Rancang Bangung Private Server Menggunakan Platform Proxmox dengan Studi Kasus: PT.MKNT

Danur Wijayanto[1,*], Arizona Firdonsyah[2], Faisal Dharma Adhinata[3] , Akhmad Jayadi[4]

[1,2] Fakultas Sains dan Teknologi, Teknologi Informasi, Universitas 'Aisyiyah Yogyakarta, Indonesia
[3] Fakultas Informatika, Program Studi Rekayasa Perangkat Lunak, Institut Teknologi Telkom Purwokerto, Indonesia
[4]Fakultas Teknik dan Ilmu Komputer, Program Studi Informatika, Universitas Teknokrat Indonesia, Lampung, Indonesia
Email: [1,*] danurwijayanto@unisayogya.ac.id, [2]arizona@unisayogya.ac.id , [3]faisal@ittelkom-pwt.ac.id
[4]akhmad.jayadi@teknokrat.ac.id

Abstrak− Teknologi virtualisasi memerankan peran penting Infrastuktur TI meliputi *private data centers* dan *platform public cloud*. Ada beberapa jenis virtualisasi yaitu Virtualisasi *Container* dan *Hypervisor*. Virtualisasi berbasis container menggunakan kernel yang sama dan bekerja dalam layer *software*. *Container* memungkinkan menjalankan beberapa *instance* sistem operasi dan perangkat keras yang sama . Berbeda dengan *container*, *hypervisor* beroperasi pada level hardware memerlukan Operasi Sistem yang terpisah dengan sistem *host*. Ada beberapa platform virtualisasi yang dapat digunakan seperti Proxmox, VMWare ESX, dan OpenStack. Proxmox mendukung hypervisor KVM (Kernel-based Virtual Machine), dan LXC Container. KVM mempunyai performa CPU yang lebih baik daripada jenis virtualisasi lainnya seperti *native*, LXC, dan Docker. Penelitian ini bertujuan untuk mengimplementasi virtualisasi di PT.MKNT menggunakan platform virtualisasi PROXMOX. Hasil menunjukkan dengan menggunakan *Platform* PROXMOX dapat membantu untuk membuat dan mengelola VM dalam *private server*.

Kata Kunci: KVM, AWS, EC2, PROXMOX, LXC, Container

Abstract− Virtualization technology plays an important role in IT infrastructure including private data centers and public cloud platforms. There are several types of virtualization like Container Virtualization and Hypervisor. Container-based virtualization uses the same kernel and works in software layers. Container allows running multiple instances of the same operating system and hardware. Unlike containers, hypervisors operating at the hardware level, which require separate System Operations with host systems. There are several virtualization platforms that can be used such as Proxmox, VMWare ESX, and OpenStack. Proxmox supports KVM (Kernel-based Virtual Machine) hypervisors, and LXC Containers. KVM has better CPU performance than other virtualization types such as native, LXC, and Docker. This research aims to implement PROXMOX virtualization platform at PT. MKNT to create private server. Results demonstrated by using the PROXMOX Platform can help to create and manage VMs at private server.

Keywords: KVM, AWS, EC2, PROXMOX, LXC, Container

1. PENDAHULUAN

Teknologi virtualisasi memerankan peran penting Infrastuktur TI meliputi *private data centers* dan platform *public cloud*. Teknologi virtualisasi mengijinkan beberapa *guest* sistem operasi untuk berbagi perangkat keras yang sama dengan linkungan yang terisolasi [1]–[3]. Ada beberapa jenis virtualisasi yaitu Virtualisasi *Container* dan *Hypervisor* [4]. Virtualisasi berbasis *container* menggunakan kernel yang sama dalam menjalankan *instance* yang banyak dalam satu sistem operasi dan perangkat keras yang sama . *Container* salah satu solusi yang ringan daripada Virtualisasi *Hypervisor*. *Container* mengujinkan untuk menjalankan proses yang terisolasi di sistem *host* dalam satu Sistem Operasi tanpa *overhead* berlebih pada CPU dan *memory* yang disebabkan oleh *hypervisor* [5], [6]. Kontainer menyediakan lingkugan virtual pada layer aplikasi [7].

Hypervisor merupakan jenis virtualisasi yang telah lama digunakan. Berbeda dengan Container, Hypervisor beroperasi pada level hardware sehingga dapat mengisolasi Virtual Machine(VM) dari sistem host. VM sendiri merupakan virtual komputer yang mempunyai CPU (Central Processing Unit), memory, network interface, media penyimpanan dan sistem operasi sendiri [8]. Hal itu memungkinkan kita menjalankan VM yang menggunakan Sistem Operasi Windows di atas Linux. [5]. Untuk dapat melakukan isolasi VM, hypervisor melakukan manajemen sumber daya fisik seperti CPU, memory, jaringan, dan media penyimpanan.

Teknologi *virtualisasi* dimanfaatkan untuk menyelesaikan masalah heterogenitas (perbedaan versi *library* atau *tools* dari aplikasi website sehingga dapat menguruangi biaya dan kompleksitas hardware serta terciptanya lingkungan yang *scalable*, *elastic*, dan biaya yang efisien [5], [6], [9], [10]. Untuk mendukung *scalable* dan *elastic* sumber daya dapat diskalakan dengan mengatur jumlah sumberdaya fisik menggunakan *Virtual Machine* (VM) apabila menerapkan *horizontal scaling* atau *Physical Machine* (PM) apabila menerapkan *Vertical Scaling* [4].

