

ANALISIS PERBANDINGAN KINERJA LAYANAN CONTAINER AS A SERVICE (CAAS) Studi Kasus : Docker dan Podman *by Chairul Mukmin*

Submission date: 08-Mar-2021 07:23AM (UTC-0800)

Submission ID: 1527449894

File name: Jurnal_KLIK_Chairul_Mukmin.docx (296.97K)

Word count: 3029

Character count: 18065

ANALISIS PERBANDINGAN KINERJA LAYANAN CONTAINER AS A SERVICE (CAAS) Studi Kasus : Docker dan Podman

Chairul Mukmin¹, Therezia Naraloka², Qodri Harun Andriyanto³

¹Fakultas Ilmu Komputer, Universitas Bina Darma

²Fakultas Ilmu Komputer, Universitas Sjakhyakirti

Jl. Jend. A Yani No.3, 0711-515582

¹chairul.mukmin@binadarma.ac.id, ²therezia.naraloka@unisti.ac.id,
qodriharun@gmail.com³

Abstract

The development of cloud computing is growing and has become a trend that is widely used in today's telecommunications world, one of the models is Container As A Service or often abbreviated as CAAS. Container As A Service can be used to run certain services or applications that are implemented on the server. This technique aims to reduce resource usage and facilitate server management. To run Container As A Service, it is necessary to have a platform, and in this research the author uses the Docker and Podman platforms. One of the factors that affect the performance of this Container is in terms of resource usage and security. There are many types of attacks against computer systems such as Denial of service attacks, these types of attacks continue to develop in various forms. This research will measure the performance of a Container service in terms of CPU, RAM and Response Time performance of the web server service during normal conditions, when running many applications, during Denial of Service attacks and when implementing Denial of service attack prevention. .

Keywords: Container As A Service, Denial of Service , Docker, Podman.

Abstrak

9
Perkembangan komputasi awan semakin berkembang dan menjadi suatu trend yang banyak digunakan di dunia telekomunikasi saat ini, salah satu modelnya yaitu Container As A Service atau yang sering disingkat CAAS. Container As A Service ini dapat digunakan untuk menjalankan layanan atau aplikasi tertentu yang di implementasikan ke dalam server. Teknik ini bertujuan untuk mengurangi penggunaan sumber daya dan memudahkan dalam manajemen server. Untuk menjalankan Container As A Service ini perlu adanya sebuah platform dan pada penelitian kali ini peneliti menggunakan platform Docker dan Podman. Salah satu faktor yang mempengaruhi kinerja dari Container yaitu dari sisi penggunaan sumber daya dan ³keamanannya. Terdapat banyak jenis serangan terhadap sistem komputer seperti serangan Denial of service, jenis serangan ini senantiasa berkembang dalam berbagai bentuk. Pada penelitian ini akan dilakukan pengukuran kinerja sebuah layanan Container dari sisi kinerja CPU, RAM dan Response Time dari layanan web server pada saat kondisi normal, pada saat menjalankan banyak aplikasi, pada saat mendapatkan serangan Denial of Service dan pada saat menerapkan pencegahan serangan Denial of service.

Kata-kata kunci: Container As A Service, Denial of Service, Docker, Podman

1. PENDAHULUAN

Perkembangan komputasi awan saat ini semakin berkembang dan menjadi suatu trend yang banyak digunakan di dunia telekomunikasi saat ini, salah satunya Container as a Service (CaaS). CaaS dapat digunakan untuk menjalankan sebuah layanan atau aplikasi tertentu yang di implementasikan ke dalam server [1]. Container merupakan bagian yang sangat penting dalam penerapan komputasi awan, karena sangat berpengaruh pada efisiensi pengelolaan sumber daya infrastruktur komputasi awan. Teknik containerisasi hadir sebagai solusi dan menjadi trend saat ini [2]. Dalam prakteknya Container as a Service perlu adanya sebuah Platform yang mendukung Container as a Service seperti Docker dan Podman.

