# RANCANG BANGUN CLUSTERING SYSTEM PADA WEB SERVER MENGGUNAKAN LOAD BALANCING BERBASIS LINUX

## SKRIPSI


Disusun Oleh :

**RIYAN ANTONO**

**NIM.0812503**





**PROGRAM STUDI TEKNIK ELEKTRO S-1**

**KONSENTRASI TEKNIK KOMPUTER**

**FAKULTAS TEKNOLOGI INDUSTRI**

**INSTITUT TEKNOLOGI NASIONAL MALANG**


**2012**

**LEMBAR PERSETUJUAN**

# RANCANG BANGUN CLUSTERING SYSTEM PADA WEB SERVER MENGGUNAKANA LOAD BALANCING BERBASIS LINUX

**SKRIPSI**

*Disusun dan Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Teknik Strata Satu (S-1)*

**Disusun oleh :**


**RIYAN ANTONO**

**0812503**


**Mengetahui,**

**Ketua Program Studi Teknik Elektro S-1**

**Ir. Yusuf Ismail Nakhoda, MT**

**NIP.Y. 1018800189**

**Diperiksa dan disetujui,**

| Dosen Pembimbing I | Dosen Pembimbing II |
|---|---|
| **Bambang Prio Hartono, ST, MT** | **Bima Aulia Firmandani, ST** |
| **NIP.Y. 1028400082** | **1121** |

**PROGRAM STUDI TEKNIK ELEKTRO S-1**

**KONSENTRASI TEKNIK KOMPUTER**

**FAKULTAS TEKNOLOGI INDUSTRI**

**INSTITUT TEKNOLOGI NASIONAL MALANG**

**2012**

# SURAT PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini :

Nama : Riyan Antono

NIM : 08.12.503

Program Studi : Teknik Elektro S-1

Konsentrasi : Teknik Komputer

Dengan ini menyatakan bahwa Skripsi yang saya buat adalah hasil karya sendiri, tidak merupakan plagiasi dari karya orang lain. Dalam Skripsi ini tidak memuat karya orang lain, kecuali dicantumkan sumbernya sesuai dengan ketentuan yang berlaku.

Demikian surat pernyataan ini saya buat, dan apabila di kemudian hari ada pelanggaran atas surat pernyataan ini, saya bersedia menerima sangsinya.

Malang, 26 September 2012

Yang membuat Pernyataan,


**Riyan Antono**
**08.12.503**

# ABSTRAK

## RANCANG BANGUN CLUSTERING SYSTEM PADA WEB SERVER MENGGUNAKAN LOAD BALANCING BERBASIS LNUX

**Riyan Antono, NIM 0812503**
**Dosen Pembimbing : Bambang Prio Hartono, ST, MT dan**
**Bima Aulia Firmandani, ST**

Dengan perkembangan internet yang begitu cepat, juga disertai banyaknya informasi-informasi mengenai berbagai macam hal di dunia ini. Salah satu cara untuk mendapatkan informasi secara cepat yaitu dengan mengunjungi situs web. Jika suatu situs terlalu banyak pengunjung, maka jika server web yang digunakan adalah server tunggal, dapat memungkinkan kegagalan dalam menampilkan informasi pada pengunjung, yang berarti situs tersebut mengalami penurunan pelayanan. Arsitektur cluster yang diterapkan sebagai server dengan performa tinggi adalah salah satu solusi yang efektif dan efisien untuk mengatasi masalah tersebut.

Teknik Load Balancing memungkinkan proses pengolahan data dibagi secara terdistribusi ke beberapa komputer server. Dengan menggunakan algoritma penjadwalan Round Robin, maka pendistribusian permintaan web yang dilakukan load balancer akan diteruskan ke beberapa web server secara bergantian. Dengan teknik ini, jika terdapat kegagalan pada suatu web server maka tidak akan berpengaruh terhadap client dalam mengakses web. Karena load balancing hanya akan meneruskan permintaan kepada server yang aktif.

Pada pengujian load balancing web server ini, dapat terlihat bahwa request yang masuk dapat diteruskan dan dibagi bebannya ke web server yang ada. Dan saat salah satu server terdapat kegagalan, load balancing tetap akan meneruskan request ke server yang aktif. Sehingga sistem load balancing web server ini dapat meningkatkan penyediaan informasi suatu web.

Kata Kunci:server, web,load balancing.

## *ABSTRACT*

*With the rapid development of the internet, also with much information about various things in this world. One way to get information quickly is by visiting the website. If a site is too many visitors, so if the web server being used is a single server, could allow failure in presenting information to visitors, which means the site is experiencing deterioration in service. Cluster architecture is implemented as a server with high performance is one of effective and efficient solutions to resolve the issue.*

*Load Balancing technique allows processes in a distributed data processing is divided into several server computers. By using the Round Robin scheduling algorithm, then the distribution of web requests that do load balancer will be forwarded to a web server in turn. With this technique, if there is a failure on a web server then it will not affect the client in accessing the web. Because the load balancing will only forward the request to the server is active.*

*On the web server load balancing test, it can be seen that the incoming request can be forwarded and shared his burden to an existing web server. And when there is a failure of one of the servers, load balancing will still forward the request to the server is active. So the web server load balancing system can improve the provision of information on the web*



*Keywords: server, web,load balancing.*

# PENGANTAR

Puji syukur ke hadirat Tuhan Yang Maha Esa atas segala limpahan rahmat-Nya sehingga penelitian berjudul Rancang Bangun Clustering System Pada Web Server Menggunakan Load Balancing Berbasis Linux dapat terselesaikan.

Penelitian ini dibuat untuk memenuhi salah satu syarat dalam memperoleh gelar sarjana teknik. Ucapan terima kasih yang sebesar-besarnya kami sampaikan pada :

1. Bapak Ir. Soeparno Jiwo, MT selaku Rektor ITN Malang.
2. Bapak Ir. H. Sidik Noertjahjono,MT selaku Dekan Fakultas Teknologi Industri ITN Malang.
3. Bapak Ir. Yusuf Ismail Nakhoda, MT selaku Ketua Program Studi Teknik Elektro S-1 ITN Malang
4. Bapak Bambang Prio Hartono, ST, MT selaku Dosen Pembimbing I.
5. Bapak Bima Aulia Firmandani, ST selaku Dosen Pembimbing II.
6. Kedua orang tua dan saudara yang telah memberi motivasi dalam penyelesaian skripsi ini.
7. Rekan-rekan Mahasiswa Elektro S-1 Angkatan 2008.
8. Rekan-rekan Asisten Laboratorium Jaringan Komputer dan Cisco ITN Malang.
9. Semua Pihak yang telah membantu dalam penulisan dan penyusunan penelitian ini.

Penulis menyadari bahwa penelitian ini masih jauh dari sempurna, untuk itu kritik dan saran dari pembaca sangat penulis harapkan untuk perbaikan penelitian ini.

Malang, Agustus 2012



Penulis

# DAFTAR ISI

# Daftar Gambar

# Daftar Tabel

# BAB I

# PENDAHULUAN

## 1.1 Latar Belakang Masalah

Tingginya akses internet melalui *web* membuat para penyedia layanan *web* harus menyediakan layanan *web* yang mempunyai kinerja tinggi. Hal ini dikarenakan tingginya akses tentu akan menyebabkan penurunan kinerja *web server* tunggal. Salah satu teknologi yang yang dapat diakukan untuk meningkatkan kemampuan *server* adalah sistem *clustering web server*. Sistem *clustering* memungkinkan peningkatan kerja dari sisi *stability* dan *reliability* sebuah *web server*. Konsep dasar dari *clustering web server* sendiri adalah menggabungkan beberapa *web server* ke dalam sistem yang bekerja secara bersama-sama dan simultan untuk menangani *request* yang tinggi.

Kehandalan dari sistem *custering web server* membuat para penyedia layanan *web* dapat memberikan layanan yang maksimal. Sebagaimana diketahui akses internet diharuskan dapat melayani secara *realtime* dan mampu bekerja aktif. Adanya sistem *clustering web server* yang dapat meningkatkan kinerja tentunya juga meningkatkan *addvalue* pada *web* itu sendiri.

Dalam membangun suatu sistem *cluster* yang handal, kita dapat memanfaatkan teknologi *Load Balancing*. *Load balancing* adalah proses pendistribusian beban terhadap sebuah servis yang ada pada sekumpulan *server* atau perangkat ketika terdapat permintaan dari pemakai. Ketika pemintaan pemakai semakin besar maka *server* tersebut akan terbebani karena harus melakukan proses pelayanan terhadap permintaan pemakai. Disinilah *load balancing* berperan, yaitu dengan membagi-bagi beban yang datang ke beberapa *server*, agar *server* tersebut memiliki *stability* dan *reliability* yang tinggi.

## 1.2 Rumusan Masalah

Permasalahan yang dihadapi dalam skripsi ini adalah :

1. Bagaimana merancang dan membangun *clustering web server* dengan menggunakan sistem *load balancing*.
2. Bagaimana mengimplementasikan algoritma Round Robin pada sistem load balancing.

## 1.3 Tujuan

Tujuan dari tugas akhir ini adalah :

1. Dapat merancang dan membangun sistem clustering web server dengan menggunakan sistem load balancing.
2. Mengimplementasikan algoritma Round Robin pada sitem Load balancing web server.

## 1.4 Batasan Masalah

Batasan masalah pada tugas akhir ini adalah sebagai berikut :

1. Sistem *clustering web server* berjalan pada sistem operasi LINUX.
2. *Load Balancing* menggunakan Piranha sebagai pembagi *traffic http* antara *client* menuju *web server.*
3. Tidak membahas tentang sistem keamanan pada *server* dan *client.*
4. Algoritma penjadwalan yang digunakan pada *Load Balancer* adalah algoritma Round Robin.

## 1.5 Metodelogi Penelitian

Adapun metode penelitian yang digunakan adalah sebagai berikut:

1. Studi literatur

   Pada tahap ini, merupakan tahap pengumpulan informasi yang diperlukan untuk perancangan sistem. Informasi tersebut diperoleh dengan membaca literatur ataupun jurnal-jurnal yang berhubungan dengan *web server load balancing*, seperti dasar-dasar teori, dokumentasi penggunaan *tool*, dan jurnal teknologi.
2. Perancangan dan Implementasi

   Pada tahap ini, dilakukan analisis kebutuhan dan perancangan sistem. Dengan harapan dapat membuat solusi yang tepat untuk merancang sistem serta kemungkinan yang dapat dilakukan untuk mengimplementasikan rancangan tersebut.
3. Eksperimen dan Evaluasi

   Pada tahap ini, model dan rancangan sistem yang telah dibuat akan diuji coba, yaitu pengujian berdasarkan kinerja, dan akan dilakukan perbaikan-perbaikan apabila diperlukan.

## 1.6 Sistematika Pembahasan

Pada penulisan skripsi ini terdiri atas lima pembahasan yaitu :

**BAB I : PENDAHULUAN**

Bab ini berisi tentang latar belakang, perumusan masalah, batasan masalah, tujuan serta metode penelitian dan sistematika penulisan.

**BAB II : TINJAUAN PUSTAKA**

Bab ini berisi tentang teori penunjang dan piranti yang dipergunakan dan berhubungan dengan sistem yang direncanakan.

**BAB III : ANALISA DAN DPERANCANGAN SISTEM**

Bab ini berisi mengenai perancangan sistem dengan meliputi *hardware* dan *software* yang diperlukan.

**BAB IV : IMPLEMENTASI DAN PENGUJIAN SISTEM**

Bab ini berisi tentang pengujian terhadap sistem pada saat implemetasi beserta analisis terhadap hasil yang didapat.

**BAB V : PENUTUP**

Bab ini berisi kesimpulan dan saran dari penulisan skripsi.

# BAB II

# TINJAUAN PUSTAKA

## 2.1 Topologi Jaringan

Topologi jaringan komputer adalah suatu cara menghubungkan komputer yang satu dengan komputer lainnya sehingga membentuk jaringan. Dalan suatu jaringan komputer jenis topologi yang dipilih akan mempengaruhi kecepatan komunikasi. Untuk itu maka perlu dicermati kelebihan / keuntungan dan kekurangan / kerugian dari masing – masing topologi berdasarkan kateristiknya.

Topologi pada dasarnya adalah peta dari sebuah jaringan. Topologi jaringan terbagi lagi menjadi dua yaitu topologi secara fisik (*physical topology*) dan topologi secara logika (*logical topology*). Topologi secara fisik menjelaskan bagaimana susunan dari label, komputer dan lokasi dari semua komponen jaringan. Sedangkan topologi secara logika menetapkan bagaimana informasi atau aliran data dalam jaringan.

Arsitektur topologi merupakan bentuk koneksi fisik untuk menghubungkan setiap *node* pada sebuah jaringan. Pada sistem LAN terdapat tiga topologi utama yang paling sering digunakan, yaitu : *Bus, Star*, dan *Ring*. Topologi jaringan ini kemudian berkembang menjadi Topologi *Tree* dan *Mesh* yang merupakan kombinasi dari *Star, Mesh*, dan *Bus*. Berikut jenis-jenis topologi Topologi (http://id.wikipedia.org/wiki/Topologi_jaringan):

1) Topologi *Bus*
2) Topologi *Ring* (Cincin)
3) Topologi *Star* (Bintang)
4) Topologi *Tree* (Pohon)
5) Topologi *Mesh* (Tak Beraturan)

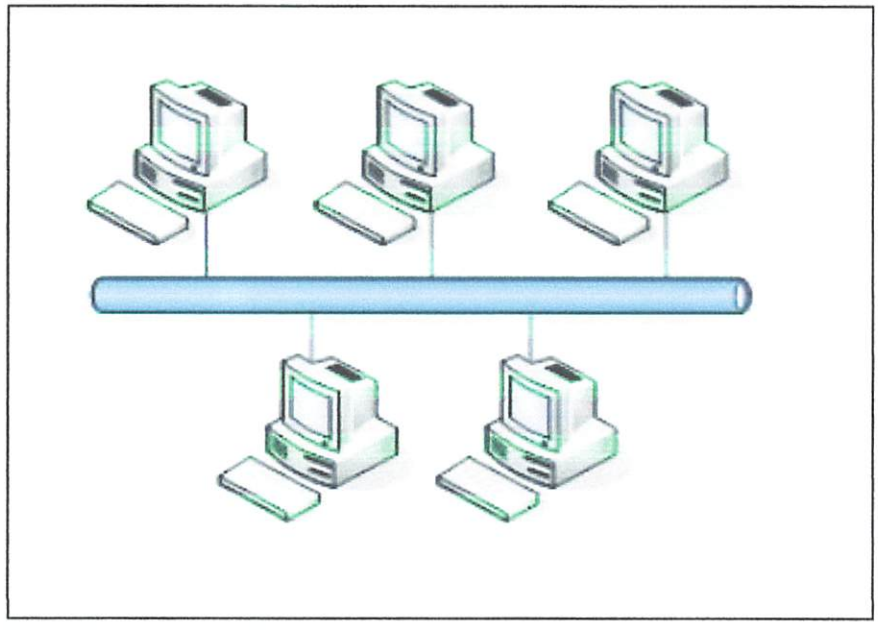
### 2.1.1 Topologi *Bus*

Topologi *bus* merupakan topologi yang banyak digunakan pada masa penggunaan kabel sepaksi menjamur. Dengan menggunakan *T-Connector* (dengan terminator 50 ohm pada ujung *network*), maka komputer atau perangkat jaringan lainnya bisa dengan mudah dihubungkan satu sama lain.

