ANALISIS PERFORMANSI FTP SERVER, WEB SERVER, DAN MAIL SERVER PADA CONTAINER DOCKER, LXC, DAN LXD

PERFORMANCE ANALYSIS OF FTP SERVER, WEB SERVER, AND MAIL SERVER ON CONTAINER DOCKER, LXC, AND LXD

Adinda Riztia Putri[1], Dr. Rendy Munadi, Ir.,M.T.[2], Ridha Muldina Negara, S.T., M.T.[3]

[1,2,3]Prodi S1 Teknik Telekomunikasi, Fakultas Teknik Elektro, Universitas Telkom
[1]adindariztia@student.telkomuniversity.ac.id, [2]rendymunadi@telkomuniveristy.ac.id,
[3]ridhanegara@telkomuniversity.ac.id

Abstrak

Teknologi *containerization* sebagi salah satu alternatif dalam virtualisasi terus berkembang. Fleksibilitas dan efektivitas yang ditawarkan oleh *containerization* bahkan kini telah terintegrasi dengan Cloud Computing, salah satunya adalah implementasi project Magnum dalam Open Stack sebagai salah satu solusi bagi *large and scalable cloud.* Container hadir dalam banyak nama, diantaranya yang paling banyak diadaptasi oleh enterprise dan paling banyak disebut adalah Doker, LXC, dan LXD.

Penelitian ini akan membahas tentang performansi Docker, LXC, dan LXD ketika menjalankan layanan server yang paling popular di cloud, yaitu FTP Server, Mail Server, dan Web Server. Pengujian dilakukan dengan tujuan untuk mengetahui overall performance dari server yang dijalankan di dalam masing-masing container dibandingkan dengan server native. Selain itu, pengujian ini juga melihat performansi masing-masing layanan server yang diujikan berupa HTTP request yang dapat dilayani oleh Web Server dalam waktu tertentu, waktu yang dibutuhkan untuk menyelesailkan FTP Request pada FTP request dan performansi berupa banyaknya pesan yang dapat dikirim sebuah mail server melalui SMTP Protocol.

Dari hasil penelitian didapatkan bahwa terdapat perbedaan performansi antara sistem native dengan sistem container Docker, LXC, dan LXD. Performansi sistem LXD menunjukkan hasil paling unggul dengan persentase 90,5% dibandingkan dengan native berdasarkan metric overall performance. Untuk pengujian FTP Server, LXD menunjukkan performansi yang unggul dari segi response time, sedangkan dari segi latency dan transfer rate menunjukkan LXC memiliki performansi yang terbaik di antara ketiganya. Pada pengujian Mail server, ketiga container memiliki selisih performansi yang cukup kecil bahkan jika dibandingkan dengan native, hanya terpaut di bawah 2%. Sedangkan pada pengujian web server, LXD menunjukkan performa yang baik jika digunakan untuk menjalankan webserver dengan concurrent level rendah. Untuk tingkat concurrent level yang lebih tinggi, LXC menunjukkan nilai yang paling stabil dan cepat diantara ketiganya.

Kata kunci : Docker, LXC, LXD, cloud computing

Abstract

Containerization technology as an alternative in virtualization is developing. Flexibility and effectivity offered by containerization technology now is integrated with cloud computing, the implementation of Magnum project in OpenStack as one of the solution for large and scalable cloud. Container has many names, among them are Docker, LXC, and LXD, the most adapted container for enterprise.

This research will reveal the performance of aforementioned container when running popular services on cloud, FTP Server, Mail Server, and Web Server. The testing is done with a purpose to know the overall performance of a server running on the top of each container compared to native server. Furthermore, the goal of this research is also to know the performance of each tested server's service, such as HTTP request that can be served by a web server in a time, time needed to complete FTP request on FTP server and performance of how many messages that a Mail Server able to process by SMTP Protocol.

Result of this research shows that there is a difference between native and Docker, LXC, and LXD performance system. Performance system of LXD show the most promising one with 90,5% compared to native based on overall performance metrix. While in FTP Server testing, LXD show the best result in term of response time, while in latency and transfer rate dominated by LXC. In email server testing, all of the container has a very little performance difference compared to native, below 2%. While in web server testing, LXD show a better result if running on a low concurrent level that is 2000 user at a time. For a higher concurrent level, LXC show a better and stable result compared to other containers.

Keywords: Docker, LXC, LXD, cloud computing.

