



9th Applied Business and Engineering Conference

IMPLEMENTASI *CLUSTERED CONTAINER* DENGAN DOCKER SARM

Suci Nurlita Putri¹, Muhammad Arif Fadhly Ridha²

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

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

E-mail: sucinurlita17ti@mahasiswa.pcr.ac.id, fadhly@pcr.ac.id

Abstract

Rapid technological developments can encourage jobs in all fields. Technology is created based on ever-increasing needs and resources. However, the problem that often occurs is that high traffic causes the server to go down because the server is still using conventional computing, so that the server workload becomes heavy and overloaded. By using cluster computing technology, incoming processes can be distributed evenly among each device or hardware node. As technology advances, clusters can be combined with containers. One of the container technology management platforms is docker. Docker has one of its products, docker swarm, which can deploy containers on multiple hosts. When the server is on standby, conventional technology uses the highest CPU and memory. Where CPU usage is 6.2 ms and memory usage is 29.4%. And when the server is busy, conventional technology uses high amounts of CPU and memory. CPU usage is 4324.1 m/s and memory usage is 45%. So that the test results obtained that swarm technology can lighten the burden of server performance compared to conventional.

Keywords: *Server, Docker, Swarm, CPU, Memory.*

Abstrak

Perkembangan teknologi yang pesat dapat mendorong pekerjaan di segala bidang. Teknologi diciptakan berdasarkan kebutuhan dan sumber daya yang terus meningkat. Namun permasalahan yang sering terjadi adalah trafik yang tinggi menyebabkan server down karena server masih menggunakan komputasi konvensional, sehingga beban kerja server menjadi berat dan overload. Dengan menggunakan teknologi komputasi cluster, proses yang masuk dapat didistribusikan secara merata di antara setiap perangkat atau node perangkat keras. Seiring kemajuan teknologi, cluster dapat digabungkan dengan container. Salah satu platform pengelola teknologi container adalah docker. Docker memiliki salah satu produknya yaitu docker swarm, yang dapat menerapkan container pada banyak host. Pada saat server dalam keadaan standby, teknologi konvensional menggunakan CPU dan memory yang paling tinggi. Dimana penggunaan CPU sebesar 6.2 ms dan penggunaan memory sebesar 29.4%. Dan ketika saat server dalam keadaan busy, teknologi konvensional menggunakan CPU dan memory dalam jumlah yang tinggi. Pada penggunaan CPU sebesar 4324.1 m/s dan penggunaan memory sebesar 45%. Sehingga didapatkan hasil pengujian bahwa teknologi swarm dapat meringankan beban kinerja sever dibandingkan dengan konvensional.

Kata Kunci: *Server, Docker, Swarm, CPU, Memory.*



PENDAHULUAN

Teknologi dan informasi di era globalisasi telah mengalami perkembangan yang sangat pesat khususnya pada penggunaan teknologi komputasi. Jenis-jenis komputasi *modern* terbagi menjadi tiga jenis yaitu komputasi bergerak, komputasi *grid* dan *cloud computing* (Alkhawarizmi, 2019).

Cloud computing adalah teknologi yang menjadikan internet sebagai server pusat untuk mengelola data dan aplikasi pengguna. *Cloud computing* memungkinkan pengguna untuk dengan mudah menjalankan program tanpa terlebih dahulu menginstal aplikasi, dan memudahkan pengguna untuk mengakses data dan informasi melalui Internet (IDCloudHost, 2019). Layanan *cloud computing* tersimpan pada server. Banyaknya aplikasi yang berasal dari berbagai platform dapat memberatkan kinerja server, oleh karena itu diperlukan sebuah *cluster computing* untuk mengoptimalkan layanan. *Cluster computing* digunakan untuk dapat meningkatkan kinerja dan ketersediaan satu komputer, dengan biaya yang jauh lebih hemat dibandingkan komputer tunggal dengan kecepatan yang sebanding. *Cluster computing* adalah dua atau lebih perangkat komputasi yang bekerja sama dalam memproses suatu masalah sehingga dapat menghasilkan kecepatan yang sangat tinggi dalam prosesnya untuk komputasi umum (IdCloudHost, 2015). Untuk membuat *cluster computing* diperlukan *container*. Diperlukan beberapa *container* untuk menjalankan *service* atau layanan yang sama untuk menjalankan aplikasi.