Ada beberapa platform virtualisasi yang dapat digunakan seperti Proxmox, VMWare ESX, dan OpenStack. [11], [12]. Proxmox mendukung *hypervisor* KVM *(Kernel-based Virtual Machine*), dan LXC Container [1]. Penulis menggunakan PROXMOX karena bersifat *open source* dan menyediakan *web interface* untuk mengelola *Virtual Machine* (VM), media penyimpanan, sumber daya seperti CPU dan RAM [13][3].

KVM menggunakan virtualisasi hardware, sehingga tidak memerlukan untuk meodifikasi sistem operasi *host*. KVM menggunakan virtualisasi hardware untuk melakukan virtualisasi *processor*, *memory* dan *I/O* [3]. KVM mempunyai performa CPU yang lebih baik daripada jenis virtualisasi lainnya seperti *native*, LXC, dan Docker [5], [14]. Hasil tersebut yang mendasari penulis untuk menggunakan Platform PROXMOX yang mendukung KVM dikarenakan aplikasi yang dijalankan di *server* membutuhkan sumber daya CPU yang tinggi untuk pengolahan data.

Penulis melakukan penelitian di PT Mitra Komunikasi Nusantara Tbk (PT MKNT), perusahaan yang bergerak di bidang distribusi produk telekomunikasi [15]. PT MKNT mengembangkan sistem *Enterprise Resource Planning* (ERP) yang berfungsi untuk memenuhi kebutuhan perusahaan dalam mengakomodasi kebutuhan manajemen data dan informasi proses bisnis dan akuntansi. Sistem tersebut sebelumnya menggunakan layanan komputasi awan dari AWS yaitu Amazon RDS (*Amazon Relational Database Service*) untuk database dan EC2 (*Elastic Computing*) untuk menjalankan aplikasi, namun karena perusahaan ingin mengelola sendiri secara *private* maka dilakukan migrasi dari aws ke server fisik.

AWS merupakan singkatan dari *Amazon Web Service* yang merupakan salah satu penyedia layanan komputasi awan dengan memberikan layanan yang sesuai kebutuhan pengguna dimana sumber daya disediakan dengan harga murah dan tidak memerlukan pembayaran di awal. Pelanggan hanya harus membayar sesuai sumberdaya yang digunakan. [16]. Penelitian ini bertujuan untuk mengimplementasi virtualisasi di PT.MKNT menggunakan platform virtualisasi PROXMOX dan dapat memberikan manfaat berupa referensi penelitian mengenai pembangunan infrastruktur server memanfaatkan platform virtualisasi PROXMOX.

2. METODE PENELITIAN

Dalam subbab ini, akan dijelaskan mengenai metode yang dilakukan penulis dalam melakukan implementasi *Platform* PROXMOX di server fisik PT MKNT. Terdapat 2 tahap yang penulis gunakan dalam melakukan migrasi, yaitu mengidentifikasi layanan AWS yang sedang digunakan, kemudian melakukan perancangan dan implementasi PROXMOX pada *server* fisik.

2.1 Identifikasi Layanan AWS

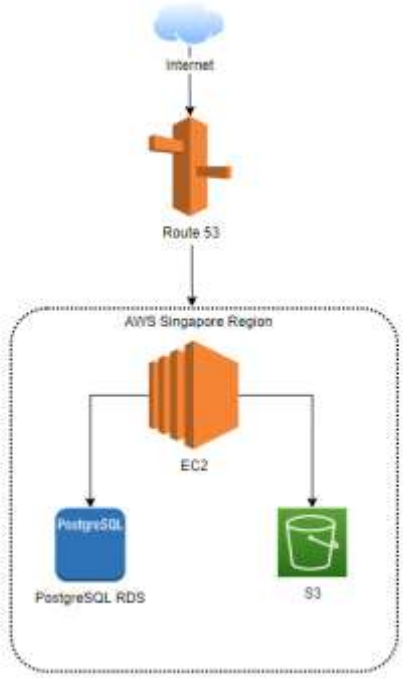
Pada tahap ini, peneliti melakukan identifikasi layanan AWS yang digunakan, kemudian dilakukan pemilihan tipe layanan yang dilakukan migrasi dan tidak. Layanan AWS yang digunakan antara lain :