1. Pendahuluan

Cloud computing bergantung pada teknologi virtualisasi yang membutuhkan sumber daya seperti server dan membagi-baginya menjadi sumber daya virtual yang disebut *Virtual Machine* (VM) [2]. Performansi dari sebuah VM mempunyai dampak yang sangat besat terhadap *overall performance* dari sebuah aplikasi cloud [13]. Level abstraksi tambahan yang digunakan ketika mengimplementasikan virtualisasi dapat mengurangi *workload performance*. Terlebih lagi VM terkenal mempunyai *overhead* yang besar sehingga sangat mempengaruhi performansinya. Hal ini menjadi pertimbangan yang cukup besar mengingat *cloud computing* membutuhkan kinerja yang baik. Dibandingkan dengan VM, container terkenal akan overheadnya yang kecil, sehingga memiliki performansi yang jauh lebih baik dari segi entitas virtualisasi [2]. Container merupakan sebuah lightweight OS yang dapat berjalan baik pada host system maupun server, yang menjalankan perintah native untuk Core CPU, mengeliminasi kebutuhan akan level emulasi atau hanya dalam kompilasi waktu. Container memungkinkan pengguna untuk menghemat konsumsi sumber daya tanpa overhead virtualisasi dan juga menyediakan isolasi [3]. Selain itu, VM akan melakukan boot up kernelnya sendiri, sedangkan container menggunakan kernel milik host [13][14] sehingga boot up time container cenderung jauh lebih cepat daripada VM [15].

Oleh karena itu, semakin ramai pengguna cloud computing beralih ke container daripada VM dalam implementasinya, menjadikan service cloud yang dijalankan lebih cepat diinisiasi dan memungkinkan untuk membangun suatu cloud application yang besar dan scalable. Dalam satu dekade terakhir ini, lightweight container-based virtualization telah menjadi alternatif untuk hypervisor-based virtualization pada cloud [16][17]. Bahkan, dalam contoh kasus pada data center di Google, sejumlah 2 milyar container dijalankan setiap minggunya, seperti yang diberitakan oleh Jack Clark, The Register UK pada 23 Mei 2016. Selain itu, perusahaan-perusahaan seperti Uber, ebey, dan Baidu pun mulai menggunakan Docker dalam menyediakan layanannya [17]. Hal ini dikarenakan kontainerisasi dinilai sebagai sebuah cara yang mudah dan efisien dalam berbagi sebuah bagian dari OS dan membagikannya ke dalam banyak aplikasi terisolasi. Dibandingkan dengan membangunkan sebuah VM yang memakan waktu dalam hitungan menit, membangun sebuah container dapat dilakukan hanya dalam hitungan detik, karena kita tidak perlu untuk menginisiasi sebuah OS.

Sejak kemunculan teknologi chroot pada 1979 [13] dan perkembangan namespace dan cgroups, para peneliti melakukan riset terkait perbandingan performansi antara hypervisor dan container. Dalam [2] [3] dan [5] secara garis besar membahas mengenai analisis komparasi antara container dan hypervisor untuk diaplikasikan dalam Cloud Computing, baik dalam IaaS maupun PaaS dan didapatkan hasil bahwa container memiliki perbedaan performansi yang sangat kecil dengan hypervisor. Selain itu, Morabito [5] juga menyarankan untuk membandingkan hasil tersebut dengan LXD. Gupta [20] membandingkan performansi Docker, VM, dan LXD yang berjalan di atas VMware ESX environment dan mendapati bahwa LXD bekerja sedikit lebih baik dari Docker dalam hal multiprocessor test. Rabindra K Barik [2] menganalisis performansi VM dalam segi Guest OS, efisiensi, keamanan, storage, isolation, networking, dan bootup time dan seperti penelitian lainnya, overhead virtualisasi pada container hampir dapat diabaikan namun memiliki overhead yang cukup signifikan pada hardware ultilization. Sampathkumar [19] melakukan beberapa benchmark untuk mengukur performansi CPU, memory, dan disk I/O pada infrastruktur LXC, Xen, dan KVM dan menyimpulkan bahwa LXC lebih disarankan untuk memvirtualisasikan infrastruktur yang secara desain bersifat dinamis dan tidak membutuhkan isolasi sumber daya yang tinggi untuk permintaan infrastrukturnya.

2. Dasar Teori

2.1. Server

Server dikenal sebagai sebuah hardware/komputer yang difungsikan untuk melayani, membatasi, dan mengontrol akses terhadap client pada jaringan komputer. Server bertugas untuk menyediakan resource untuk digunakan oleh komputer-komputer client.

2.1.1. FTP Server

File Transfer Protocol mulai diperkenalkan pada tahun 1971 sebagai sebuah protokol yang memungkinkan user bertukar data dengan network host. Walaupun FTP sudah banyak digantikan oleh protokol-protokol terbaru yang diklaim lebih secure seperti HTTP, HTTPS, SCP, dan lain sebagainya, per tahun 2016 ada sekitar 13,8 juta perangkat berbasiskan pengalamatan IPv4 yang difungsikan sebagai FTP Server dan 21,8 juta yang merespon pada port 21 atau port yang digunakan oleh FTP.[7]

Mekanisme kerja FTP adalah saat client mengirim permintaan untuk mengakses sesuatu dengan perintah "<command> [arguments] \r\n" ke server dan mengekstraksi return code 3 digit, server akan merespon apakah permintaan tersebut dikabulkan atau tidak, dengan memanfaatkan dua jenis koneksi. Pertama, untuk mengontrol messages atau pesan antara client-server, dan kedua untuk mengirimkan data yang diminta oleh client.

