

ANALISIS PENINGKATAN KINERJA FTP SERVER MENGGUNAKAN LOAD BALANCING PADA CONTAINER

Januar Al Amien ¹, Doni Winarso ²

1 Program Studi Teknik Informatika Universitas Muhammadiyah Riau

2 Program Studi Sistem Informasi Universitas Muhammadiyah Riau
Indonesia

Email: januaralamien@umri.ac.id¹, doniwinarsi@umri.ac.id²

ABSTRAK

Cloud computing merupakan teknologi yang menjawab tantangan akan kebutuhan teknologi komputasi yang efisien. Terdapat banyak hal yang dapat diimplementasikan menggunakan teknologi cloud computing seperti web service, layanan penyimpanan, aplikasi dan lain-lain. Penerapan cloud computing dengan menggunakan teknologi container dapat membantu dalam pengelolaan aplikasi serta mengoptimalkan penggunaan sumber daya berupa penggunaan memory dan prosesor pada server. Dalam penelitian ini penerapan docker container diimplementasikan menggunakan layanan aplikasi FTP (File Transfer Protocol). Layanan FTP dibuat menjadi 3 container didalam satu computer server. Untuk menangani permasalahan beban kinerja pada FTP server terhadap permintaan yang terlalu berat (overload) digunakan load balancing. Load balancing merupakan metode untuk meningkatkan kinerja sekaligus mengurangi beban kinerja pada FTP server. Berdasarkan hasil pengujian, penerapan multi container serta load balancing didalam FTP server dalam penanganan beban memiliki hasil penggunaan memory yang lebih kecil dan pemanfaatan penggunaan prosesor yang merata.

Kata Kunci: *Cloud Computing, Docker, FTP, Load Balancing, HAProxy.*

ABSTRACT

Cloud computing is a technology that answers the challenge of the need for efficient computing technology. There are many things that can be implemented using cloud computing technologies such as web services, storage services, applications and others. Use of cloud computing using container technology can help in the management of applications and optimize the use of resources in the form of memory and processor usage on the server. In this research docker containers implemented by service of FTP (File Transfer Protocol). The FTP service is made into 3 containers within a single server computer. To handle load problems performance on the FTP server against overload requests, load balancing is used. Load balancing is a method to improve performance while reducing the performance load on FTP servers. Based on the test results, the use of multi container and load balancing in the FTP server in load handling has result of smaller memory usage and utilization of processor usage evenly.

Keywords: *Cloud Computing, Docker, FTP, Load Balancing, HAProxy.*

1. PENDAHULUAN

Akhir-akhir ini *cloud computing* menjadi teknologi yang dikembangkan oleh perusahaan besar dalam memberikan fasilitas kepada para penggunaannya seperti halnya Google dengan aplikasinya yaitu Google driver. Google drive memberikan layanan penyimpanan kepada penggunaannya yang seolah-olah memiliki tempat penyimpanan sendiri yang bisa dibawa kemana saja dan dapat diakses melalui berbagai macam media elektronik seperti *smartphone*, *tablet* atau komputer.

Dalam pengertiannya *cloud* (awan) merupakan gambaran dari jaringan komputer / internet yang diabstraksikan yang mana didalamnya terdapat berbagai macam infrastruktur komputerisasi yang kompleks. Pada *cloud computing* sumber daya seperti prosesor, tempat penyimpanan, jaringan, perangkat lunak menjadi abstrak (virtual) dan diberikan layanan di jaringan / internet. Dengan menggunakan teknologi *cloud computing* kita dapat menggabungkan beberapa perangkat menjadi satu kesatuan dan dapat membuat banyak server pada satu perangkat komputer dengan menggunakan virtualisasi [1].

Layanan pada *cloud computing* dapat menangani banyak permintaan (request) dalam satu waktu. Apalagi pada sebuah instansi yang memerlukan lalu lintas data yang tinggi yang dapat menyebabkan kinerja pada sistem menjadi sangat lambat dikarenakan kelebihan beban (*overload*). Untuk

penggunaan server dengan sistem terdistribusi membutuhkan suatu metode agar dapat membagi beban dengan merata disetiap server yaitu dengan menerapkan metode *load balancing* [2].

Penerapan *load balancing* pada server bertujuan untuk menangani server yang sibuk sehingga dapat meningkatkan kinerja pada sistem terdistribusi [2].

Docker *container* tidak seperti seperti Virtual mesin, dimana docker tidak menggunakan *hardware* atau virtualisasi. Program yang berjalan pada docker *container* berhubungan langsung dengan kernel linux pada *host* sistem operasi. *Container* memungkinkan mengisolasi lingkungan program, sehingga program dapat berjalan tanpa gangguan dari permasalahan di sistem operasi [2].

Universitas Muhammadiyah Riau program studi Teknik Informatika, merupakan sebuah instansi yang memerlukan sebuah sistem penyimpanan data terpadu, akan tetapi infrastruktur yang ada belum cukup memadai untuk memenuhi kebutuhan penyimpanan data para Dosen. Pengembangan sistem *cloud computing* dengan memanfaatkan sumber daya komputer yang dimiliki oleh instansi menjadi solusi untuk mengatasi permasalahan tersebut. Dalam penelitian berfokus pada uji coba terhadap sebuah aplikasi FTP dengan mengukur kemampuan kinerja satu container dan multi container menggunakan load balancing terhadap penggunaan memori dan CPU.

2. TINJAUAN PUSTAKA

2.1. Cloud Computing

Komputasi awan (Cloud Computing) merupakan gabungan dari teknologi komputasi dan pengembangan berbasis internet (awan). Awan (cloud) adalah metafora dari internet, sebagaimana awan sering digambarkan di diagram jaringan komputer. Cloud computing juga merupakan abstraksi dari infrastruktur kompleks yang disembunyikannya. Ia adalah suatu metode komputasi dimana kapabilitas terkait teknologi informasi disajikan sebagai suatu layanan (as a service), sehingga pengguna dapat mengaksesnya melalui internet tanpa mengetahui apa yang ada didalamnya, ahli dengannya, atau memiliki kendali terhadap infrastruktur teknologi yang membantunya [3].