a. Amazon EC2, merupakan layanan komputasi pada *cloud* yang dapat diubah ukuran dan spesifikasinya. Layanan ini termasuk dalam jenis Iaas (*Infrastructur as a* Service) [7]. Amazon EC2 menyediakan kontrol penuh pada sumber daya komputasi. Amazon EC2 juga menawarkan beragam pilihan *processor*, media penyimpanan, jaringan, sistem operasi, dan model pembelian. [17].
b. Amazon RDS, merupakan layanan database relasional untuk MySQL, oracle dan MS SQL Server. AWS RDS memudahkan untuk membuat, mengoperasikan *database* relasional pada *cloud* yang menyediakan *cost-efficient* dan perubahan kapasitas sewaktu-waktu serta terdapat fitur *backup* otomatis [18], [19].
c. Amazon Route53, merupakan layanan DNS (*Domain Name System*) yang mendukung *high available* dan *scalable*. Amazon Route53 didesain untuk memberikan pengembang aplikasi sebuah layanan yang dapat diandalkan dan *cost effective* untuk mengarahkan pengguna ke aplikasi dengan cara mentranslasikan nama seperti www.example.com ke alamat IP seperti 192.169.2.1 [20].
d. Amazon S3 (*Simple Storage Service*), merupakan salah satu layanan penyimpanan yang menawarkan ketersedian data, keamanan, dan performa. Layanan ini dapat digunakan oleh semua jenis pelanggan dan industri untuk menyimpan dan melindungi data dari beragam *use case* seperti *Big Data*, website, aplikasi mobile, *backup* dan *restore, archive*, aplikasi enterprise, perangkat IoT, dan *Big Data Analitics* [21].

Tabel 1 menunjukkan daftar layanan AWS yang digunakan dan status apakah akan dilakukan migrasi atau tidak, sedangkan Gambar 1 menunjukkan Topologi layanan AWS yang digunakan.

Tabel 1. Daftar Layanan

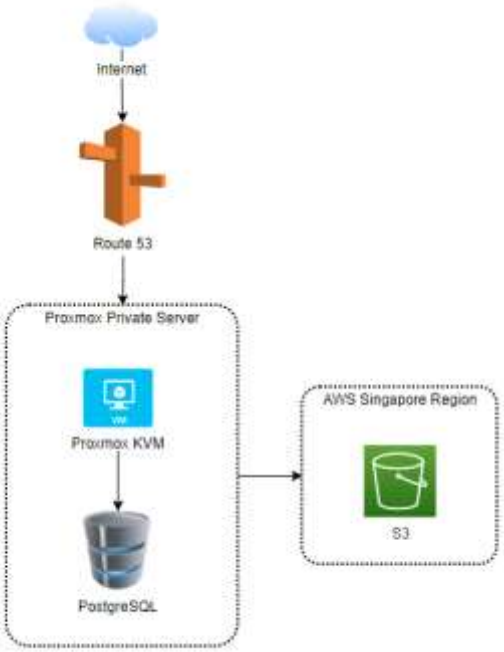
Layanan	Migrasi
Amazon EC2	Ya
Amazon RDS	Ya
Amazon Route53	Tidak
Amazon S3	Tidak



Gambar 1. Topologi Layanan AWS

2.2 Merancang dan Implementasi PROXMOX pada *Server* Fisik

Setelah mengidentifikasi layanan yang akan dimigrasi dari AWS, penulis melakukan desain ulang topologi. Tujuan dari mendesain ulang topologi adalah untuk menyesuaikan kebutuhan yang ada sehingga *environtment* server dapat berjalan dengan baik, dikarenakan ada beberapa layanan yang masih menggunakan AWS. Topologi baru yang digunakan ditunjukkan pada Gambar 2.



Gambar 2. Rancangan Topologi Private Server dengan Layanan AWS

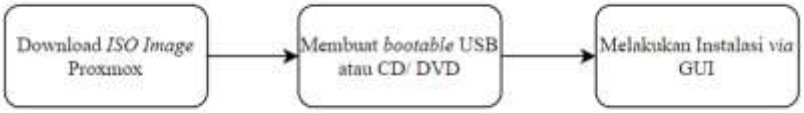
Pada Gambar 2 menunjukkan gambaran topologi Private Server yang terhubung dengan layanan AWS. Private Server menggunakan platform virtualisasi Proxmox yang kemudian dibuat dua *Virtual Machine* berbasis KVM,

yang digunakan untuk menjalankan aplikasi *front end* dan *back end* serta database PostgreSQL. Tujuan dari pemisahan database dan aplikasi antara lain adalah untuk memudahkan dalam pengeloaan, dan *scaling up*.

Setelah selesai melakukan desain ulang topologi server, langkah selanjutnya adalah melakukan implementasi. Server yang digunakan adalah DELL PowerEdge R740 dengan spesifikasi yang ditunjukkan pada Tabel 2. Sedangkan versi Proxmox yang digunakan adalah Proxmox versi 6.2. Sebelum melakukan instalasi Proxmox di server, kita harus mengunduh *ISO image* di website : https://www.proxmox.com/en/downloads yang kemudian membuat *bootable* USB atau CD/ DVD sebelum dilakukan proses instalasi. Cara instalasi proxmox ditunjukkan pada Gambar 3.