FTP Server terhubung dengan FTP client dan akan memberikan hak akses berupa mengunduh, upload, rename, delete, dan lain-lain terhadap suatu file yang dikehendaki oleh FTP client. Dalam penggunaannya, FTP menggunakan koneksi TCP dan mendengarkan port 20 dan 21.

2.1.2. Web Server

Web Server dapat ditujukan kepada komputer/perangkat keras dan software yang bekerja bersama dalam menyimpan, mengantarkan, dan mengontrol hal-hal yang dapat membuat sebuah web page tersedia di perangkat client. Pada sisi perangkat keras, sebuah web server adalah komputer atau perangkat keras yang menyimpan file-file komponen dari sebuah website, misalnya dokumen HTML, CSS stylesheets, dan file JavaScript, dan mengantarkannya ke end-user. Perangkat keras ini terhubung ke internet dan dapat diakses melalui sebuah domain name. [8]

2.1.2. Mail Server

Setiap tahunnya, ada sejumlah 12 triliyun email yang dikirim setiap harinya. Sejak ditemukan pada 1971, pemanfaatan email sebagai opsi utama dalam komunikasi elektronik terus meningkat, termasuk pertumbuhan mail server [12]. Mail server adalah sebuah server yang bertanggung jawab dalam menerima dan mengantarkan e-mail melalui jaringan atau internet. Sebuah mail server dapat menerima e-mail dari komputer client dan mengirimkannya ke mail server yang lain, dan juga mengantarkan email yang diterima ke komputer client.

2.2. Container

Dibandingkan dengan virtualisasi server secara tradisional, container merupakan proposisi yang menarik dalam hal application environment yang membutuhkan web-scale. Docker, LXC, dan LXD adalah tiga container yang berbasiskan Linux kernel pada awalnya. Saat ini Docker diusung untuk dapat berjalan dalam berbagai jenis OS seperti Windows dan Mac.

Konsep fundamental dari container adalah untuk membuat virtual instance, membagi sebuah OS bersamaan dengan library, driver, dan binary yang ada di dalamnya. Hal ini sangat berbeda dengan konsep virtualisasi tradisional berupa VM yang menginisiasi sebuah OS baru.

Container dimaksudkan untuk menyediakan isolasi dan resource management pada Linux environment [3]. Container menjanjikan level isolasi dan keamanan yang sama dan lebih terintegrasi dengan host OS.

Sebelum mengenal tentang container, kita perlu memahami linux cgroups dan dan namespace terlebih dahulu. Dua hal ini adalah fitur dari Linux kernel yang menciptakan tembok pembatas antar-container dan proses-proses lain yang berlangsung dalam host.

Linux cgroup pertamakali dikembangkan oleh Google. Linux cgroup adalah sebuah konsep dari linux kernel yang mengatur isolasi dan penggunaan sumber daya sistem seperti CPU, memory, untuk proses-proses tertentu. Hampir semua container menggunakan cgroup sebagai mekanisme pokok dalam manejemen sumber daya.

Namespace pertamakali dikembangkan oleh IBM untuk membungkus sebuah set dari sumber daya sistem dan memberikannya kepada sebuah proses agar tampak seperti dedicated. Singkatnya, namespace membuat sebuah penghalang antara container dengan berbagai level. Pid namespace memungkinkan setiap container memiliki process id nya masing-masing, net namespace memungkinkan setiap container untuk memiliki kelengkapan jaringannya tersendiri, seperti routing table, iptable, maupun loopback interface. IPC namespace menyediakan isolasi untuk berbagai mekanisme IPC yang dinamakan semaphore, message queues, dan shared memory segments. Mnt namespace menyediakan mountpoint bagi tiap container, sedangkan UTS namespace memastikan bahwa container yang berbeda dapat menampilkan dan mengganti hostname nya.

Gagasan awal untuk teknologi container pertamakali bermula dari dari chroot. Lalu, mulai berkembang FreeBSD Jails yang dikenal sebagai teknologi container yang paling pertama. Kemudian, berkembang Linux-VServer project yang memisahkan user space environment. Namun, Linux-VServer ini belum terdapat migration process dan clustering.