Adapun 3 jenis layanan yang terdapat didalam cloud computing antara lain [4]:

a) *Software as a Service (SaaS).*

Kebanyakan karyawan kantor berkecimpung dengan mengolah kata, Spreadsheet maupun alat bantu presentasi. Untuk yang berbasis Linux mungkin tidak menjadi masalah karena tidak berbayar, sementara yang berbasis Windows perlu membayar biaya lisensinya. Dengan SaaS, dimanapun, kapanpun, pekerja dapat mengakses file di lokal karena baik data maupun program tersedia di cloud. Contohnya adalah Google Doc dengan menyediakan layanan aplikasi Word Processing, Spreadsheet dan Presentation, selain tentu saja tempat penyimpanannya. Bahkan beberapa vendor sudah menyediakan layanan lengkap untuk sistem enterprise, seperti Customer Resource Management (CRM) yang mampu mengelola sistem persediaan suatu perusahaan. Jadi SaaS berbeda dengan PaaS, karena SaaS pengguna hanya tinggal menggunakan, sedangkan PaaS harus membuat sendiri.

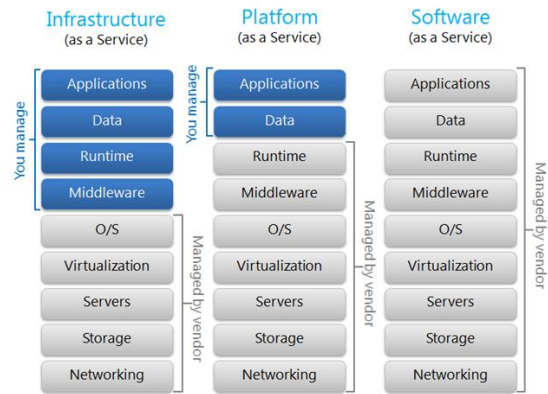
b) *Platform as a Service (PaaS).*

Pada perkantoran dalam beroperasi, karyawan menggunakan aplikasi baik berbasis Windows maupun Linux seperti aplikasi penjualan, pembelian, stok, pendaftaran siswa, dan sebagainya. Dalam perjalanannya biasanya sangat mengandalkan pembelian perangkat lunak dengan harga yang tidak murah. PaaS mengatasi masalah tersebut dengan memberikan layanan aplikasi, misalnya Google dengan Google App dimana pengguna dapat membuat aplikasi berbasis web sendiri tanpa memusingkan server dan domain name.

c) *Infrastructure as a Service (IaaS).*

Biasanya dalam membangun Sistem Informasi, suatu institusi membangun infrastruktur standar berupa sistem client – server baik berbasis web maupun desktop. Jaringan perlu disediakan baik wireless maupun wire dengan server terpusat. Dengan IaaS, semua diserahkan kepada cloud, baik sistem komunikasi maupun servernya (Email, Web, dan Database). Karena cenderung menangani sarana dan prasarana, IaaS sering disebut CloudWare. Cloud computing sebagai servis terhadap infrastruktur harus memperhatikan pula aspek keamanan.

Adapun perbedaan SaaS, PaaS, IaaS seperti pada Gambar 1 berikut [5].



Gambar 1. Stack Layanan Cloud Computing

Dimulai dari sebelah kanan yaitu SaaS, seluruh stack merupakan tanggung jawab penyedia layanan cloud. Konsumen benar-benar hanya mengkonsumsi aplikasi yang disediakan. Pada bagian tengah yaitu PaaS, penyedia layanan cloud bertanggung jawab mengelola Networking hingga Runtime.

Konsumen memiliki kendali dan tanggung jawab membuat aplikasi dan juga skema database-nya. Pada bagian kiri yaitu IaaS, penyedia layanan cloud bertanggung jawab untuk Networking hingga Virtualization. Konsumen sudah mulai bertanggung jawab untuk Operating System keatas.

Bentuk penyebaran cloud computing antara lain [6]:

a) *Private Cloud*

Infrastruktur cloud yang ditetapkan secara eksklusif untuk digunakan oleh satu organisasi yang membawahi beberapa konsumen (misalnya : unit bisnis). Dapat dimiliki, diatur dan dikendalikan oleh organisasi, pihak ketiga, atau paduan antar keduanya, dan dapat berada atau diluar lokasi.

b) *Community Cloud*

Infrastruktur cloud yang ditetapkan secara eksklusif untuk digunakan oleh komunitas konsumen dari organisasi yang memiliki kepedulian yang sama (misalnya : misi, kebutuhan keamanan, kebijakan, dan pertimbangan kepatuhan). Mungkin dimiliki, diatur dan dijalankan oleh satu atau lebih organisasi dalam komunitas, pihak ketiga, atau paduan antar keduanya, dan dapat berada atau diluar lokasi.

c) *Public Cloud*

Infrastruktur cloud yang ditetapkan untuk bebas digunakan oleh masyarakat luas. Mungkin dimiliki, diatur, dan dijalankan oleh organisasi bisnis, akademis atau pemerintahan, atau kombinasi diantaranya. Berada pada lokasi penyedia layanan cloud

d) *Hybrid Cloud*

Infrastruktur cloud yang merupakan komposisi dari dua atau lebih infrastruktur cloud yang berbeda (private, community, atau public), yang masing-masing tetap berdiri sendiri sebagai entitasnya, namun terikat dengan standar atau kepemilikan teknologi yang sama, yang memungkinkan portabilitas data dan aplikasi (seperti : cloud bursting untuk menyeimbangkan beban antar cloud).

