PENERAPAN PENGOLAHAN PARALEL MODEL CLUSTER SEBAGAI WEB SERVER

Maman Somantri, Ari Darmariyadi
Teknik Elektro Fakultas Teknik Universitas Diponegoro
Jl. Prof. Sudharto, Tembalang Semarang
Email : mmsomantri@elektro.ft.undip.ac.id

Intisari

Pengolahan paralel merupakan suatu cara yang dilakukan untuk meningkatkan kecepatan pengolahan data dengan melakukan lebih dari satu pengolahan data tersebut secara bersamaan. Salah satu bentuk pengolahan paralel adalah model cluster. Pengolahan paralel model cluster ini akan digunakan untuk mengolah data Web, dengan membangun server Web yang di-cluster. Cluster server Web ini menggunakan teknologi Linux Virtual Server (LVS) yang dapat dilakukan dengan NAT, IP tunneling, dan direct routing yang memiliki empat algoritma penjadwalan.

Pada penelitian ini akan digunakan teknologi LVS untuk membuat cluster Web Server dengan menggunakan NAT, diterapkannya teknologi Network File System, dan Network Block Device yang digunakan sebagai media penyimpanan dalam jaringan.. Dalam pengujian sistem cluster ini, pertama dilakukan pengujian jaringan yang digunakan untuk mengetahui kinerja sistem, dan pengujian sistem cluster dalam mengolah data Web dengan perangkat lunak WebBench dan script benchmark.

Kata kunci : *Pengolahan data, Cluster, Web server.*

1. PENDAHULUAN

Dewasa ini internet menjadi suatu kebutuhan yang sangat penting bagi seluruh lapisan masyarakat di dunia, baik itu bagi kalangan pelajar, ilmuwan, dan usahawan. Hal ini menyebabkan semakin meningkatnya permintaan akan kebutuhan informasi dalam internet, sehingga trafik dalam internet semakin padat oleh permintaan-permintaan akan informasi. Semakin meningkatnya trafik dalam internet menyebabkan beban kerja pada server penyedia layanan internet tersebut juga meningkat seiring dengan bertambahnya permintaan yang masuk, sehingga server tersebut akan kelebihan beban dalam waktu yang pendek, terutama untuk server yang menyediakan layanan yang populer. Sehingga untuk mengatasi masalah kelebihan beban server tersebut, terdapat dua penyelesaian. Pertama dengan meningkatkan server yang ada dengan server berkinerja tinggi, hal ini untuk sementara memang dapat menyelesaikan masalah yang ada sekarang ini, tetapi ketika permintaan meningkat lebih tinggi lagi maka akan dengan cepat kelebihan beban kembali, sehingga memerlukan peningkatan kembali, proses peningkatan ini sangat rumit dan memerlukan biaya yang besar. Cara kedua dengan membangun server yang mempunyai skalabilitas yang tinggi dengan membuat server yang di-cluster, dengan kata lain jika beban kerja meningkat, maka dapat dengan mudah menambahkan sebuah server yang baru atau lebih ke dalam sistem cluster untuk memenuhi peningkatan permintaan yang terus menerus bertambah.

Sistem server cluster yang dibangun dengan menggunakan *load balancer* untuk menyalurkan beban di antara server-server yang ada dalam sebuah cluster. Layanan paralel dari server dapat dibuat tampak sebagai layanan virtual pada alamat IP tunggal, maka pengguna akan melihat virtual server, bukan server cluster. Penyebaran penjadwalan adalah per koneksi, yang dapat membuat beban yang seimbang di antara server-server. Kegagalan dapat disamarkan ketika satu atau lebih server gagal. Hal inilah yang membuat saya ingin menguji penerapan sistem server cluster ini dalam melayani permintaan halaman web yang ada sebagai penelitian.

2. TINJAUAN PUSTAKA

2.1 Konsep Dasar Pengolahan Paralel

Sistem pemrosesan dengan lebih dari satu prosesor dilakukan untuk meningkatkan kinerja dari sistem. Peningkatan kecepatan pemrosesan yang sempurna hampir tidak mungkin dicapai, disebabkan semua komponen dalam sistem memiliki proses yang berurutan, fase inisialisasi yang berbeda setiap prosesnya, pembacaan data dan pengumpulan hasil yang juga membutuhkan waktu proses.

Sistem multi prosesor secara garis besar dapat dibagi berdasarkan empat hal, antara lain:

a. Struktur logika

Struktur logika di sini mengenai bagaimana tanggung jawab pengontrolan diserahkan pada elemen sistem. Dua bentuk hubungan yang sangat jelas yaitu vertikal dan horisontal. Pada sistem vertikal, elemen secara hirarki terstruktur, sedangkan pada sistem horisontal, elemen secara logika sama.

b. Struktur fisik

Struktur fisik dari sistem multi prosesor mengacu pada metode pertukaran informasi dan merupakan fungsi dari pengaturan komunikasi antar prosesor dan topologi interkoneksi.

c. Mode operasi

Secara umum, komputer digital diklasifikasi ke dalam empat kategori, menurut keragaman dari aliran (*stream*) instruksi dan data.