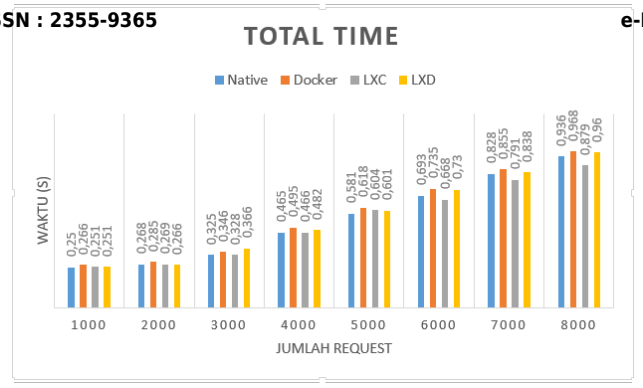
3. Pembahasan

Bab ini akan membahas mengenai hasil penelitian berupa data yang menunjukkan performansi Web Server, FTP Server, dan Mail Server ketika dijalankan di container yang berbeda. Setiap container diuji dengan menjalankan Web Server, FTP Server, dan Mail Server secara bergantian. Pada Web Server, pengujian yang dilakukan adalah dengan membangkitkan sejumlah request yang jumlahnya ditingkatkan secara bertahap untuk melihat waktu respon server terhadap request tersebut menggunakan benchmark ab (Apache Bench). Pada FTP Server, pengujian yang dilakukan adalah dengan menguji waktu yang dibutuhkan bagi server untuk menyelesaikan sebuah FTP Request, berikut data rate dan latencynya menggunakan jmeter dan wireshark. Sedangkan pada mail server, server akan diuji dengan membangkitkan 10 concurrent thread untuk memaksa ultilisasi server hingga mendekati 100% dan akan dilihat seberapa banyak jumlah pesan yang dapat dikirimkan melalui protocol SMTP.

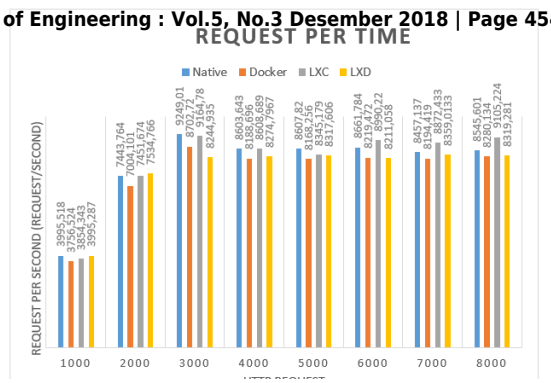
Lalu, pada masing-masing container dilakukan uji untuk melihat overall performance matric (IOzone read, IOzone write, ramspeed, dan CPU performance) menggunakan aplikasi Phoronix Test Suite dan sysbench. Pengujian ini dilakukan dengan menambah beban dari container yang digunakan sehingga terjadi stress dan memakan sumberdaya perangkat semaksimal mungkin (hingga 99%). Kemudian, pengujian yang sama dilakukan di atas host machine tanpa layer virtualisasi, atau native. Untuk meminimalisir anomali yang dihasilkan dari perangkat keras yang digunakan, maka setiap pengujian diambil sebanyak 30 kali dan hasil yang digunakan adalah rata-rata dari 30 data tersebut.

3.1. Pengujian Web Server

Hasil pengujian dari apache benchmark ini disajikan dalam satuan second (s). Pengukuran ini dilakukan untuk mengetahui kecepatan server dalam melayani HTTP Request dengan jumlah request yang sudah ditentukan. Semakin kecil nilainya, makan akan semakin baik kinerja web server dalam melayani HTTP Request.



Gambar 3. 1 Hasil pengujian total time web server



Gambar 3. 2 Hasil pengujian request per time web server

Dari hasil pengujian, didapatkan bahwa docker memiliki kinerja yang cenderung kurang baik dibandingkan dengan platform container lain yang diujikan pada penelitian ini. Dibandingkan dengan native, penurunan performansi total time pada docker berkisar 3,37% hingga 6,38%. Sedangkan pada LXC penurunan performansinya adalah berkisar mulai 0,06% hingga 3,9%, dan 0,16% hingga 12,6% pada platform LXD.

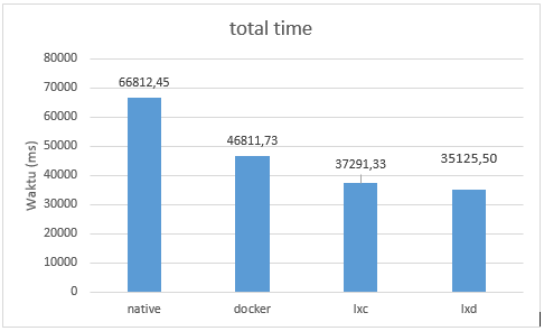
Request per time menunjukkan jumlah request yang dapat ditangani selama satu detik pengujian yang bergantung pada beban HTTP request yang diminta. Pada pengujian ini, semakin besar nilainya menunjukkan bahwa semakin banyak HTTP request yang dapat ditangani web server yang bersangkutan, maka semakin baik nilainya.