Docker dapat digunakan untuk mengatasi masalah tersebut. Docker adalah salah satu platform yang dibangun berdasarkan teknologi *container*. Docker merupakan sebuah project *open-source* yang menyediakan platform terbuka untuk *developer* maupun *sysadmin* untuk dapat membangun, mengemas, dan menjalankan aplikasi dimanapun sebagai sebuah wadah (*container*) yang ringan. Docker memiliki salah satu produk yaitu *docker swarm* untuk mendeploy *container* pada *multihost*. *Docker swarm* adalah *container orchestration* dan *clustering tool* untuk mengatur *docker hosts*, yang merupakan bagian dari *docker engine*. *Docker swarm* merupakan *original clustering* yang dibuat dan disediakan oleh Docker sehingga menjamin *high availability* dan *high performance* aplikasi yang berjalan. Kelebihan Docker Swarm adalah jika salah satu

host down maka service docker akan digantikan dengan *host* yang sedang aktif (Bella, Data, & Yahya, 2019).

Berdasarkan uraian tersebut, maka dilakukan penelitian mengenai “Implementasi Clustered Container dengan Docker Swarm” dengan tujuan membangun *clustered computer* menggunakan docker swarm untuk membagi beban pada masing masing node, sehingga meminimalkan terjadinya layanan yang tidak dapat diakses, dan membandingkan kinerja CPU dan *memory* antara komputasi konvensional dengan *clustered container*.

METODE PENELITIAN

Adapun metode penelitian yang dilakukan yaitu sebagai berikut:

A. Pengujian *High-Availability*

Pengujian yang dilakukan pada web server akan diuji dengan cara, yaitu:

1. Pengujian *failover* bertujuan untuk menguji apakah layanan tetap diberikan meskipun hanya satu mesin yang digunakan.
2. Pengujian *failback* bertujuan untuk menguji apakah layanan akan berpindah dari satu *node* ke *node* lainnya.
3. Pengujian *Load Balancing* bertujuan untuk mendistribusikan beban kinerja server secara merata sehingga server tidak akan kelebihan beban saat menerima permintaan dari *client*.

B. Pengujian *Performance*

Pengujian *performance* diuji dengan beberapa cara yaitu:

1. Pengujian *Stress Testing*

Tujuan dari pengujian ini adalah untuk menentukan dan mengukur kekuatan web server saat menggunakan *tools* JMeter.

2. Pengujian performa virtualisasi web server dalam keadaan *standby* dan *busy*
Pengujian ini dilakukan untuk melihat persentase penggunaan CPU dan memory yang terpakai pada web server tidak di akses *client (standby)* dan di akses *client (busy)* pada server konvensional dan *clustered container* swarm dengan menggunakan aplikasi protokol SNMP.

C. Pengujian *Scalability*

Pengujian ini dilakukan untuk melihat kapan beban layanan meningkat, untuk mengetahui pengujian scalability dapat dilakukan dengan cara berikut:

- a. Menambahkan *container* jika jumlah *client* yang mengakses *website* semakin banyak dan sebaliknya.
- b. Menambahkan *node* jika jumlah *client* yang mengakses *website* semakin banyak dan sebaliknya.