Terdapat empat organisasi komputer berdasarkan mode operasi sebagai berikut:

1. SISD (*Single Instruction stream Single Data stream*)
2. SIMD (*Single Instruction stream Multiple Data stream*)
3. MISD (*Multiple Instruction stream Single Data stream*)
4. MIMD (*Multiple Instruction stream Multiple Data stream*)

d. Mode interaksi.

Berdasarkan mode interaksi sebagai dasar, sistem dapat diklasifikasikan dengan tingkat hubungan (*coupling*) dan cara komunikasi antara prosesor-prosesor, dibagi menjadi dua yaitu:

a. Sistem *loosely coupled*
   Sistem ini juga dikenal sebagai jaringan komputer.

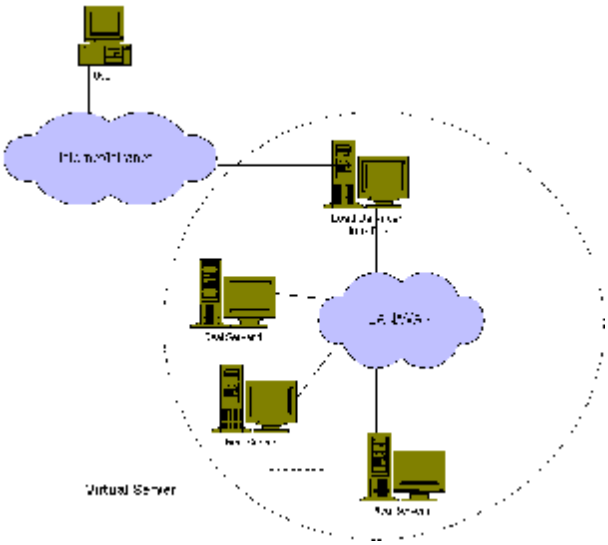
b. Sistem *tightly coupled*
   Sistem ini juga dikenal sebagai sistem multi prosesor.

2.2 Virtual Server Cluster

Server cluster dimana server-servernya dihubungkan oleh jaringan yang cepat dan merupakan suatu arsitektur untuk membangun server yang mempunyai kinerja dan tingkat ketersediaan layanan yang tinggi. Bentuk rancangan server cluster (*loose coupled*) ini lebih mudah untuk dikembangkan, lebih efektif dalam biaya dan lebih handal dibandingkan dengan prosesor tunggal atau sistem multiprosesor *tightly-coupled.*

Virtual server merupakan server yang mudah dikembangkan dan tingkat ketersediaannya tinggi yang dibangun dari cluster server independen. Aplikasi client berinteraksi dengan cluster seperti seandainya menggunakan sebuah server dengan kinerja dan tingkat ketersediaan yang tinggi. *Client* tidak terpengaruh oleh interaksi yang ada dalam cluster dan tidak memerlukan suatu modifikasi. Arsitektur virtual server yang umum dapat dilihat pada gambar 1.

Pada gambar dapat dilihat bahwa *real server* yang merupakan server yang sebenarnya pada virtual server cluster dihubungkan dengan menggunakan LAN/WAN pada sebuah *director* (*load balancer*), yang mengatur jadwal permintaan pada server yang berbeda dan melakukan layanan paralel dari cluster agar dilihat sebagai sebuah virtual server dengan alamat IP tunggal. Peningkatan atau pengembangan server secara transparan dilakukan dengan menambahkan atau menghilangkan sebuah *real server* dalam cluster. Tingkat ketersediaan layanan yang tinggi dapat dilakukan dengan pendeteksian *real server* atau kesalahan daemon dan pengkonfigurasian sistem dengan benar.



Gambar 1: Arsitektur virtual server secara umum

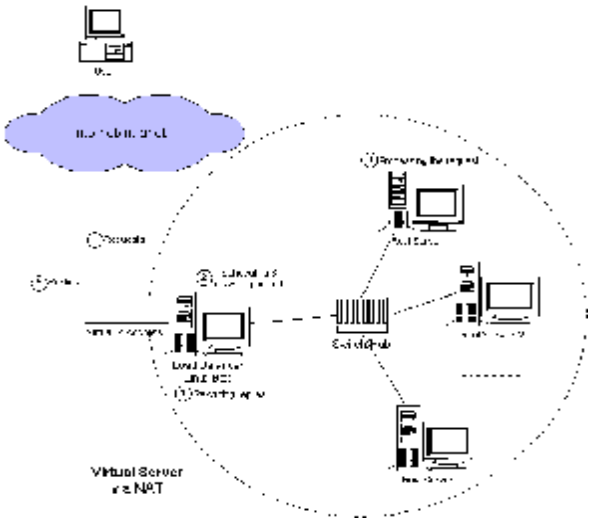
2.2.1 Arsitektur Virtual Server

Virtual server dapat diimplementasikan dengan tiga metode, yaitu virtual server via *Network Address Translation* (NAT), virtual server via *direct routing*, dan virtual server via *IP tunneling*.