Berdasarkan hasil pengujian seperti yang terlihat pada gambar 4.2, Docker mempunyai perbedaan performansi berkisar di antara 3,1% hingga 5,9% dibandingkan native. Sementara itu, penurunan performansi 0,91% hingga 3,5% pada LXC dan 0,005% hingga 10,8% pada LXD.

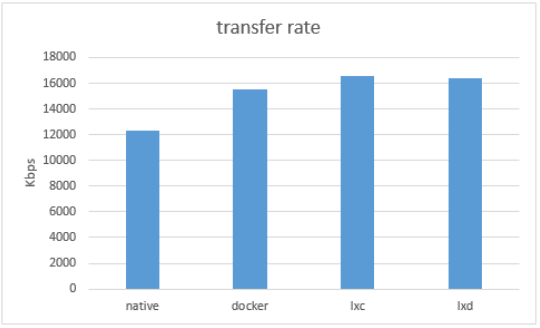
Secara umum, dapat dilihat bahwa performansi docker cenderung kurang dibandingkan dengan LXC dan LXD. Hal ini disebabkan karena Docker cenderung mengalami bottleneck pada beberapa pengujian karena interface network yang digunakan [17].

3.2. Pengujian FTP Server

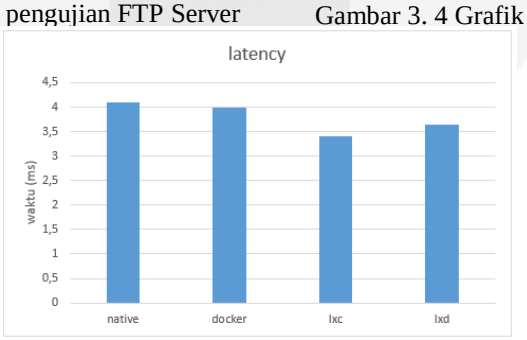
Pengujian FTP Server ini diamati dari performansinya dalam melayani FTP Request dengan parameter total time (dalam s), transfer rate (dalam byte per second), dan latency (dalam ms). Ukuran file yang digunakan dalam pengujian ini adalah 500 MB.



Gambar 3. 3 Grafik total time pengujian FTP Server



Gambar 3. 4 Grafik transfer rate pengujian FTP Server



Gambar 3. 5 Grafik latency pengujian FTP Server

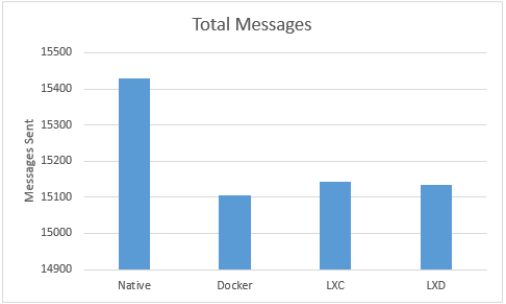
Gambar 3.3 menunjukkan hasil total waktu yang diperlukan FTP server untuk melayani FTP Request yang diajukan oleh host. Nilai dari total time ini semakin baik jika waktunya semakin kecil atau semakin cepat. Dibandingkan dengan performa FTP server yang dijalankan di atas server native, platform container membutuhkan waktu lebih cepat. Docker menunjukkan 29,9% lebih cepat dibanding native, lxc dengan 44,18% dan lxd dengan 47,42%.

Gambar 3.4 menunjukkan transfer rate ftp server yang bersesuaian. Hal ini ikut menjelaskan perolehan total time FTP server yang dijalankan pada masing-masing container. Docker memiliki gap time yang cenderung lebih besar daripada LXC dan LXD, dikarenakan terdapat bottleneck pada interface yang digunakan. Dari gambar 4.4 dapat diketahui bahwa performa paling baik adalah oleh LXC, yaitu 34,04% lebih baik dibanding native. Docker dan LXD memiliki performansi 25,8% dan 33,05% lebih baik dibandingkan dengan native.

Gambar 4.5 menunjukkan jeda waktu yang dibutuhkan dalam pengantaran paket FTP. Berdasarkan gambar 4.5 di atas, latency time yang diperoleh dalam pengujian ini tergolong ke dalam low latency karena bernilai di bawah 1000 ms.