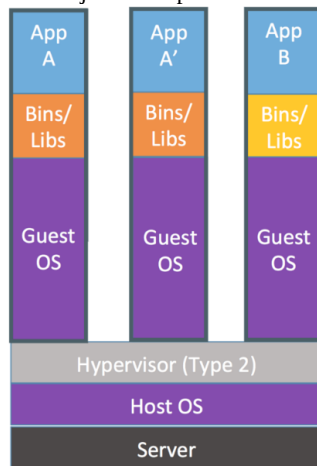
2.2. Docker

Docker merupakan sebuah tool yang memudahkan untuk membuat, menyebarkan dan menjalankan aplikasi dengan menggunakan container. Container memungkinkan para developer untuk mengemas sebuah aplikasi dengan keseluruhan komponen yang dibutuhkan, seperti libraries dan kebutuhan lainnya dan dikemas menjadi sebuah paket. Sehingga developer dapat memastikan bahwa aplikasi akan berjalan pada mesin Linux yang lainnya tanpa perlu

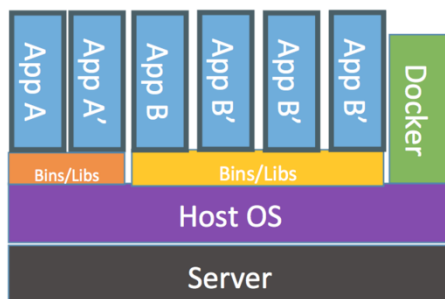
memperhatikan pengaturan yang dimiliki mesin tersebut yang mungkin berbeda dengan mesin yang digunakan untuk menulis dan menguji kode [7].

Pada dasarnya docker sedikit mirip dengan virtual machine. Tetapi tidak seperti sebuah virtual machine, dari pada membuat keseluruhan sistem operasi virtual, docker memungkinkan aplikasi untuk digunakan pada kernel linux yang sama dengan sistem yang dijalankan, dan hanya membutuhkan aplikasi yang dibungkus dengan segala sesuatu yang dibutuhkan tidak berjalan pada komputer host. Ini memberikan peningkatan kinerja dan mengurangi ukuran pada aplikasi^[7].

Perbedaan Docker dengan virtual machine adalah containers dalam docker memiliki sumber daya yang terisolasi sama dan manfaat alokasi seperti virtual machine tetapi perbedaan arsitektur yang berbeda memungkinkan containers untuk menjadi lebih portabel dan efisien [8].



Gambar 2. Sistem Kerja Virtual Machine



Gambar 3. Sistem Kerja Docker

Perbedaan antara virtual machine dan docker seperti gambar 2, virtual machine membutuhkan banyak ruang untuk Guest OS (Sistem Operasi). Sedangkan docker pada gambar 3, docker tidak membutuhkan banyak sistem operasi, karena satu sistem operasi dapat digunakan bersama pada seluruh container yang mana masing-masing container saling terisolasi satu sama lain yang disesuaikan dengan Bin/Libs [8].

2.3. FTP Server

FTP (File Transfer Protocol) server adalah aplikasi yang memberika akses atau pertukaran transfer data antara dua komputer (client dan server). Transer file atau data dapat terjadi antara komputer di jaringan lokal atau melalui internet. FTP menggunakan port koneksi 20 dan port 21 dan berjalan exclusively melalui TCP bukan UDP. FTP server mendengarkan pada port 21 untuk incoming connection dari

FTP client. Biasanya port 21 adalah command port dan port 20 adalah data port [1].

FTP bisa berjalan pada mode aktif atau mode pasif, yang menentukan bagaimana koneksi data terbentuk [9]:

a. Mode Aktif

Pada mode aktif, client membuat kontrol koneksi TCP ke server dan mengirimkan server alamat IP client dan client port number, dan kemudian menunggu hingga server memulai koneksi data TCP ke alamat IP client dan port number client.

b. Mode Pasif

Pada mode pasif, client menggunakan kontrol koneksi untuk mengirim perintah PASV ke server dan menerima alamat IP server dan port number dari server. Kemudian digunakan client untuk membuka koneksi data dari port client ke alamat IP server dan port number server yang diterima. Mode pasif dapat digunakan pada situasi dimana client berada dibelakang firewall dan tidak bisa menerima koneksi TCP yang masuk.

2.4. Load Balancing

Load Balancing adalah sebuah metodologi jaringan komputer untuk mendistribusikan beban kerja di beberapa komputer atau cluster komputer untuk mencapai pemanfaatan optimal sumber daya, memaksimal throughput meminimalkan waktu respon dan menghindari kelebihan beban. Load balancer dapat melakukan berbagai fungsi seperti load balancing, traffic engineering, dan switching cerdas trafik. Load balancer dapat melakukan pemeriksaan kesehatan pada server, aplikasi, dan konten yang meningkatkan ketersediaan layanan dan pengelolaannya. Dalam penerapan load balancer ada berbagai metode atau algoritma yang dapat digunakan untuk mendistribusikan beban kepada setiap server didalam sekumpulan server [10].

Ada dua algorithma perbandingan [11]:

1. Algoritma least Connection

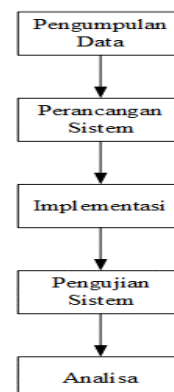
Mengalokasikan permintaan ke instance dengan paling sedikit koneksi aktif.

2. Algoritma Round Robin

Memutar permintaan secara merata di antara beberapa kejadian.

I. PERANCANGAN

1.1 Metode Penelitian

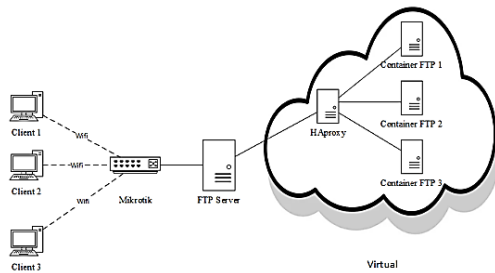


Gambar 4. Kerangka Konseptual Penelitian

Berdasarkan gambar 4 Kerangka konseptual merupakan suatu bentuk kerangka berpikir yang dapat digunakan sebagai pendekatan ilmiah dalam memecahkan masalah yang menggambarkan hubungan antar variabel dalam proses analisis.