Tabel 2. Spesifikasi Server

Jenis	Spesifikasi
Processor	Dual Processor CPU(s) Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz
RAM (*Random Access Memmory)*	384 GB
Disk	SSD 2 TB
Amazon S3	Tidak



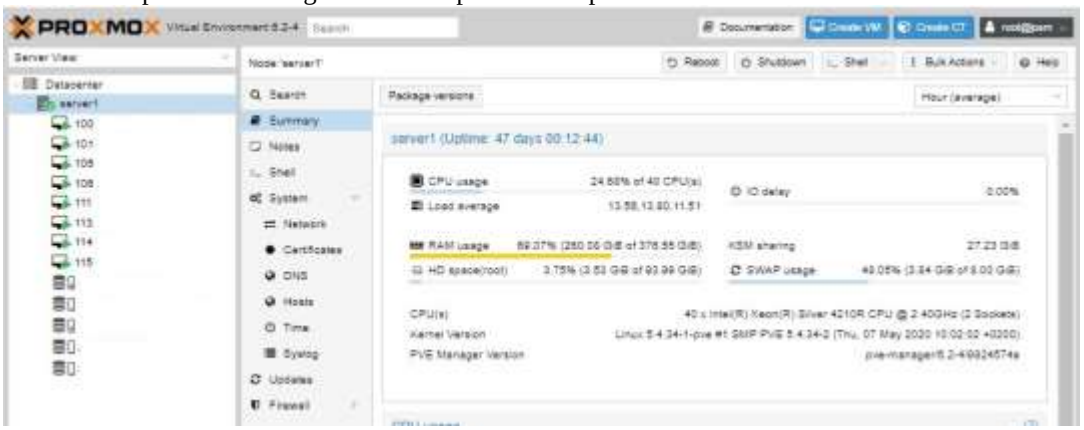
Gambar 3. Langkah – langkah Instalasi Proxmox

3. HASIL DAN PEMBAHASAN

Pada bab ini penulis menyajikan hasil implementasi dan pengujian. Pengujian bertujuan untuk mengetahui apakah platform Proxmox dapat berjalan dengan baik. Pengujian dilakukan yang akan dilakukan adalah sebagai berikut : pengujian konektivitas internet VM yang terdiri dari konektivitas dengan pengguna, antar VM dan dengan AWS. Dalam penyajian gambar di bab ini, penulis akan menyamarkan informasi mengenai alamat *domain*, alamat IP (*Internet Protocol*), dan beberapa VM yang tidak digunakan dalam penelitian ini dikarenakan data tersebut bersifat rahasia.

3.1 Hasil Implementasi

a. Tampilan Web Platform Proxmox
Gambar 4 menunjukkan tampilan *Control Panel* Proxmox berbasis web. Secara *default*, alamat web tersebut dapat diakses dengan alamat https://alamat-ip-server:8006.

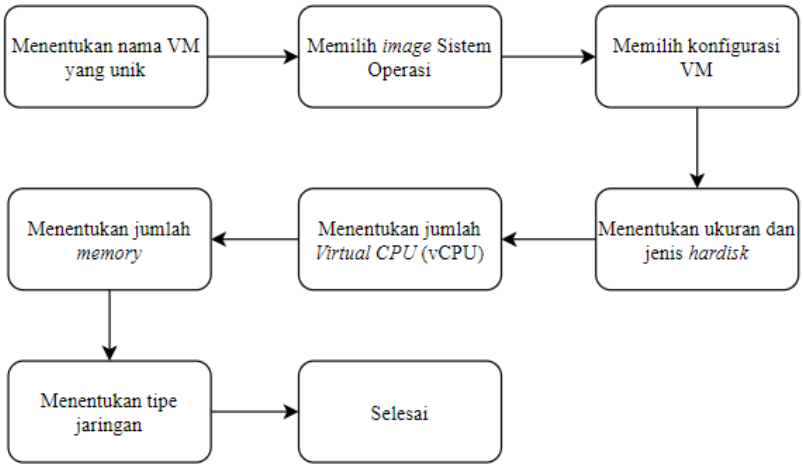


Gambar 4. Tampilan *Control Panel* Proxmox

b. Implementasi VM
Hasil dari implementasi ini berupa pembuatan dua buah VM dengan *hypervisor* KVM yang sudah ditunjukkan pada Gambar 2 yang berfungsi untuk menjalankan aplikasi dan Database PostgreSQL. Alur dalam membuat VM ditunjukkan pada Gambar 5. Langkah pertama adalah menentukan nama VM, nama VM yang digunakan harus unik, tidak boleh menggunakan nama yang sudah digunakan. Langkah kedua

yaitu memilih *image* sistem operasi yang digunakan. Pada penelitian ini, VM menggunakan Sistem Ubuntu 20.04 LTS (*Long Term Support*). Langkah ketiga yaitu mengkonfigurasi VM seperti *Graphic Card* dan *Disk Controller*. Penelitian ini menggunakan settingan default untuk *Graphic Card* dan VirtIO SCSI untuk *Disk Controller*.

Langkah keempat yang mengkonfigurasi ukuran dan jenis *hardisk*. Penulis menggunakan ukuran yang berbeda untuk kedua VM. Untuk VM dengan ID 105 mempunyai alokasi *disk* sebesar 400GB, sedangkakan untuk VM dengan ID 121 mempunyai alokasi *disk* sebesar 200GB. Langkah kelima yaitu menentukan *Virtual CPU* atau vCPU. Untuk VM dengan ID 105 mempunyai alokasi vCPU sebesar 20 vCPU, sedangkan untuk VM dengan ID 121 mempunyai alokasi 6 vCPU. Langkah keenam adalah mentukan jumlah *memory*, dimana VM dengan ID 105 mempunyai alokasi *memory* sebesar 170GB RAM, sedangkan untuk VM dengan ID 121 mempunyai alokasi *memory* sebesar 48GB RAM. Sedangkan langkah terakhir yaitu mengkonfigurasi perangkat jaringan yang sama – sama menggunakan tipe *bridge*. VM dengan ID 105 mempunyai alokasi *disk, memory,* dan *vCPU* lebih banyak daripada VM dengan ID 121 disebabkan VM dengan ID 105 digunakan sebagai *database server*. Detail spesifikasi VM dapat dilihat pada Tabel 3.



