IMPLEMENTASI *KUBERNETES CLUSTER* MENGGUNAKAN KVM

Nurul Khairina Surbakti[1], Muhammad Arif Fadhly Ridha[2]

[1]Program Studi Teknik Informatika, Politeknik Caltex Riau, Jl. Umban Sari (Patin) No. 1 Rumbai, Pekanbaru, 28265

[2]Program Studi Teknik Informatika, Politeknik Caltex Riau, Jl. Umban Sari (Patin) No. 1 Rumbai, Pekanbaru, 28265

E-mail: nurul17ti@mahasiswa.pcr.ac.id, fadhly@pcr.ac.id

Abstract

Rapid technological development requires services that can be accessed at any time. The increasing number of client requests will result in the server being down due to more workloads. Therefore, clusters are present to handle increased server workloads and evenly divide each server. However, to run a cluster requires cluster orchestration software that is Kubernetes. Kubernetes is used for container management. The virtualization technology used in Kubernetes is Kernel-Based Virtual Machine (KVM) virtualization. In this research, we compare latency, CPU usage, and memory usage when opening web pages, between conventional technology and Kubernetes cluster. The results of the highest latency test are 16614 ms by Kubernetes cluster and the lowest latency is 1522 ms by conventional technology. In conventional systems, the response from the user is directly received by the web server, processed and then sent. In Kubernetes clusters, there are nodes running on top of virtual machines, so it takes longer to respond. Good latency is low latency because it can open web pages faster. In the standby state, the lowest CPU usage and memory usage occurred in conventional were 0.07% and 28% because there were no nodes running on the virtual machine. In a busy state, the lowest CPU usage occurs on the Kubernetes cluster server is 1.06% due to the division of workload on virtual machines. The lowest memory usage occurs on a conventional server is 57% because there are no nodes running on a virtual machine.

Keywords: *Kubernetes cluster, KVM, Latency, CPU, Memory.*

Abstrak

Perkembangan teknologi yang pesat membutuhkan layanan yang dapat diakses kapan saja. Semakin meningkatnya jumlah *request client* akan mengakibatkan *server* mengalami down karena beban kerja yang lebih. Oleh karena itu, *cluster* hadir untuk menangani beban kerja server yang meningkat dan membagi secara merata setiap server. Namun, untuk menjalankan sebuah *cluster* dibutuhkan *software cluster orchestration* yaitu *Kubernetes*. *Kubernetes* digunakan untuk manajemen *container*. Teknologi virtualisasi yang digunakan dalam *Kubernetes* adalah virtualisasi *Kernel-Based Virtual Machine* (KVM). Pada penelitian ini membandingkan *latency*, pemakaian CPU, dan pemakaian memori saat membuka halaman web, antara teknologi konvensional dan *Kubernetes cluster*. Diperoleh hasil pengujian *latency* tertinggi yaitu 16614 ms oleh *Kubernetes cluster* dan *latency*

ISSN: 2339 – 2053

Pekanbaru, 25 Agustus 2021

terendah yaitu 1522 ms oleh teknologi konvensional. Pada sistem konvensional, respons dari pengguna secara langsung diterima oleh *web server*, diproses lalu dikirim. Pada *Kubernetes cluster* terdapat *node-node* yang berjalan di atas mesin virtual, sehingga membutuhkan waktu yang lebih lama dalam memberikan respons. *Latency* yang baik yaitu *latency* yang rendah karena dapat membuka halaman web lebih cepat. Dalam keadaan *standby*, pemakaian CPU dan pemakaian memori terendah terjadi pada konvensional adalah 0,07% dan 28% karena tidak ada *node-node* yang berjalan di atas mesin virtual. Dalam keadaan *busy*, pemakaian CPU terendah terjadi pada server *Kubernetes cluster* adalah 1,06% karena adanya pembagian beban kerja terhadap mesin virtual. Pemakaian *memory* terendah terjadi pada server konvensional adalah 57% karena tidak ada *node-node* yang berjalan di atas mesin virtual.

Kata Kunci: *Kubernetes Cluster, KVM, Latency, CPU, Memory.*

PENDAHULUAN

Perkembangan zaman telah memberikan dampak besar pada dunia teknologi, salah satunya adalah teknologi komputasi. Teknologi komputasi adalah kegiatan menggunakan dan mengembangkan teknologi komputer, perangkat keras, dan perangkat lunak komputer. Jenis -jenis komputasi modern terbagi tiga macam, yaitu komputasi mobile, komputasi grid, dan *cloud computing* (Alkhawarizmi, 2019). Menurut *National Institute of Standards and Technology* (NIST) *cloud computing* adalah sebuah model yang memungkinkan akses di mana saja, memberikan kenyamanan, akses jaringan sesuai permintaan (*on-demand*) untuk membagi dan memanfaatkan secara bersama-sama beberapa *resource* yang telah dilakukan pengaturan (Mell & Grance, 2011).