1.2 Topology Jaringan

Topology jaringan menggambarkan bentuk konsep komputer yang terhubung ke jaringan yang saling terkoneksi. Adapun bentuk rancangan topology jaringan pada penelitian yang akan dibuat seperti pada Gambar 5.



Gambar 5. Topologi Jaringan

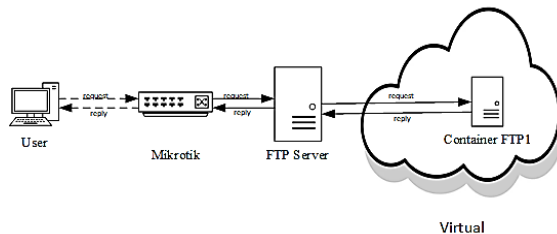
Berdasarkan gambar 5, didalam FTP server terdapat 3 container FTP dan HAProxy digunakan sebagai balancer yang akan menangani proses load balancing. Dari ketiga user yang melakukan request terhadap FTP server, proses request tersebut akan dibagikan ke dalam 3 container FTP yang akan membuat penanganan request yang dilakukan oleh 3 user dibagi rata kedalam container FTP.

1.3 Skenario

Perencanaan skenario pengujian menggambarkan urutan pengujian yang akan dilakukan, adapun pada pengujian ini menggunakan beberapa skenario dalam melakukan pengujian, yaitu:

a) Skenario satu

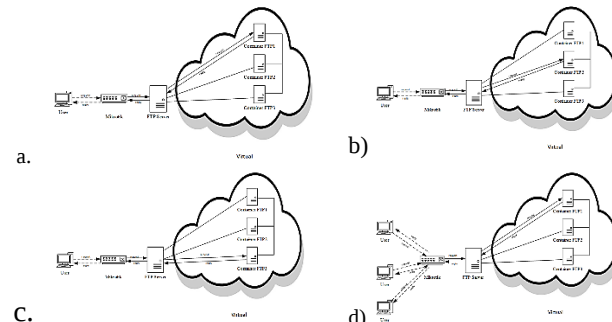
Bertujuan untuk menguji kinerja dan mengukur penggunaan CPU dan memory pada salah satu container dengan cara melakukan login kedalam container dan melakukan proses upload data.



Gambar 6. Topologi Skenario satu

b) Skenario dua

bertujuan untuk menguji sinkronisasi data pada setiap container.

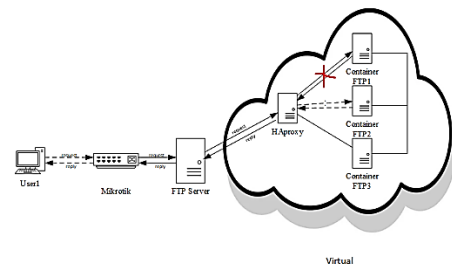


Gambar 7. Topologi Skenario dua

- Pengujian dilakukan dengan menggunakan 1 account user yang terdaftar pada container FTP1 dan melakukan pengujian login dan upload data kedalam container FTP1.
- Pengujian login kedalam container FTP2 menggunakan account user yang terdaftar pada FTP1, kemudian mengecek isi data didalam container FTP2 yang telah diupload melalui container FTP1.
- Pengujian login kedalam container FTP3 menggunakan account user yang terdaftar pada FTP1, kemudian mengecek isi data didalam container FTP3 yang telah diupload melalui container FTP1.
- Pengujian menggunakan 3 user yang terdaftar pada masing-masing container untuk melakukan proses upload data kedalam salah satu container dan melihat kinerja penggunaan CPU dan memory pada server FTP dan 1 container.

c) Skenario tiga

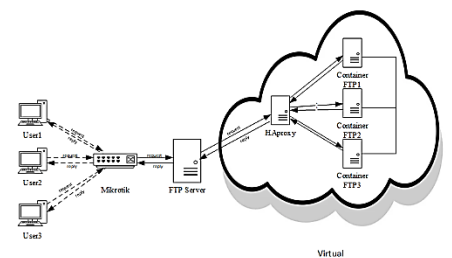
Bertujuan untuk menguji penanganan load balancing yang akan diuji dengan menggunakan 1 user yang melakukan login kedalam salah satu container FTP dan kemudian koneksi atau service yang digunakan oleh user diputus dan user dialihkan kedalam container FTP yang lain.



Gambar 8. Topologi Skenario tiga Pengujian terhadap gangguan salah satu Container FTP

d) Skenario empat

Bertujuan mendapatkan hasil kinerja penggunaan CPU dan memory pada saat penanganan loadbalancing dengan 3 user yang melakukan aktifitas upload data kedalam container FTP.



Gambar 9. Topologi Skenario empat Pengujian Kinerja Load Balancing Container FTP

II. IMPLEMENTASI DAN PENGUJIAN

2.1 Instalasi Docker

Untuk dapat melakukan perancangan container, langkah awal adalah melakukan instalasi docker. Docker digunakan untuk menjalankan multi container FTP. Untuk melakukan instalasi docker pertama kali yang harus dilakukan adalah melakukan menambahkan repository docker kedalam repository milik host sever dengan menggunakan perintah berikut.

```
$ sudo add-apt-repository \
```

```
"deb [arch=amd64]
https://download.docker.com/linux/ubuntu \

$(lsb_release -cs) \

stable"
```

Setelah menambahkan repository docker kedalam list repository pada host server, langkah selanjutnya adalah melakukan instalasi docker dengan menggunakan perintah berikut.

```
$ sudo apt-get install docker-ce
```