Gambar 5. Alur pembuatan VM

3.2 Pengujian Konektivitas

Pada subbab ini akan disajikan hasil pengujian berupa pengujian konektivitas dengan pengguna yaitu pengelola server, konektivitasi antar VM dan konektivitas VM dengan AWS S3. Untuk keperluan pengujian, dibutuhkan 2 VM dengan detail yang ditunjukkan pada Tabel 3 dimana penulis hanya akan menampilkan ID VM dan Hostname keamanan dan kerahasiaan alamat IP Publik.

Tabel 3. Detail VM untuk Pengujian

VM-ID	Hostname	Alamat IP Lokal	OS	Disk	vCPU
105	hp-db-primary	10.0.1.83	Ubuntu 20.04 LTS	400GB	20
121	fe-node-01	10.0.1.76	Ubuntu 20.04 LTS	200GB	6

3.2.1 Konektivitas dengan Pengguna

Pengujian konektivitas dengan pengguna dilakukan dengan menggunakan pengujian pengiriman paket ICMP (Internet Control Message Protocol) dari VM ke suatu domain (pada penelitian ini menggunakan alamat google.com) dan akses server memanfaatkan protokol SSH (Secure Shell) menggunakan aplikasi putty.

a. Pengiriman Paket ICMP

Gambar 6 menunjukkan pengiriman paket ICMP dari VM dengan ID 105. sedangkan Gambar 7 menunjukkan pengiriman paket ICMP dari VM dengan ID 121. Dari kedua hasil pengujian tersebut,

dihasilkan *packet loss* sebesar 0%. Sehingga dapat disimpulkan pengiriman paket ICMP sukses dan VM dapat mengakses jaringan internet.

```
root@hp-db-primary:~# ping google.com
PING forcesafesearch.google.com (216.239.38.120) 56(84) bytes of data.
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=1 ttl=118 time=12.7 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=2 ttl=118 time=12.2 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=3 ttl=118 time=12.2 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=4 ttl=118 time=12.2 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=5 ttl=118 time=12.2 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=6 ttl=118 time=12.2 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=7 ttl=118 time=12.3 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=8 ttl=118 time=12.3 ms
^C
--- forcesafesearch.google.com ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7009ms
rtt min/avg/max/mdev = 12.166/12.274/12.669/0.156 ms
root@hp-db-primary:~#
```

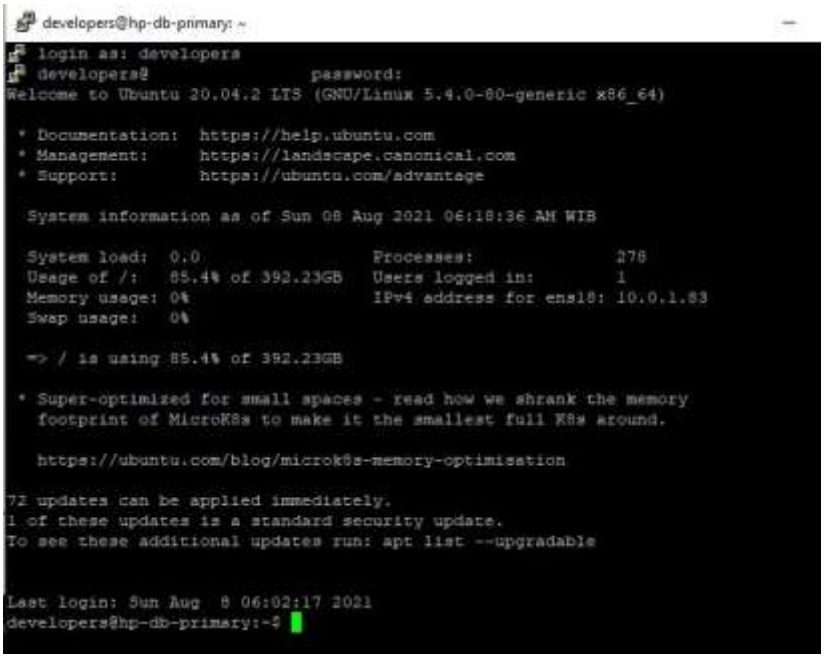
Gambar 6. Hasil Pengiriman Paket ICMP VM dengan ID 105

```
root@fe-node-01:~# ping google.com
PING forcesafesearch.google.com (216.239.38.120) 56(84) bytes of data.
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=1 ttl=118 time=12.4 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=2 ttl=118 time=12.3 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=3 ttl=118 time=12.3 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=4 ttl=118 time=12.2 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=5 ttl=118 time=12.3 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=6 ttl=118 time=12.3 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=7 ttl=118 time=12.4 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=8 ttl=118 time=12.4 ms
64 bytes from any-in-2678.1e100.net (216.239.38.120): icmp_seq=9 ttl=118 time=12.4 ms
^C
--- forcesafesearch.google.com ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8010ms
rtt min/avg/max/mdev = 12.235/12.350/12.442/0.059 ms
root@fe-node-01:~#
```