Penelitian ini mengukur kinerja platform Docker dan Podman, karena kedua platform ini merupakan penerapan teknologi container terpopuler. Kedua platform ini merupakan perangkat lunak virtualisasi jenis sistem operasi level virtualisasi berbasis Linux Container (LXC) yang memungkinkan pengembang aplikasi membuat, menguji dan menjalankan aplikasi dalam sebuah lingkungan yang berbeda, terisolasi dan fleksibel. Dalam prakteknya, komputasi awan masih memerlukan penelitian maupun percobaan lebih lanjut karena beberapa teknologinya masih tergolong baru.

Aspek penelitian yang banyak dikaji adalah mengenai kinerja layanan di atas Container as a Service, salah satu faktor yang mempengaruhi performansi sebuah teknik virtualisasi yaitu dari sisi keamanannya [3]. Di antara berbagai serangan, Denial of Service (DoS) merupakan serangan flood TCP SYN yang paling umum dan dikenal sebagai salah satu metode DoS yang paling kuat untuk mempengaruhi kinerja dari target. DoS adalah salah satu contoh jenis serangan yang dapat mengganggu infrastruktur dari jaringan komputer, serangan jenis ini memiliki suatu pola khas, dimana dalam setiap serangannya akan mengirimkan sejumlah paket data secara terus-menerus kepada target serangannya [4]. Hal ini menarik, Peneliti melakukan penelitian secara lebih mendalam bagaimana kinerja yang dihasilkan oleh platform Container tersebut seperti CPU, RAM, dan Response Time layanan webserver dari container pada saat container berjalan normal, menjalankan beberapa aplikasi, pada saat mendapatkan serangan, dan pada saat pencegahan serangan DoS dijalankan. Parameter kinerja yang dihasilkan akan dibandingkan antara Docker dan Podman.

2. METODOLOGI PENELITIAN

2.1 Metode PPDIIO

Metode penelitian menggunakan PPDIIO yang dikembangkan oleh Cisco dalam perancangan sistem jaringan. Tahapan dalam metode PPDIIO meliputi Prepare, Plan, Design, Implement, Operate dan Optimize [5].

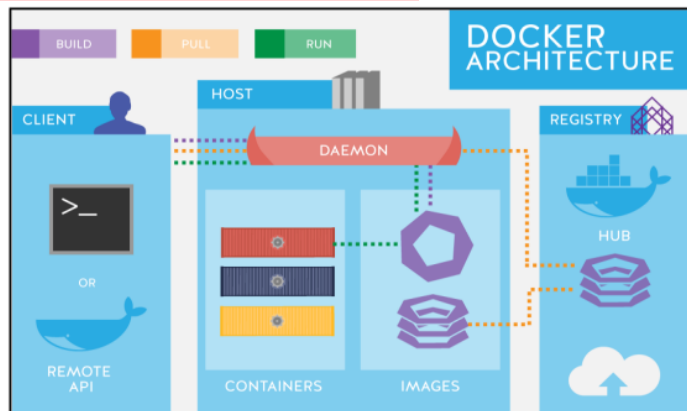
2.2 Skenario Pengujian

1. Pengujian 1 merupakan pengujian kinerja layanan *container* ketika normal tanpa menjalankan aplikasi dan tanpa adanya serangan DoS
2. Pengujian 2 merupakan pengujian kinerja layanan *container* ketika menjalankan banyak aplikasi

- 3 Pengujian 3 merupakan pengujian kinerja ketika layanan *container* mendapatkan serangan DoS selama 15 menit
- 4 Pengujian 4 merupakan pengujian kinerja ketika layanan *container* mendapatkan serangan DoS selama 30 menit
- 5 Pengujian 5 merupakan melakukan serangan DoS Flood terhadap Web Server
- 6 Pengujian 6 merupakan pengujian kinerja ketika layanan *container* mendapatkan serangan DoS dan adanya pencegahan.

