and run applications so that every application running on Kubernetes can create a greater number of connections than on OpenStack.

**Table 3.** Comparison Result of Cloud Network Connection Testing on OpenStack and Kubernetes.

Total Connection	OpenStack	Kubernetes
Min	0 ms	67 ms
Mean	2 ms	198 ms
Standard Deviation (SD)	1.7 ms	168.0 ms
Med	1 ms	276 ms
Max	12 ms	801 ms



**Fig. 10.** The comparison of Cloud Network Connection on OpenStack and Kubernetes.

## 5 Conclusion

Based on the tests carried out, it can be concluded that in the comparison between OpenStack and Kubernetes in system architecture, especially from the perspective of DevOps and Cloud network connections, OpenStack has advantages in terms of DevOps compared to Kubernetes, through the ease of creating data directly via the Dashboard and the use of Pods and Podman. Meanwhile, in terms of cloud network connections, Kubernetes has an advantage over OpenStack, because Kubernetes uses container orchestration to manage and run applications, thereby increasing the number of connections on the cloud network.

## Acknowledgment

The authors would like to express their gratitude to Telkom University and Udayana University for their technical and financial support during this research.

## References

1. Y. Suryanata, "Membangun Sistem Informasi Manajemen Perpustakaan Dengan CDS/ISIS," *Jurnal Pustakawan Indonesia*, Vol.12, No.1, 2020.
2. A. Wiedemann, et al., "Understanding How DevOps Aligns Development and Operations: A Tripartite Model of Intra-IT Alignment," *European Journal of Information Systems*, Vol. 10, 2020. DOI: 10.1080/0960085X.2020.1782277.
3. E. Suhendar, "Tinjauan Sistematis : Implementasi Cloud Computing Terhadap Keamanan Layanan Publik," *Jurnal Smart Comp*, Vol.11, No.4, 2022.
4. R.L. Rahardian, et al., "Implementasi Layanan Cloud Computing Software As a Service Pada Usaha Mikro Kecil dan Menengah," *Majalah Ilmiah Teknologi Elektro*, Vol.17, No.3, 2018.
5. I. Barokah, and A. Asriyanik, "Analisis Perbandingan Serverless Computing Pada Google Cloud Platform," *Jurnal Teknologi Informatika dan Komputer*, Vol.7, No.2, 2021.
6. Openstack, "The Most Widely Deployed Open Source Cloud Software in the World," Open Infra Foundation. Retrieved: <https://openstack.org/>
7. OpenNebula, "The Open Source Cloud & Edge Computing Platform," OpenNebula System. Retrieved: <https://opennebula.io/>
8. Red Hat OpenShift, "OpenShift Documentation," Red Hat. Retrieved: <https://docs.openshift.com/>
9. Cloud Native Computing Foundation, "Kubernetes Documentation," CNCF. Retrieved: <https://kubernetes.io/id/>
10. A.N. Huda, and S.S. Kusumawardani, "Kubernetes Cluster Management for Cloud Computing Platform: A Systematic Literature Review," *JUTI: Jurnal Ilmiah Teknologi Informasi*, Vol.20, No.2, 2022. pp.75-83.
11. H. Triyanto, et al., "Analisa Perbandingan Performa Openstack dan Apache Cloudstack dalam Model Cloud Computing Berbasis Infrastructure As a Service," *JUSTIN: Jurnal Sistem dan Teknologi Informasi*, Vol.8, No.1, 2020.
12. R.P. Anugrah, et al., "Private Cloud Computing (Studi Kasus Untuk Fasilitas Mahasiswa UTDI)," *Jurnal of Information System Management (Joism)*, Vol.4, No.1, 2022.
13. I. Rosyadi, et al, "Implementation Autoscaling Container Web Server using Kubernetes Promox Based on Server University of Darussalam Gontor," *Jurnal Unida Gontor*, Vol.6, No.1,2019. pp.31-38.
14. T. Tohirin, et al., "Implementasi DevOps pada Pengembangan Aplikasi e-Skrining Covid-19," *Jurnal Multinetics*, Vol.6, No.1, 2020. pp.15-20.
15. E. Kurniawan, "Penerapan Teknologi Cloud Computing pada Universitas (Studi Kasus: Fakultas Teknologi Informasi UKDW)," *Jurnal Eksis*, Vol.8, No.1, 2015. pp.29-36.
16. O. Pramadika, and D.W. Chandra, "Provisioning Google Kubernetes Engine Cluster Dengan Menggunakan Terraform Dan Jenkins Pada Dua Environment," *Jurnal Ilmiah Penelitian dan Pembelajaran Informatika (JIPI)*, Vol.8, No.2, 2023. Pp.597-606.
17. L. Sulistyowati, et al., "Implementasi Cloud Computing sebagai Infrastructure as a Service untuk Penyediaan Web Server," *Jurnal Teknologi Informasi Aiti*, Vol.9, No.2. pp.185-201.

18. I.M.A.S. Darma, and I.G.O.G. Atitama, "Implementasi Load Balancing Pada Openstack dengan Metode Round Robin," *Prosiding Seminar Nasional Informatika (SENAPATI) 2019*, pp. 115-119.
19. I.P.A.E. Pratama, "Infrastructure as Code (IaC) Menggunakan OpenStack untuk Kemudahan Pengoperasian Jaringan Cloud Computing (Studi Kasus: Smart City di Provinsi Bali)," *Jurnal Ilmu Pengetahuan Dan Teknologi Komunikasi (IPTEKKOM)*, Vol.23, No.1, 2021. pp.93-105.
20. P.G.S.C. Nugraha, et al., "Implementasi Private Cloud Computing Sebagai Layanan Infrastructure as a Service (IaaS)," *Jurnal Ilmiah Ilmu Komputer*, Vol.8, No.2, 2015. pp.7-14.
21. M. Fauzan, et al., "Analisis Dan Perancangan Infrastruktur Private Cloud Dengan Openstack," *Jurnal Pseudocode*, Vol.4, No.2, 2017. pp.180-189. <https://doi.org/10.33369/pseudocode.4.2.180-189>.
22. J.R. Panggabean, J. R., et al., "Layanan Infrastruktur Komputasi Multitenant dengan Open-Stack di Lingkungan MaaS," *Jurnal Teknologi dan Sistem Komputer (JTSiskom)*, Vol.5, No.4. <https://doi.org/10.14710/jtsiskom.5.4.2017.142-146>.
23. M.A. Nugroho, et al., "Analisis Cluster Container pada Kubernetes dengan Infrastruktur Google Cloud Platform," *Jurnal Ilmiah Penelitian dan Pembelajaran Informatika (JIPI)*, Vol.3, No.2, 2018. pp.84-93. <https://doi.org/10.29100/jipi.v3i2.651>.
24. L. Widyawati, et al., "Server Deployment Overview," *Jurnal Informatika Teknologi dan Sains (JINTEKS)*, Vol.3, No.1, 2021. pp.267-271.
25. A. Saputra, et al., "Implementasi Infrastructure as a Service pada Cloud Computing Menggunakan Metode Load Balancing," *Jurnal Sistem dan Teknologi Informasi (JUSTIN)*, Vol.8, No.4, 2020.
26. Pratama, I.P.A.E., Adhitya, I.G.N.A.K., "Post Quantum Cryptography: Comparison between RSA and McEliece," *9th International Conference on ICT for Smart Society: Recover Together, Recover Stronger and Smarter Smartization, Governance and Collaboration, ICISS 2022 Proceeding*, 2022.

**Open Access** This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