2.2 Instalasi Docker Compose

Docker compose merupakan sebuah alat untuk menentukan dan menjalankan multi-container pada docker. Dengan menggunakan compose, file dengan format YAML merupakan file untuk mengkonfigurasi service aplikasi yang akan dijalankan. Dan dengan menggunakan satu instruksi dapat membuat sekaligus menjalankan semua service yang sudah dikonfigurasi. Untuk dapat menjalankan service pada docker compose, terlebih dahulu melakukan instalasi dengan menggunakan perintah berikut.

```
$ sudo curl -L
https://github.com/docker/compose/releases/download/1.19.
0/docker-compose - `uname -s` - `uname -m` -o
/usr/local/bin/docker-compose
```

2.3 Pembuatan Container FTP

Pada tahapan pembuatan container FTP yang harus dilakukan pertama kali adalah membuat docker images. Docker images merupakan serangkaian lapisan instruksi yang ada pada Dockerfile.

Dalam pembuatan container FTP pada penelitian ini menggunakan images yang ada pada repository di dockerhub <https://hub.docker.com/r/stilliard/pure-ftp/>.

Dalam penelitian ini repository tersebut dikembangkan dengan menggunakan Dockerfile yang berada pada directory bernama FTP. Didalam directory FTP selain terdapat file dengan nama Dockerfile terdapat juga directory lain dengan nama ftpserver2 dan ftpserver3 dengan masing-masing directory memiliki file dengan nama Dockerfile. Berikut adalah isi Dockerfile pada directory FTP.

```
FROM stilliard/pure-ftp

EXPOSE 20 21 2020 30000 30001 30002 30003 30004
30005 30006 30007 30008 30009 30010 30011 30012
30013 30014 30015

FROM /run.sh -c 30 -C 10 -l puredb:/etc/pure-
ftpd/pureftpd.pdb -x -E -j -R -P $PUBLICHOST -p
30000:30015
```

Berikut adalah isi Dockerfile pada directory ftpserver2.

```
FROM stilliard/pure-ftp

EXPOSE 20 21 2121 30016 30017 30018 30019 30020
30021 30022 30023 30024 30025 30026 30027 30028
30029 30030
```

```
FROM /run.sh -c 30 -C 10 -l puredb:/etc/pure-
ftpd/pureftpd.pdb -x -E -j -R -P $PUBLICHOST -p
30016:30030
```

Berikut adalah isi Dockerfile pada directory ftpserver3.

```
FROM stilliard/pure-ftp

EXPOSE 20 21 2222 30031 30032 30033 30034 30035
30036 30037 30038 30039 30040 30041 30042 30043
30044 30045

FROM /run.sh -c 30 -C 10 -l puredb:/etc/pure-
ftpd/pureftpd.pdb -x -E -j -R -P $PUBLICHOST -p
30031:30045
```

Selanjutnya membuat file docker-compose.yml didalam directory FTP untuk membuat multi container FTP. Berikut adalah isi pada docker-compose.yml.

```
version: '3'

services:
  ftpd_server1:
    build: .
    container_name: ftpd_server1
    ports:
      - "2020:21"
    environment:
      PUBLICHOST: 192.168.100.251
      ADDED_FLAGS: -O w3c:/var/log/pure-
ftpd/transfer.log
    networks:
      ftp:
        ipv4_address: 172.16.1.11
    volumes:
      - ftp-db-volume:/etc/pure-ftp/passwd
      - /ftp_data/mondy/data_users:/home/ftpusers
    expose:
      - "21"
      - "30000"
      - "30001"
      - "30002"
      - "30003"
      - "30004"
      - "30005"
      - "30006"
      - "30007"
      - "30008"
      - "30009"
      - "30010"
      - "30011"
      - "30012"
      - "30013"
      - "30014"
      - "30015"
    restart: always

  ftpd_server2:
    build: ./ftpserver2
    container_name: ftpd_server2
    ports:
      - "2121:21"
    environment:
      PUBLICHOST: 192.168.100.251
```

```

    ADDED_FLAGS: -O w3c:/var/log/pure-
ftpd/transfer.log
networks:
  ftp:
    ipv4_address: 172.16.1.12
volumes:
  - ftp-db-volume:/etc/pure-ftpd/passwd
  - /ftp_data/mondy/data_users:/home/ftpusers
expose:
  - "21"
  - "30016"
  - "30017"
  - "30018"
  - "30019"
  - "30020"
  - "30021"
  - "30022"
  - "30023"
  - "30024"
  - "30025"
  - "30026"
  - "30027"
  - "30028"
  - "30029"
  - "30030"
restart: always

ftpd_server3:
  build: ./ftpserver3
  container_name: ftpd_server3
  ports:
    - "2222:21"
  environment:
    PUBLICHOST: 192.168.100.251
    ADDED_FLAGS: -O w3c:/var/log/pure-
ftpd/transfer.log
networks:
  ftp:
    ipv4_address: 172.16.1.13
volumes:
  - ftp-db-volume:/etc/pure-ftpd/passwd
  - /ftp_data/mondy/data_users:/home/ftpusers
expose:
  - "21"
  - "30031"
  - "30032"
  - "30033"
  - "30034"
  - "30035"
  - "30036"
  - "30037"
  - "30038"
  - "30039"
  - "30040"
  - "30041"
  - "30042"
  - "30043"
  - "30044"
  - "30045"
restart: always

volumes:
  ftp-db-volume:
    external:
      name: ftp-db-volume

```

```

ftp:
  driver: bridge
ipam:
  driver: default
  config:
    - subnet: 172.16.1.0/24

```

2.4 Instalasi HAProxy

HAProxy (High Availability Proxy) merupakan aplikasi yang digunakan sebagai load balancer dan proxy solution yang bisa dijalankan pada system operasi linux.