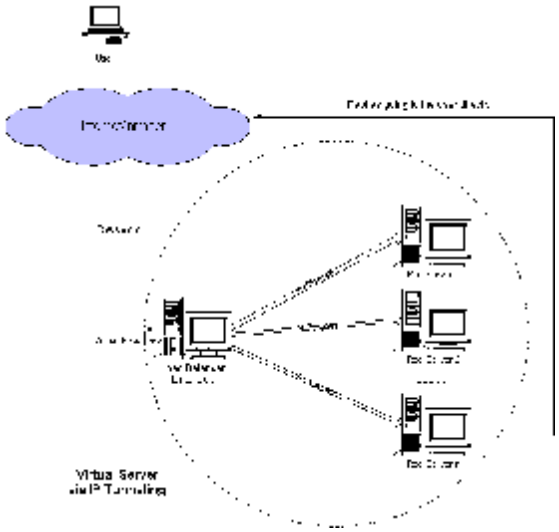
Virtual Server via NAT

Berdasarkan keterbatasan alamat IP dalam ipv4 dan beberapa alasan keamanan, banyak jaringan menggunakan alamat IP lokal yang tidak dapat digunakan di luar jaringan (atau dalam internet). Kebutuhan akan n*etwork address translation* timbul ketika hosts dalam jaringan lokal ingin mengakses internet atau ingin diakses melalui internet. NAT berdasar pada fakta bahwa *header* untuk protokol internet dapat diatur secara benar, karena itu client merasa menghubungi satu alamat IP, tetapi server dengan alamat IP yang berlainan merasa dihubungi secara langsung oleh client. Hal ini dapat digunakan untuk membuat sebuah virtual server, misalnya layanan paralel pada alamat IP yang berbeda dapat tampak seperti layanan pada sebuah alamat IP tunggal.

Arsitektur virtual server via NAT dapat dilihat pada gambar 2. *Director* dan *real server* dihubungkan oleh sebuah switch atau hub. *Real server* biasanya menjalankan layanan yang sama dan mempunyai layanan yang isinya sama. *Director* bertugas sebagai pembagi permintaan pada *real server* yang berbeda melalui NAT.



Gambar 2: Arsitektur virtual server via NAT



Gambar 3: Arsitektur virtual server via *IP tunneling*

Aliran kerja dari virtual server via NAT adalah sebagai berikut: ketika *user* mengakses layanan yang disediakan oleh server cluster, paket permintaan yang diarahkan pada alamat IP maya (alamat IP eksternal untuk *director*) datang pada *director*. *Director* akan memeriksa alamat tujuan paket dan nomor port. Jika sesuai dengan layanan virtual server menurut tabel aturan virtual server. *Real server* yang melayani dipilih dalam cluster dengan algoritma penjadwalan (*schedulling*), dan hubungan ditambahkan pada tabel hash yang menyimpan semua koneksi yang berlangsung. Kemudian, alamat tujuan dan port dari paket ditulis ulang pada *real server* yang dipilih, dan paket dikirimkan pada *real server* tersebut. Ketika paket datang melalui koneksi ini (koneksi yang berlangsung dapat ditemukan dalam tabel hash), paket akan ditulis ulang dan dikirimkan ke server yang dipilih. Ketika tanggapan paket diterima, *director* menulis ulang alamat asal port dari paket untuk layanan tersebut. Ketika koneksi dibatalkan atau habis waktunya, catatan koneksi akan dihilangkan dari tabel hash.

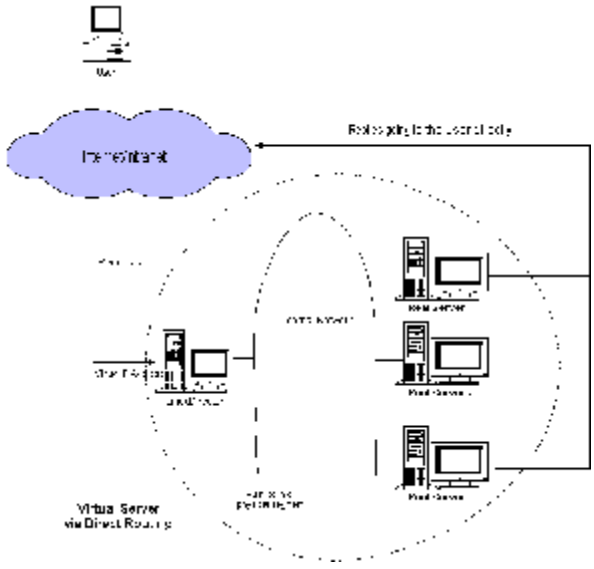
Virtual Server via IP Tunneling

IP tunneling (*IP encapsulation*) merupakan teknik untuk enkapsulasi datagram IP dari semua datagram IP, yang mengijinkan datagram yang dituju untuk satu alamat IP diselubungi dan diarahkan pada alamat IP lainnya. Teknik ini dapat digunakan untuk penerapan virtual server, dimana *director* mengarahkan paket permintaan pada server yang berbeda, server akan memproses permintaan dan mengembalikan hasilnya secara langsung pada client, dan layanan akan tampak sebagai layanan pada alamat IP tunggal. Arsitektur virtual server via IP tunneling dapat dilihat pada gambar 3.