3.3. Pengujian Mail Server

Pengujian performansi mail server ini dilakukan dengan menggunakan tool bernama postal. Mail server diatur sedemikian rupa agar dibebankan dengan 10 concurrent thread yang masing-masing akan mengirimkan sejumlah pesan sampai dengan 500 pesan, yang masing-masing berukuran 25 MB dengan menggunakan protokol SMTP dalam waktu satu menit. Adapun hasil pengujian ini disajikan dalam gambar berikut:



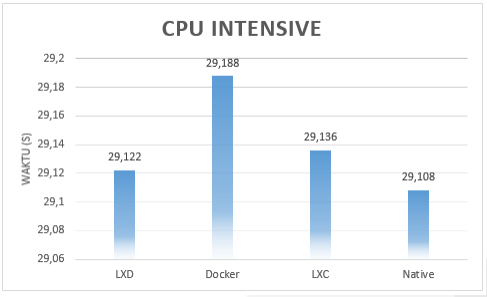
Gambar 3. 6 Hasil pengujian mail server

Berdasarkan gambar 3.6, semakin besar nilai yang diperoleh maka semakin baik performansinya. Perbedaan performansi atau overhead yang diperoleh dari pengujian ini cukup kecil, yaitu sekitar 1,8% hingga 2,09% lebih kecil dibandingkan native. Bottleneck pada docker masih dapat dijumpai dalam pengujian ini, ditunjukkan dengan overheadnya yang mencapai 2,09%, sedikit lebih besar dibandingkan LXC dengan overhead 1,85% dan LXD dengan overhead 1,91%.

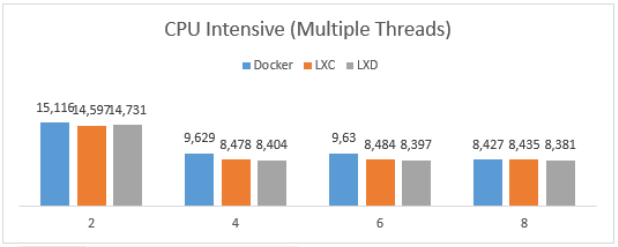
3.4. Pengujian *Overall Performance System*

3.4.1. CPU Performance

Skema pengujian ini dilakukan dengan memberikan beban kepada CPU berupa sejumlah kalkulasi yang didefinisikan dengan maximum prime number. Hasil dari pengujian ini menunjukkan berapa waktu yang dibutuhkan bagi CPU untuk melakukan kalkulasi tersebut. Dalam pengujian ini, semakin kecil nilai output yang dihasilkan, maka semakin baik performansinya.



Gambar 3. 4 Grafik hasil performansi CPU Single Thread

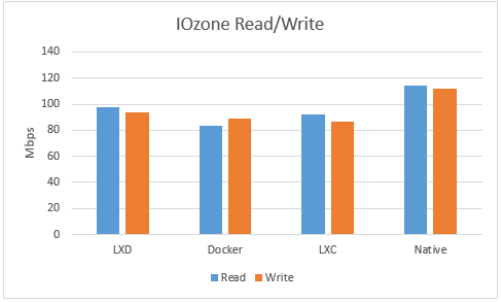


Gambar 3.8 Grafik hasil performansi CPU Multiple Thread

Hasil performansi di atas merupakan perbandingan antara native dan sistem kontainerisasi Docker, LXC, dan LXD dari segi Single Thread CPU Performance. Selisih performansi antara container dan native adalah 0,047% untuk LXD, 0,094% untuk LXC, dan 0,27% untuk Docker. Untuk mengetahui performansi CPU dalam melakukan multiple thread, maka dilakukan pengujian dengan menambahkan jumlah thread menjadi 2, 4, 6, dan 8.

Berdasarkan tabel 4.2, nampak bahwa kinerja CPU dari masing-masing platform container mengalami saturasi pada thread ke-6 dan ke-8. Hal ini dapat menjelaskan fungsi container yang difungsikan untuk melakukan seminimal mungkin aktivitas dalam satu waktu, contohnya seperti microservice.

3.4.2. IOzone Read/Write



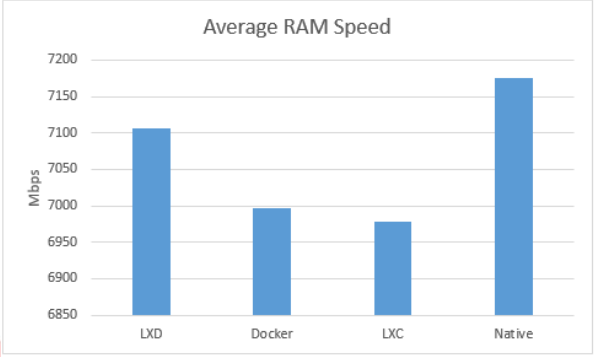
Gambar 3.8 Grafik hasil performansi CPU Multiple Thread

Dalam pengujian ini IOzone Read, semakin besar nilai yang didapat maka semakin baik performa sistem container dalam melakukan proses baca harddisk. Berdasarkan gambar 4.9, performa native mengalami penurunan rata-rata sebesar 20% yang disebabkan oleh overhead virtualisasi dari container itu sendiri dan tidak adanya optimasi terhadap kinerja filesystem container. Perbandingan persentase IOzone read masing-masing adalah 85,41%

atau 97,93 Mbps untuk LXD, 82,22% atau 83,088 Mbps untuk Docker dan 80,52% atau 92,32 Mbps untuk LXC.