Pada penelitian ini HAProxy digunakan sebagai pembagi beban kinerja kedalam 3 container FTP. Untuk dapat menggunakan HAProxy terlebih dahulu melakukan instalasi dengan menggunakan perintah berikut.

```
$ sudo apt-get install haproxy
```

Setelah selesai melakukan instalasi selanjutnya melakukan konfigurasi HAProxy agar dapat melakukan pembagian kinerja kedalam container FTP dengan mengkonfigurasi file haproxy.cfg. Berikut isi dari haproxy.cfg.

```

defaults
    log global
    mode tcp
    option tcplog
    timeout connect 5000
    timeout client 5000
    timeout server 5000

frontend haproxy
    bind *:21
    default_backend ftp_server_pool

frontend ftpd_server1
    bind *:30000-30015
    default_backend ftpd_server1

frontend ftpd_server2
    bind *:30016-30030
    default_backend ftpd_server2

frontend ftpd_server3
    bind *:30031-30045
    default_backend ftpd_server3

backend ftp_server_pool
    balance leastconn

    server ftpd_server1 172.16.1.11 check port 21
inter 10s rise 1 fall 2
    server ftpd_server2 172.16.1.12 check port 21
inter 10s rise 1 fall 2
    server ftpd_server3 172.16.1.13 check port 21
inter 10s rise 1 fall 2

backend ftpd_server1
    server ftpd_server1 172.16.1.11 check port 21
inter 10s rise 1 fall 2

backend ftpd_server2

```

```
server ftpd_server2 172.16.1.12 check port 21
inter 10s rise 1 fall 2
```

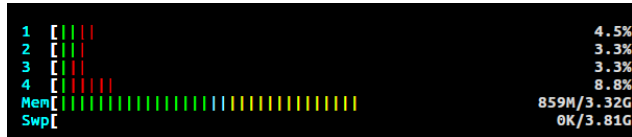
```
backend ftpd_server3
```

```
server ftpd_server3 172.16.1.13 check port 21
inter 10s rise 1 fall 2
```

2.5 Pengujian dan Hasil

Pada pengujian ini mengacu kepada skenario pengujian yang telah dirancang.

a) Pengujian Skenario satu



Gambar 10. Uji Performance prosesor dan memory pada server FTP

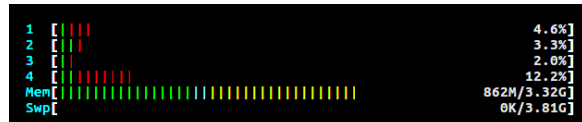
CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %
6cace20057a9	ftpd_server1	3.27%	6.656MiB / 3.323GiB	0.20%
6c4e9fa6e43e	ftpd_server2	0.00%	2.984MiB / 3.323GiB	0.09%
29ce25f5d08f	ftpd_server3	0.00%	2.738MiB / 3.323GiB	0.08%

Gambar 11. Monitoring Kinerja setiap Container

Tabel 1. Hasil Pengamatan Skenari satu

Htop	Memory usage	Prosesor 1	Prosesor 2	Prosesor 3	Prosesor 4
1 User	859 MB	4.5%	3.3%	3.3%	8.8%
Container Stats	Memory Usage		CPU Usage		
Container FTP1	6.7 MB		3.3%		
Container FTP2	2.9 MB		0%		
Container FTP3	2.7 MB		0%		

b) Pengujian Skenario dua



Gambar 12. Uji Performance dan memori 1 Container 3 user pada server FTP

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %
6cace20057a9	ftpd_server1	5.14%	10.48MiB / 3.323GiB	0.31%
6c4e9fa6e43e	ftpd_server2	0.00%	3MiB / 3.323GiB	0.09%
29ce25f5d08f	ftpd_server3	0.00%	2.738MiB / 3.323GiB	0.08%

Gambar 13. Monitoring Kinerja pada setiap Container

Tabel 2. Hasil Pengamatan Skenario dua

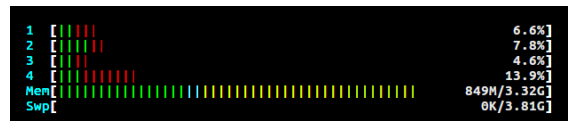
Htop	Memory usage	Prosesor 1	Prosesor 2	Prosesor 3	Prosesor 4
3 User	862 MB	4.6%	3.3%	2.0%	12.2%
Container Stats	Memory Usage		CPU Usage		
Container FTP1	10.5 MB		5.1%		
Container FTP2	3.0 MB		0%		
Container FTP3	2.7 MB		0%		

c) Pengujian Skenario tiga

```
haproxy[6834]: 192.168.100.5:52477 [05/Mar/2018:21:51:45.245] ftpd_server1 ftpd_server1/ftpd_server1 1/
haproxy[6834]: 192.168.100.5:52476 [05/Mar/2018:21:51:44.842] haproxy ftp_server_pool/ftpd_server1 1/0
haproxy[6834]: 192.168.100.5:52480 [05/Mar/2018:21:52:50.998] haproxy ftp_server_pool/ftpd_server2 1/0
```

Gambar 14. Tampilan proses penanganan pada HAProxy

d) Pengujian Skenario empat



Gambar 15. Pengujian performance memory dan prosesor pada server FTP menggunakan load balancing

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %
6cace20057a9	ftpd_server1	1.03%	4.141MiB / 3.323GiB	0.12%
6c4e9fa6e43e	ftpd_server2	2.55%	3.305MiB / 3.323GiB	0.10%
29ce25f5d08f	ftpd_server3	2.05%	4.172MiB / 3.323GiB	0.12%