HASIL DAN PEMBAHASAN

A. Hasil Pengujian *High-Availability*

Pengujian ini dilakukan sebanyak 5 kali pengujian agar dapat melihat tingkat keberhasilan dari pengujian *load balancer*, *failover* dan *failback*. Pada pengujian *failover* ini dilakukan skenario jika worker2 diofflinekan apakah node lainnya menggantikan fungsi layanan dari worker2 dan web server masih dapat diakses oleh *client* atau tidak. 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	Tidak Aktif	Aktif
2	Aktif	Aktif	Tidak Aktif	Aktif
3	Aktif	Aktif	Tidak Aktif	Aktif
4	Aktif	Aktif	Tidak Aktif	Aktif
5	Aktif	Aktif	Tidak Aktif	Aktif

Pada Tabel 1 menunjukkan bahwa status web server tetap aktif dan dapat diakses *user* ketika *worker2* telah *diofflinekan*. Layanan web server diambil fungsi oleh *worker1*. Berikut hasil pengujian tingkat keberhasilan *failback* dapat dilihat pada Tabel 2.

Tabel 2
Hasil Pengujian *Failback*

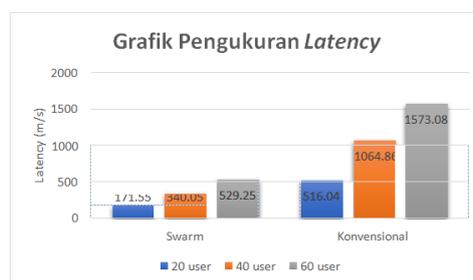
Pengujian Ke -	Node			Status Web Server
	Master	Worker 1	Worker 2	
1	Aktif	Aktif	Aktif	Aktif
2	Aktif	Aktif	Aktif	Aktif
3	Aktif	Aktif	Aktif	Aktif
4	Aktif	Aktif	Aktif	Aktif
5	Aktif	Aktif	Aktif	Aktif

Pada Tabel 2 melakukan simulasi *user* sebanyak 20, 40, dan 60 user. Percobaan dilakukan sebanyak 5 kali.

B. Hasil Pengujian *Performance*

1. Hasil *Stress Testing*

Pengukuran tingkat *latency* dilakukan dengan 3 kali tahapan pengujian yaitu dengan 20 *client*, 40 *client* dan 60 *client* dengan masing-masing pengujian dilakukan 5 kali percobaan pengambilan data.



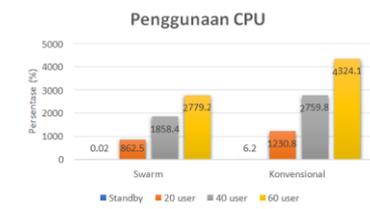
Gambar 1. Grafik pengukuran *latency*

Berdasarkan pengujian yang telah dilakukan diperoleh grafik seperti pada gambar 1 yang merupakan jumlah *latency* dari masing-masing server. Pada teknologi konvensional menunjukkan jumlah *latency* paling tinggi yaitu mencapai 1573.08 m/s.

Hal ini disebabkan karena dilakukannya *stress request* yang tinggi menyebabkan tingkat *delay* pada web server yang semakin besar. Sedangkan jumlah *latency* paling sedikit adalah teknologi swarm hanya 171.55m/s. Hal ini disebabkan karena adanya pembagian beban kerja pada *node-node* saat *stress testing* yang menerima *request* dari *client*, sehingga tingkat *delay* pada web server semakin kecil.

2. Hasil CPU dan *Memory*

Berdasarkan pengujian yang dilakukan pada saat server keadaan *standby* dan *busy* diperoleh data penggunaan CPU. Penggunaan CPU berikut dilakukan dengan keadaan *standby*, 20 klien, 40 klien dan 60 klien dengan masing-masing pengujian dilakukan 5 kali percobaan pengambilan data.