2.3 Dasar Teori

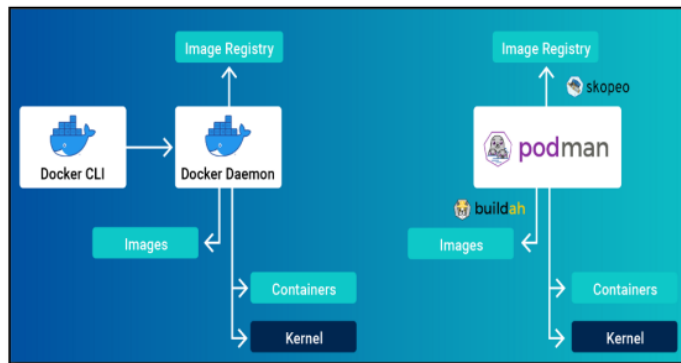
Docker adalah suatu platform terbuka bagi pengembang perangkat lunak dan pengelola sistem jaringan untuk membangun, mengirimkan dan menjalankan aplikasi-aplikasi terdistribusi [5]. Docker dengan teknik container banyak dilirik para pengembang aplikasi karena dianggap sebagai solusi trend saat ini [6]. Docker merupakan open source software di bawah Lisensi Apache Versi 2.0 yang bisa dipergunakan secara gratis. Docker menggunakan arsitektur client server. Docker client menghubungi Docker daemon, yang melakukan pekerjaan berat, menjalankan, dan mendistribusikan docker container anda. Kedua Docker client dan daemon dapat berjalan pada sistem yang sama. Docker client dan daemon berkomunikasi via sockets atau lewat API yang disediakan Docker.



Gambar 2.1 Docker Arsitektur [7].

Docker memberikan berbagai macam keuntungan dan kemudahan dalam proses deployment software. Platform ini dapat digunakan untuk membangun, mempersiapkan, dan menjalankan aplikasi. Aplikasi dapat di bungkus dalam container, dan aplikasi dapat berjalan pada lingkungan apapun dimana saja. Kelebihan docker pada layanan cloud ini dapat membantu permasalahan pada aplikasi yang menggunakan multi container dan layanan pada server cluster shared.

Podman merupakan platform container open-source yang digunakan untuk menjalankan maupun manajemen ekosistem container seperti pods, containers, container images, dan container volumes menggunakan libpod library. Dengan Podman, sebuah layanan dapat berjalan sebagai proses terisolasi, independen terhadap lingkungan sekitarnya [8].



Gambar 2.2 Docker vs Podman [9].

Podman juga memiliki Command Line Interface (CLI) yang kompatibel dengan docker serta pengintegrasian yang lebih baik dengan sistem. Podman memudahkan para developer untuk mencari, menjalankan, dan membagikan container.

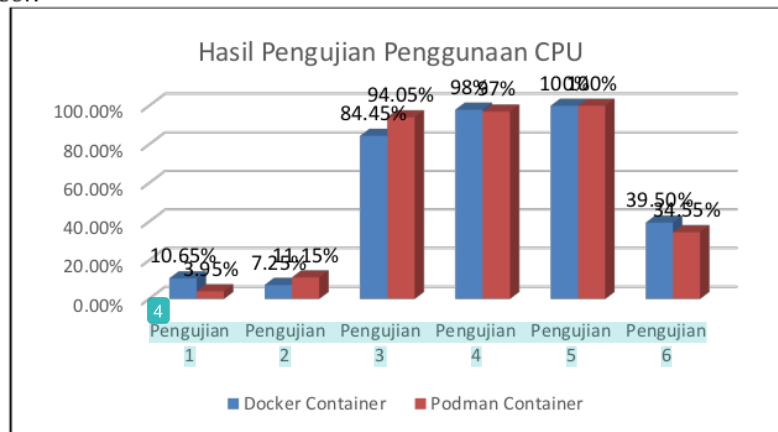
3. HASIL DAN PEMBAHASAN

3.1 Hasil Penelitian

Setelah melakukan pengujian sesuai dengan tahapan metodologi penelitian maka dapat dijelaskan hasil dan pembahasan sebagai berikut :

3.1.1 Penggunaan CPU

Pengukuran pada parameter CPU bertujuan untuk mengetahui berapa beban kerja pada suatu sistem yang dapat dilakukan oleh sebuah komputer. Hasil pengujian yang lebih rendah menunjukkan performa yang lebih baik, karena hasil pengujian ini dilakukan dengan banyaknya operasi dan intruksi yang dilakukan processor.