Urutan kerja virtual server via IP tunneling adalah sebagai berikut: ketika *user* mengakses layanan yang disediakan oleh server cluster, paket yang diarahkan pada alamat IP dari *director* tiba, kemudian *director* akan memeriksa alamat tujuan paket dan port, jika sesuai untuk layanan virtual server, satu virtual server dipilih dari sistem cluster dengan algoritma penjadwalan, dan koneksi ditambahkan ke dalam tabel hash yang mencatat semua koneksi yang terjadi. Kemudian, *director* mengenkapsulasi paket dengan datagram IP dan mengarahkannya pada server yang dipilih sebelumnya. Ketika paket datang melalui hubungan ini dan *real server* yang dipilih dapat ditemukan dalam tabel hash, paket akan dienkapsulasi dan diarahkan pada real server. Ketika *real server* menerima paket terenkapsulasi, *real server* tersebut akan mendekapsulasi paket dan memproses permintaan, terakhir mengembalikan hasil pada *user* secara langsung. Ketika koneksi dibatalkan atau habis waktunya, catatan koneksi akan dihilangkan dari tabel hash.

Virtual Server via Direct Routing

Teknik pengiriman permintaan *Direct Routing* ini menggunakan alamat virtual IP yang sama dari *real server* dan *director*. Alamat virtual IP pada *director* digunakan untuk menerima paket permintaan, dan kemudian secara langsung mengirimkan paket pada server yang telah dipilih. Semua *real server* menggunakan perangkat *alias non-arp* sebagai alamat virtual IP. *Director dan semua real server* harus memiliki perangkat ethernet yang dihubungkan dengan sebuah hub/switch. Arsitektur virtual server via *direct routing* dapat dilihat pada gambar 4.



Gambar 4. Arsitektur virtual server via *direct routing*

Urutan kerja virtual server via *direct routing* adalah sebagai berikut: ketika *user* mengakses layanan virtual yang disediakan oleh sistem cluster, paket yang ditujukan untuk alamat virtual IP sampai pada *director*. *Director* kemudian memeriksa alamat dan port tujuan paket. Jika sesuai dengan layanan virtual, sebuah *real server* akan dipilih dari cluster dengan algoritma penjadwalan, dan koneksi dimasukkan pada tabel hash yang menyimpan koneksi yang berlangsung. Kemudian, *director* secara langsung mengirimkan paket tersebut pada server yang dipilih. Ketika paket untuk koneksi layanan ini dan server yang dipilih dapat ditemukan dalam tabel hash, paket akan langsung dikirimkan lagi pada server. Ketika server menerima paket yang ditujukan untuknya, server akan menemukan bahwa paket adalah untuk alamat perangkat *alias*-nya, maka permintaan akan diproses dan mengirimkan hasilnya secara langsung pada *user*. Setelah koneksi dihentikan atau waktu habis, catatan koneksi akan dihapus dari tabel hash.

2.2.2 Algoritma Penjadwalan

Dalam memilih *real server* dari sistem cluster diterapkan empat algoritma penjadwalan, antara lain: Round-Robin, Weighted Round-Robin, Least-Connection dan Weighted Least-Connection.

Penjadwalan Round-Robin

Algoritma penjadwalan Round-Robin, dalam arti kata, mengarahkan koneksi jaringan pada server yang berbeda dalam bentuk round-robin, yang memperlakukan semua *real server* sama menurut jumlah koneksi atau waktu respon.

Penjadwalan Weighted Round-Robin

Penjadwalan weighted round-robin ini memperlakukan *real server* dengan kapasitas proses yang berbeda. Masing-masing server dapat diberikan bobot, yaitu bilangan integer yang menunjukkan kapasitas proses (bobot awal adalah 1).

Penjadwalan Least-Connection

Algoritma penjadwalan least-connection mengarahkan koneksi jaringan pada server dengan jumlah koneksi aktif yang paling sedikit. Penjadwalan ini merupakan salah satu algoritma penjadwalan dinamik, karena memerlukan penghitungan koneksi aktif untuk masing-masing *real server* secara dinamik. Pada virtual server dimana terdapat sekumpulan server dengan kinerja yang mirip, penjadwalan least-connection baik untuk melancarkan pendistribusian ketika beban permintaan bervariasi banyak, karena semua permintaan panjang tidak akan mempunyai kesempatan untuk diarahkan pada server.

Penjadwalan Weighted Least-Connection

Penjadwalan weighted least-connection merupakan sekumpulan penjadwalan least-connection, dimana dapat ditentukan bobot kinerja pada masing-masing real server. Server dengan nilai bobot yang lebih tinggi akan menerima persentase yang lebih besar dari koneksi-koneksi aktif pada satu waktu. Bobot pada masing-masing *real server* dapat ditentukan dan koneksi jaringan dijadwalkan pada masing-masing server dengan persentase jumlah koneksi aktif untuk masing-masing server sesuai dengan perbandingan bobotnya (bobot awal adalah 1).