Kesulitan utama dari penggunaan kabel sepaksi adalah sulit untuk mengukur apakah kabel sepaksi yang digunakan benar-benar matching atau tidak. Karena kalau tidak sungguh-sungguh diukur secara benar akan merusak *NIC* (*network interface card*) yang digunakan dan kinerja jaringan menjadi terhambat, tidak mencapai kemampuan maksimalnya. Topologi ini juga sering digunakan pada jaringan dengan basis *fiber optic* (yang kemudian digabungkan dengan topologi *star* untuk menghubungkan dengan *client* atau *node*).

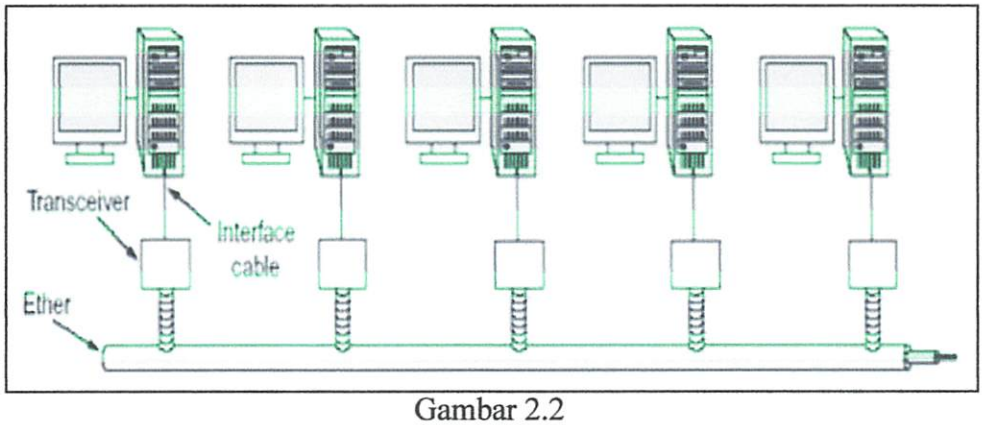
Pada topologi *bus* dua ujung jaringan harus diakhiri dengan sebuah *terminator*. *Barel connector* dapat digunakan untuk memperluasnya. Jaringan hanya terdiri dari satu saluran kabel yang menggunakan kabel *BNC*. Komputer yang ingin terhubung ke jaringan dapat mengkaitkan dirinya dengan men-tap *Ethernetnya* sepanjang kabel.

Instalasi jaringan *Bus* sangat sederhana, murah dan maksimal terdiri atas 5-7 komputer. Kesulitan yang sering dihadapi adalah kemungkinan terjadinya tabrakan data karena mekanisme jaringan relatif sederhana dan jika salah satu *node* putus maka akan mengganggu kinerja dan trafik seluruh jaringan.



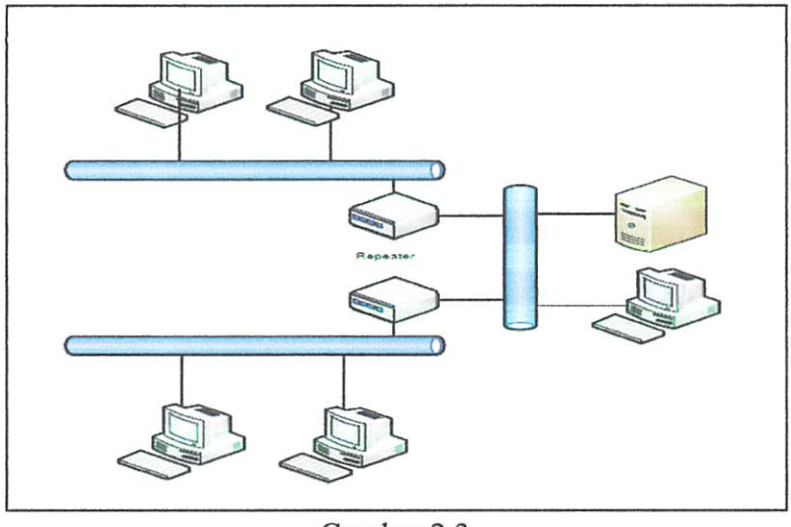
Gambar 2.1
Topologi *Bus*

- ✓ Pada titik tertentu diadakan sambungan (tap) untuk setiap terminal.
- ✓ Wujud dari tap ini bisa berupa kabel *transceiver* bila digunakan *thick coax* sebagai media transmisi.
- ✓ Atau berupa *BNC T-connector* bila digunakan *thin coax* sebagai media transmisi.
- ✓ Atau berupa konektor RJ-45 dan *Hub* bila digunakan kabel *UTP.*
- ✓ Transmisi data dalam kabel bersifat *full duplex*, dan sifatnya *broadcast,* semua terminal bisa menerima transmisi data.




Gambar 2.2
Koneksi Kabel-*Transceiver* Pada Topologi *Bus*

- ✓ Suatu *protocol* akan mengatur transmisi dan penerimaan data, yaitu *Protocol Ethernet* atau *CSMA/CD.*
- ✓ Melihat bahwa pada setiap segmen (bentang) kabel ada batasnya maka diperlukan *"Repeater"* untuk menyambungkan segmen-segmen kabel.




Gambar 2.3
Perluasan Topologi *Bus* Menggunakan *Repeater*

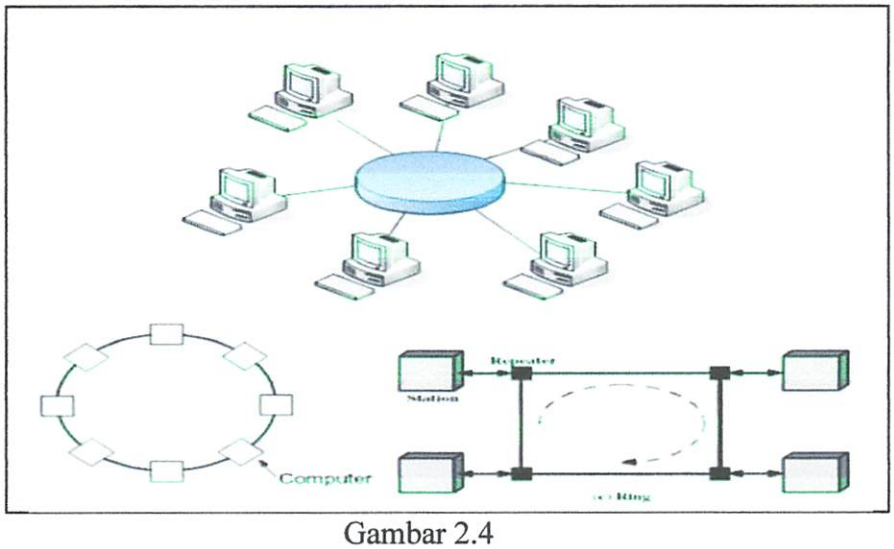
Table 2.1
Kelebihan dan Kekurangan Topologi *Bus*

| | |
|---|---|
| **Kelebihan Topologi Bus** | ✓ Instalasi relatif lebih murah.<br>✓ Kerusakan satu komputer *client* tidak akan mempengaruhi komunikasi antar *client* lainnya. |
| **Kelemahan Topologi Bus** | ✓ Jika kabel utama (bus) atau *backbone* putus maka komunikasi gagal.<br>✓ Bila kabel utama sangat panjang maka pencarian gangguan menjadi sulit. |

### 2.1.2 Topologi *Ring* (cincin)

Topologi *ring* biasa juga disebut sebagai topologi cincin karena bentuknya seperti cincin yang melingkar. Semua komputer dalam jaringan akan di hubungkan pada sebuah cincin. Cincin ini hampir sama fungsinya dengan *concentrator* pada topologi *star* yang menjadi pusat berkumpulnya ujung kabel dari setiap komputer yang terhubung.

Secara lebih sederhana lagi topologi cincin merupakan untaian media transmisi dari satu terminal ke terminal lainnya hingga membentuk suatu lingkaran, dimana jalur transmisi hanya "satu arah". Tiga fungsi yang diperlukan dalam topologi cincin : penyelipan data, penerimaan data, dan pemindahan data.




Gambar 2.4
Prinsip Koneksi Topologi *Ring*

- ✓ Penyelipan data adalah proses dimana data dimasukkan kedalam saluran transmisi oleh terminal pengirim setelah diberi alamat dan *bit-bit* tambahan lainnya.
- ✓ Penerimaan data adalah proses ketika terminal yang dituju telah mengambil data dari saluran, yaitu dengan cara membandingkan alamat yang ada pada paket data dengan alamat terminal itu sendiri. Apabila alamat tersebut sama maka data kiriman disalin.
- ✓ Pemindahan data adalah proses dimana kiriman data diambil kembali oleh terminal pengirim karena tidak ada terminal yang menerimanya (mungkin akibat salah alamat). Jika data tidak diambil kembali maka data ini akan berputar-putar dalama saluran. Pada jaringan bus hal ini tidak akan terjadi karena kiriman akan diserap oleh "*terminator*".
- ✓ Pada hakekatnya setiap terminal dalam jaringan cincin adalah "*repeater*", dan mampu melakukan ketiga fungsi dari topologi cincin.
- ✓ Sistem yang mengatur bagaimana komunikasi data berlangsung pada jaringan cincin sering disebut *token-ring*.

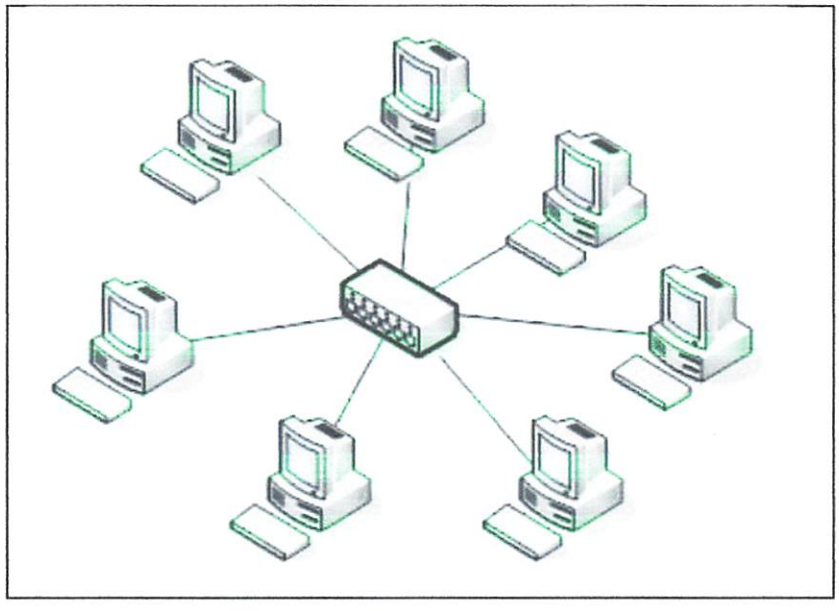
Table 2.2
Kelebihan dan Kelemahan Topologi *Ring*

| | |
|---|---|
| **Kelebihan Topologi *Ring*** | ✓ Hemat kabel.<br>✓ Tidak akan terjadi tabrakan pengiriman data (*collision*), karena pada satu waktu hanya satu *node* yang dapat mengirimkan data. |
| **Kelemahan Topologi *Ring*** | ✓ Peka kesalahan, sehingga jika terdapat gangguan di suatu *node* mengakibatkan terganggunya seluruh jaringan.<br>✓ Pengembangan jaringan lebih kaku.<br>✓ Sulit mendeteksi kerusakan.<br>✓ Dapat terjadi *collision* (dua paket data tercampur).<br>✓ Diperlukan penanganan dan pengelolaan khusus. |

### 2.1.3 Topologi *Start* (Bintang)

Disebut topologi *star* karena bentuknya seperti bintang, sebuah alat yang disebut *concentrator* bisa berupa hub atau switch menjadi pusat, dimana semua komputer dalam jaringan dihubungkan ke *concentrator* ini.

- ✓ Pada topologi Bintang *(Star)* sebuah terminal pusat bertindak sebagai pengatur dan pengendali semua komunikasi yang terjadi. Terminal-terminal lainnya melalukan komunikasi melalui terminal pusat ini.
- ✓ Terminal kontrol pusat bisa berupa sebuah komputer yang difungsikan sebagai pengendali tetapi bisa juga berupa "*HUB*" atau *"MAU" (Multi Accsess Unit).*



Gambar 2.5
Prinsip Kerja Topologi *Star*

- ✓ Terdapat dua alternatif untuk operasi simpul pusat.

  a. Simpul pusat beroperasi secara "*broadcast*" yang menyalurkan data ke seluruh arah. Pada operasi ini walaupun secara fisik kelihatan sebagai bintang namun secara logik sebenarnya beroperasi seperti *bus*. Alternatif ini menggunakan *HUB*.

  b. Simpul pusat beroperasi sebagai "*switch*", data kiriman diterima oleh simpul kemudian dikirim hanya ke terminal tujuan (bersifat

*point-to-point*), akternatif ini menggunakan *MAU* sebagai pengendali.

- ✓ Bila menggunakan HUB maka secara fisik sebenarnya jaringan berbentuk topologi Bintang namun secara logis bertopologi Bus. Bila menggunakan MAU maka baik fisik maupun logis bertopologi Bintang.

