CoMBInES
CONFERENCE ON MANAGEMENT, BUSINESS, INNOVATION, EDUCATION AND SOCIAL SCIENCES

Volume 1 No 1 (2021)

Diterima : February 01, 2021
Disetujui : February 05, 2021
Diterbitkan: February 24, 2021

Conference on Management, Business, Innovation, Education and Social Science
https://journal.uib.ac.id/index.php/combines

Analisis Perbandingan Performa *Web Server Docker Swarm* dengan *Kubernetes Cluster*

Stefanus Eko Prasetyo[1], Yulfan Salimin[2]
Email korespondensi : stefanus.eko@uib.edu, 1731047.yulfan@uib.edu

[1]Fakultas Ilmu Komputer, Universitas Internasional Batam, Kota Batam, Indonesia
[2]Fakultas Ilmu Komputer, Universitas Internasional Batam, Kota Batam, Indonesia

Abstrak

Sebuah infrastruktur jaringan yang baik sangat dibutuhkan bagi perusahaan. Infrastruktur dikatakan baik bila dapat mengatasi ratusan maupun ribuan request dari client dan tidak mengalami down. Pada penelitian Mohamad Rexa, performa *web server* dalam *Docker Swarm* meningkatkan availability layanan sebesar 31% percontainer yang ada. Dan juga pada penelitian Imron, *Kubernetes Cluster* mampu menangani jumlah permintaan *client* 19 kali lipat dibanding *single server.* Oleh karena itu penulis merancang infrastruktur dari *Docker Swarm* dan *Kubernetes*. Kemudian melakukan *performance testing* kepada dua infrastruktur tersebut dengan tujuan untuk mengetahui perbandingan dari performa dua kontainerisasi tersebut. Dari hasil perbandingan, perusahaan maupun instansi dapat memilih infrastruktur jaringan yang cocok sesuai kebutuhan

Kata Kunci:

Docker Swarm, Kubernetes Cluster, Performance Test, Web Server.

Pendahuluan

Pertumbuhan kebutuhan data dan informasi diera digital ini semakin tinggi. Untuk mengatasi server yang overload maka diterapkan konsep clustering. Konsep clustering berupa sekumpulan server yang bekerja bersamaan dalam sebuah sistem untuk memberikan high availability. Hal ini berarti kebutuhan server untuk pemberi layanan juga semakin banyak. Namun tuntutan pemanfaatan hardware yang hemat energi dicapai oleh sebuah teknologi yang dikenal dengan virtualisasi containerisasi(Khan et al., 2019).

Containerisasi telah menjadi salah satu tren dalam pengembangan perangkat lunak disaat ini. Metode containerisasi ini lebih efisien, fleksibel dan ringan dibanding metode yang lama. Para developer dan tim operasional mulai meninggalkan metode lama dalam mengembangkan aplikasi karena containerisasi memberikan kemudahan proses kerja dan juga mengurangi sumber daya yang diperlukan dari sistem dan aplikasi yang mereka kembangkan. Metode containerisasi pada dasarnya adalah melakukan pengemasan koding dan dependensi agar aplikasi tersebut dapat bekerja secara seragam dan konsisten pada infrastruktur lainnya. Hasil dari pengemasan koding dan depedensi tersebut disebut dengan container. Salah satu

virtualisasi berbasis container yang paling populer digunakan saat ini adalah Docker(Azab, 2017).

Docker merupakan aplikasi open source yang mengotomatisasi penyebaran aplikasi kedalam container. Pengemasan ini membuat aplikasi tersebut terisolasi, artinya kita dapat memasang banyak aplikasi dalam sebuah server tanpa mempermasalahan bentrokan depedensi antar aplikasi. Namun, mengelola container untuk menciptakan banyak layanan adalah tugas yang berat bagi Docker(Afis, Data, & Yahya, 2019). Oleh karena itu diperlukan sebuah alat orkestrasi yang dapat mengatur container-container.