Gambar 16. Monitoring Kinerja setiap Container

```
haproxy[1346]: 192.168.100.7:51981 [07/Mar/2018:00:52:55.850] ftpd_server1 ftpd_server1/ftpd_server1 1/
haproxy[1346]: 192.168.100.10:50103 [07/Mar/2018:00:53:00.029] ftpd_server2 ftpd_server2/ftpd_server2 1/
haproxy[1346]: 192.168.100.7:51985 [07/Mar/2018:00:53:04.834] ftpd_server1 ftpd_server1/ftpd_server1 1/
haproxy[1346]: 192.168.100.8:61559 [07/Mar/2018:00:52:55.505] haproxy ftp_server_pool/ftpd_server3 1/0
haproxy[1346]: 192.168.100.7:51984 [07/Mar/2018:00:53:04.179] ftpd_server2 ftpd_server2/ftpd_server2 1/
haproxy[1346]: 192.168.100.8:61557 [07/Mar/2018:00:52:52.793] haproxy ftp_server_pool/ftpd_server3 1/0
haproxy[1346]: 192.168.100.8:61503 [07/Mar/2018:00:53:00.494] ftpd_server3 ftpd_server3/ftpd_server3 1/
haproxy[1346]: 192.168.100.8:61502 [07/Mar/2018:00:53:00.250] ftpd_server3 ftpd_server3/ftpd_server3 1/
haproxy[1346]: 192.168.100.10:50101 [07/Mar/2018:00:52:58.517] haproxy ftp_server_pool/ftpd_server2 1/0
haproxy[1346]: 192.168.100.10:50104 [07/Mar/2018:00:52:01.082] ftpd_server2 ftpd_server2/ftpd_server2 1/
```

Gambar 17. Hasil Logs pada HAProxy

Tabel 3. Hasil Pengamatan Skenario empat

Htop	Memory usage	Prosesor 1	Prosesor 2	Prosesor 3	Prosesor 4
Balancing 3 User	849 MB	6.6%	7.8%	4.6%	13.9%
Container Stats	Memory Usage		CPU Usage		
Container FTP1	4.1 MB		1.0%		
Container FTP2	3.3 MB		2.5%		
Container FTP3	4.1 MB		2.0%		

e) Pengujian Respon Time Least Connection

Tabel 4. Hasil Pengujian Respon Time Least Connection

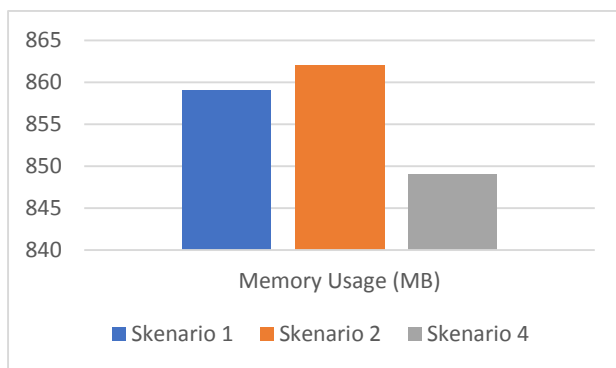
Percobaan	Total Waktu			
	S1	S2	S3	Total
Video 1	68309	52027	35354	155689
Video 2	0	52030	75425	127454
Video 3	55945	52022	70262	178228
RAR 1	42792	52023	52691	147505
RAR 2	54361	52048	45823	152232
RAR 3	52026	47316	54520	153861
DOC 1	2931	0	6005	8936
DOC 2	5334	54063	5356	64753
DOC 3	2071	3226	0	5297
PDF1	0	338	337	675
PDF2	54066	339	340	54745
PDF3	343	54078	344	54765

- f) Pengujian Respon Time Round Robin
Tabel 5. Hasil Pengujian Respon Time Round Robin

Percobaan	Total Waktu			
	S1	S2	S3	Total
Video 1	64237	0	0	64237
Video 2	0	45582	63690	109272
Video 3	47316	54520	153861	453598
RAR 1	0	52017	39432	91449
RAR 2	45776	52017	53976	151768
RAR 3	52020	45546	54017	151582
DOC 1	2078	0	54039	56117
DOC 2	2107	0	54290	56397
DOC 3	10938	0	0	10938
PDF1	0	183	186	369
PDF2	56644	124	124	56892
PDF3	0	121	121	243

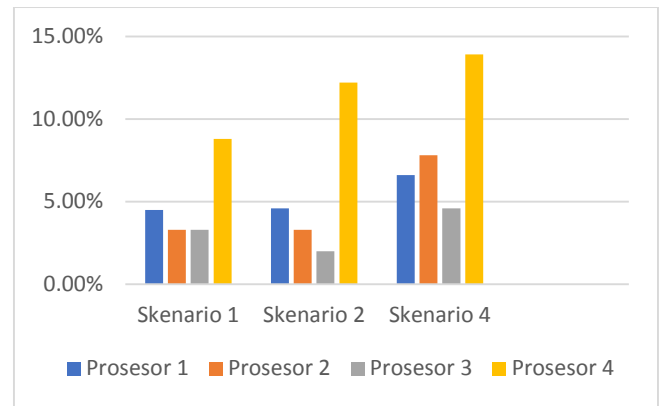
- g) Grafik Hasil

Pada grafik hasil akan menampilkan hasil perbandingan kinerja berdasarkan data yang telah ada pada skenario 1, skenario 2, dan skenario 4 pada penggunaan memory usage dan prosesor usage di FTP server.



Gambar 18. Perbandingan hasil Penggunaan Memory Usage pada sisi FTP server

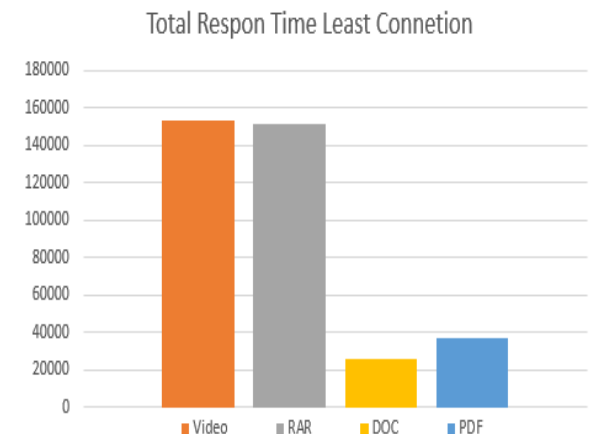
Berdasarkan gambar 18, terjadinya peningkatan penggunaan memory dari skenario 1 ke skenario 2 dan terdapat penurunan penggunaan memory pada skenario 4. Dimana pada skenario 4 tersebut telah menggunakan load balancing, sehingga pada penggunaan memory terjadi penurunan.