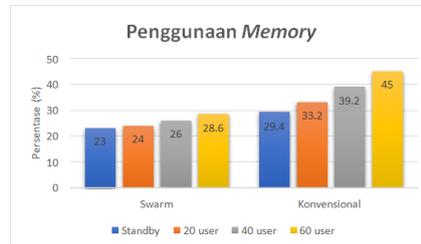
Gambar 2. Grafik penggunaan CPU saat *standby* dan *busy*

Berdasarkan pengujian yang telah dilakukan diperoleh grafik seperti pada gambar 2 yang merupakan penggunaan CPU keadaan *standby* dan *busy*. Pada saat keadaan *busy*, teknologi konvensional menunjukkan penggunaan CPU paling tinggi yaitu mencapai 4324.1% dikarenakan *request client* meningkat saat dilakukannya *stress testing*. Sedangkan penggunaan CPU paling sedikit saat menerima *request client* adalah teknologi swarm hanya 2779.2%. Hal ini disebabkan karena pembagian beban kerja pada *node* saat menerima request dari klien.

Pada saat keadaan *standby*, teknologi konvensional menunjukkan CPU paling tinggi yaitu mencapai 6.2%. Sedangkan penggunaan CPU paling sedikit adalah teknologi swarm hanya 0.02%.

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

naik penggunaan CPU. Selanjutnya adalah analisa untuk penggunaan *memory* pada masing-masing server pada saat *standby* dan diakses oleh 60 *client* secara bertahap,



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

Berdasarkan pengujian yang telah dilakukan diperoleh grafik seperti gambar 3 yang merupakan penggunaan memory keadaan *standby* dan *busy*.

Pada pengujian ini ternyata menunjukkan hasil yang sama dengan penggunaan *memory*, yaitu pada saat keadaan *busy* teknologi konvensional menunjukkan penggunaan *memory* paling tinggi yaitu mencapai 45%. Tentu penyebabnya sama yaitu karena request klien meningkat saat dilakukannya stress testing. Pada saat keadaan *standby*, teknologi konvensional menunjukkan *memory* paling tinggi yaitu mencapai 29.4%. Sedangkan penggunaan *memory* paling sedikit adalah teknologi konvensional hanya 23%.

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

SIMPULAN

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

1. Teknologi Swarm menunjukkan jumlah *latency* terendah dibandingkan dengan konvensional berdasarkan semua pengujian yang dilakukan.
2. Teknologi konvensional menggunakan sumber daya lebih besar seperti CPU dan *memory* dibandingkan dengan swarm berdasarkan semua pengujian yang dilakukan.



9th Applied Business and Engineering Conference

3. Dengan menggunakan teknologi swarm, kinerja server menjadi lebih ringan dibandingkan dengan konvensional.

DAFTAR PUSTAKA

- Alkharizmi. (2019). Komputasi Modern. Retrieved 11 20, 2020, from mamz.weebly: <https://mamz.weebly.com/komputasi-modern.html>
- Bella, M. R., 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*, 10.
- IDCloudHost. (2019, September 16). *Mengenal Apa itu CloudComputing : Defenisi, Fungsi, dan Cara Kerja*. Retrieved 11 18, 2020, from idcloudhost.com: <https://idcloudhost.com/mengenal-apa-itu-cloud-computing-defenisi-fungsi-dan-cara-kerja/>
- IdCloudHost. (2015). Retrieved from Mengenal Pengertian Komputer Kluster: <https://idcloudhost.com/mengenal-pengertian-komputer-cluster/>
- Bik, M. F. (2017). IMPLEMENTASI DOCKER UNTUK PENGELOLAAN BANYAK APLIKASI WEB (Studi Kasus : Jurusan Teknik Informatika UNESA). *Jurnal Manajemen Informatika. Volume 7 Nomor 2 Tahun 2017, 46-50*, 5.
- Sagala, M. A. (2010). Implementasi Load Balancing Pada Web Server. *Departemen Teknik Elektro Fakultas Teknik Universitas Sumatera Utara*.
- Lestaringati, S. I., & Rozak, F. (2014). Pembangunan Aplikasi Monitoring Jaringan Berbasis Web Menggunakan Simple Network Management Protocol (SNMP). *Majalah Ilmiah UNIKOM*.
- Lysy C. Moleong, A. M. (2013). Implementasi Cluster Computing Untuk Render Animasi. *e-Jurnal Teknik Elektro dan Komputer*, 8.