Terdapat dua alat orkestrasi container yang dikenal luas, yaitu Docker Swarm dan Kubernetes Cluster. Docker Swarm merupakan alat orkestrasi yang dikembangkan oleh docker sendiri. Cluster Docker Swarm memungkinkan seseorang untuk menambahkan jumlah node yang tidak terbatas, dan Docker memungkinkan seseorang untuk menjalankan container tanpa batas pada node, hal ini memberikan pengujian skala penuh yang mendekati kondisi nyata(Shichkina, Kupriyanov, & Moldachev, 2018). Sedangkan Kubernetes Cluster yang dikembangkan oleh Google mengatur node dengan tingkat penggunaan sumber daya yang seimbang. Apalagi di kluster Kubernetes, administrator dapat menentukan skema penempatan multi-layer yang menggabungkan berbagai prioritas agar proses seleksi container dapat menyeluruh(Fu et al., 2019). Keduanya memberikan banyak fitur dalam mengelola sumber daya, kebijiakan keamanan, akses jaringan dan administrasi sistem.

Berdasarkan latar belakang yang telah diberikan di atas, penulis akan melakukan penelitian untuk mengetahui efisiensi web server cluster container dari Docker Swarm dan Kubernetes Cluster dengan menganalisa performa kedua web server tersebut. Penelitian ini dengan judul "Analisis Perbandingan Performa Web Server Docker Swarm dengan Kubernetes Cluster". Penulis berharap dengan adanya penelitian analisis performa ini, para penyedia layanan dapat menggunakan alat orkerstrasi cluster container yang cocok sesuai kebutuhan mereka.

Tinjauan Pustaka

Menurut Demirkol & Demirkol (2018), dengan penelitian berjudul "Energy Efficiency with an Application Container". Penggunaan teknik containerisasi sangat diperlukan dalam memperluas IT yang ramah lingkungan dan cost-effective. Pada penelitian ini, performa container dibandingkan dengan virtual machine kernel dan hasilnya container lebih unggul pada benchmark yang dilakukan terhadap *CPU performance*, kecepatan RAM, dan apache. Sedangkan virtual machine kernel unggul pada *Loopback* performa jaringan TCP dan *AIO-stress*.

Menurut Ramana, Ponnavaikko, & Subramanyam (2019), menggunakan server cluster dapat meningkatkan kinerja *web server* dan efektivitas ini dicapai tergantung pada proses pendistribusian permintaan *client*. Pendistribusian permintaan client harus terjadi dengan cara yang transparan bagi pengguna diantara beberapa *node server*, yang mempengaruhi *availability* dan *scalability* dalam sistem *web server* terdistribusi.

Pada penelitian Rexa, Data, & Yahya (2019), *failover* dan *load balancer* berdasarkan *memory* dapat bekerja dalam *Docker Swarm* dan berhasil mendistribusikan *traffic web server* secara rata antar *host*. Dan juga penelitian Rosyadi, (2019), dalam menguji performa *web server clustering* menggunakan *Kubernetes*. Performa *web server* ini dapat menangani jumlah permintaan *client* 19 kali lipat daripada *web server* yang tidak menggunakan virtualisasi dan *autoscaling*.

Penelitian "MARKA: A Microservice Architecture-Based Application Performance Comparison Between Docker Swarm and Kubernetes" yang dilakukan Günaydın, Cebeci, & Suba¸sı (2020), performa *docker swarm* dan *Kubernetes Cluster* dibandingkan dengan cara membebani server dengan meningkatkan jumlah pengguna. Dari hasil penelitian ini dapat diketahui bahwa docker swarm lebih efisien ketika jumlah pengguna semakin meningkat dibanding dengan Kubernetes cluster. Namun pada penelitan Marathe, Gandhi, & Shah (2019), tentang "Docker Swarm and Kubernetes in Cloud Computing Environment". Performa CPU dan memory pada Kubernetes lebih baik dari docker swarm.

Dalam melakukan load testing, penelitian Abbas, Sultan, & Bhatti (2017), yang berjudul "Comparative Analysis of Automated Load Testing Tools: Apache JMeter, Microsoft Visual Studio (TFS), LoadRunner, Siege" menyimpulkan bahwa keempat testing tools bekerja dengan baik dalam melakukan uji otomatis. Tapi Apache JMeter memberikan hasil yang lebih baik dari tools lain karena JMeter termasuk metode pengujian rasio dan juga memiliki permeriksaan konsistensi.