Layanan *cloud computing* tersimpan pada *server*. Semakin meningkatnya jumlah *request* yang mencapai ribuan bahkan jutaan pada waktu yang bersamaan pada beban *server* akan mengakibatkan *overload* karena layanan harus bekerja secara terus menerus dalam menyediakan data yang dibutuhkan oleh *client*. Selain itu, banyaknya aplikasi yang berasal dari berbagai platform juga dapat memberatkan kinerja *server*. Dalam meningkatkan kualitas kinerja *server* dibutuhkan solusi ketika terjadi masalah dalam satu *node*, maka mesin virtual di dalamnya akan berpindah ke *node* lain untuk meminimalkan terjadinya layanan yang tidak dapat diakses. Oleh karena itu, diperlukan sebuah teknologi *cluster computing* yang bertujuan mengurangi beban kerja *server* (Apriliana dkk., 2018).

ISSN: 2339 – 2053

Dengan seiring perkembangan teknologi, *container* hadir sebagai virtualisasi yang dapat menjalankan sebuah aplikasi, sehingga membutuhkan beberapa *container* untuk menjalankan layanan yang sama. Oleh karena itu, dibutuhkan sebuah *platform* yang berfungsi sebagai wadah untuk *container* beserta seluruh hal yang dibutuhkan oleh aplikasi agar aplikasi dapat berfungsi dengan baik. Salah satu *platform* untuk mengelola teknologi *container* adalah *Docker*. *Docker* merupakan project *open-source* yang menyediakan *platform* terbuka bagi pengembang dan administrator sistem untuk dapat membangun, mengemas, dan menjalankan aplikasi dimanapun sebagai sebuah *container* yang ringan (IDCloudHost, 2020). Dalam membangun suatu layanan dibutuhkan sebuah *software orchestration* yaitu *Kubernetes* (K8s).

Kubernetes merupakan *platform open-source* untuk manajemen beban kerja aplikasi dalam *container*, serta menyediakan konfigurasi dan otomatisasi secara deklaratif. *Kubernetes* berada dalam ekosistem yang besar dan berkembang pesat. *Service, support,* dan perkakas *Kubernetes* tersedia secara meluas (Kubernetes, 2020). Dengan terciptanya teknologi tersebut, tidak perlu menginstal banyak *server* yang di dalamnya ada banyak *environment*, aplikasi, juga kebutuhan lainnya, sebab semuanya sudah ditampung dalam sebuah *container* yang terdapat dalam *Kubernetes*. *Kubernetes* akan manajemen *container* dalam jumlah besar untuk membangun layanan *microservices*.

Salah satu teknologi virtualisasi yang berjalan di dalam *Kubernetes* adalah virtualisasi *Kernel-Based Virtual Machine* (KVM). Virtualisasi KVM menggunakan jenis virtualisasi *Full virtualization*. KVM adalah teknologi virtualisasi yang dikembangkan oleh Linux dengan tipe *hardware* x86 (64-bit). KVM diimplementasikan sebagai modul kernel *loadable* yang mengubah kernel Linux menjadi *bare metal hypervisor* (IDreg, 2013).

Berdasarkan uraian di atas, maka dilakukan penelitian mengenai "Implementasi *Kubernetes Cluster* menggunakan KVM" dengan tujuan membangun *Kubernetes cluster* menggunakan virtualisasi KVM untuk membagi beban pada masing-masing mesin virtual, sehingga meminimalkan terjadinya layanan yang tidak dapat diakses, dan membandingkan kinerja sumber daya perangkat keras (CPU dan *memory*) pada saat

ISSN: 2339 – 2053

Pekanbaru, 25 Agustus 2021

virtualisasi *web server* keadaan *standby* dan *busy* antara teknologi konvensional dengan teknologi *Kubernetes cluster*.