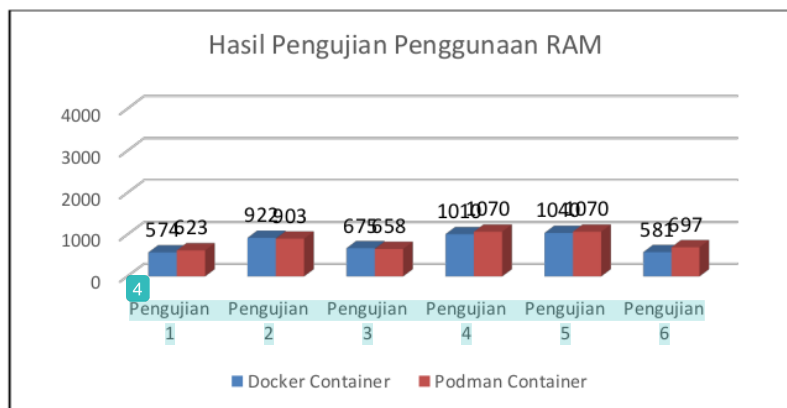
Gambar 3.1 Hasil Pengujian Penggunaan CPU

Berdasarkan hasil pengamatan pada parameter CPU yang dapat dilihat pada Gambar 3.1 didapatkan bahwa penggunaan processor container docker pada pengujian 1 sebesar 10.65% dan 3.95% untuk container podman, pada pengujian 2 sebesar 7.25% untuk container docker dan 11.15% untuk container podman, pada

pengujian 3 sebesar 84.45% untuk *container* docker dan 94.05% untuk *container* podman, pada pengujian 4 sebesar 98% untuk *container* docker dan 97% untuk *container* podman, pada pengujian 5 memiliki nilai yang sama yaitu sebesar 100%, pada pengujian ke 6 sebesar 39.50% untuk *container* docker dan 34.55 untuk *container* podman. Dari 6 pengujian maka hasil rata-rata penggunaan CPU pada *container* docker sebesar 56.65% sedangkan pada *container* podman sebesar 56,75%.

3.1.2 Penggunaan RAM

Pada parameter *Random Access Memory (RAM)* yaitu waktu *container* berjalan normal, *container* menjalankan banyak aplikasi, *container* mendapat serangan D₁₀ dan *container* mendapat serangan DoS setelah adanya pencegahan. Untuk nilai yang lebih kecil menunjukkan performa yang lebih baik.