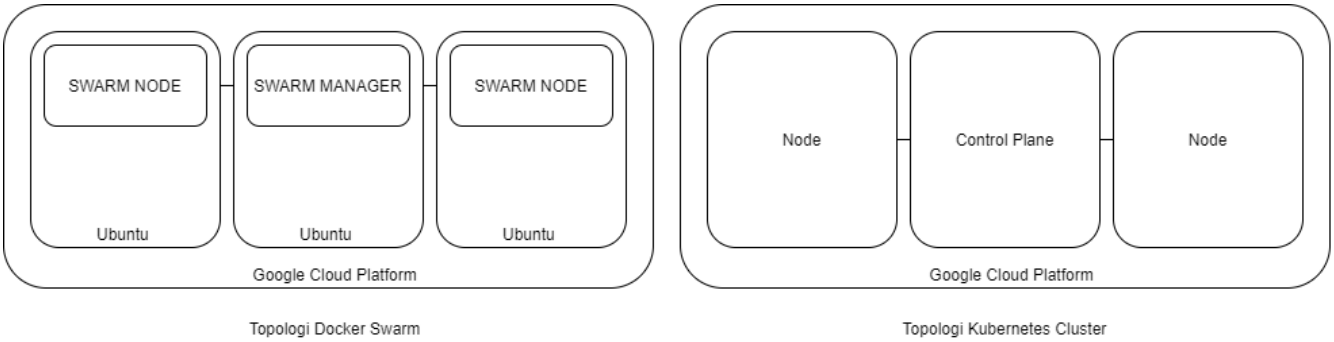
Berdasarkan kajian penelitan yang telah ada, maka peneliti merancang sebuah penelitian untuk menganalisis perbandingan performa antara *Docker Swarm* dan *Kubernetes Cluster* menggunakan tools *JMeter* untuk mengetahui bagaimana performa dari masing-masing virtualisasi container tersebut.

Metodologi Penelitian

Penelitian ini dilakukan dengan metode load testing, metode ini mengukur respon server dalam berbagai kondisi load. Parameter pengukuran respon server yang digunakan dalam load testing ini adalah sebagai berikut:

1. Response Time: Jumlah total waktu yang diperlukan untuk menanggapi permintaan layanan. Response time terbagi menjadi tiga, Average - waktu rata-rata dari serangkaian hasil, Min - waktu terpendek yang client tunggu agar server merespon, Max - waktu terlama yang client tunggu agar server merespon.
2. Standard Deviation: Kestabilan response time, bisa juga dikatakan dengan bagaimana rata-rata waktu respon tersebar.
3. Error: Persentase permintaan yang tidak terpenuhi
4. Throughput: Jumlah permintaan yang diproses per detik
5. CPU Utilization: Penggunaan sumber daya CPU komputer, atau jumlah pekerjaan yang ditangani oleh CPU pada untuk menanggapi permintaan layanan.

Server *docker swarm* dan server *Kubernetes cluster* diuji dalam dua kondisi load, yang pertama adalah load 300 user dengan loop sebanyak 20 kali, sehingga menghasilkan total data pengujian sebanyak 6000 kali. Sedangkan yang kedua adalah load 2000 user dengan loop sebanyak 20 kali, sehingga menghasilkan total data pengujian sebanyak 40000 kali. Dalam pelaksanaan penelitian ini dibangun dua mesin dengan spesifikasi yang sama menggunakan google cloud platform sebagai platform penyedia layanan cloud. Berikut adalah desain topologi dari masing-masing sistem:



Gambar 1 Topologi Sistem

Beberapa perangkat keras yang digunakan dalam penelitian ini adalah sebagai berikut:

1. Google Cloud Server, sebagai platform cloud server dari google untuk tempat pengimplementasian docker swarm dan Kubernetes cluster. Dengan tiap instance memiliki spesifikasi utama:
   Processor : 2 Core
   Memory : 4 GB
   Harddisk : 50 GB
   Total 6 instance, tiga untuk docker swarm (satu manager dan dua worker) dan tiga untuk Kubernetes cluster (satu control plane dan dua node).
2. Sebuah laptop untuk melakukan analisa performa docker swarm dan Kubernetes cluster.

Beberapa perangkat lunak yang digunakan dalam penelitian ini adalah sebagai berikut:

1. Docker swarm sebagai alat orkestrasi container, menggunakan sistem operasi Ubuntu 16.04 LTS.
2. Kubernetes Cluster sebagai alat orkestrasi container, menggunakan sistem operasi Ubuntu 16.04 LTS.
3. Nginx version 1.1.3 sebagai aplikasi yang akan diimplementasikan pada kedua cluster, berupa aplikasi web server.
4. Apache Jmeter version 5.4.1 sebagai alat untuk mengukur performa aplikasi web server yang terpasang pada container masing-masing.