Table 2.3
Kelebihan dan kelemahan Topologi *Star*

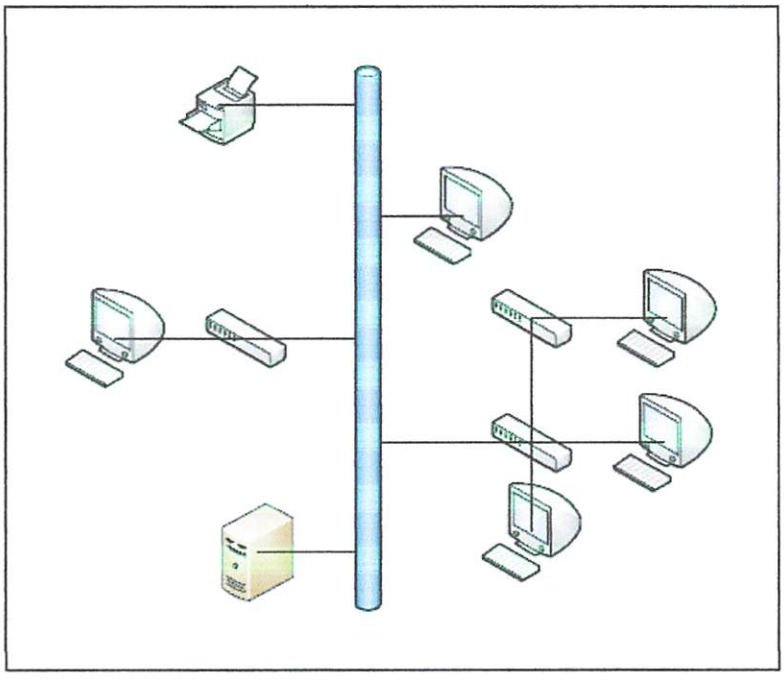
| | |
|---|---|
| **Kelebihan Topologi *Star*** | ✓ Karena setiap komponen dihubungkan langsung ke simpul pusat maka pengelolaan menjadi mudah, kegagalan komunikasi mudah ditelusuri.<br>✓ Kegagalan pada satu komponen /terminal tidak mempengaruhi komunikasi terminal lain. |
| **Kelemahan Topologi *Star*** | ✓ Kegagalan pusat kontrol (simpul pusat) memutuskan semua komunikasi<br>✓ Bila yang digunakan sebagai pusat kontrol adalah *HUB* maka kecepatan akan berkurang sesuai dengan penambahan komputer, semakin banyak semakin lambat. |

### 2.1.4 Topologi *Tree* (Pohon)

Topologi Pohon adalah kombinasi karakteristik antara topologi bintang dan topologi *bus*. Topologi ini terdiri atas kumpulan topologi bintang yang dihubungkan dalam satu topologi *bus* sebagai jalur tulang punggung atau *backbone*. Komputer-komputer dihubungkan ke *hub*, sedangkan *hub* lain di hubungkan sebagai jalur tulang punggung.

Topologi jaringan ini disebut juga sebagai topologi jaringan bertingkat. Topologi ini biasanya digunakan untuk interkoneksi antar sentral dengan hirarki yang berbeda. Untuk hirarki yang lebih rendah digambarkan pada lokasi yang

rendah dan semakin keatas mempunyai hirarki semakin tinggi. Topologi jaringan jenis ini cocok digunakan pada sistem jaringan komputer.



Gambar 2.6
Topologi *Tree*

Pada jaringan pohon, terdapat beberapa tingkatan simpul atau *node*. Pusat atau simpul yang lebih tinggi tingkatannya, dapat mengatur simpul lain yang lebih rendah tingkatannya. Data yang dikirim perlu melalui simpul pusat terlebih dahulu. Misalnya untuk bergerak dari komputer dengan *node-3* kekomputer *node-7* seperti halnya pada gambar, data yang ada harus melewati *node-3, 5* dan *node-6* sebelum berakhir pada *node-7.*

Ada dua kesulitan pada topologi ini:

1) Karena bercabang maka diperlukan cara untuk menunjukkan kemana data dikirim, atau kepada siapa transmisi data ditujukan.

2) Perlu suatu mekanisme untuk mengatur transmisi dari terminal terminal dalam jaringan.

Table 2.4
Kelebihan dan Kelemahan Topologi *Tree*

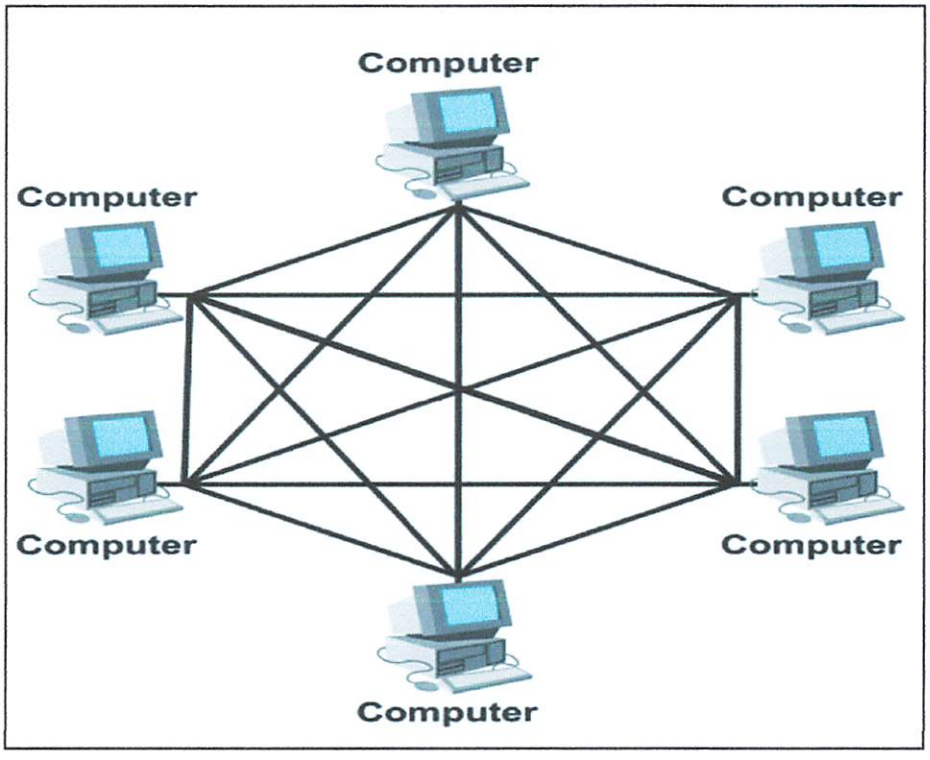
| | |
|---|---|
| **Kelebihan Topologi Tree** | ✓ dapat terbentuknya suatu kelompok yang dibutuhkan pada setiap saat. |
| **Kelemahan Topologi Tree** | ✓ apabila simpul yang lebih tinggi kemudian tidak berfungsi, maka kelompok lainnya yang berada dibawahnya akhirnya juga menjadi tidak efektif.<br>✓ Cara kerja jaringan pohon ini relatif menjadi lambat. |

### 2.1.5 Topologi *Mesh*

Topologi *mesh* adalah suatu bentuk hubungan antar perangkat dimana setiap perangkat terhubung secara langsung ke perangkat lainnya yang ada di dalam jaringan. Akibatnya, dalam topologi mesh setiap perangkat dapat berkomunikasi langsung dengan perangkat yang dituju (*dedicated links*).

Karakteristik Topologi Mesh :

- ✓ Topologi Mesh adalah topologi yang tidak memiliki aturan dalam koneksi. Topologi ini biasanya timbul akibat tidak adanya perencanaan awal ketika membangun suatu jaringan.
- ✓ Karena tidak teratur maka kegagalan komunikasi menjadi sulit dideteksi, dan ada kemungkinan boros dalam pemakaian media transmisi.
- ✓ Topologi ini menerapkan hubungan antar sentral secara penuh. Jumlah saluran yang harus disediakan untuk membentuk jaringan Mesh adalah jumlah sentral dikurangi 1.
- ✓ Tingkat kerumitan jaringan sebanding dengan meningkatnya jumlah sentral yang terpasang.
- ✓ Disamping kurang ekonomis juga relatif mahal dalam pengoperasiannya.




Gambar 2.7
Topologi *Mesh*

Table 2.5
Kelebihan dan Kelemahan Topologi *Mesh*

| | |
|---|---|
| **Kelebihan Topologi Mesh** | ✓ Hubungan *dedicated links* menjamin data langsung dikirimkan ke komputer tujuan tanpa harus melalui komputer lainnya sehingga dapat lebih cepat karena satu link digunakan khusus untuk berkomunikasi dengan komputer yang dituju saja (tidak digunakan secara beramai-ramai/sharing).<br>✓ Memiliki sifat *Robust*, yaitu Apabila terjadi gangguan pada koneksi komputer A dengan komputer B karena rusaknya kabel koneksi (links) antara A dan B, maka gangguan tersebut tidak akan mempengaruhi koneksi komputer A dengan komputer lainnya.<br>✓ *Privacy* dan *security* pada topologi mesh lebih terjamin, karena komunikasi yang terjadi antara dua komputer tidak akan dapat diakses oleh komputer lainnya.<br>✓ Memudahkan proses identifikasi permasalahan pada saat terjadi |

| | |
|---|---|
| | kerusakan koneksi antar komputer.<br>✓ Keuntungan utama dari penggunaan topologi *mesh adalah fault tolerance (Toleransi Kesalahan.* |
| **Kelemahan Topologi Mesh** | ✓ Membutuhkan banyak kabel dan Port I/O. semakin banyak komputer di dalam topologi mesh maka diperlukan semakin banyak kabel links dan port I/O (lihat rumus penghitungan kebutuhan kabel dan Port).<br>✓ Sulitnya pada saat melakukan instalasi dan melakukan konfigurasi ulang saat jumlah komputer dan peralatan-peralatan yang terhubung semakin meningkat jumlahnya.<br>✓ Hal tersebut sekaligus juga mengindikasikan *bahwa* topologi jenis ini membutuhkan biaya yang relatif mahal.<br>✓ Karena setiap komputer harus terkoneksi secara langsung dengan komputer lainnya maka instalasi dan konfigurasi menjadi lebih sulit.<br>✓ Banyaknya kabel yang digunakan juga mengisyaratkan perlunya space yang memungkinkan di dalam ruangan tempat komputer-komputer tersebut berada. |

## 2.2 *Cluster*

Yang dimaksud dengan *Computer Cluster* adalah sekumpulan komputer yang terhubung di dalam suatu jaringan komputer bekerja secara bersama dan saling berhubungan satu sama lain untuk mendukung suatu kerja yang biasa ditangani oleh sebuah komputer tunggal (Turbo Linux User Guide, 2000: 33).

Pada suatu jaringan komputer, pembentukan suatu komputer *cluster* biasanya ditujukan untuk meningkatkan kemampuan komputasi dan kinerja sistem komputer didalam menyelesaikan suatu permasalahan dengan cara mendistribusikan beban pekerjaan yang ada pada beberapa komputer. Jadi komputer *Cluster* merupakan salah satu bentuk implementasi dari komputasi paralel pada jaringan komputer.

Dengan peningkatan jumlah pengguna dan beban kerja, pengelola *cluster* seringkali khawatir bagaimana sistemnya dapat terus mengikuti perkembangan zaman. Oleh karena itu hardware dan software yang terlibat dalam pembuatan *cluster* harus memenuhi persyaratan berikut (Utdiratatmo : 2004 ):

- *Skalability*, yaitu system dapat menyesuaikan diri menangani beban yang ditanggungya.
- 24x7 *availability*, yaitu sistem harus dapat bekerja 24 jam 7 hari penuh meskipun terjadi kerusakan atau kegagalan di sebagian *hardware* atau *software* dalam *system.*
- *Manageability,* yaitu *system* harus terjangkau dengan mudah meskipun ukurannya besar.
- *Cos-effectiveness,* yaitu *system* harus terjangkau secara ekonomi dan dapat diekspansi.

Didalam suatu komputer *cluster* umumnya akan dijumpai satu komputer yang berperan sebagai *Cluster Manager* serta komputer lainnya yang berperan sebagai *Cluster Node* pada *cluster* tersebut. *Cluster Manager* bertugas untuk mendistribusikan pekerjaan yang harus diselesaikan oleh komputer lainnya yang dikonfigurasikan sebagai *cluster node* tergantung pada tipe cluster yang digunakan. Beberapa tipe komputer *cluster* yang digunakan dalam implementasi dari cluster antara lain (Turbo Linux User Guide, 2000: 44).

*1) Shared Processing*

Tipe ini dipakai oleh *Beowulf Clustering* dimana melalui *system clustering,* kemampuan komputasi dari beberapa *system computer* digabungkan untuk memperoleh suatu *system* dengan kemampuan komputasi yang lebih besar.

*2) Load Balancing*

Tipe ini menyerupai tipe *shared processing* diatas namun setiap *cluster node* akan memproses bagian pekerjaan yang diberikan kepadanya oleh *cluster manager* tanpa melakukan komunikasi dengan *cluster node* lainnya dalam menyelesaikan pekerjaan tersebut.

3) *Fail Over*

Tipe ini menyerupai tipe *load balancing* diatas *Fail-over* hampir mirip dengan *load balancing*, namun proses tidak didistribusikan pada seluruh cluster *node* tetapi hanya pada suatu *node* tertentu saja. *Cluster node* lainnya bertugas mengambil alih proses bila *node* yang sebelumnya bertugas *down* atau gagal menyelesaikan pekerjaan yang diberikan oleh *cluster manager.*

4) *High availability*

Merupakan metode dimana seluruh fasilitas komputasi yang dimiliki sistem dijaga keberadaannya dan dapat dimanfaatkan atau mengambil alih proses komputasi yang dilakukan oleh bagian lain bilamana dibutuhkan. *High availability* dapat dimplementasikan baik secara *software* maupun *hardware. High availability* dapat dicapai pada komputer *cluster* tipe *load balancing* dan *fail over* diatas.