Gambar 7. Hasil Pengiriman Paket ICMP VM dengan ID 121

b. Akses Server Menggunakan *Putty*

Gambar 8 menunjukkan hasil *remote* ke VM dengan ID 105 menggunakan *putty*, sedangkan Gambar 9 menunjukkan hasil *remote* ke VM dengan ID 121 menggunakan *putty*. Dari kedua hasil pengujian tersebut, administator *server* sukses mengakses kedua VM. Sehingga dapat disimpulkan pengujian akses server sukses dan VM dapat dikelola secara *remote* atau jarak jauh.

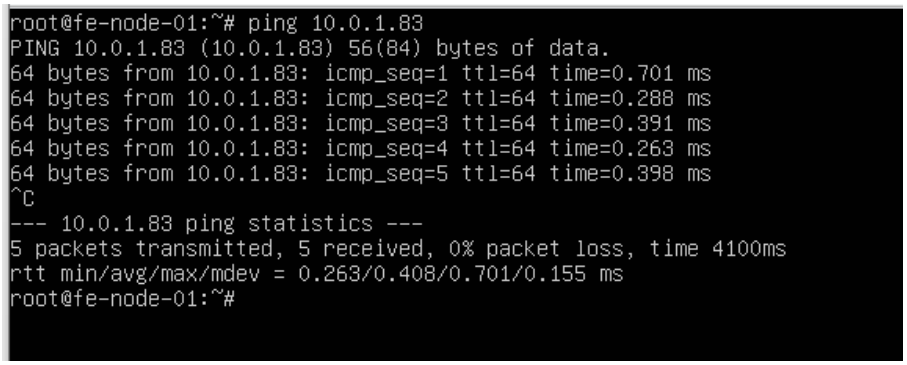


Gambar 8. Hasil Remote menggunakan putty ke VM dengan ID 105

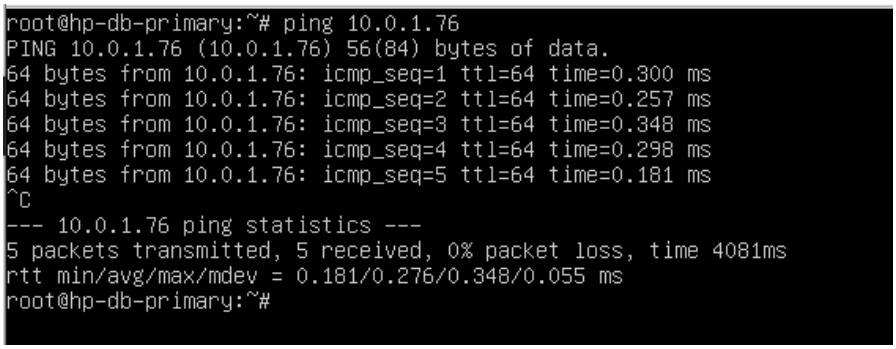
Gambar 9. Hasil Remote menggunakan putty ke VM dengan ID 121

3.2.2 Konektivitas antar VM

Pengujian konektivitas antar VM dilakukan dengan menggunakan pengujian pengiriman paket ICMP antar kedua VM. Dalam pengujian konektivitas ini, IP yang digunakan adalah IP Lokal, karena VM berada dalam satu jaringan dan *server* yang sama. Alamat kedua VM ditunjukkan pada Tabel 3. Hasil yang ditunjukkan pada Gambar 10 dan Gambar 11 menunjukkan paket ICMP dapat terkirim dengan sempurna yang dibuktikan dengan *packet loss* sebesar 0%. Sehingga dapat disimpulkan pengiriman paket ICMP sukses dan VM dapat saling berkomunikasi.