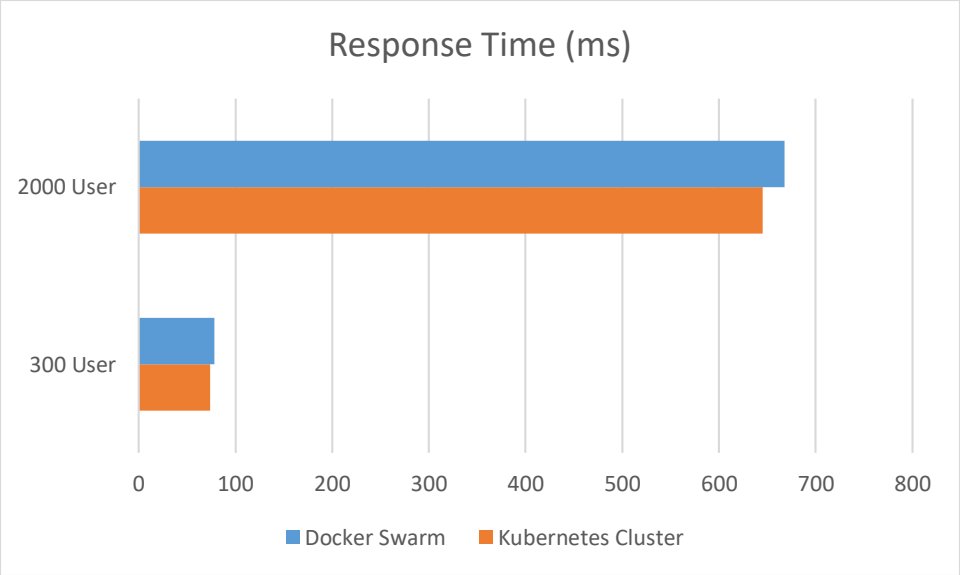
Hasil dan Pembahasan

Sebelum pengujian dilakukan kepada dua server cluster dengan alat orkestrasi yang berbeda. Kedua server cluster dibangun didalam Google Cloud Platform dengan spesifikasi server yang sama. Kemudian docker swarm manager dihubungkan dengan dua swarm node, Kubernetes cluster control plane dihubungkan dengan dua node. Setelah itu installasi Nginx sebagai web server pada swarm manager dan control plane, dan direplikasi ke node lainnya. Kedua server cluster kemudian diuji load test menggunakan aplikasi Apache Jmeter pada laptop yang telah disiapkan. Hasil data pengukuran disusun berdasarkan parameter yang digunakan pada penelitian ini.

Tabel 1 Hasil Response Time

Server	Response Time					
	300 User			2000 User		
	Min	Max	Avg	Min	Max	Avg
Docker Swarm	25	1195	78	25	42779	668
Kubernetes Cluster	26	1364	74	26	43336	645

Response time menunjukkan waktu yang dibutuhkan oleh server untuk merespon klien, sehingga semakin cepat response time yang ada maka server semakin bagus. Dari data yang terkumpul dapat diketahui bahwa dari kedua load test, Kubernetes Cluster lebih unggul karena rata rata response time yang lebih rendah.