Gambar 19. Perbandingan Hasil penggunaan Prosesor Usage pada sisi FTP server

Berdasarkan gambar 19, pada skenario 1 dan skenario 2 pemanfaatan penggunaan prosesor hanya terdapat pada prosesor 4 saja, dimana pada grafik tersebut terlihat peningkatan yang signifikan pada prosesor 4. Sedangkan pada skenario 4 pemanfaatan penggunaan prosesor terjadi pada keseluruhan prosesor yang ada, sehingga pemanfaatan penggunaan prosesor jadi lebih optimal yang mana akan berpengaruh pada penggunaan memory seperti pada gambar 18 sebelumnya.

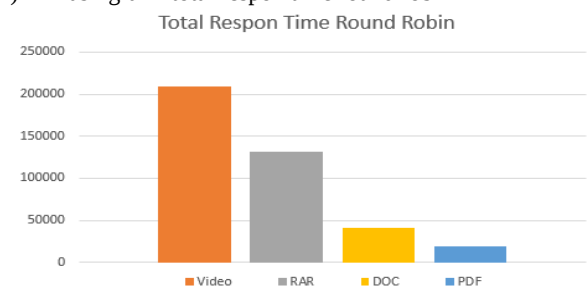
- h) Hasil grafik total respon time least connection



Gambar 20. Perbandingan Hasil pengukuran Total Least Connection Respon Time FTP server

Berdasarkan gambar 20, rata waktu pada video di dapatkan 153791 ms, RAR 151199, Dokumen 26329 dan PDF 36728. Dengan masing-masing jumlah file rata-rata 1GB

- i) Hasil grafik total respon time round robin



Gambar 21. Perbandingan Hasil pengukuran Total Round Robin Respon Time FTP server

III. KESIMPULAN

Berdasarkan hasil yang diperoleh melalui pengamatan dan ujicoba, maka dapat ditarik kesimpulan sebagai berikut:

- 1) Penerapan server FTP pada Universitas Muhammadiyah Riau program studi Teknik Informatika membantu dalam memenuhi kebutuhan Dosen akan penyimpanan data yang terpusat, dikarenakan masing-masing dosen memiliki akses kedalam sesuai dengan akun masing-masing tanpa tercapur dengan yang lain.
- 2) Penerapan load balancing dapat mengalihkan service yang mati ke service yang lain, sehingga proses pengiriman data tidak terganggu dikarenakan service yang mati.
- 3) Penerapan multi container dan balancing pada server FTP membantu mengoptimalkan kinerja pada server FTP dalam penanganan request atau beban yang banyak dan mampu membagi kerja pada setiap container untuk menjaga kestabilan memory dan prosesor yang tetap membuat sistem berjalan baik.

DAFTAR PUSTAKA

- [1] Y. L. Oktavianus, "Membangun Sistem Cloud Computing dengan Implementasi Load Balancing dan Pengujian Algoritma Penjadwalan Linux Virtual Server pada FTP Server," *J. Nas. Tek. Elektro*, vol. 2, no. 1, pp. 25–30, 2013.
- [2] M. A. Nugroho and R. Kartadie, "Analisis Kinerja Penerapan Container untuk Load Balancing Web Server," *JUPI (Jurnal Ilm. Penelit. dan Pembelajaran Inform.)*, vol. 1, no. 02, pp. 7–15, 2016.
- [3] M. Jamil, A. Khairan, and A. Fuad, "Implementasi Aplikasi Telemedicine Berbasis Jejaring Sosial dengan Pemanfaatan Teknologi Cloud Computing," *J. Edukasi dan Penelit. Inform.*, vol. 1, no. 1, 2017.
- [4] E. Kurniawan, "PENERAPAN TEKNOLOGI CLOUD COMPUTING DI UNIVERSITAS Studi Kasus : Fakultas Teknologi Informasi UKDW," *Eksis*, vol. 08, no. 01, pp. 29–36, 2015.
- [5] setiawan herri ashari ahmad, "Cloud Computing : Solusi ICT," *Sist. Inf.*, vol. 29, no. 6, pp. 1–5, 2009.
- [6] M. A. F. Ridha, "Implementasi Cloud Computing Menggunakan Openvz dalam Perkuliahan Praktikum Sistem Operasi," *J. Nas. Tek. Elektro*, vol. 5, no. 1, p. 145, 2018.
- [7] K. Devendra, K. Swati, K. Shruti, and J. Shreejeet, "HPC Cloud Burst Using Docker," *Int. Res. J. Eng. Technol.*, vol. 4, no. 3, pp. 1378–1382, 2017.
- [8] A. Kurniawan, H. N. Palit, and J. Adjarwirawan, "Eksplorasi Pemanfaatan Docker untuk Mempermudah Pengelolaan Instalasi Komputer di Laboratorium Komputer Teknik Informatika Universitas Kristen Petra," pp. 2–7, 2016.
- [9] C. M. Babu and O. L. Chandana, "File Transfer Protocol in Cloud Computing," vol. 2, no. March, pp. 665–667, 2014.
- [10] S. D. Riskiono, S. Sulistyono, and T. B. Adji, "Kinerja Metode Load Balancing dan Fault Tolerance Pada Server Aplikasi Chat," *Pros. Semin. Nas. ReTII*, 2017.
- [11] D. Wadhwa and N. Kumar, "Performance Analysis of Load Balancing Algorithms in," vol. 4, no. 1, pp. 59–66, 2014.