Gambar 10. Hasil ping dari VM dengan ID 121 ke VM dengan ID 105

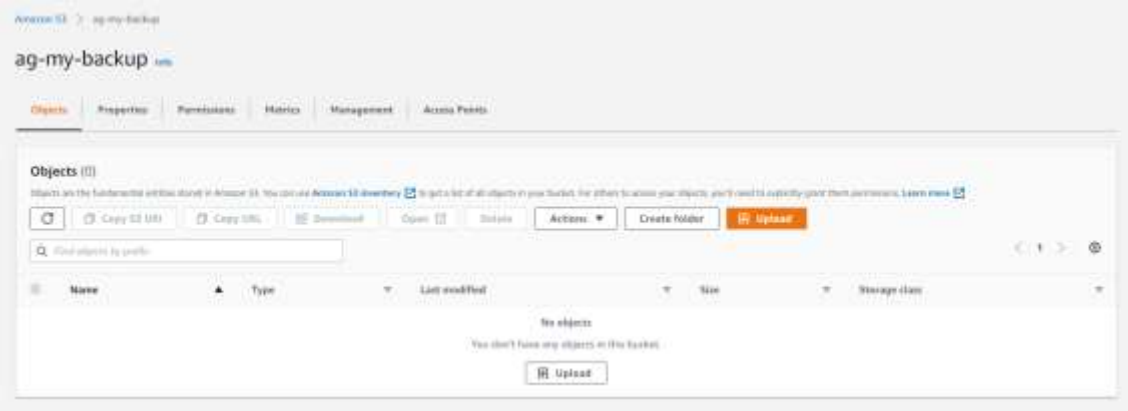


Gambar 11. Hasil ping dari VM dengan ID 105 ke VM dengan ID 121

3.2.3 Konektivitas dengan AWS S3

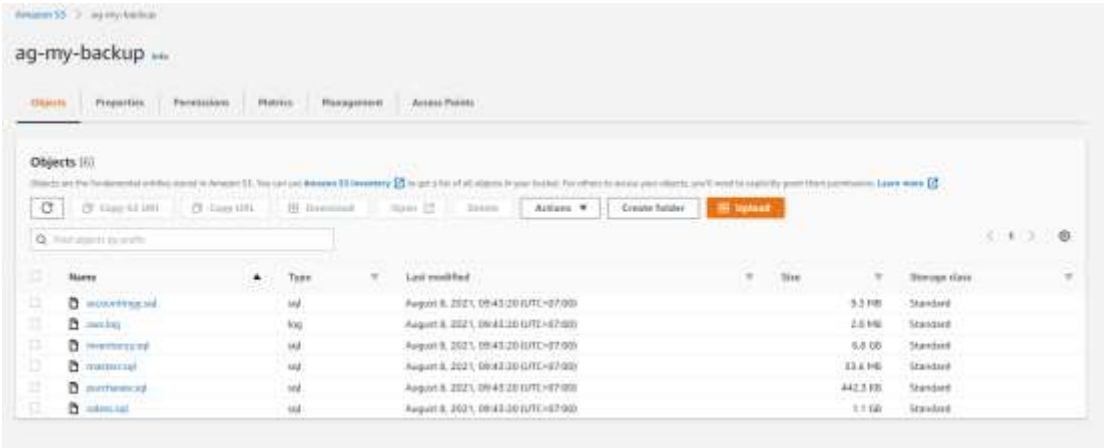
Pengujian konektivitas dengan AWS dilakukan dengan melakukan *upload* dan *download* file ke AWS S3. Pengujian ini hanya menggunakan VM dengan ID 105 yang membutuhkan koneksi ke AWS S3 untuk

keperluan backup. Sebelum dilakukan pengujian konektivitas ini, perlu dilakukan instalasi AWS *Command Line Interface* (CLI) supaya dapat terhubung ke AWS S3 serta membuat *bucket* di AWS S3. Gambar 12 menyunjukkan *bucket* yang telah dibuat dengan nama ag-my-backup, kemudian pada Gambar 13 menunjukkan proses ujicoba konektivitas dengan melakukan *upload* beberapa file ke *bucket* AWS yang beralamat di s3://ag-my-backup. Setelah proses *upload* selesai, dapat dilihat pada Gambar 14 bahwa file – file tersebut sudah sukses dan terbaca pada *bucket*. Dengan demikian, dapat disimpulkan bahwa VM sukses terkoneksi dengan AWS S3.



Gambar 12. S3 *Bucket*

Gambar 13. Proses Transfer Data ke AWS S3



Gambar 14. Isi dari S3 *Bucket* setelah Proses Transfer Data

Tabel 4 menunjukkan rangkuman ujicoba yang telah dilakukan. Dari tabel tersebut, dapat disimpulkan bahwa keseluruhan ujicoba dilakukan dengan sukses.

Tabel 4. Rangkuman Ujicoba

No	Pengujian	Hasil
1	Konektivitas dengan pengguna	sukses terhubung
2	Konektivitas antar VM	sukses terhubung
3	Konektivitas dengan AWS s3	sukses terhubung

4. KESIMPULAN

Dari hasil implementasi dan pengujian yang telah dilakukan dapat ditarik kesimpulan bahwa dengan menggunakan Platform Proxmox dapat dilakukan implementasi topologi yang mirip dengan topologi yang digunakan perusahaan pada AWS, serta dapat membantu untuk membuat dan mengelola VM dalam *private server*. Dalam lingkungan Proxmox juga memungkinkan antar VM untuk saling berkomunikasi menggunakan IP *Local*, sehingga tidak perlu memerlukan IP *Public*. Penelitian selanjutnya diharapkan dapat meneliti mengenai *High Availability* atau *Load Balancing* pada Platform Proxmox

UCAPAN TERIMAKASIH

Terima kasih disampaikan kepada Ginanjar Budhiraharja, S.I.P., M.M. selaku IT *Project Manager* di PT. Mitra Komunikasi Nusantara Tbk yang telah mengijinkan penulis untuk menyelesaikan penelitian ini serta pihak-pihak yang telah mendukung terlaksananya penelitian ini.