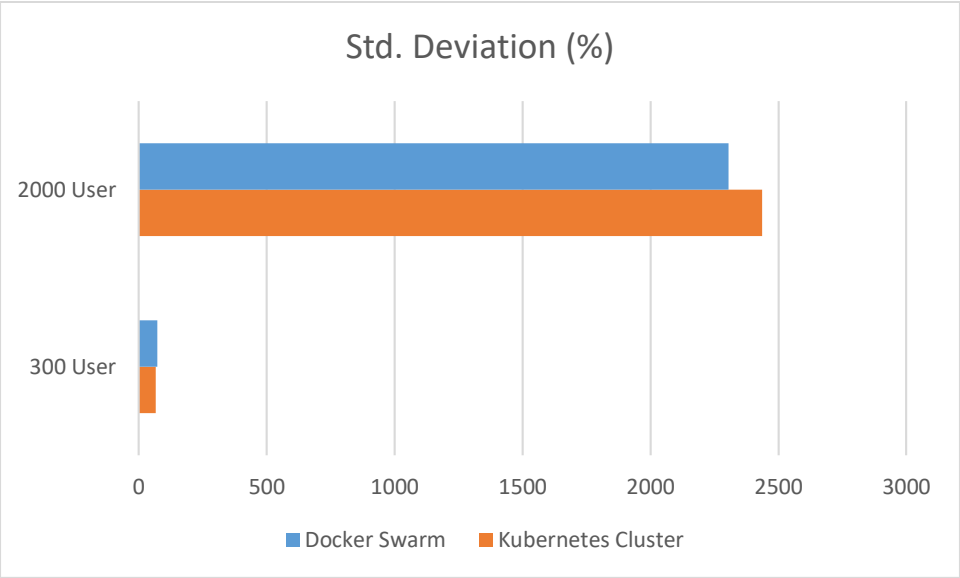


Gambar 2 Chart Response Time (ms)

Tabel 2 Hasil Standard Deviation

Server	Deviation	
	300 User	2000 User
Docker Swarm	72.68	2305.85
Kubernetes Cluster	65.62	2436.28

Standard Deviation merupakan kestabilan response time, bisa juga dikatakan dengan bagaimana rata-rata waktu respon tersebar. Semakin kecil persentase Std. deviation, semakin stabil server dalam merespon permintaan. Pada data std. deviation Kubernetes cluster tetap menunjukkan performa yang lebih bagus dari docker swarm pada saat load test 300 user, namun untuk load test 2000 user performa docker swarm menyalip Kubernetes cluster.

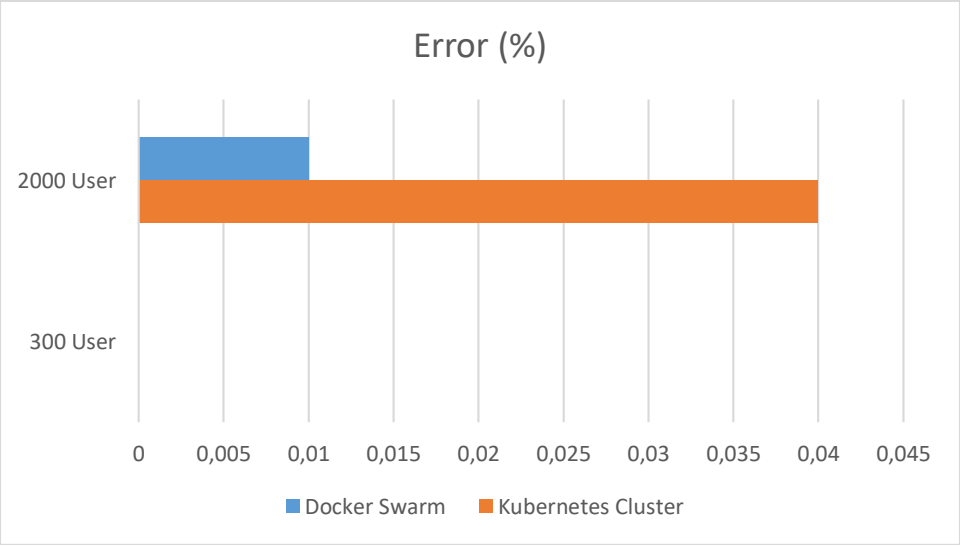


Gambar 3 Chart Std. Deviation (%)

Tabel 3 Hasil Error

Server	Error					
	300 User			2000 User		
	Total Req.	Comp. Req.	Error (%)	Total Req.	Comp. Req.	Error (%)
Docker Swarm	6000	6000	0	20000	19998	0.01
Kubernetes Cluster	6000	6000	0	20000	19992	0.04

Error berupa hasil persentase permintaan yang tidak terpenuhi kedua server. Pada load 300 user kedua server memiliki 0% error, tapi pada load 2000 user, Kubernetes cluster lebih banyak error dibanding docker swarm.