## 2.3 *Load Balancing*

*Load balancing* adalah suatu metode untuk mendistribusikan beban kepada beberapa *host* sehingga beban kerja menjadi lebih ringan. Ini bertujuan agar waktu rata-rata mengerjakan tugas menjadi singkat dan dapat menaikkan *utilitas processor.*

*Load balancing* dapat diimplementasikan dengan *hardware, software* maupun gabungan keduanya. Konfigurasi standart yang ada memberikan gambaran bahwa satu mesin ditempatkan diantara *client* dan *server*, mesin ini disebut sebagai *director* karena tugasnya adalah memberikan *balancing* pada *request* dari *client* ke *server.*

Sebuah *load balancer* adalah perangkat jaringan yang dipasang antara *client* dan *server,* bekerja sebagai saklar untuk *request* dari *client. Load balancer* mengimplementasikan beberapa metode penjadwalan yang akan menentukan kea rah *server* mana request dari *client* akan diteruskan. Beberapa keuntungan yang diperoleh dari teknik load balancing sebagai berikut (Nasution, A.H:2011):

a. Flexibility : *server* tidak lagi menjadi inti system dan resource utama, tetapi menjadi bagian dari banyak *server* yang membentuk cluster. Hal ini

berarti bahwa performa per unit dari cluster tidak terlalu diperhitungkan, tetapi performa cluster secara keseluruhan. Sedangkan untuk meningkatkan performa dari cluster, *server* atau unit baru dapat ditambahkan tanpa mengganti unit yang lama.

b. Scalability : System tidak memerlukan desain ulang seluruh atsitektur system untuk mengadaptasikan system tersebut ketika terjadi perubahan pada komponen system.

c. Security : Untuk semua trafik yang melewati load balancer, aturan keamanan dapat diimplementasikan dengan mudah. Dengan privat network digunakan untuk real *servers*, alamat IP-nya tidak akan diakses secara langsung dari luar system cluster.

d. *High-availability* : Load balancer dapat mengetahu kondisi real *server* dalam system secara otomatis, jika terdapat real *server* yang mati maka akan dihapus dari daftar real *server*, dan jika real *server* tersebut kembali aktif maka akan dimasukkan kedalam daftar real *server*. Load balancer juga dapat dikonfigurasi secara redundant dengan load balancer yang lain.

## 2.4 *Web Server*

*Web Server* merupakan sebuah perangkat lunak dalam *server* yang berfungsi menerima permintaan (*request*) berupa halaman *web* melalui *HTTP* atau *HTTPS* dari *client* yang dikenal dengan *browser web* dan mengirimkan kembali (*response*) hasilnya dalam bentuk halaman-halaman *web* yang umumnya berbentuk dokumen *HTML* (http://www.apache.org/).

*Web server*, untuk berkomunikasi dengan *client*-nya (*web browser*) mempunyai *protocol* sendiri, yaitu *HTTP* ( *Hypertext Transfer Protocol* ). Dengan *protocol* ini, komunikasi antar *web server* dengan *client*-nya dapat saling dimengerti dan lebih mudah.

*Server HTTP Apache* atau *server Web/WWW Apache* adalah *server web* yang dapat dijalankan dibanyak system operasi ( Unix, BSD, Linux, Microsoft Windows serta platform lainnya ) yang berguna untuk melayani dan memfungsikan situs *web*. *Protocol* yang digunakan untuk melayani fasilitas *web/www* ini menggunakan *HTTP*.

*Apache* merupakan perangkat lunak terbuka yang dikembangkan oleh komunitas terbuka yang terdiri dari pengembang-pengembang dibawah naungan *Apache software foundation.*


Gambar 2.8
Logo *Apache*

## *2.5 CentOS*

*CentOS* adalah *system* operasi berbasis kernel linux yang bersifat *freeware* dan *open-source.* CestOS yang merupakan kependekan dari *Community ENTerprise Operating System* adalah salah satu distro linux turunan dari *Red Hat Enterprise Linux (RHEL)* (http://www.centos.org/).

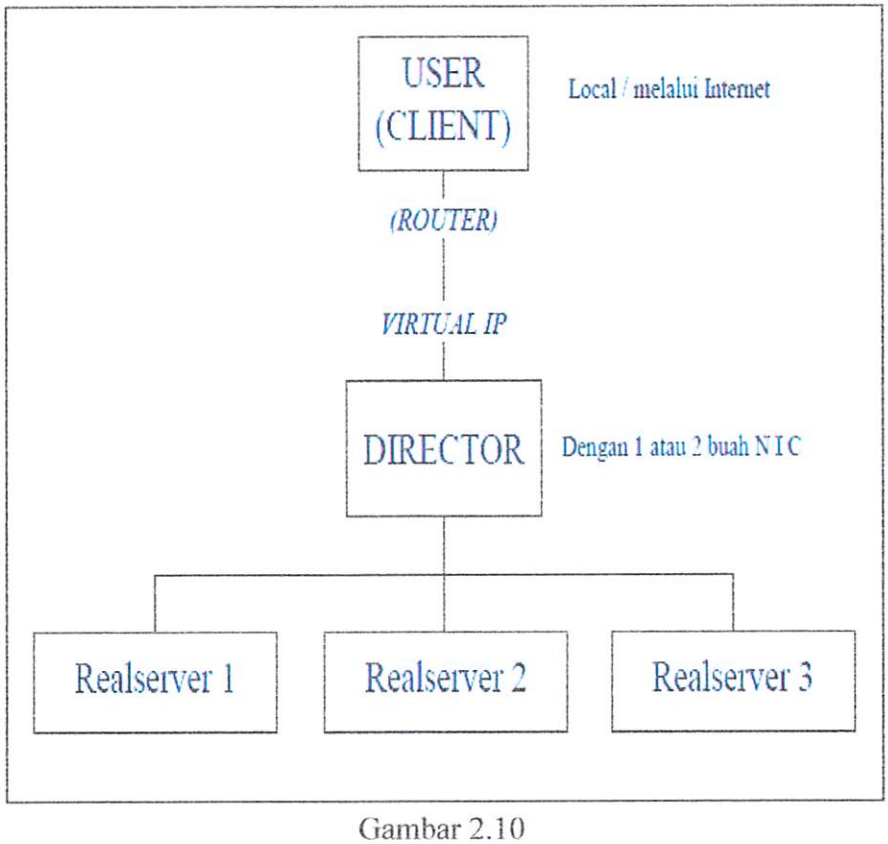
.


Gambar 2.9
Logo CentOS

## *2.6 LVS (Linux Virtual Server)*

*Linux Virtual Server* atau disingkat *LVS* merupakan suatu teknologi *clustering* yang dapat digunakan untuk membangun suatu *server* dengan menggunakan kumpulan dari beberapa buah *real server. LVS* merupakan implementasi dari komputer *cluster* dengan metode *High Availbalility. LVS*

mengimbangi berbagai bentuk dari *service* jaringan pada banyak mesin dengan memanipulasi paket sebagaimana diproses *TCP/IP stack*. Satu dari banyak peran yang paling umum dari *Linux Virtual Server* adalah bertindak sebagai *server* yang berada pada garis terdepan dari kelompok *server web*. Seperti ditunjukkan pada Gambar 2.10. Linux Virtual *Server* atau *LVS* ini terdiri dari sebuah *Director* dan beberapa *realserver* yang bekerja bersama dan memberikan *service* terhadap permintaan *user*. Permintaan *user* diterima oleh *Director* yang seolah olah berfungsi sebagai *IP Router* yang akan meneruskan paket permintaan *user* tersebut pada *realserver* yang siap memberikan *service* yang diminta .( Gozali F, Alex:2002).




Gambar 2.10
Struktur Dasar *Linux Virtual Server* dengan 3 Buah *Real Server*

Dengan demikian *virtual server* akan terdiri dari beberapa *computer* yang mempunyai *image* yang sama tetapi ditempatkan pada *IP* yang berbeda. *User* dapat mengakses *virtual server* tersebut dengan bantuan suatu *Director*, yang bertugas untuk melakukan pemetaan IP dari *server* dan komputer lainnya yang berperan sebagai *virtual server*.

*LVS Director* adalah modifikasi dari sistem Linux yang bertanggung jawab untuk mendistribusikan permintaan *user/client* terhadap *realserver* pada kelompok *server*. *Realserver* melakukan pekerjaan untuk memenuhi permintaan serta memberikan atau membuat laporan balik kepada *user/client*.

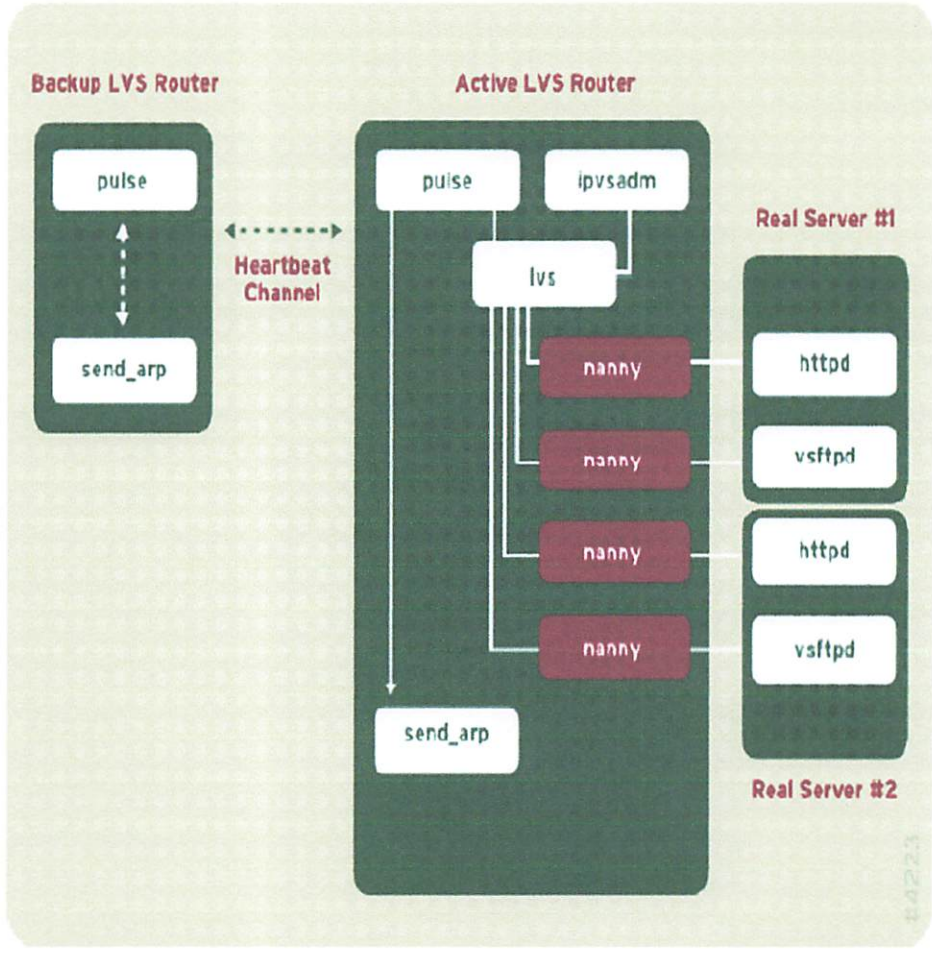
*LVS Director* memelihara rekaman daripada sejumlah permintaan yang telah ditangani oleh masing-masing *realserver* dan menggunakan informasi ini ketika memutuskan *server* mana yang akan ditugaskan untuk menangani suatu permintaan berikutnya.

*LVS Director* memelihara rekaman daripada sejumlah permintaan yang telah ditangani oleh masing-masing *realserver* dan menggunakan informasi ini ketika memutuskan *server* mana yang akan ditugaskan untuk menangani suatu permintaan berikutnya.

*Service* yang dapat didukung oleh *LVS* (Turbo Linux User Guide, 2000: 53), antara lain:

1. *Web site (HTTP, HTTPS)*
2. *FTP (File Transfer Protocol)*
3. *Email (SMTP, POP3, dan IMAP)*
4. *News (NNTP)*
5. *DNS (Domain Name Service)*
6. *LDAP (Lightweight Directory Access Protocol)*

Suatu *LVS* terdiri dari beberapa komponen pendukung yang bertugas untuk menyimpan informasi yang dibutuhkan, mengatur proses kerja *LVS* serta proses lainnya. Pembagian tugas dari tiap komponen dapat dilihat pada tabel 1. sedangkan hubungan antar komponen *LVS* dapat dilihat pada Gambar 2.11.




Gambar 2.11
Hubungan antara komponen *LVS*

Table 2.6
Tugas Komponen *LVS*

| | |
|---|---|
| *Pulse* | Mengontrol proses jalannya daemon yang sesuai dengan kebutuhan proses sekaligus menjadi heatbeat yang mengirimkan sinyal pada *active director*. *Pulse* bekerja pada *LVS director*. |
| *LVS* | merupakan *LVS daemon* yang bekerja pada *director* yang bertugas untuk membaca file konfigurasi dan memanggil *ipvsadm* untuk menampilkan *routing table* dari *IPVS*. |
| *Nanny* | Merupakan komponen yang bertugas untuk memonitor *daemon* yang bekerja pada *active director*. |
| */etc//sysconfig/ha/lvs.cf* | Merupakan file konfigurasi dari *LVS cluster file* konfigurasi ini terdapat pada *director*. |

| | |
|---|---|
| *Ipvsadm* | Berfungsi untuk meng-update *IPVS routing table.* Komponen ini terdapat pada *director.* |
| *Piranha configuration tool* | *Tool* yang berbasis *web* untuk memonitor, mengkonfigurasi dan administrasi *LVS cluster.* |
| *Send-arp* | Bertugas untuk mengirimkan *ARP (Address Resolution Protocol ) broadcast* ketika alamat *virtual* sever berubah dari satu *node* ke *node* lainnya |

### 2.6.1 Metode *Forwarding* Pada *LVS*

a. *LVS-NAT*

Pada metode *LVS-NAT, load bal;ancer* berfungsi pula sebagai *Network Address Translation (NAT).* Komputer yang berfungsi sebagai *load balancer* minimal memiliki 2 *NIC (Network Interface Card)* yaitu *eth0* dan *eth1.*

b. *LVS-TUN*

*LVS-TUN* menggunakan metode *IP Tunneling* agar paket dapat dienkapsulasi oleh *director* dan diteruskan menuju *realserver.* Setelah diproses oleh *realserver, client request* langsung dikembalikan ke *client* tanpa melalui *director. LVS-TUN* cocok untuk *director* dan *realserver* yang tidak berada dalam jaringan yang sama atau dihubungkan *via internet.*

c. *LVS-DR*

Pada *LVS-DR (LVS-Direct Routing), realserver* mengembalikan *client request* yang telah diprosesnya langsung ke *client* yang memintanya. Perbedaannya dengan topologi lainnya adalah *LVS-DR* tidak memerlukan tunneling *server,* namun *director* dan *realserver* harus berada di dalam jaringan yang sama.