REFERENCES

[1] S. A. Algarni, M. R. Ikbal, R. Alroobaea, A. S. Ghiduk, and F. Nadeem, "Performance Evaluation of Xen, KVM, and Proxmox Hypervisors:," *Int. J. Open Source Softw. Process.*, vol. 9, no. 2, pp. 39–54, Apr. 2018, doi: 10.4018/IJOSSP.2018040103.

[2] P. China Venkanna Varma, V. K. C. K., V. Valli Kumari, and S. Viswanadha Raju, "Analysis of a Network IO Bottleneck in Big Data Environments Based on Docker Containers," *Big Data Res.*, vol. 3, pp. 24–28, Apr. 2016, doi: 10.1016/j.bdr.2015.12.002.

[3] A. Kovari and P. Dukan, "KVM & OpenVZ virtualization based IaaS open source cloud virtualization platforms: OpenNode, Proxmox VE," in *2012 IEEE 10th Jubilee International Symposium on Intelligent Systems and Informatics*, Subotica, Serbia, Sep. 2012, pp. 335–339. doi: 10.1109/SISY.2012.6339540.

[4] X. Wan, X. Guan, T. Wang, G. Bai, and B.-Y. Choi, "Application deployment using Microservice and Docker containers: Framework and optimization," *J. Netw. Comput. Appl.*, vol. 119, pp. 97–109, Oct. 2018, doi: 10.1016/j.jnca.2018.07.003.

[5] R. Morabito, J. Kjallman, and M. Komu, "Hypervisors vs. Lightweight Virtualization: A Performance Comparison," in *2015 IEEE International Conference on Cloud Engineering*, Tempe, AZ, USA, Mar. 2015, pp. 386–393. doi: 10.1109/IC2E.2015.74.

[6] C. de Alfonso, A. Calatrava, and G. Moltó, "Container-based virtual elastic clusters," *J. Syst. Softw.*, vol. 127, pp. 1–11, May 2017, doi: 10.1016/j.jss.2017.01.007.

[7] M. Uehara, "Performance Evaluations of LXC Based Educational Cloud in Amazon EC2," in *2016 30th International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, Crans-Montana, Switzerland, Mar. 2016, pp. 638–643. doi: 10.1109/WAINA.2016.24.

[8] "What is a virtual machine (VM)?" https://www.redhat.com/en/topics/virtualization/what-is-a-virtual-machine (accessed Aug. 28, 2021).

[9] Sulastri Apridayanti, Isnawaty, and Rizal Adi Saputra, "Desain Dan Implementasi Virtualisasi Berbasis Docker Untuk Deployment Aplkasi Web," Oct. 2018, doi: 10.5281/ZENODO.1407862.

[10] W. Li and A. Kanso, "Comparing Containers versus Virtual Machines for Achieving High Availability," in *2015 IEEE International Conference on Cloud Engineering*, Tempe, AZ, USA, Mar. 2015, pp. 353–358. doi: 10.1109/IC2E.2015.79.

[11] M. Riasetiawan, A. Ashari, and I. Endrayanto, "Distributed Replicated Block Device (DRDB) implementation on cluster storage data migration," in *2015 International Conference on Data and Software Engineering (ICoDSE)*, Yogyakarta, Indonesia, Nov. 2015, pp. 93–97. doi: 10.1109/ICODSE.2015.7436978.

[12] A. Arfriandi, "Perancangan, Iimplementasi, dan Analisis Kinerja Virtualisasi Server Menggunakan PROXMOX, VMWARE ESX, dan OPENSTACK," *J. Teknol.*, vol. 5, no. 2, pp. 182–192, 2012.

[13] L. Chen, W. Huang, A. Sui, D. Chen, and C. Sun, "The online education platform using Proxmox and noVNC technology based on Laravel framework," in *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*, Wuhan, China, May 2017, pp. 487–491. doi: 10.1109/ICIS.2017.7960041.

[14] W. Felter, A. Ferreira, R. Rajamony, and J. Rubio, "An updated performance comparison of virtual machines and Linux containers," in *2015 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, Philadelphia, PA, USA, Mar. 2015, pp. 171–172. doi: 10.1109/ISPASS.2015.7095802.

[15] "About Us," *About PT Mitra Komunikasi Nusantara Tbk*. https://www.mknt.id/about (accessed Jul. 28, 2021).

[16] S. Narula, A. Jain, and Prachi, "Cloud Computing Security: Amazon Web Service," in *2015 Fifth International Conference on Advanced Computing & Communication Technologies*, Haryana, India, Feb. 2015, pp. 501–505. doi: 10.1109/ACCT.2015.20.

[17] "AWS EC2," *Amazon EC2*. https://aws.amazon.com/ec2 (accessed Jul. 28, 2021).

[18] S. Mukherjee, "Benefits of AWS in Modern Cloud," *SSRN Electron. J.*, 2019, doi: 10.2139/ssrn.3415956.

[19] "AWS RDS," *Amazon RDS*. https://aws.amazon.com/rds (accessed Jul. 28, 2021).

[20] "AWS Route53," *Amazon Route53*. https://aws.amazon.com/route53 (accessed Jul. 28, 2021).

[21] "AWS S3," *Amazon S3*. https://aws.amazon.com/s3 (accessed Jul. 28, 2021).