Sama halnya dengan pengujian IOzone Read, hasil pengujian ini menunjukkan performa yang semakin baik jika hasilnya semakin besar. Dari gambar 4.10, nampak bahwa sistem yang diuji di atas platform mengalami penurunan performansi dari performa nativenya rata-rata 20%. Adapun persentase performa yang dicapai oleh masing-masing container dibandingkan dengan nativenya adalah 83,54% atau 93,61 Mbps untuk LXD, 79,38% atau 88,95 Mbps untuk Docker, dan 77% atau 86,27 Mbps untuk LXC.

3.4.3. RAM Speed



Gambar 3.9 Grafik hasil pengujian RAM Speed

Berdasarkan gambar 3.9, LXD memiliki performa paling baik dibandingkan dengan container-container lainnya yang diujikan dalam tugas akhir ini dengan persentase 99,03% atau sebesar 7106,172 Mbps. Docker dan LXC memiliki selisih performa yang sangat tipis, yaitu 97,52% untuk Docker (6997,45 Mbps) dan 97,25% atau 6978,58 Mbps untuk LXC.

Tabel 3.1 Hasil pengujian metrik overall performance

Parameter	Sistem			
	Native	Docker	LXC	LXD
IOzone Read	1	0,74	0,81	0,85
IOzone Write	1	0,82	0,77	0,83
RAM Speed	1	0,98	0,97	0,99
CPU Intensive	1	0,73	0,91	0,95
Summary	4,00	3,27	3,46	3,62
Overall Performance	100%	81,7%	86,5%	90,5%

Berdasarkan table 3.1, LXD unggul dalam nilai keseluruhan untuk metric overall performance dengan performansi 90,5%. LXD merupakan versi peningkatan dari container LXC dan terbukti secara sistem LXD memang memiliki performa lebih unggul dari LXC walaupun hanya 4,62%.

4. Kesimpulan

Dari hasil pengujian dan analisis performansi FTP Server, Web Server, dan Mail Server pada container Docker, LXC, dan LXD didapatkan bahwa server-server tersebut dapat dijalankan pada ketiga container tersebut namun dengan perbedaan performansi. Berdasarkan hasil analisis performansi sistem dan performansi dari masing-masing server yang bersesuaian, didapatkan hasil berikut ini:

1. Terdapat perbedaan performansi sistem antara container dengan native dikarenakan adanya overhead virtualisasi. Hasil overall performance dalam penelitian ini menunjukkan LXD memiliki nilai yang paling unggul dari Docker dan LXD dengan persentase 90,5%. Hasil ini menunjukkan bahwa LXD sebagai peningkatan dari LXC mampu meningkatkan performansi sistem linux container secara keseluruhan.
2. Jika dibandingkan dengan performansi sistem native, semua parameter yang diujikan dalam metrik overall performance menunjukkan bahwa native masih unggul, walaupun hanya terpaut sedikit saja. Contohnya pada parameter CPU Intensive, perbedaan performansi antara native dan container hanya terpaut 0,137%. Pada parameter lainnya, seperti IOzone Read, IOzone Write, dan RAM Speed test, didapati perbedaan performansi berkisar mulai dari 2,067% (pada RAM Speed) dan 20% (pada test IOzone). Perbedaan performansi ini dikarenakan adanya isolasi dari cgroups agar proses yang terjadi di dalam container tidak bercampur dengan host systemnya.
3. Pada pengujian performansi FTP Server, LXD menunjukkan nilai yang lebih unggul dibandingkan dengan Docker dan LXC. Dari segi parameter total time, LXD unggul dengan nilai 35,125 s. Namun pada parameter transfer rate, LXC tercatat 0,98% lebih baik dibandingkan dengan LXD, dan juga memiliki latency 5,7% lebih kecil dari LXD.
4. Pada pengujian mail server dengan parameter total pesan yang dapat dikirim oleh server yang bersesuaian melalui protokol SMTP, LXC menunjukkan nilai paling unggul dengan overhead hanya 1,85% dibandingkan dengan native. Perbedaan performansi di antara ketiga container cukup kecil yaitu di bawah 0,05%. Hal ini menunjukkan bahwa standalone mail server dapat berjalan dengan performansi hampir sama

dengan native pada platform container Docker, LXC, dan LXD.

5. Pada pengujian web server, pada web server yang tergolong hanya menangani request dengan jumlah sedikit (hingga 2000 concurrent level), LXD memiliki performa yang lebih baik dibanding Docker dan LXC dari segi jumlah request yang dapat ditangani perdetiknya. Bahkan performansinya hanya terpaut di bawah 1% dibandingkan dengan native. Untuk tingkat concurrent level yang lebih tinggi, LXC menunjukkan performansi yang lebih baik dibandingkan dengan Docker dan LXD, yang berkisar 0,9% hingga 3.9% dibandingkan dengan native. Web server yang dijalankan di atas ketiga platform container ini dapat diimplementasikan untuk menunjang bisnis Ecommerce Cloud seperti website perbankan, karena response timenya telah memenuhi batas minimum Service Level Agreement, yaitu di bawah 3s [22].