### 2.6.2 Algoritma Penjadwalan

Beberapa jenis algoritma penjadwalan yang dapat diterapkan pada system *Linux Virtual Server (LVS)* pada proses distribusi request kepada *realserver* antara lain yaitu (Administration Linux Virtual Server : 2009):

1. *Round Robin (rr)*

   Mekanisme penjadwalan pada *LVS* dikerjakan oleh sebuah patch kernel yang disebut modul *IP Virtual Server* atau *IPVS modules*. Modul ini mengaktifkan layer 4 yaitu *transport layer switching* yang dirancang dapat bekerja dengan baik pada *multi server* dalam *IP address* tunggal (*virtual IP address*). *IPVS* membuat *IPVS table* pada kernel untuk menelusuri dan merutekan paket ke *realserver* secara efisien. *Table* ini digunakan oleh *load balancer* yang sedang aktif untuk meneruskan *client request* dari *virtual IP address* ke *realserver*. *IPVS table* secara rutin diperbarui menggunakan *software ipvsadm*. Pada penjadwalan tipe *round-robin*, manager mendistribusikan *client request* sama rata ke seluruh *realserver* tanpa memperdulikan kapasitas *server* ataupun beban *request*. Jika ada tiga *realserver* (A,B,C), maka *request* 1 akan diberikan manager kepada *server* A, request 2 ke *server* B, request 3 ke *server* C dan *request* 4 kembali ke *server* A. mekanisme ini dapat dilakukan jika seluruh real *server* menggunakan spesifikasi computer yang sama.
2. *Weighted Round Robin (wrr)*

   Penjadwalan ini memperlakukan *realserver* dengan kapasitas proses yang berbeda. Masing-masing *realserver* dapat diberi bobot bilangan integer yang menunjukkan kapasitas proses, dimana bobot awal adalah 1.
3. *Least connection (lc)*

   Merupakan algoritma penjadwalan yang mengarahkan koneksi jaringan pada *server* aktif dengan jumlah koneksi yang paling sedikit. Penjadwalan ini termasuk salah satu algoritma penjadwalan dinamis, karena memerlukan perhitungan koneksi aktif untuk masing-masing *realserver* secara dinamis.

4. *Weighted Least Connection (wlc)*

   Merupakan sekumpulan penjadwalan *least connection* dimana dapat ditentukan bobot kinerja pada masing-masing *real server.*

5. *Locality Based Least Connection (lblc)*

   Metode penjadwalan yang akan mendistribusikan lebih banyak request kepada real *server* yang memilik koneksi kurang aktif.

6. *Destination Hashing (dh)*

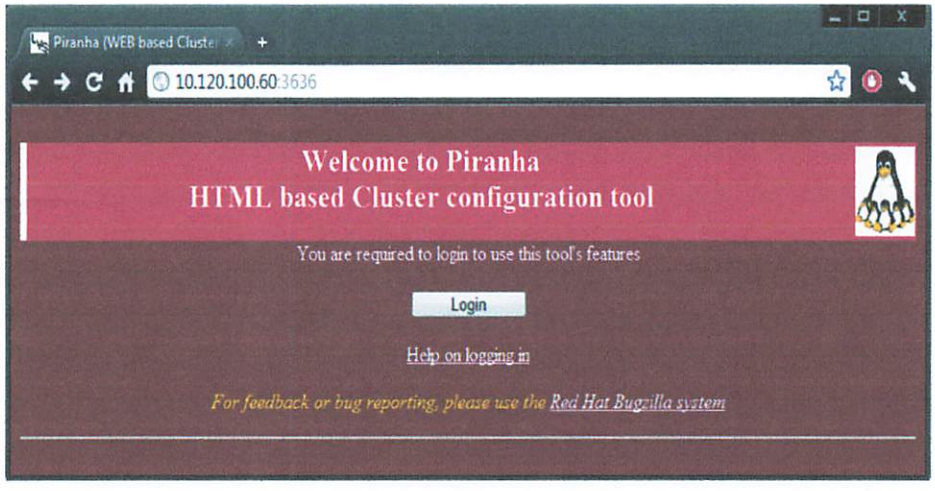
   Merupakan algoritma penjadwalan statis yang dapat meneruskan request dari *client* kepada satu real *server* tertentu sesuai dengan layanan yang diminta. Terdapat suatu table hash berisi alamat tujuan dari masing-masing *real server* beserta layanan yang tersedia pada setiap real *server.*

7. *Source Hasing (sh)*

   Hampir sama dengan metode destination hasing tetapi pada metode ini table berisi mengenai informasi alamat adalah paket yang dikirimkan oleh *client.*

### 2.6.3 *Piranha*

*Piranha* merupakan *tool* yang digunakan untuk mengkonfigurasi suatu *system cluster* secara *GUI* yang berjalan di system operasi linux. Dalam sebuah *system load balancing*, *piranha* berada pada *node Director.* Fungsi utama dari *piranha* yaitu memulai dan menghentikan sesuatu yang dinamakan dengan *pulse* (nadi), juga dapat mengatur algoritma penjadwalan yang akan digunakan.




Gambar 2.12
Tampilan Piranha

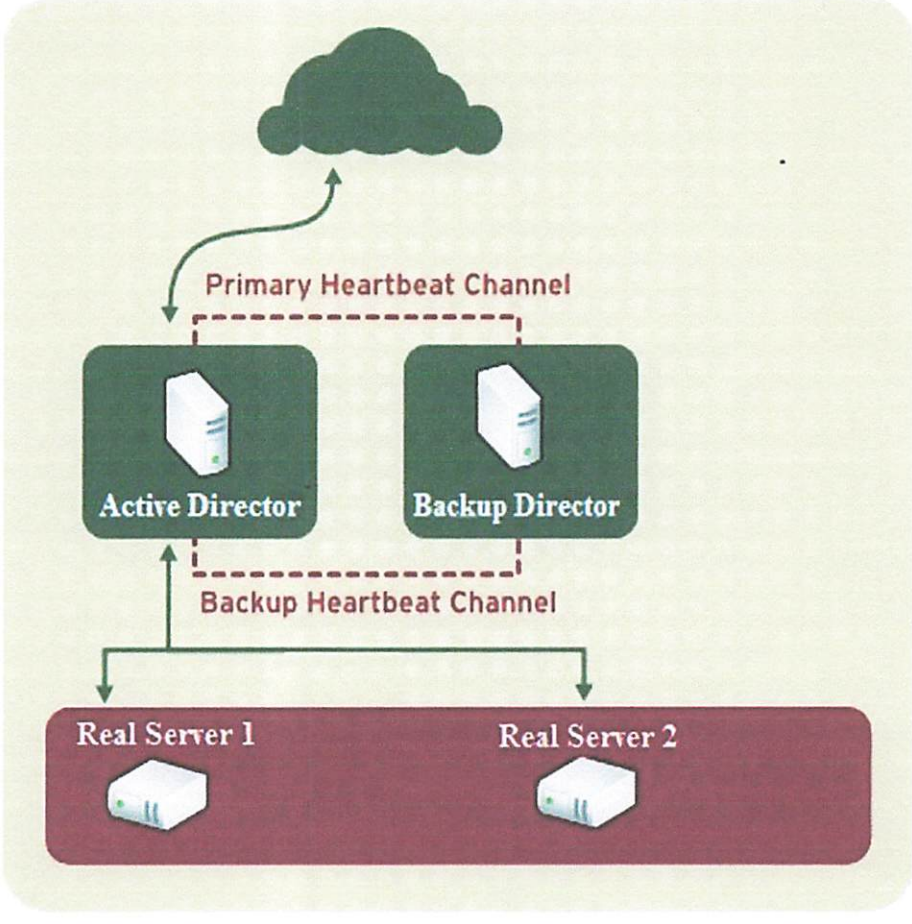
# BAB III

# ANALISA DAN PERANCANGAN

Dalam bab ini akan diuraikan perencanaan dari system load balancing web server yang akan dibangun. Pembahasan dimulai dari perancangan system dan topologi jaringan untuk load balancing beserta instalasi dan konfigurasinya.

## 3.1 Gambaran Umun Sistem

Pada tugas akhir ini akan dibangun web server system load balancing. Sebelum membangun system yang dikehendaki, diperlukan perencanaan spesifikasi hardware dan topologi jaringan yang akan digunakan dalam tugas akhir ini. Model system yang dirancang secara jelas digambarkan pada gambar 3.1.




Gambar 3.1
Rancangan Model Sistem Load Balancing

Pada system Load Balancing diatas terdapat komponen penting penyusun utama, yaitu :

- ✓ Director, sebagai sebuah server yang bertugas untuk meneruskan paket dari client kepada real server. Client dari jaringan luar akan menganggap director seolah sebagai server tunggal.
- ✓ Real server, sebagai server yang melayani permintaan client, dalam hal ini sebagai web server yang memberikan layanan http.

System diatas yang dirancang menggunakan metode Direct Routing dengan beban pada real server diseimbangkan oleh load balancer/Director. Director dan real server dihubungkan oleh switch dan real server memberikan layanan sama, yaitu http.

Heartbeat merupakan suatu teknologi awal dari failover. Fungsi dari Heartbeat adalah memeriksa layanan dari unit server, jika service tersebut mati maka akan langsung menggantikannya dengan server backup service. Berikut peran dari Heartbeat :

- ✓ Memonitor server online (bekerja atau tidak) bias menggunakan Ethernet maupun serial port.
- ✓ Melakukan failover dimana server master akan dialihkan ke server slave jika terjadi kegagalan pada server master.

## 3.2 Analisa Kebutuhan Software/Hardware

### 3.2.1 Komponen Perangkat Keras/Hardware

Sistem linux virtual cluster diuji dengan cara membangkitkan request secara simultan dalam jumlah yang dikehendaki. Untuk melakukan hal tersebut diperlukan jumlah client yang banyak pula. Maka digunakan request generator agar cukup dengan satu buah computer seolah dapat menghasilkan request yang dihasilkan oleh banyak client secara simultan.

Berikut pada table 3.1 adalah spesifikasi yang digunakan dalam tugas akhir ini :

Table 3.1

Komponen Perangkat Keras Sistem

| | |
|---|---|
| Director aktif dan Director Backup | - Intel Pentium 4, 3Ghz.<br>- Memory 512mb.<br>- HD 80Gb.<br>- Operating System Linux CentOs 6. |
| Real Server 1 | - Intel Pentium 4, 3Ghz.<br>- Memory 512mb.<br>- HD 80Gb.<br>- Operating System Linux CentOs 6. |
| Real Server 2 | - Intel Pentium 4, 3Ghz.<br>- Memory 512mb.<br>- HD 80Gb.<br>- Operating System Linux CentOs 6. |
| Client | - Intel Core I5, 2,3 Ghz.<br>- Memory 2Gb.<br>- HD 640Gb.<br>- Operating System Windows 7. |

### 3.2.2 Komponen Perangkat Lunak/Software

- ✓ System Operasi

  Komputer sebagai director menggunakan system operasi linux CentOS 6. Begitu pula dengan computer real server.

- ✓ Kernel

  Kernel merupakan initi dari suatu system operasi yang berhubungan langsung dengan perangkat keras. Pada bagian kernel terdapat berbagai pilihan untuk memilih dan mengaktifkan driver perangkat keras dan algoritma penjadwalan yang akan digunakan. Kernel yang digunakan saat penelitian akhir ini adalah kernel versi 2.6.3 untuk computer director maupun real server.Ipvsadm.

- ✓ Web Server

  Perangkat lunak untuk web server yaitu apache.

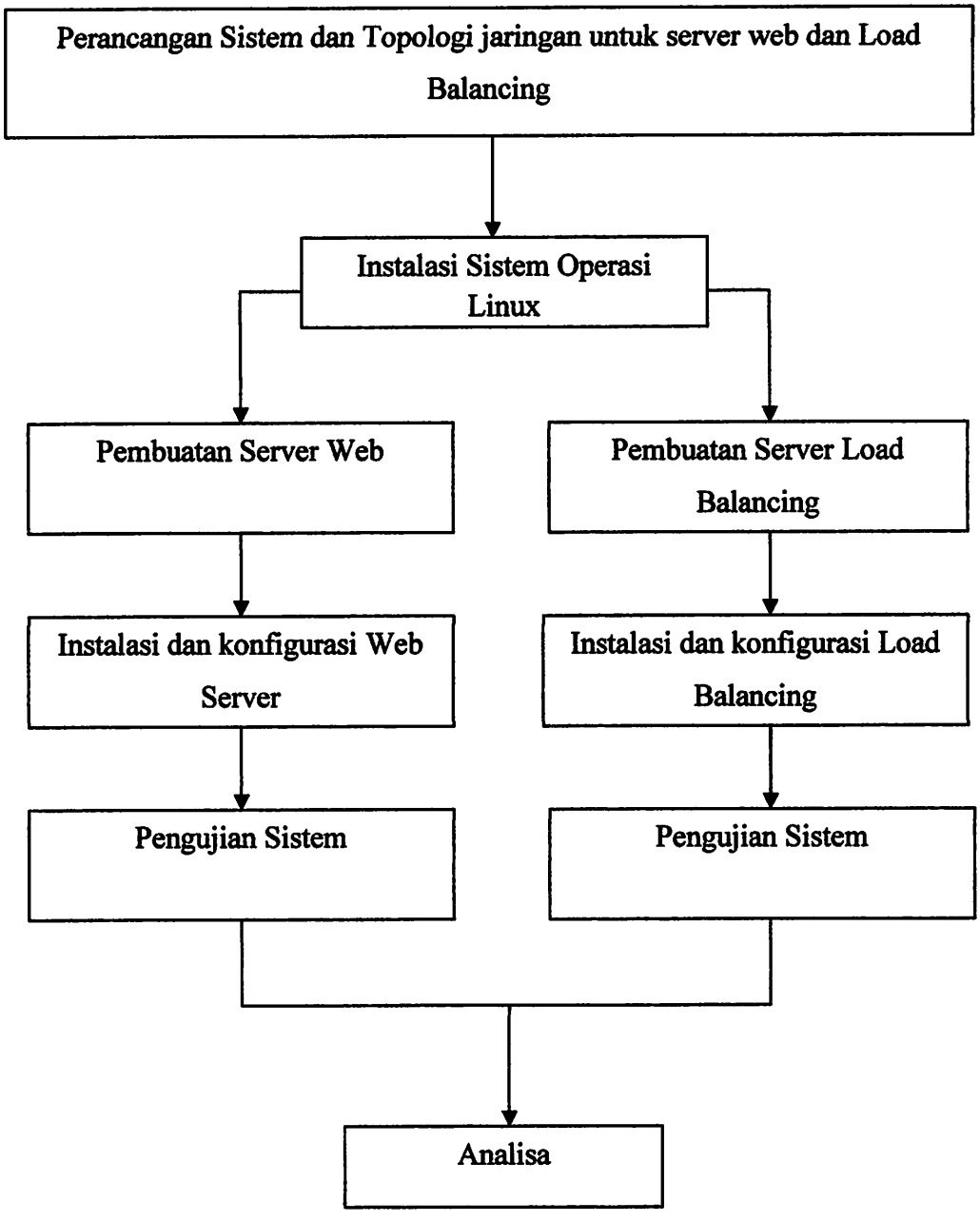
- ✓ Ipvsadm

  IP Virtual Server Administration (Ipvsadm) digunakan untuk mengatur kerja director sehingga dapat menambahkan layanan-layanan apa saja yang dapat diberikan, dan memilih algoritma penjadwalan yang digunakan, dan meneruskan request kepada real server yang sedang aktif.