METODE PENELITIAN

Metode penelitian yang dilakukan dengan beberapa cara yaitu sebagai berikut:

A. Pengujian *High-Availability*

Pengujian dilakukan sebanyak 5 kali pengujian. Pengujian yang dilakukan pada *web server* akan diuji dengan beberapa cara, yaitu: pengujian *failover* bertujuan untuk menguji apakah layanan tetap diberikan meski hanya satu *machine* yang digunakan. Pengujian *failback* bertujuan untuk menguji apakah layanan akan berpindah dari satu *node* ke *node* yang lainnya dan pengujian *load balancing* untuk membagi beban kinerja *server* secara seimbang, agar *server* tidak mengalami *overload* saat menerima *request* dari *client.*

B. Pengujian *Performance*

Pengujian *performance* akan diuji dengan beberapa cara yaitu: pengujian *stress testing* bertujuan untuk mengetahui dan mengukur kinerja sebuah *web server* dalam menangani *request client* dalam jumlah yang sangat banyak secara bersamaan dalam satu waktu pada *server* dengan menggunakan *tools* JMeter. Pengujian kinerja virtualisasi *web server* dalam keadaan *standby* dan *busy* dilakukan untuk memantau persentase penggunaan CPU dan *memory* yang terpakai pada *web server* tidak di akses *client* (*standby*) dan di akses *client* (*busy*) pada *server* konvensional dan *Kubernetes cluster* dengan menggunakan aplikasi protokol SNMP dan pengujian *scalability* bertujuan untuk melihat ketika beban layanan meningkat akan melakukan pembesaran melalui penambahan *pod*, dan pada saat beban menurun layanan akan melakukan pengurangan terhadap *pod* sesuai kebutuhan layanan tersebut.

HASIL DAN PEMBAHASAN

A. Hasil Pengujian *High-Availability*

Pengujian *failover*, *failback* dan *load balancing* dilakukan sebanyak 5 kali percobaan pada server *Kubernetes cluster* dengan skenario mematikan salah satu *node*

ISSN: 2339 – 2053

Pekanbaru, 25 Agustus 2021

yaitu worker2 untuk melihat tingkat keberhasilan dari *failover* dan menghidupkan kembali worker2 untuk melihat tingkat keberhasilan dari *failback*. Adapun *load balancer* diimplementasikan pada *pod*, sehingga saat *pod* di-*replicas* secara otomatis oleh *horizontal pods autoscale* (HPA), *load balancer* akan melakukan distribusi *pod* secara seimbang pada setiap *node*. Berikut hasil pengujian tingkat keberhasilan *failover* dan *load balancing* dapat dilihat pada Tabel 1.

Tabel 1
Hasil Pengujian *Failover* dan *Load Balancing*

Pengujian Ke -	Node			Status Web Server
	Master	Worker 1	Worker 2	
1	Aktif	Aktif	Mati	Aktif
2	Aktif	Aktif	Mati	Aktif
3	Aktif	Aktif	Mati	Aktif
4	Aktif	Aktif	Mati	Aktif
5	Aktif	Aktif	Mati	Aktif

Pada Tabel 1 menunjukkan bahwa status *web server* tetap aktif, sehingga dapat meminimalkan layanan yang tidak dapat di akses karena jika salah satu *node* mati yaitu worker 2 maka *web server* masih dapat diakses karena terdapat *node* master dan worker1 yang masih dalam keadaan *running*. Berikut hasil pengujian tingkat keberhasilan *failback* dapat dilihat pada Tabel 2.

Tabel 2
Hasil Pengujian *Failback*

Pengujian Ke -	Node			Status Web Server
	Master	Worker 1	Worker 2	
1	Aktif	Aktif	Aktif	Aktif
2	Aktif	Aktif	Aktif	Aktif
3	Aktif	Aktif	Aktif	Aktif
4	Aktif	Aktif	Aktif	Aktif
5	Aktif	Aktif	Aktif	Aktif

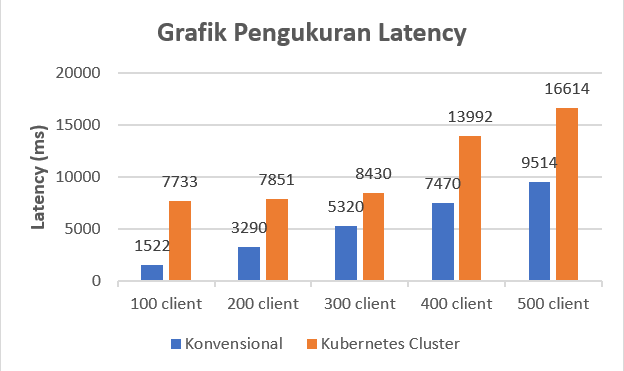
Pada Tabel 2 menunjukkan bahwa status *web server* tetap aktif dan dapat diakses *client* ketika worker2 dihidupkan kembali. Dimana terjadi pembagian beban pada ketiga *node* yang running yaitu master, worker 1 dan worker 2, sehingga layanan *web server* dapat berjalan pada setiap *node*.