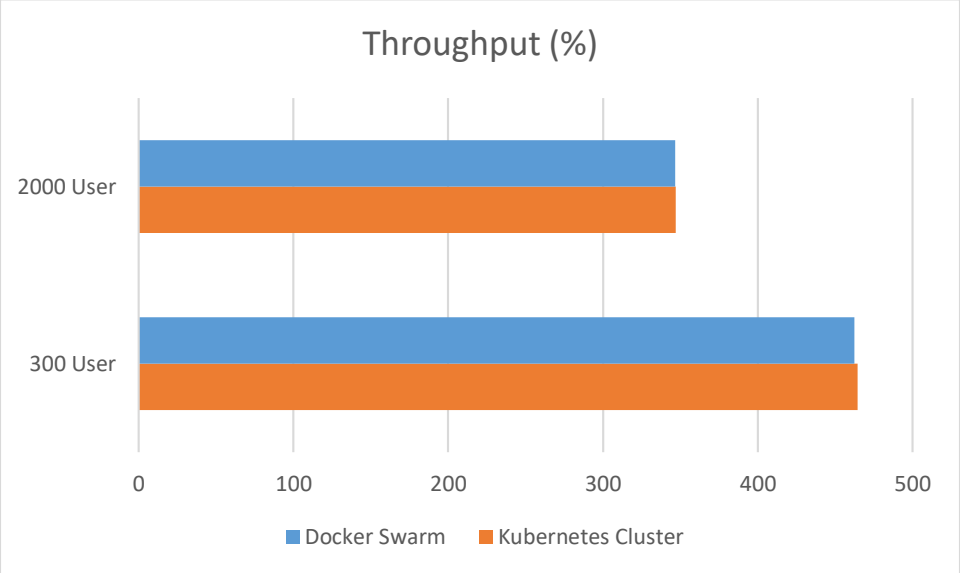


Gambar 4 Chart Error (%)

Tabel 4 Hasil Throughput

Server	Throughput	
	300 User	2000 User
Docker Swarm	462.6	346.7
Kubernetes Cluster	464.5	346.9

Throughput merupakan parameter yang umum dijadikan acuan dalam menentukan kualitas koneksi jaringan. Semakin tinggi nilai throughput semakin bagus juga kualitas jaringannya. Pada test yang telah dilakukan, tidak ada hasil yang significant dari kedua load test.

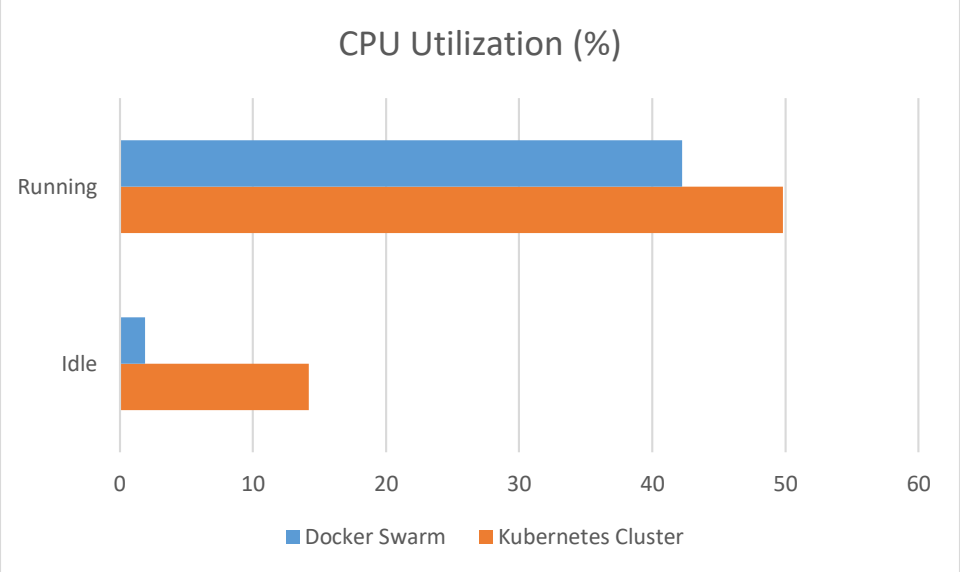


Gambar 5 Chart Throughput (%)

Tabel 5 Hasil CPU Utilization

Server	CPU Utilization	
	Idle	Running
Docker Swarm	1.91	42.23
Kubernetes Cluster	14.19	49.82

CPU Utilization berupa penggunaan komputer dalam pemrosesan sumber daya, atau jumlah pekerjaan yang ditangani oleh CPU. Tingginya tingkat persentasi CPU Utilization dapat menunjukkan kinerja aplikasi yang buruk. Docker swarm lebih hemat sumber daya CPU komputer dari pada Kubernetes cluster pada saat server idle maupun running.



Gambar 6 Chart CPU Utilization (%)

Berdasarkan komparasi parameter data yang dihasilkan oleh penelitian didapatkan hasil bahwa performa Kubernetes cluster lebih baik dibanding dengan docker swarm. Response time kubernetes cluster lebih unggul 23ms dan Std. Deviation 7.06% lebih kecil pada load test 300 thread. Namun dalam penggunaan resource CPU, docker swarm lebih jauh unggul dibanding Kubernetes cluster dengan penggunaan CPU saat running 7.59% lebih rendah dari Kubernetes Cluster.