- ✓ Apache

  Apache Web Server merupakan program aplikasi yang berjalan di server, berfungsi untuk menjalankan aplikasi web sehingga bisa diakses oleh client baik melalui jaringan intranet maupun internet.

## 3.3 Tahap – Tahap Perencanaan Sistem




Gambar 3.2
Tahap Perencanaan Sistem

Dalam gambar 3.2 diatas merupakan tahap-tahap perencanaan sistem load balancing web server. Berikut adalah penjelasan tahap-tahap yang terdapat pada gambar 3.2.

- ✓ Tahap 1 yaitu perancangan sistem dan Topologi jaringan untuk server web dan Load Balancing. Dalam tahap ini penulis meneliti dan mencari topologi yang tepat dalam pembuatan sistem load balancing ini. Sehingga didapatkan topologi dengan 2 buah server load balancing, master dan slave. Dan 2 buah web server.

- ✓ Tahap 2 yaitu instalasi sistem operasi linux pada server load balancing dan web server.

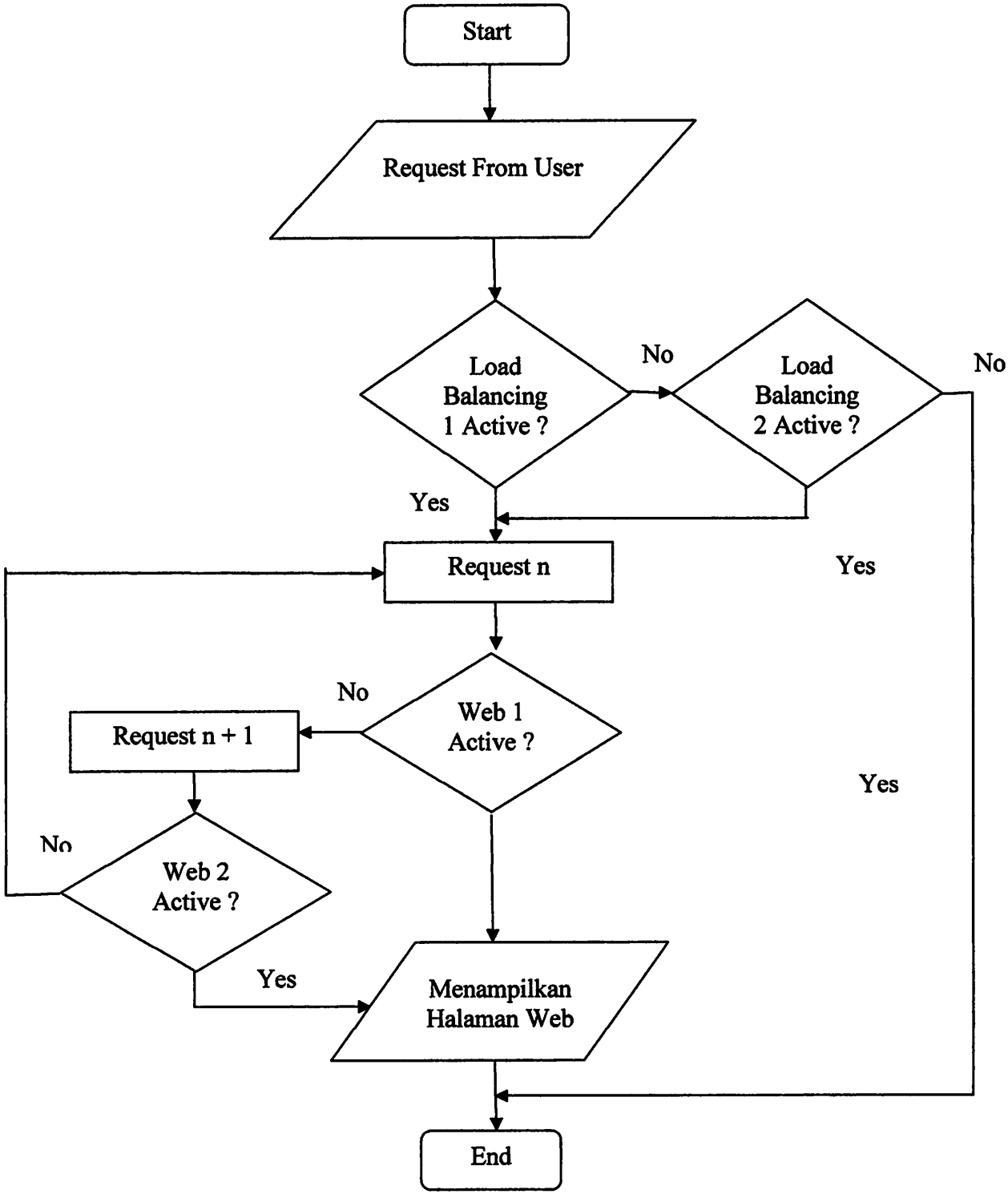
- ✓ Pada tahap 3 yaitu pembuatan werver load balancing dan web server dengan sistem operasi dasar adalah Linux CentOS 6.

- ✓ Tahap 4 adalah konfigurasi web server dan load balancing. Dalam konfigurasi web server, komponen lunak/software yang digunakan adalah apache 2. Apache 2 ini merupakan software yang menyediakan service web. Dan pada konfigurasi load balancing, komponen lunak yang digunakan adalah sebuah tool yaitu Piranha. Dengan piranha konfigurasi load balancing semakin mudah dengan didukungnya interface secara GUI kepada administrator.

- ✓ Tahap 5 yaitu pengujian sistem. Pada tahap ini hal yang diuji yaitu apakah algoritma yang digunakan sudah berjalan dengan baik atau belum. Dan pengujian saat salah satu server mengalami kegagalan apakah halaman web tetap bias diakses.

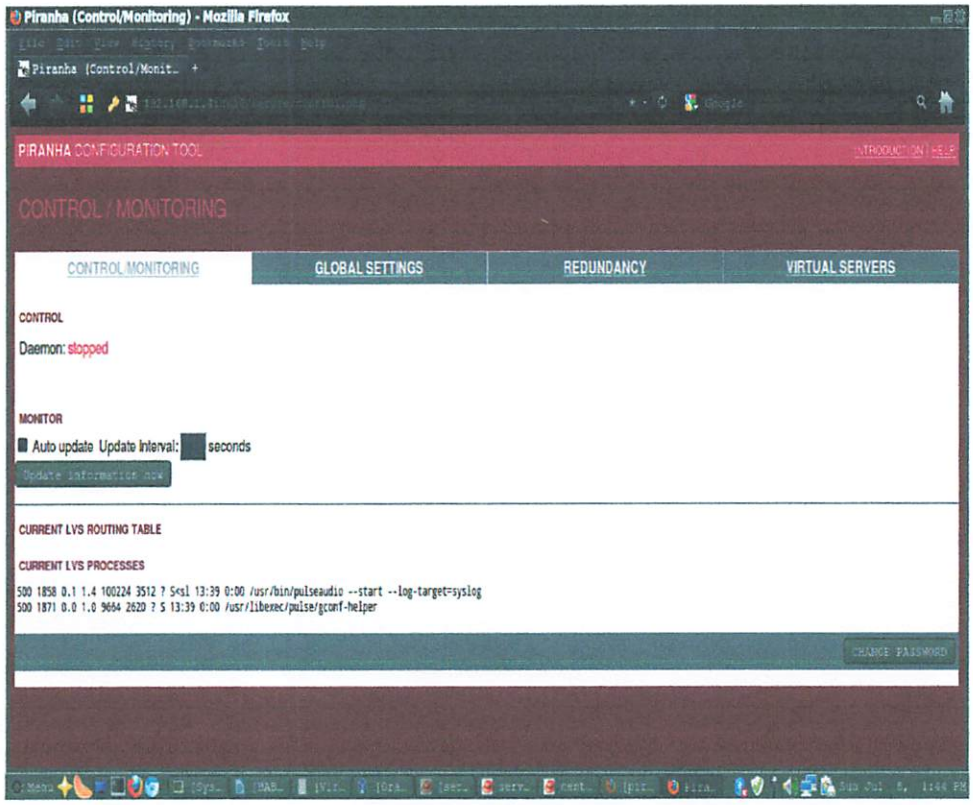
- ✓ Pada tahap terakhir ini yaitu analisa mengenai apakah permintaan halaman web dari client benar tertuju ke real server atau tidak, dan algoritma berjalan dengan baik.

### 3.4 Flowchart Sistem




Gambar 3.3
Flowchart Sistem

Di dalam Director terdapat komponen penting yang digunakan dalam mendukung terciptanya suatu sistem cluster load balancing. Komponen ini sudah dijelaskan penulis pada bab 2. Dan untuk tool yang digunakan yaitu piranha. Gambar 3.4 menunjukkan menu utama piranha dan penjelasannya sebagai berikut :




Gambar 3.4
Tampilan Menu Piranha

Pada table 3.2 berisikian mengenai fungsi dari tiap-tiap menu utama dari Piranha.

Table 3.2
Menu Utama Piranha

| | |
|---|---|
| Control/Monitoring | Pada menu ini administrator dapat memantau daemon Pulse dan Table Routing dari proses yang berjalan. |
| Global Setting | Berisi tentang pengaturan server primary. |

| | |
|---|---|
| Redundancy | Pengaturan backup dari server utama. Hal ini dinamakan teknologi Heartbeat. |
| Virtual Servers | Pada menu ini berisikan tentang pengaturan server virtual, dan real server. |

Dari skenario gambar 3.3, sebelum load balancing meneruskan permintaan ke real server, terdapat algoritma yang digunakan yaitu Round Robin. Prinsip sederhana dari algoritma ini adalah pembagian beban secara merata tiap real server. Setiap request yang datang akan diarahkan pada real server yang aktif. Misal terdapat 4 request web dari client, yang terdiri dari 2 server web A dan B. seperti prinsip algoritma round robin, maka pada saat request pertama akan diarahkan ke server A, request kedua diarahkan ke server B, request ketiga diarahkan ke server A, dan request keempat ke server B.

Dalam algoritma ini dapat dikonfigurasi melalui piranha, yaitu berada dalam menu virtual server. Sedangkan melalui penambahan script, dapat mengisikan script ke dalam direktori /etc/sysconfig/ha/lvs.cf.

Pada umumnya, semua situs web di dunia ini digerakkan oleh suatu program aplikasi yang berjalan di server. Program tersebut, yang sering dipakai adalah Apache, karena selain berlisensi GPL (general Public Lisence) atau free software, juga mudah dikonfigurasikan. Sedangkan aplikasi yang menjalankan program apache tersebut biasa dinamakan Web Server atau Httpd.

Apache Web Server merupakan program aplikasi yang berjalan di server, berfungsi untuk menjalankan aplikasi web sehingga bisa diakses oleh client baik melalui jaringan intranet maupun internet.

# BAB IV

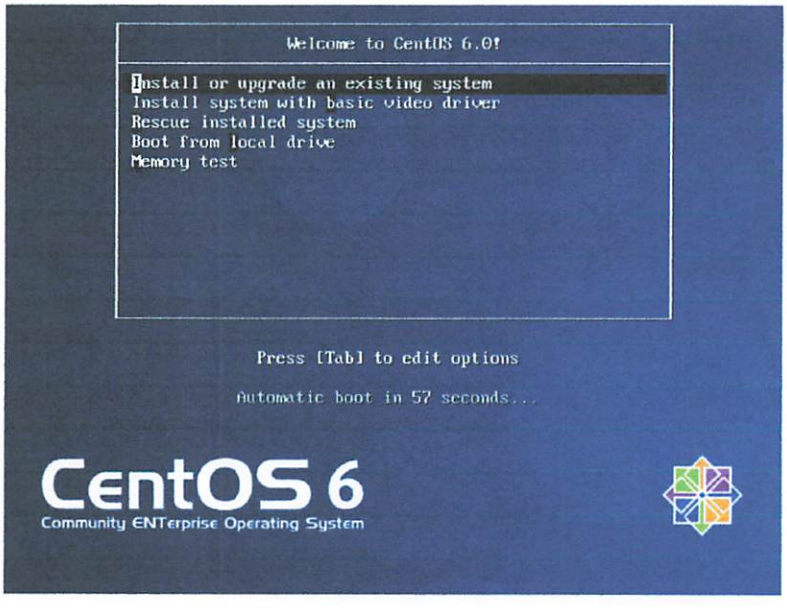
# IMPLEMEMTASI DAN PENGUJIAN

## 4.1 Implementasi Sistem

Dalam bab ini akan dilakukan tentang implementasi dan pembahasan mengenai pengujian analisis performa dari load balancing yang telah dibangun. Tugas akhir ini memerlukan sistem operasi yang nantinya dapat digunakan untuk membangun cluster yang beroperasi sebagai server web. Di atas sistem ini nantinya dapat berjalan software-software yang digunakan untuk menjalankan dan mengelola server web. Sebelum melakukan analisis performasi sistem dan jaringan, maka perlu dilakukan instalasi sistem operasi dan software lainnya.