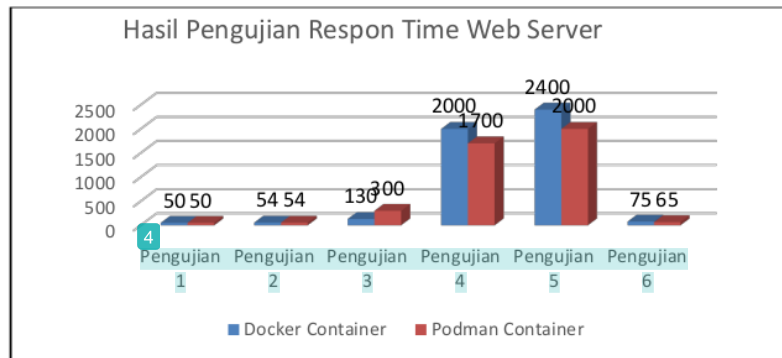


Gambar 3.2 Hasil Pengujian Penggunaan RAM

Gambar 3.2 Penggunaan RAM *container* docker pada pengujian 1 sebesar 574 dan *container* podman sebesar 623, pada pengujian 2 untuk *container* docker sebesar 922 dan *container* podman sebesar 903, pada pengujian 3 *container* docker sebesar 675 dan *container* podman sebesar 658, pada pengujian 4 *container* docker sebesar 1010 dan *container* podman sebesar 1070, pada pengujian 5 *container* docker sebesar 1040 dan *container* podman sebesar 1070, pada pengujian 6 *container* docker sebesar 581 dan *container* podman sebesar 697. Dari 6 pengujian maka hasil rata-rata penggunaan RAM pada *container* docker sebesar 800.4 sedangkan pada *container* podman sebesar 836.9.

3.1.3. Respon Time Web Server

Parameter *Respon Time* dari web server yang dijalankan pada *container*. Nilai pengukuran yang lebih kecil menunjukkan performa yang lebih baik.



Gambar 3.3 Hasil Pengujian Respon Time Web Server

Gambar 3.3 Nilai Response Time dari web server pada pengujian 1 container docker menghasilkan nilai respon time yang sama yaitu sebesar 50 milidetik dan container podman sebesar 50 milidetik, pada pengujian 2 masih sama yaitu container docker sebesar 54 milidetik dan podman *container* sebesar 54 milidetik, pada pengujian 3 container docker sebesar 130 milidetik dan container podman sebesar 300 milidetik, pada pengujian 4 container docker sebesar 2000 milidetik dan container podman sebesar 1700 milidetik, pada pengujian 5 container docker sebesar 2400 milidetik dan container podman 2000 milidetik, pada pengujian ke 6 container docker 75 milidetik dan container podman 65 milidetik. Dari 6 pengujian maka hasil rata-rata respon time pada container docker sebesar 776.5 milidetik sedangkan pada container podman sebesar 694.9 milidetik.

3.2. Pembahasan

3.2.1. Kinerja Dari Layanan *Container as a Service* Pada Saat Kondisi Normal Dan Pada Saat Menjalankan Banyak Aplikasi

Kinerja layanan *Container as a Service* pada saat berjalan normal menurut hasil pengukuran tidak terlalu memakan sumber daya yang besar, pernyataan ini berdasarkan penggunaan CPU dan RAM yang dijelaskan pada gambar 3.1 dan gambar 3.2 yaitu dengan data *container* docker hanya sebesar 10.65% untuk penggunaan CPU dan untuk penggunaan RAM sebesar 574 Mb, sedangkan untuk *container* podman sebesar 3,95% untuk penggunaan CPU dan untuk penggunaan RAM sebesar 623 Mb.

Kinerja layanan *container* pada saat menjalankan beberapa aplikasi sebanyak enam aplikasi juga tergolong tidak terlalu memakan sumber daya yang besar juga ¹⁴ pernyataan ini berdasarkan pada hasil pengukuran yang sudah dijelaskan pada gambar 3.1 dan gambar 3.2 yaitu *container* docker 7.25% untuk penggunaan CPU dan RAM sebesar 922 Mb untuk penggunaan RAM, sedangkan untuk *container* podman sebesar 11.15% untuk penggunaan CPU dan untuk penggunaan RAM sebesar 903 Mb.

Dari kinerja layanan diambil dari Response Time Web Server yang dijalankan pada *container*. Hasil penelitian Response Time dari web server dijelaskan pada gambar 3.3. Hasil pengukuran menunjukkan response time yang cepat untuk kedua platform *container*, mempunyai response time yang hampir sama dan tidak menunjukkan perbedaan yang signifikan, ketika *container* berjalan normal response

time selama 50 milidetik dan ketika *container* menjalankan beberapa aplikasi mempunyai response time selama 54 milidetik.