Daftar Pustaka:

[1] Golden, Bernard, *Virtualization for Dummies 3rd HP Special Edition:* Wiley, 2011

[2] Rabindra K Barik, Rakesh K. Lenka, K. Rahul Rao, Devam Ghose, "*'Performance Analysis of Virtual Machine and Containers in Cloud Computing'*," International Conference on Computing, Communication and Automation, pp. 1204-1210, 2016

[3] Rajdeep Dua, A. Reddy Raja, Dharmesh Kakadia, "'*Virtualization vs Containerization to support PaaS*'," IEEE International Conference on Cloud Engineering, pp. 610-614, 2014

[4] SDxCentral, "*ContainerCon North America 2016", https://www.sdxcentral.com/event/containercon-north-america-2016/*, 2016

[5] Roberto Morabito, Jimmy Kjallman, Miika Komu, "*Hypervisor vs Lightweight Virtualization: a Performance Comparison',"* IEEE International Conference on Cloud Engineering, pp. 386-393, 2015

[6] Aswarizha, Revan Faredha, "Analisis Performansi Network Function Virtualization pada Hypervisor XEN, KVM, dan VMWare ESXi", Open Library Telkom University, 2016.

[7] Drew Springall, Zakir Durumeric, J. Alex Alderman, *"FTP: The Forgotten Cloud,"* IEEE International Conference on Dependable System and Networks, pp. 503-513, 2016

[8] Mozilla Developer Network, "What is Web Server?", https://developer.mozilla.org/en-US/docs/Learn/Common_question/What_is_a_web_server/, 2017

[9] Nikhil Tripathi, Neminath Hubballi, Yogera Singh, "*How Secure are Web Servers?*", IEEE International Conference on Availability, Realibility, and Security, pp. 454-463, 2016

[10**]** SDxCentral, "Chapter 1- Containers War: LXC vs Docker", https://www.sdxcentral.com/reports/linux-container-ecosystem/chapter-1-container-wars-lxcs-vs-docker/, 2017

[11] Pivotal, "Moments in Container History", https://content.pivotal.io/infographics/moments-in-container-history, 2017

[12] ServerWatch, "You've got Mail – Messaging Servers Trends and Must-haves", http://www.serverwatch.com/tutorials/article.php/3461461/Youve-Got-Mail--Messaging-Server-Trends-and-MustHaves.htm, 2017

[13] Kovacs, Akos, "*Comparison of Different Linux Container",* IEEE 40th International Conference on Telecommunication and Signal Processing, pp. 47-51, 2017

[14] Amr A. Mohallel, Julian M. Bass, Ali Dehghantaha, "Experimenting with Docker: Linux container and base OS attack surfaces", 2016 International Conference on Information Society (i-Society), pp. 17-21, 2016

[15] Miguel G. Xavier, Marcelo V. Neves, Fabio D. Rossi, Tiago C. Ferreto, "*Performance Evaluation of Container-Based Virtualization for High Performance Computing Environments*", 2013 21st Euromicro International Conference on Parallel. Distributed, and Network-Based Processing, pp. 233-240. 2013

[16] S. Soltesz, H. Potzl, M. E. Fiuczynski, A. Bavier, and L. Peterson, "*Container-based Operating System Virtualization: A Scalable, High Performance Alternative to Hypervisor"*, Eurosys '07, pp. 275-287, 2007

[17] Zheng Li, Maria Kihl, Qinghua Lu, and Jens A. Anderson, "*Performance Overhead Comparison between Hypervisor and Container based Virtualization"*, IEEE 31st International Conference on Advanced Information Networking and Application, pp. 955-962, 2017

[18] R. Rosen. "*Resource Management: Linux kernel, Namespaces, and cgroups"*, Haifux, 2013

[19] A. Sampathkumar, "*Virtualizing Intelligent riverR: A Comparative Study of Alternative Virtualization Technologies",* Master's thesis Clemenson University, 2013

[20] Sapan Gupta, Deepanshu Gera, "A Comparison of LXD, Docker, and Virtual Machine", International Journal of Scientific & Engineering Research, pp. 1414-1417, 2016

[21] Joris Claassen, Ralph Koning, Paola Grosso, "Linux containers networking: performance and scalability of kernel modules", IEEE/IFIP Network Operations and Management Sysposium, pp 713-717, 2016

[22] Busalim, Abdelsalam,"Service Level Agreement Parameters for Ecommerce Cloud", International Journal od Scientific Knowledge Computing and Information Technology, pp 8-20, 2014