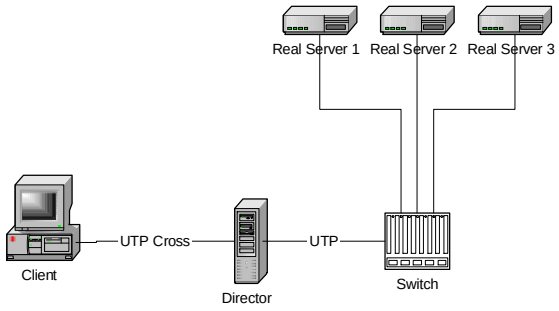
3. PERANCANGAN DAN KONFIGURASI SISTEM KOMPUTER CLUSTER SEBAGAI WEB SERVER

Virtual server cluster merupakan satu grup server dengan *director* yang tampak dari dunia luar (*client* pada internet) sebagai sebuah server. Virtual server dapat menawarkan beberapa jenis layanan, atau layanan dengan kapasitas/*throughput* yang lebih tinggi, atau layanan dengan kelebihan dimana sebuah server dapat dilepaskan dari cluster untuk perawatan. Virtual server cluster ini memerlukan komponen-komponen sebagai berikut:

a. *Client*, sebagai sebuah komputer dengan telnet *client* dan http *client*.
b. *Director*, sebagai sebuah server yang tampak dari dunia luar (*client* pada internet), yang mengarahkan permintaan *client* agar sampai pada *real server*.
c. *Real server*, sebagai server yang melayani permintaan *client*, baik itu layanan telnet, http, ftp, dan lain-lain.

3.1 Sistem Virtual Server Cluster

Sistem virtual server cluster yang dirancang adalah virtual server via *network address translation* (NAT) dengan beban pada *real server* diseimbangkan oleh *director* dengan menggunakan metode penjadwalan *Round-Robin*. Arsitektur virtual server via NAT yang diterapkan dapat dilihat pada gambar 5. *Director* (*load balancer*) dan *real server* dihubungkan oleh sebuah switch atau hub. *Real server* menjalankan layanan yang sama yaitu http. Halaman web yang dilayani oleh layanan ini disimpan dalam sebuah *network filesystem* yang disediakan oleh *director*. *Director* membagi permintaan pada *real server* yang berbeda via NAT. Untuk hubungan *client* dengan *director* digunakan kabel UTP *cross*.



Gambar 5: Arsitektur virtual server yang digunakan

3.2 Perangkat Lunak

Perangkat lunak yang digunakan untuk menerapkan virtual server via NAT ini adalah Red Hat Linux 6.2.

3.2.1 *Director*

Director menggunakan sistem operasi Red Hat Linux 6.2 dengan menggunakan kernel 2.2.17 yang telah diberi *patch* untuk dapat menangani virtual server, *director* pada sistem yang dirancang ini bertindak juga sebagai *reverse address resolution protocol* server (server RARP), *network file server* (server NFS) dan *network block device server* (server NBD).

3.2.2 *Real server*

Real server menggunakan sistem operasi Red Hat Linux 6.2 dengan kernel 2.2.17. Sistem operasi ini berada pada *network file system* yang disediakan oleh *director*, sehingga akan menggunakan sumber daya yang ada pada *director* dalam hal ini media penyimpanan.

3.2.3 *Client*

Client pada dasarnya dapat menggunakan sistem operasi apa saja yang menggunakan http *client*. Pada pengujian digunakan sistem operasi Windows NT 4.0, karena memiliki kemampuan untuk jaringan yang lebih baik dibandingkan dengan Windows 95/Windows 98.

3.3 Teknik Konfigurasi Sistem Komputer pada Cluster

Sistem virtual server yang dirancang ini menggunakan *real server* yang sistem operasinya berada pada *director* dengan menggunakan fasilitas *network file system* dan *network block device* yang disediakan oleh linux. Untuk itu diperlukan konfigurasi-konfigurasi pada *director* agar dapat berlaku sebagai server RARP, server NFS dan server NBD juga pada *real server* agar dapat memanfaatkan fasilitas yang disediakan oleh *director*.

3.3.1 /etc/hosts

File /etc/hosts ini berisi alamat IP beserta nama host yang ada pada sistem komputer cluster yang dirancang. File ini berguna untuk memudahkan dalam mengingat nama dari suatu host untuk alamat IP pada sistem komputer cluster ini.

3.3.2 *Domain Name System*

Domain Name System (DNS) ini digunakan untuk memetakan nama suatu host pada alamat IP maupun sebaliknya. DNS ini mempunyai komponen *resolver*, yang merupakan bagian dari aplikasi yang berfungsi menjawab pertanyaan aplikasi tentang domain. Pada linux, *resolver* ini dikonfigurasi pada file /etc/resolve.conf yang digunakan oleh linux untuk menentukan name server untuk mnguraikan suatu IP atau nama.