3.2.2. Dampak Serangan DoS SYS Flood Terhadap Layanan *Container as a Service*

Pengujian dengan melakukan serangan *Denial of Service* terhadap aplikasi layanan *container*, didapatkan hasil bahwa *container* memakan sumber daya yang besar dari segi penggunaan CPU, hasil ini dijelaskan pada gambar 3.1 dan 3.2 yaitu untuk layanan *container* docker sebesar 84.45% dari penggunaan CPU dan untuk penggunaan RAM sebesar 675 Mb, sedangkan untuk layanan podman sebesar 94.05 % dari penggunaan CPU dan untuk penggunaan RAM sebesar 658 Mb.

Dari sisi kinerja layanan dapat diambil data yang telah didapatkan setelah dilakukannya penelitan dari Response Time dari web server yang dijalankan pada *container* saat mendapatkan serangan *Denial of Service (DoS)*, dan hasil penelitian untuk Response Time dari web server dijelaskan pada gambar 3.3 dan dapat disimpulkan bahwa layanan kedua platform *container* mengalami penurunan kinerja dari Response Time web server yang sedang berjalan, untuk *container* docker memberikan Response Time selama 130 milidetik, sedangkan untuk *container* podman kinerja dari layanannya mengalami penurunan yang sangat jauh ketika mendapatkan serangan *Denial of Service (DoS)* itu dilihat pada Response Time yang diberikan dari web server yaitu selama 300 milidetik.

3.2.3. Pencegahan Penurunan Kinerja Dari *Container as a Service* Ketika Terjadi Serangan *Denial of Service*

Setelah mengetahui dampak dari serangan *Denial of Service (DoS)* untuk kinerja layanan *container*, dilakukan pencegahan penurunan kinerja layanan *container* dari serangan *Denial of Service (DoS)* dengan menggunakan *rules iptables* yang diterapkan pada server host. Hasil penelitian dapat diketahui kinerja dari layanan *Container as a Service* pada saat berjalan normal tidak memakan sumber daya yang besar, didapatkan hasil bahwa sumber daya yang digunakan oleh *container* mengalami penurunan, dari *container* docker CPU yang digunakan sebesar 39.50% dan 581 Mb untuk penggunaan RAM, sedangkan *container* podman penggunaan CPU sebesar 34.55% dan 697 Mb untuk penggunaan RAM.

Dari segi kinerja layanan *container* setelah dilakukan pencegahan terhadap serangan *Denial of Service (DoS)* didapatkan hasil bahwa pencegahan tersebut berhasil mencegah terjadinya penurunan kinerja layanan dari Response Time web server pada *container*. Untuk *container* docker memberikan Response Time selama 75 milidetik, sedangkan *container* podman memberikan Response Time selama 65 milidetik. Dari data tersebut penelitian ini berhasil mencegah penurunan kinerja layanan dari kedua platform *container* selama 50% dari waktu *container* mendapat serangan *Denial of Service (DoS)* tanpa ada pencegahan.

3.2.4. Perbandingan Kinerja Antara Kedua Layanan Dari Platform *Container as a Service*

Kinerja layanan container Docker Setelah dilakukan 6 skenario pengujian, hasil dari pengujian layanan *container* docker ditunjukkan pada tabel dibawah ini:

Tabel 1. Kinerja Layanan Container Docker

Pengujian	CPU		RAM		Respon Time	
	< 30%	> 30%	< 600 mb	> 600 mb	< 100 ms	> 100 ms
Pengujian 1	2,65 %	-	574 mb	-	50 ms	-
Pengujian 2	7,25 %	-	-	922 mb	54 ms	-
Pengujian 3	-	84,45 %	-	675 mb	-	130 ms
Pengujian 4	-	98.00%	-	1.010 mb	-	2000 ms
Pengujian 5	-	100.00%	-	1.040 mb	-	2400 ms
Pengujian 6	-	39,45 %	581 mb	-	75 ms	-

Dari 1 sampai 6 skenario pengujian, penggunaan CPU *container* docker lebih unggul pada pengujian 1,2, dan 3. Kemudian dari segi penggunaan RAM *container* docker lebih unggul pada pengujian 1,4,5 dan 6, sedangkan untuk Response Time dari web server *container* docker lebih unggul pada pengujian 3.