B. Hasil Pengujian *Performance*

1. Hasil *Stress Testing*

ISSN: 2339 – 2053

Berdasarkan pengujian yang dilakukan pada saat *stress testing* terhadap *web server* pada masing-masing *server*. Pengukuran tingkat *latency* yang dilakukan dengan 5 kali tahapan pengujian yaitu 100 *client*, 200 *client*, 300 *client*, 400 *client* dan 500 *client* dengan masing-masing pengujian dilakukan 5 kali percobaan dalam mengambil data.



Gambar 1. Grafik pengukuran *latency*

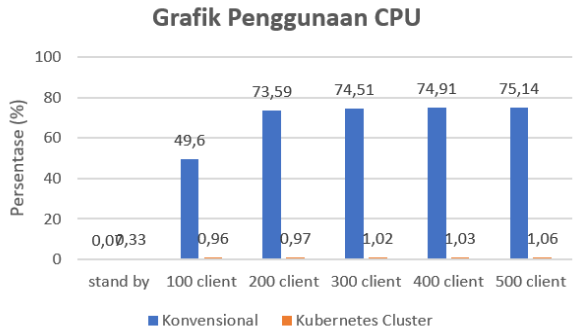
Berdasarkan pengujian yang telah dilakukan diperoleh grafik seperti pada Gambar 1 yang merupakan hasil rata-rata jumlah latency dari semua pengujian yang dilakukan pada *server* konvensional dan *Kubernetes cluster*.

Pada teknologi *Kubernetes cluster* menunjukkan jumlah *latency* paling tinggi yaitu 16614 ms, sedangkan jumlah *latency* paling rendah adalah teknologi konvensional yaitu 1522 ms. Pada sistem konvensional, respons dari pengguna secara langsung yang diterima oleh *web server*, di proses lalu dikirim, sedangkan pada *Kubernetes cluster* terdapat *node-node* yang berjalan di atas mesin virtual yang di dalamnya terdapat *web server* pada masing-masing *node*, sehingga membutuhkan waktu lebih lama dalam memberikan respons. *Latency* yang baik yaitu *latency* yang rendah karena dapat membuka halaman web lebih cepat.

2. Hasil CPU dan *Memory*

Berdasarkan pengujian yang dilakukan pada saat keadaan *server standby* dan *busy* diperoleh data penggunaan CPU dan *memory*. Penggunaan CPU dan *memory* dilakukan dengan keadaan standby, 100 *client*, 200 *client*, 300 *client*, 400 *client* dan 500 *client* dengan masing-masing pengujian dilakukan 5 kali percobaan dalam mengambil data.

ISSN: 2339 – 2053

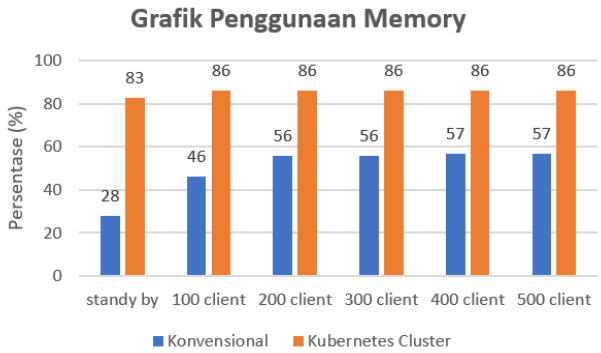


Gambar 2. Grafik penggunaan CPU saat *standby* dan *busy*

Berdasarkan pengujian yang telah dilakukan diperoleh grafik seperti pada Gambar 2 yang merupakan hasil rata-rata *persentase* penggunaan CPU pada *server* konvensional dan *Kubernetes cluster* dalam keadaan *standby* dan *busy*.

Pada saat keadaan *standby*, teknologi konvensional menunjukkan penggunaan CPU terendah yaitu 0,07% karena tidak ada *node-node* yang berjalan di atas mesin virtual, sedangkan penggunaan CPU tertinggi adalah teknologi *Kubernetes cluster* yaitu 0,33% karena terdapat *node-node* yang berjalan di atas mesin virtual.