DNS ini pada sistem linux diimplementasikan oleh software BIND (*Barkeley Internet Name Domain*). Dalam menjalankan server DNS pada suatu host, diperlukan file-file sebagai berikut:

Konfigurasi file /etc/named.conf

File ini berisi jenis server apa saja yang akan dibuat, apa saja nama-nama file domain dan *reverse*-nya.

Konfigurasi /etc/named.boot

File ini berisi informasi inisialisasi domain, yaitu tipe server DNS yang dijalankan, daftar zone tempat server DNS ini memiliki wewenang, serta lokasi file atau server lain tempat server DNS ini bisa mendaptkan data awalnya.

Konfigurasi file database domain

File ini berisi nama-nama host yang berada dalam sebuah domain. Sebuah domain biasanya mempunyai beberapa subdomain lagi dan sebaiknya ditulis dalam file database yang berbeda. File ini selain berisi informasi nama-nama host juga ada bagian yang menyatakan pendelegasian, pengaturan *mail exchanger* dan *aliasing*

- Pendelegasian

  Hal ini dilakukan apabila ingin membuat subdomain dan memberikan otoritas penuh

pada sebuah DNS lain untuk memegang databasenya

- *Mail Exchanger*
  Mail exchanger adalah sebuah server yang menyediakan layanan untuk menerima dan meneruskan mail untuk root dari host lainnya.
- Aliasing
  Aliasing berfungsi untuk memberikan nama lain kepada sebuah host yang sudah mempunyai nama.

Konfigurasi file database reverse domain

File ini berisi database kebalikan dari database domain. Dalam database domain sebelumnya dinyatakan dari nama host ke alamat IP sedangkan pada reverse database sebaliknya, yaitu dari alamat IP ke nama host.

Pengkonfigurasian DNS digunakan pada jaringan komputer cluster yang dirancang ini berguna untuk mempercepat dalam pencarian nama-nama host yang terdapat dalam domain pada jaringan lokal.

3.3.3 *Reverse Address Resolution Protocol*[4][18]

Reverse Address Resolution Protocol (RARP) merupakan protokol yang berguna untuk mengadakan translasi dari alamat perangkat keras yang diketahui menjadi alamat IP. RARP ini memanfaatkan alamat perangkat keras ethernet yang berbeda antara satu dengan yang lain untuk mendapatkan alamat IP masing-masing. Dengan RARP ini, server RARP yang memiliki alamat perangkat keras ethernet dalam cache-nya akan menunggu *client* yang menggunakan alamat perangkat keras ethernet yang terdapat pada cache server RARP, jika terdapat *client* dengan alamat ethernet yang sama dengan yang terdapat pada cache server RARP maka server RARP akan memberikan alamat IP yang telah ditunjukkan pada cache server RARP.

3.3.4 *Network File System*

RARP di atas digunakan untuk mendapatkan alamat IP yang diperoleh dari pengenalan alamat ethernet, sedangkan agar sistem tanpa harddisk tersebut dapat memperoleh sistemnya sendiri digunakan *network file system* yang diletakkan pada server NFS, di sini yang bertindak sebagai server NFS yaitu *director*.

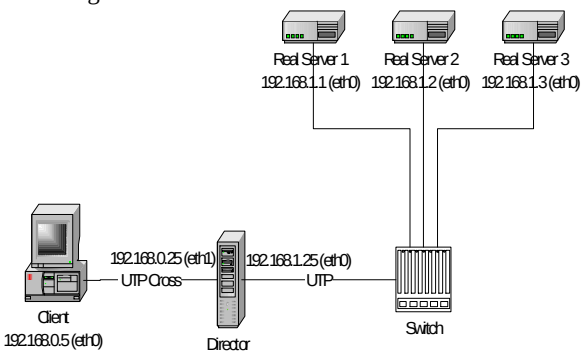
3.3.5 *Network Block Device*

NBD merupakan konsep menggunakan media penyimpanan yang terdapat pada jaringan atau dalam hal ini pada server untuk dijadikan sebagai tempat penyimpanan bagi yang memerlukan media penyimpanan tambahan. NBD ini hampir mirip dengan NFS hanya saja NBD ini dapat menyimpan sistem file apapun termasuk sistem file DOS/Windows, Windows NT, dan sebagainya. Pada sistem komputer cluster ini NBD akan digunakan sebagai *swap memory* bagi *real server*, yang bertindak sebagai server NBD adalah *director*.