Kinerja layanan *Container* Podman setelah dilakukan 6 skenario pengujian, hasil dari pengujian layanan *container* podman ditunjukkan pada tabel dibawah ini:

Tabel 2. Kinerja Layanan Container Podman

Pengujian	CPU		RAM		Response Time	
	< 30%	> 30%	< 600 mb	> 600 mb	< 100 ms	> 100 ms
Pengujian 1	3,95 %	-	-	623 mb	50 ms	-
Pengujian 2	11,15 %	-	-	903 mb	54 ms	-
Pengujian 3	-	94,05 %	-	658 mb	-	300 ms
Pengujian 4	-	97.00 %	-	1.070 mb	-	1700 ms
Pengujian 5	-	100.00%	-	1.070 mb	-	2000 ms
Pengujian 6	-	34,55 %	-	697 mb	65 ms	-

Untuk pengujian 1 sampai 6 dari segi penggunaan CPU *container* podman unggul pada pengujian 4, 5 dan 6, kemudian dari segi penggunaan RAM *container* docker unggul pada pengujian 2 dan 3, sedangkan untuk Response Time dari web server *container* docker unggul pada pengujian 4, 5 dan 6.

Dari perbandingan kedua platform *container* yaitu *container* docker dan *container* podman *container* docker lebih banyak unggul dari beberapa pengujian dari pada *container* podman. Untuk rekomendasi penggunaan teknologi *Container as a Service* (CaaS) ini *container* docker lebih bisa diandalkan karena pada saat terjadi serangan *Denial of Service (DoS)* kinerja dari layanannya tidak banyak mengalami penurunan, selain itu *container* docker saat ini adalah teknologi *container* yang paling populer dan banyak digunakan sesuai dengan pernyataan yang sudah dikutip sebelumnya.

4. SIMPULAN

Dari penelitian ini dapat ditarik beberapa kesimpulan sebagai berikut :

1. Untuk kinerja layanan container pada saat container berjalan normal yaitu dengan menjalankan 1 aplikasi saja container tidak banyak memakan sumber daya untuk menjalankan aplikasi dalam pengujian dan container docker lebih unggul dari container podman pada pengujian ini.
2. Pada saat container menjalankan banyak aplikasi yaitu sebanyak 6 aplikasi container juga masih belum terlalu memakan sumber daya yang besar, dan pada saat menjalankan pengujian ini container docker masih unggul dari container podman.
3. Saat container mendapatkan serangan Denial of Service kinerja dari kedua layanan container mengalami penurunan kinerja, dan sumber daya pada server mengalami peningkatan yang sangat besar.
4. Setelah pencegahan untuk serangan container diterapkan, layanan dari container podman lebih unggul dari container docker dari segi penggunaan sumber daya dan kinerja layanan.

Untuk penelitian yang akan datang terkait dengan penelitian yang berjudul "Analisis Perbandingan Kinerja Layanan Container As A Service (Caas) Studi Kasus : Docker dan Podman" ini:

1. Untuk hasil yang lebih maksimal disarankan untuk menggunakan container yang berapa pada cloud secara langsung yang dapat disewa pada penyedia layanan.
2. Agar dapat melihat kinerja layanan container diberbagai serangan, dapat digunakan jenis serangan selain serangan yang telah digunakan pada penelitian ini.