Kesimpulan

Berdasarkan penelitian yang telah dilakukan, dapat disimpulkan bahwa dalam mengorkestrasi cluster container, docker swarm dan Kubernetes memiliki kelebihan masing-masing. Dari load testing 300 thread dan 2000 thread untuk web server nginx yang dilakukan, Kubernetes cluster memberikan performa yang bagus dengan response time yang lebih cepat dibanding docker swarm. Namun docker swarm dapat memberikan performa yang tidak jauh beda dari Kubernetes cluster dengan menggunakan resource yang lebih efisien. Ada juga beberapa saran untuk penelitian kedepannya seperti menganalisis performa email server maupun file server dalam pengimplementasian cluster container.

Daftar Pustaka

Abbas, R., Sultan, Z., & Bhatti, S. N. (2017). Comparative analysis of automated load testing tools: Apache JMeter, Microsoft Visual Studio (TFS), LoadRunner, Siege. *International Conference on Communication Technologies, ComTech 2017*, 39–44. https://doi.org/10.1109/COMTECH.2017.8065747

Afis, D. S., Data, M., & Yahya, W. (2019). Load Balancing Server Web Berdasarkan Jumlah Koneksi Klien Pada Docker Swarm. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer (J-PTIIK) Universitas Brawijaya*, *3*(1), 925–930.

Azab, A. (2017). Enabling docker containers for high-performance and many-task computing. *Proceedings - 2017 IEEE International Conference on Cloud Engineering, IC2E 2017*, 279–285. https://doi.org/10.1109/IC2E.2017.52

Demirkol, Ô. E., & Demirkol, A. (2018). Energy Efficiency with an Application Container.

Turkish Journal of Electrical Engineering and Computer Sciences, *26*(2), 1129–1139. https://doi.org/10.3906/elk-1801-284

Fu, Y., Zhang, S., Terrero, J., Mao, Y., Liu, G., Li, S., & Tao, Di. (2019). Progress-based Container Scheduling for Short-lived Applications in a Kubernetes Cluster. *Proceedings - 2019 IEEE International Conference on Big Data, Big Data 2019*, 278–287. https://doi.org/10.1109/BigData47090.2019.9006427

Günaydın, T., Cebeci, G., & Suba¸sı, Ö. (2020). *MARKA: A Microservice Architecture-Based Application Performance Comparison Between Docker Swarm and Kubernetes*.

Khan, A. A., Zakarya, M., Buyya, R., Khan, R., Khan, M., & Rana, O. (2019). An energy and performance aware consolidation technique for containerized datacenters. *IEEE Transactions on Cloud Computing*, (April), 1–1. https://doi.org/10.1109/tcc.2019.2920914

Marathe, N., Gandhi, A., & Shah, J. M. (2019). Docker swarm and kubernetes in cloud computing environment. *Proceedings of the International Conference on Trends in Electronics and Informatics, ICOEI 2019*, *2019-April*(Icoei), 179–184. https://doi.org/10.1109/icoei.2019.8862654

Ramana, K., Ponnavaikko, M., & Subramanyam, A. (2019). A Global Dispatcher Load Balancing (GLDB) Approach for a Web Server Cluster. In *Proceedings of the International Conference on Communications and Cyber Physical Engineering 2018* (Vol. 500). https://doi.org/10.1007/978-981-13-0212-1

Rexa, M., Data, M., & Yahya, W. (2019). Implementasi Load Balancing Server Web Berbasis Docker Swarm Berdasarkan Penggunaan Sumber Daya Memory Host. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer (J-PTIIK) Universitas Brawijaya*, *3*(4), 3478–3487.

Rosyadi, I. (2019). *Implementasi Autoscaling Container Web Server Menggunakan Kubernetes Berbasis Proxmox pada Server Universitas Darussalam Gontor*.

Shichkina, Y. A., Kupriyanov, M. S., & Moldachev, S. O. (2018). Application of Docker Swarm cluster for testing programs, developed for system of devices within paradigm of Internet of things. *Journal of Physics: Conference Series*, *1015*(3). https://doi.org/10.1088/1742-6596/1015/3/032129