3.4 Teknik Konfigurasi Komputer Cluster sebagai Web Server

Setelah komputer yang akan digunakan sudah selesai dikonfigurasi, maka menuju langkah selanjutnya menghubungkan komputer yang ada pada jaringan lokal sebagai sistem komputer cluster yang nantinya akan melaksanakan tugasnya sebagai Web Server. Konfigurasi dari sistem komputer cluster yang digunakan dapat dilihat pada gambar 6

Pada bagian berikut akan dijelaskan, cara-cara melakukan konfigurasi komputer cluster dengan menggunakan tipe *forwarding* NAT dengan arsitektur sebagai berikut:



Gambar 6 Konfigurasi sistem komputer cluster

3.4.1 Apache Server

Untuk menjalankan tugasnya sebagai suatu Web Server, maka real server harus menjalankan perangkat lunak Web Server. Pada penelitian ini digunakan Web Server Apache. Web Server Apache mempunyai kelebihan sebagai berikut:

- Termasuk perangkat lunak dalam kategori *freeware*
- Dapat beroperasi pada berbagai sistem operasi seperti FreeBSD, Linux, Solaris, Windows dan lain-lain.
- Mudah dalam pengkonfigurasian, hanya httpd.conf untuk versi 1.3.3 ke atas.

3.4.2 Ipvsadm[13][14][18]

Ipvsadm ini sebagai media antarmuka antara pengguna dengan virtual server cluster yang merupakan sebuah utility untuk melakukan administrasi layanan IP virtual server yang disediakan oleh kernel Linux. Hal-hal yang dapat dilakukan oleh ipvsadm dalam melakukan administrasi virtual server cluster adalah sebagai berikut:

- Mengkonfigurasi tipe *forwarding* yang akan digunakan, antara lain: NAT, *direct routing*, dan *tunneling*.

- Mengkonfigurasi layanan dan server yang akan diarahkan oleh *director*.
- Memberikan bobot pada masing-masing *real server*, yang berguna jika terdapat server lebih cepat dibandingkan dengan yang lain.
- Mengatur algoritma penjadwalan, antara lain: *Round-Robin*, *Weighted Round-Robin*, *Least-Connection* dan *Weighted Least-Connection.*

3.4.3 *Forwarding* dengan *Network Address Translation*

Untuk melakukan *forwarding* ini digunakan ipchains yang berguna untuk mengkonfigurasi, menjaga dan mengecek aturan IP *firewall* dalam kernel linux. ipchains ini dapat mengatur 3 kategori yang berbeda, antara lain: *IP input chain*, *IP output chain*, dan *IP forwarding chain*.

- *IP input chain* digunakan untuk mengecek semua paket yang datang memasuki firewall server.
- *IP output chain* digunakan untuk mengecek semua paket dikirimkan keluar firewall server.
- *IP forwarding chain* digunakan untuk mengecek semua paket yang ingin melewati *firewall* server pada mesin atau jaringan yang lain.

3.5 Perangkat Lunak Penguji Sistem Web Server Cluster

3.5.1 NetPIPE

Analisa kinerja jaringan diperlukan untuk memperkirakan kinerja dari aplikasi komunikasi baik itu dalam jaringan maupun protokol yang digunakan. Untuk menganalisa kinerja dari jaringan sistem komputer cluster pada penelitian ini digunakan perangkat lunak NetPIPE (*Network Protocol Independent Performance Evaluator*).

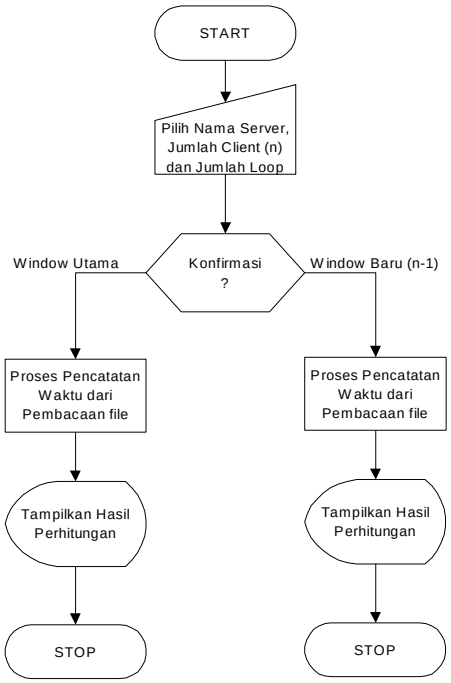
NetPIPE ini merupakan alat pengukur kinerja dari protokol independen yang memvisualisasikan kinerja jaringan dengan kondisi yang bervariasi. NetPIPE bekerja pada hubungan jaringan dengan menggunakan protokol TCP/IP. NetPIPE hanya akan mengirimkan blok data pada jaringan tanpa mengalirkan blok data lain dalam pesan yang dikirimkan, sehingga akan menghasilkan waktu transfer dari sebuah blok tunggal, juga menyediakan informasi lain yang diperlukan seperti ukuran blok mana yang memberikan *throughput* terbaik, *throughput* yang diberikan oleh sebuah blok berukuran k (Bytes/bits). NetPIPE ini akan menghasilkan keluaran berupa file yang terdiri atas lima kolom antara lain: *latency* (sec), *throughput* (bps), jumlah bit dalam blok (bit), jumlah Bytes dalam blok (Bytes) dan variance. Di mana yang dijadikan parameter yang berguna dalam pengukuran kinerja dengan menggunakan NetPIPE adalah *latency*, *throughput* dan ukuran paket. *Latency* (waktu transfer) merupakan waktu yang dibutuhkan CPU untuk mengirimkan paket dan menerima jawaban dari paket, *throughput* adalah besarnya paket yang dapat dikirimkan per detik, sedangkan ukuran paket merupakan besarnya paket yang dikirimkan pada waktu pengujian.