DAFTAR PUSTAKA

- [1] J. Sharif, "Membangun Private Cloud Computing dan Analisa Terhadap Serangan DoS, Study Kasus SMKN 6 Jakarta," *J. Telekomun. dan Komput.*, vol. 12 no. 3, p. 270, 2017, doi: 10.22441/incomtech.v6i3.1160.
- [2] F. Adiputra, "Container dan Docker: Teknik Virtualisasi dalam Pengelolaan Banyak Aplikasi Web," *J. SimanteC*, 2015.
- [3] C. Fiddin, R. Mayasari, and R. Munadi, "Analisis Performansi Virtualisasi Container Menggunakan Docker Dibawah Serangan Networked Denial of Service," *e-Proceeding Eng.*, vol. 5, no. 1, pp. 281-290, 2018.
- [4] O. Wijaya, J. Jusak, A. Sukmaaji, P. Studi, / Jurusan, and S. Komputer, "Pemodelan Karakteristik Denial of Service Attack Melalui Analisis Data Trafik," *Pemodelan Karakteristik Denial Serv. Attack Melalui Anal. Data Trafik*, 2014.
- [5] M. K. Imammuddin, Januar Al-Amien, "Membangun Cloud Menggunakan Docker Pada Implementasi Load Balancing dan Pengujian Algoritma Round Robin Pada Web Server," *Pros. Semin. Nas. Comput. Technol. its Apl.*, vol. 1, no. 1, pp. 17-21, 2019.

- [6] R. Khalida, A. Muhajirin, and S. Setiawati, "Teknis Kerja Docker Container untuk Optimalisasi Penyebaran Aplikasi," *PIKSEL Penelit. Ilmu Komput. Sist. Embed. Log.*, 2019, doi: 10.33558/piksel.v7i2.1819.
- [7] I. Harfiansyah, "Mengenal Teknologi Docker - Codepolitan," *codepolitan.com*, 2016. .
- [8] M. A. Aziz, A. Bhawiyuga, and F. A. Bakhtiar, "Implementasi Container Live Migration Antar- Cloud Provider Menggunakan Podman dan CRIU," vol. 4, no. 9, pp. 3246–3254, 2020.
- [9] A. Sheka, A. ¹³rsenev, and V. Samun, "Containerization in scientific calculations," in *SIBIRCON 2019 - International Multi-Conference on Engineering, Computer and Information Sciences, Proceedings, 2019*, doi: ¹¹1109/SIBIRCON48586.2019.8958324.
- [10] C. Mukmin and W. Cholil, "Perbandingan Openvz Dengan Kernel Based Virtual Machine (Kvm)," *J. Ilm. Matrik*, vol. 20, no. 2, pp. 129–135, 2019, doi: 10.33557/jurnalmatrik.v20i2.115.

ANALISIS PERBANDINGAN KINERJA LAYANAN CONTAINER AS A SERVICE (CAAS)

Studi Kasus : Docker dan Podman

ORIGINALITY REPORT

16%

SIMILARITY INDEX

15%

INTERNET SOURCES

1%

PUBLICATIONS

5%

STUDENT PAPERS

PRIMARY SOURCES

1

media.neliti.com

Internet Source

4%

2

klik.ulm.ac.id

Internet Source

1%

3

libraryproceeding.telkomuniversity.ac.id

Internet Source

1%

4

Submitted to Universitas Brawijaya

Student Paper

1%

5

norisahrunedukasi.files.wordpress.com

Internet Source

1%

6

simantectrunojoyo.files.wordpress.com

Internet Source

1%

7

doku.pub

Internet Source

1%

8

eprints.umm.ac.id

Internet Source

1%

9

www.scribd.com

10

juti.if.its.ac.id

Internet Source

1%

11

jurnal.univbinainsan.ac.id

Internet Source

1%

12

ojs.uajy.ac.id

Internet Source

1%

13

Denis Shishmintsev, Alexander Bokov.
"Calibration and Evaluation Methods of
Performance of DRFM-technology Radar Echo
Simulator", 2020 Ural Symposium on
Biomedical Engineering, Radioelectronics and
Information Technology (USBREIT), 2020

Publication

1%

14

repository.its.ac.id

Internet Source

1%

15

Ery Setiyawan Jullev Atmadji, Beki Maryuni
Susanto, Rahardian Wiratama. "Pemanfaatan
IPTables Sebagai Intrusion Detection System
(IDS) dan Intrusion Prevention System (IPS)
Pada Linux Server", Teknika, 2017

Publication

1%

16

core.ac.uk

Internet Source

1%

Exclude quotes Off

Exclude matches Off

Exclude bibliography Off