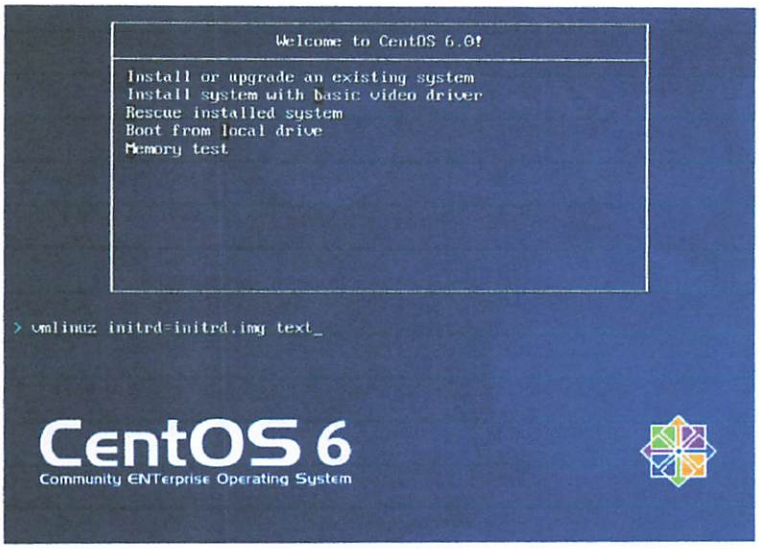
### 4.1.1 Instalasi Sistem Operasi CentOS.

Langkah pertama yaitu boot DVD CentOS. Pilih 'install or upgrade an existing system".



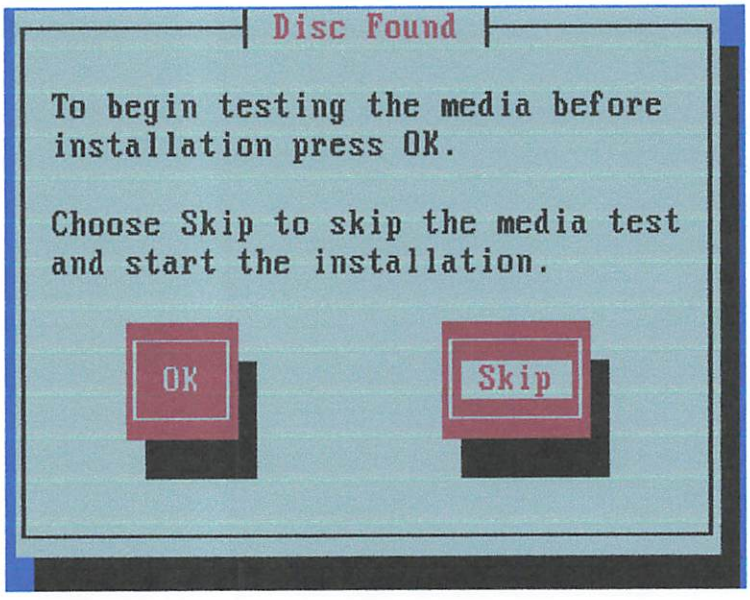

Gambar 4.1
Tampilan awal instalasi

Tekan <tab>, kemudian ketik "text" (tanpa tanda kutip). Untuk proses instalasi akan berjalan dalam mode text.



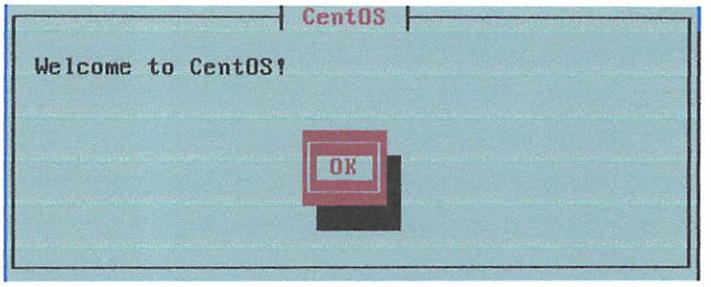

Gambar 4.2
Pilihan Instalasi

Saat muncul pilihan pada gambar 4.3, pengecekan media instalasi, tekan ''OK untuk pengecekan, dan tekan "SKIP" untuk melanjutkan proses pengistalan tanpa pengecekan.



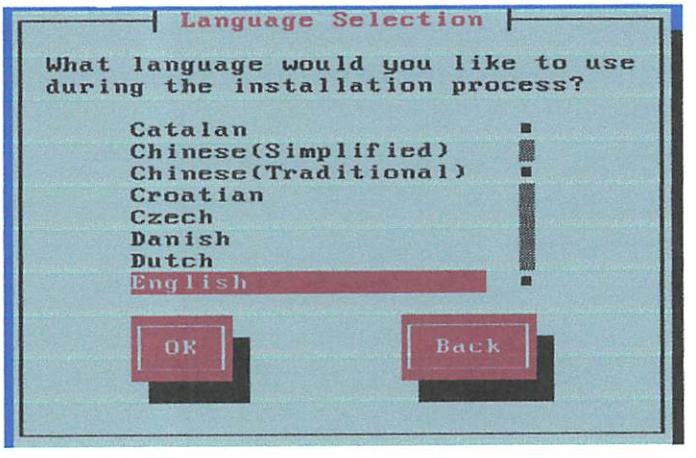

Gambar 4.3
Pengecekan Media Instalasi

Kemudian saat muncul "Welcome to CentOS". Klik "OK".



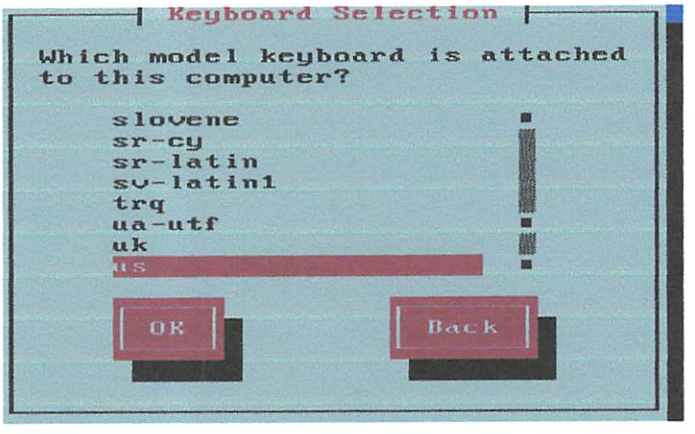

Gambar 4.4
Tampilan Awal Instalasi

Pilih bahasa. Yaitu bahasa yang akan digunakan. Penulis menggunakan bahasa inggris.



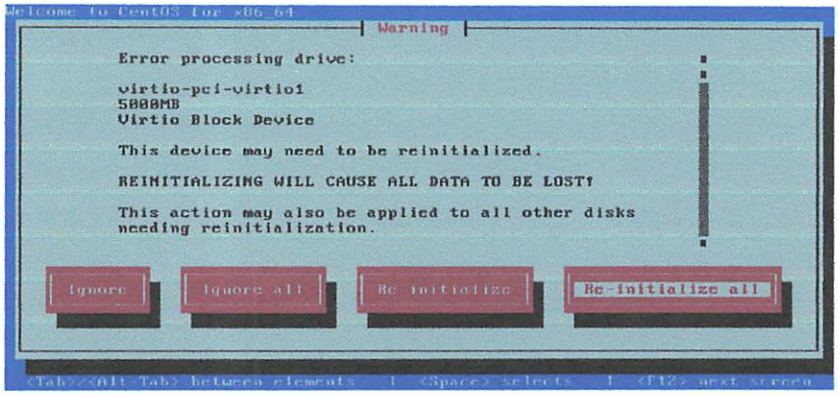

Gambar 4.5
Pemilihan Bahasa

Pilih layout keyboard.



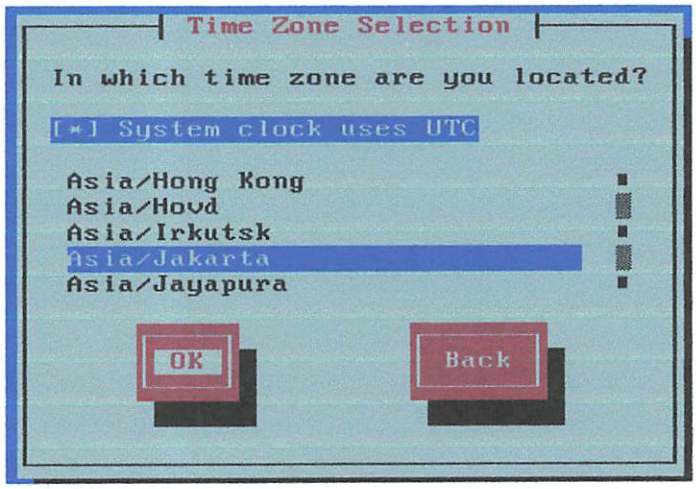

Gambar 4.6
Pemilihan Layout Keyboard

Pilih Re-Initialize all.



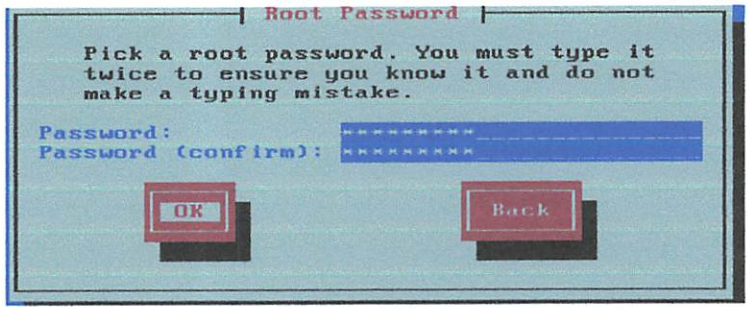

Gambar 4.7
Pemilihan cara instalasi

Kemudian pilih zona waktu.



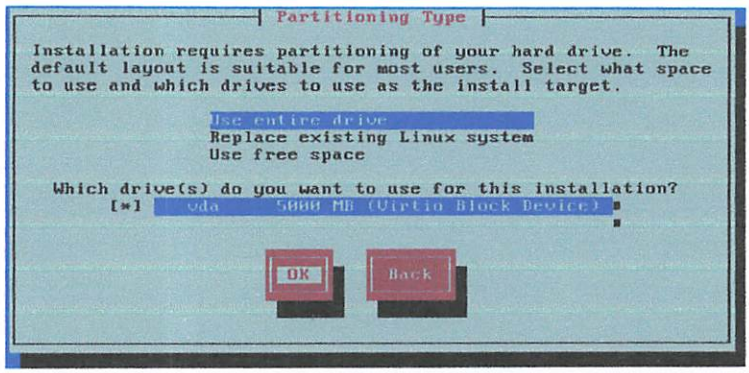

Gambar 4.8
Pemilihan Zona Waktu

Masukkan password root.




Gambar 4.9
Pengisian Password

Pilih "use entire drive" untuk mesin tunggal. Lalu "OK". Jika ingin menggunakan dual boot, pilihlah "use free space".

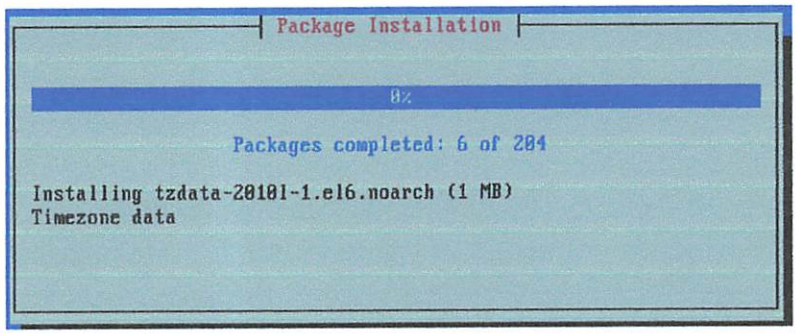



Gambar 4.10
Pemilihan instalasi mesin tunggal

Jika sudah yakin dengan konfigurasi, klik "write changes to disk".

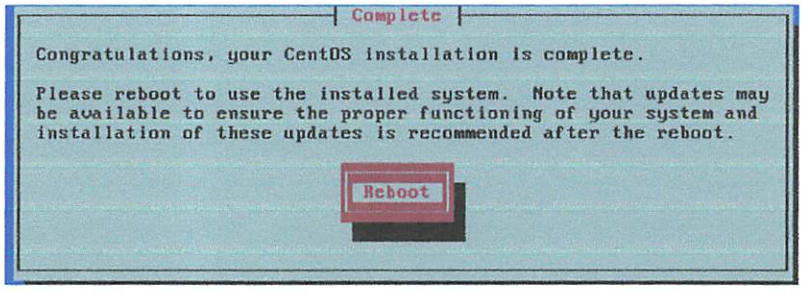

Gambar 4.11
Pilihan disk yang digunakan

Proses instalasi.



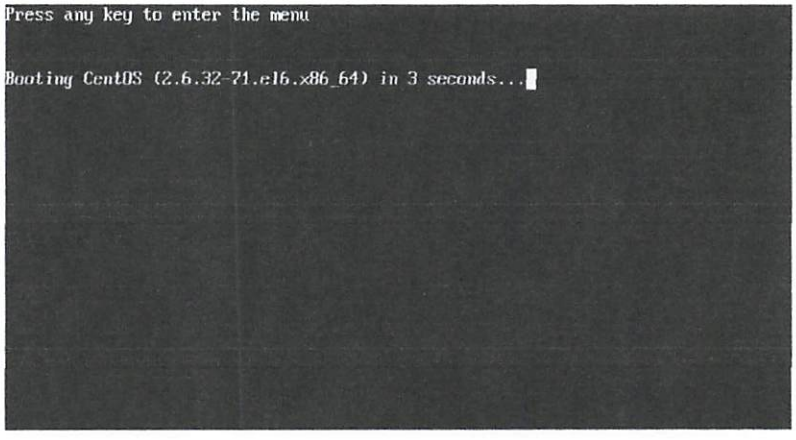

Gambar 4.12
Proses Instalasi

Jika instalasi selesai, klik "reboot".




Gambar 4.13
Instalasi Selesai

Berikut adalah proses booting CentOS.