Pada saat keadaan *busy*, teknologi *Kubernetes cluster* menunjukkan penggunaan CPU terendah yaitu 1,06% karena adanya pembagian beban kerja pada setiap *node* saat menerima *request client*, sedangkan penggunaan CPU tertinggi adalah teknologi konvensional yaitu 75,14% karena tidak adanya pembagian beban kerja terhadap mesin virtual. Selanjutnya adalah analisis penggunaan *memory* pada masing-masing *server*.



Gambar 3. Grafik penggunaan memory saat *standby* dan *busy*

ISSN: 2339 – 2053

Pekanbaru, 25 Agustus 2021

Berdasarkan pengujian yang telah dilakukan diperoleh grafik seperti pada Gambar 3 yang merupakan hasil rata-rata *persentase* penggunaan *memory* pada *server* konvensional dan *Kubernetes cluster* dalam keadaan *standby* dan *busy*.

Pada saat keadaan *standby*, teknologi konvensional menunjukkan penggunaan *memory* terendah yaitu 28% karena tidak ada *node-node* yang berjalan di atas mesin virtual, sedangkan penggunaan *memory* tertinggi adalah teknologi *Kubernetes cluster* yaitu 83% karena terdapat *node-node* yang berjalan di atas mesin virtual.

Pada saat keadaan *busy*, teknologi konvensional menunjukkan penggunaan *memory* terendah yaitu 57% karena tidak ada *node-node* yang berjalan di atas mesin virtual, sedangkan penggunaan *memory* tertinggi adalah teknologi *Kubernetes cluster* yaitu 86% karena terdapat *node-node* yang berjalan di atas mesin virtual.

Pengujian dengan membedakan jumlah *client* ternyata memberikan hasil penggunaan CPU dan *memory* yang berbeda. Semakin banyak *client* yang mengakses maka akan naik penggunaan CPU dan *memory*.

SIMPULAN

Berdasarkan penelitian yang telah dilakukan, ada beberapa hal yang dapat disimpulkan dari penelitian ini, yaitu:

1. *Kubernetes cluster* dapat dibangun di dalam virtualisasi KVM.
2. Penerapan *cluster* dapat mengurai layanan yang tidak dapat diakses apabila terjadi masalah. Dengan menggunakan *cluster*, apabila *node* yang sedang menjalankan servicenya gagal atau mati, maka masih terdapat node yang lain.
3. *Latency* tertinggi yaitu 16614 ms oleh *Kubernetes cluster* dan *latency* terendah yaitu 1522 ms oleh teknologi konvensional berdasarkan semua pengujian yang dilakukan.
4. Dalam keadaan *standby*, pemakaian CPU dan pemakaian *memory* terendah adalah konvensional yaitu 0,07% dan 28% dibandingkan dengan teknologi *Kubernetes cluster* berdasarkan semua pengujian yang dilakukan.

ISSN: 2339 – 2053

5. Dalam keadaan *busy*, pemakaian CPU terendah adalah *Kubernetes cluster* adalah 1,06% dan pemakaian *memory* terendah adalah konvensional yaitu 57% berdasarkan semua pengujian yang dilakukan.

DAFTAR PUSTAKA

Alkhawarizmi. (2019). *Komputasi Modern*. weebly.com. https://mamz.weebly.com/komputasi-modern.html

Apriliana, L., Darusalam, U. D., & Nathasia, N. D. (2018). Clustering Server Pada Cloud Computing Berbasis Proxmox VE Menggunakan Metode High Availability. *JOINTECS (Journal of Information Technology and Computer Science)*. https://doi.org/10.31328/jointecs.v3i1.498

IDCloudHost. (2020). *Mengenal Apa itu Docker, Definisi, Fungsi, Keunggulan dan Cara Kerjanya*. idcloudhost.com. https://idcloudhost.com/mengenal-apa-itu-docker-definisi-fungsi-keunggulan-dan-cara-kerjanya/

IDreg. (2013). *Pengertian KVM, XEN, OPENVZ*. idreg.net. https://www.idreg.net/pengertian-kvm-xen-openvz/

Kubernetes. (2020). *Apa itu Kubernetes?* kubernetes.io. https://kubernetes.io/id/docs/concepts/overview/what-is-kubernetes/

Mell, P., & Grance, T. (2011). The NIST Definition of Cloud Computing. In *Application Performance Management (APM) in the Digital Enterprise* (hal. 267–269). Elsevier. https://doi.org/10.1016/B978-0-12-804018-8.15003-X

ISSN: 2339 – 2053

Pekanbaru, 25 Agustus 2021