Gambar 4.14
Proses Booting

Instalasi selesai.

### 4.1.2 Instalasi Web Sever

Hal pertama yang kita lakukan terhadap komputer web server yaitu menginstall Apache web browser, dalam CentOS untuk menginstall apache dengan mengetikkan :

```
# yum install httpd
```

Jika proses instalasi sudah selesai, maka pengaturan IP address untuk web server 1 dan 2.

```
# vi /etc sysconfig/network-scripts/ifcfg-eth0
```

Kemudian isikan dengan skrip berikut untuk web server 1:

```
DEVICE=eth0
BOOTPROTO=static
BROADCAST=192.168.1.255
IPADDR=192.168.1.4
NETMASK=255.255.255.0
NETWORK=192.168.1.0
ONBOOT=yes
TYPE=Ethernet
```

Simpan dan keluar dengan menggunakan ":wq"(tanpa tanda kutip).

Dan untuk setting IP vurtual dengan perintah sebagai berikut :

```
# ip a a 192.168.1.100/24 dev eth0 label eth0:0
```

Untuk web server 2 :

```
DEVICE=eth0
BOOTPROTO=static
BROADCAST=192.168.1.255
IPADDR=192.168.1.5
NETMASK=255.255.255.0
NETWORK=192.168.1.0
ONBOOT=yes
TYPE=Ethernet
```

Simpan dan keluar dengan menggunakan ":wq"(tanpa tanda kutip).

*# ip a a 192.168.1.100/24 dev eth0 label eth0:0*

Pada web server 1 dan 2, ketikkan perintah berikut pada terminal :

```
# vi /etc/sysctl.conf
```

kemudian tekan "i" untuk mengedit script tersebut. Edit script :

```
# net.ipv4.ip_forward=0
```

menjadi :

```
# net.ipv4.ip_forward=1
```

dan pada baris akhir tambahkan scrip berikut :

```
# net.ipv4.conf.eth0.arp_ignore=1
# net.ipv4.conf.eth0.arp_announce=2
# net.ipv4.conf.all.arp_ignore=1
# net.ipv4.conf.all.arp_announce=2
```

Simpan dan keluar dengan menggunakan ":wq"(tanpa tanda kutip).

Dan tahap akhir dari konfigurasi web server 1 dan 2 yaitu dengan mengaktifkan apache. Dengan perintah :

```
# /etc/init.d/httpd start
```

jika terdapat tulisan "OK" berwarna hijau, maka start apache berjalan sukses, seperti pada gambar 4.15.


Gambar 4.15
Aktivasi Apache

### 4.1.3 Instalasi Load Balancer

Dalam konfigurasi load balancer, ada beberapa komponen yang harus diikutsertakan, antara lain piranha sebagai tool load balancing. Didalam piranha sudah terdapat, ipvsadm, apache, httpd dan pulse. Untuk menginstal piranha, buka terminal dan digunakan perintah berikut kemudian masukkan password yang dapat dilihat pada gambar 4.16 :

```
# yum install piranha
```

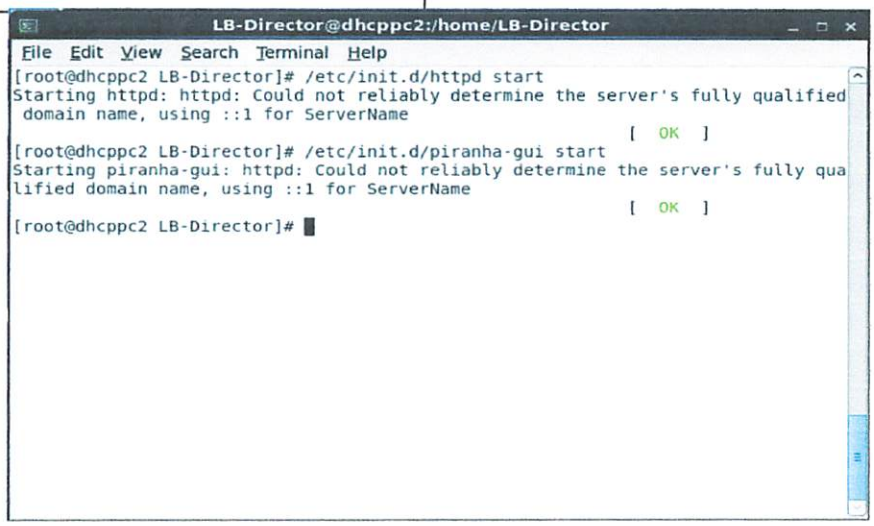
```
#piranha-passwd
```


Gambar 4.16
Input Password Piranha

Setelah proses instalasi selesai, kita masukkan password piranha yang dikehendaki. Kemudian aktifkan apache web server dan piranha GUI. Piranha GUI dimaksudkan agar piranha dapat dikonfigurasi melalui web browser. Dengan perintah sebagai berikut atau dapat dilihat pada gambar 4.17:

```
# /etc/init.d/httpd start
```

```
# /etc/init.d/piranha-gui start
```




Gambar 4.17
Aktivasi Piranha

Jika proses instalasi dan konfigurasi piranha sudah selesai, kemudian pengaturan IP yang akan digunakan. Dalam hal ini instalasi dan pengaturan IP dilakukan pada server load balancing *master* dan *slave*. Buka terminal kemudian ketikkan perintah :

```
# vi /etc sysconfig/network-scripts/ifcfg-eth0
```

Kemudian isikan dengan skrip berikut untuk server *master* :

```
DEVICE=eth0

BOOTPROTO=static

BROADCAST=192.168.1.255
```

```
IPADDR=192.168.1.2
```

*NETMASK=255.255.255.0*

*NETWORK=192.168.1.0*

*ONBOOT=yes*

*TYPE=Ethernet*

Simpan dan keluar dengan menggunakan ":wq"(tanpa tanda kutip).

Dan pada server *slave* isikan script berikut :

*DEVICE=eth0*

*BOOTPROTO=static*

*BROADCAST=192.168.1.255*

*IPADDR=192.168.1.3*

*NETMASK=255.255.255.0*

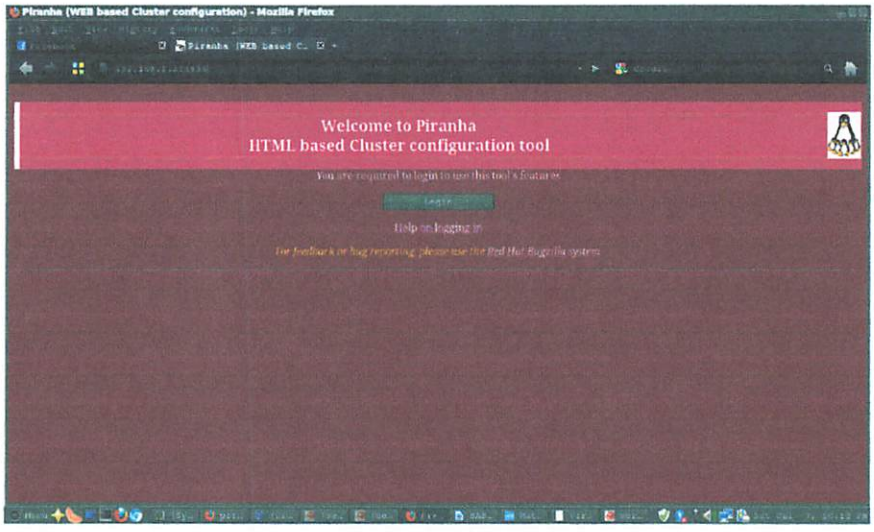
*NETWORK=192.168.1.0*

*ONBOOT=yes*

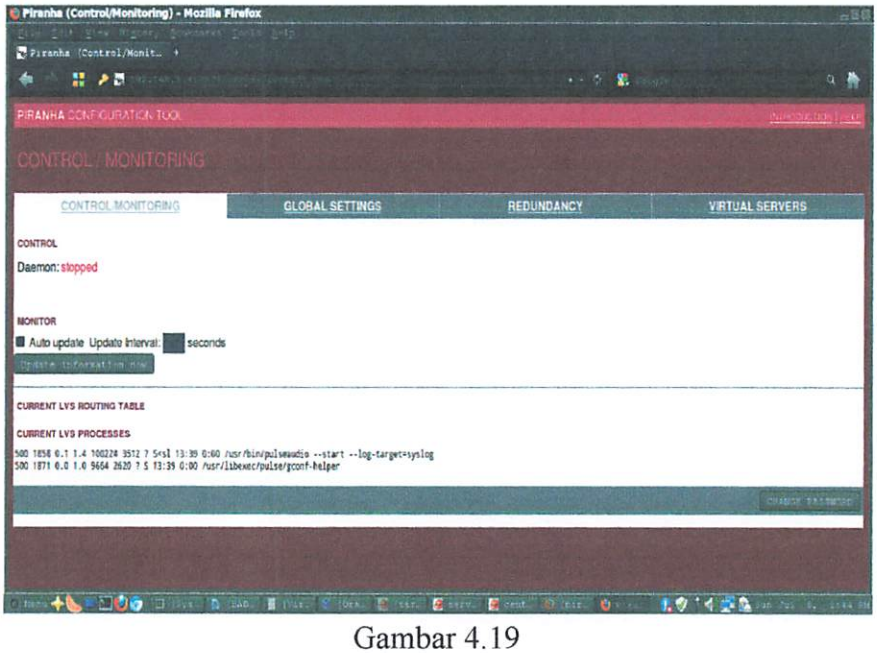
*TYPE=Ethernet*

Dalam proses diatas merupakan dasar dari sistem load balancing yaitu penginstalan komponen yang akan digunakan dan pengaturan IP. Dalam load balancing, diperlukan sebuah tool yang dinamakan PIRANHA. Piranha merupakan tool yang mendukung terhadap suatu teknologi Load Balancing. Dengan tampilan secara GUI (Graphical User Interface) melalui web browser, akan semakin mempermudah administrator untuk mengkonfigurasi loadbalancing. Selain secara GUI, kita juga dapat menambahkan script kedalam direktori /etc/sysconfig/ha/lvs.conf. Dan script tersebut akan saling berhubungan dengan piranha.

Agar dapat mengakses piranha melalui web browser, dapat memasukkan alamat ip address load balancing diikuti titik dua (:)3636 seperti pada gambar 4.18.

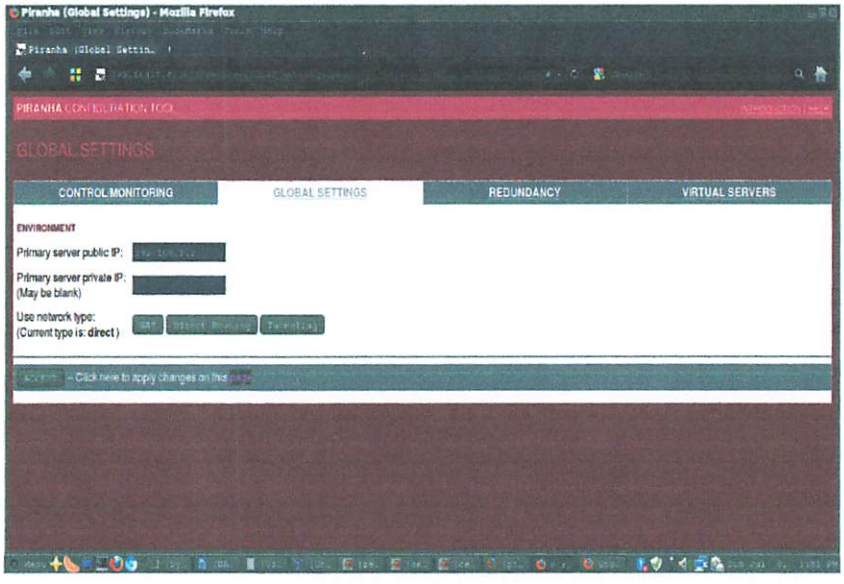



Gambar 4.18
Halaman Awal Piranha



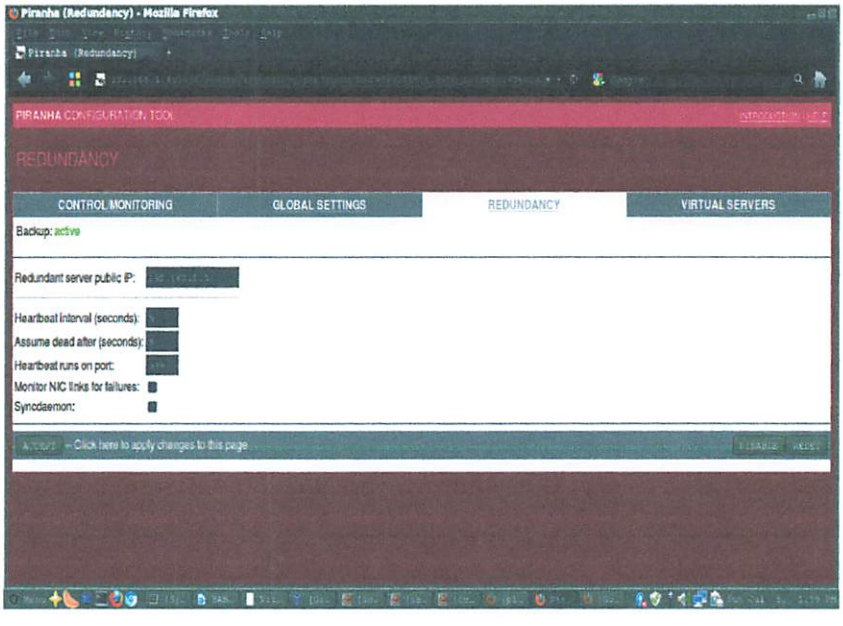

Gambar 4.19
Halaman Control Monitoring

Pada gambar 4.18 merupakan halaman awal piranha saat diakses. Dan pada gambar 4.19 merupakan Control/Monitoring yang menampilkan status daemon dan menunjukkan LVS table routing.




Gambar 4.20
Halaman Global Setting

Dalam gambar 4.20, Global setting, yaitu tempat informasi mengenai server Load Balancer primary atau Load Balancer Active. Global setting dapat diisi dengan IP yang menjadi server *master.*




Gambar 4.21
Halaman Redundancy

Pada gambar 4.21 atau redundancy, menyediakan pengaturan untuk LVS Heatbeat. Dalam pengaturan ini penulis mengisikan IP address server *load balancer slave* yaitu 192.168.1.3.