3.5.2 WebBench

Untuk menganalisa kinerja dari sistem cluster yang dirancang dalam kerjanya sebagai web server digunakan perangkat lunak WebBench 4.0.1. WebBench 4.0.1 ini merupakan sebuah program berlisensi Ziff Davis benchmark untuk mengukur kinerja dari web server. WebBench ini terdiri dari sebuah controller dan dapat mensimulasikan banyak client. WebBench ini menggunakan PC sebagai client untuk mengirimkan permintaan pada server, dengan ukuran file yang bervariasi. Pada WebBench ini disimulasikan beberapa client yang mensimulasikan web browser yang dapat ditentukan maksimal 60 client per satu PC. Ketika server selesai menanggapi permintaan client, client akan menyimpan informasi seperti lama server dalam melayani dan besarnya data yang dikirimkan pada client, seteleh selesai disimpan maka akan mengirimkan permintaan baru. Ketika tes yang dilakukan berakhir, WebBench akan menghitung dua nilai keseluruhan server (*request per second* dan *throughput* (*bytes per second*).

3.5.3 Script Benchmark

Selain menggunakan WebBench untuk menganalisa kinerja cluster yang dirancang pada penelitian ini sebagai web server, digunakan juga script benchmark yang ditulis dalam bahasa pemrograman PHP, yang menggunakan dua client secara paralel, script ini merupakan hasil modifikasi dari script ezhttpbench. Diagram alir dari script benchmark dapat dilihat pada gambar 7.



Gambar 7: Diagram alir script benchmark

Proses kerja dari script benchmark ini adalah sebagai berikut:

- memilih nama server yang akan dites pada pilihan, jumlah client dan juga disertai jumlah loop yang ingin dilakukan dalam pembacaan halaman web dari server yang ingin dites. Kemudian tekan tombol OK.
- Setelah tombol OK ditekan akan keluar konfirmasi untuk proses selanjutnya. Kemudian tekan tombol Benchmark.
- Ketika menekan tombol Benchmark, jika lebih dari 1 client akan muncul sejumlah window baru sesuai dengan jumlah *client* dikurangi satu (n-1) yang digunakan untuk melakukan pembacaan halaman web, pembacaan halaman web juga dilakukan pada window utama. Pembacaan halaman web/file dibatasi sebesar 100000 bytes, hal ini ditujukan untuk mendapatkan *throughput* maksimum dari server yang diuji pada jaringan yang digunakan.
- Kemudian akan ditampilkan pada masing-masing window hasil pengujian halaman web, yang antara lain berisi lama pengujian, *throughput* dari web server, dan jumlah halaman web maksimal yang mampu dilayani.

4. PENGUJIAN DAN ANALISA KINERJA SISTEM CLUSTER SEBAGAI WEB SERVER

Pada dasarnya sistem komputer cluster ini dirancang untuk mempercepat proses pelayanan suatu Web Server terhadap permintaan halaman web oleh beberapa client. Pengujian dan analisa yang dilakukan terhadap sistem komputer cluster ini dimaksudkan untuk menunjukkan bahwa beberapa komputer lama juga dapat berguna untuk melakukan pelayanan dengan cepat.

4.1 Perangkat Pengujian Sistem Cluster

Peralatan yang dipergunakan dalam penyusunan penelitian ini untuk menguji dan menganalisa kinerja dari sistem cluster berupa tujuh buah komputer, sebuah sebagai director, tiga buah komputer sebagai real server, sebuah komputer client, dan dua buah computer untuk membandingkan hasil dari server cluster. Spesifikasi alat uji yang digunakan sebagai berikut:

1. Director (PII/350, 64MB, 100Mbps dan 10Mbps)
2. Real Server 1 (486DX4/100, 12MB, 0KB cache, 10Mbps)
3. Real Server 2 (486DX4/100, 12MB, 256KB cache, 10Mbps)
4. Real Server 3 (486DX2/66, 12MB, 256 cache, 10Mbps)
5. Client (PII/300, 64MB, 100Mbps)
6. Pembanding 1 (P233, 48MB, 10Mbps)
7. Pembanding 2 (PII/300, 64MB, 10Mbps)

4.2 Pengujian dan Analisa Kinerja Jaringan Lokal Komputer Cluster

4.2.1 Pengujian dan Analisa Kinerja Jaringan tanpa Layanan Virtual Server

Hubungan *client – director* menggunakan jaringan 100Mbps, sedangkan hubungan *director – real server*, pembanding menggunakan jaringan 10Mbps. Hasil pengujian dengan menggunakan NetPIPE didapatkan hasil sebagai berikut:

Hubungan	*Throughput Max*	*Latency* pada Saturasi
Client – director	80Mbps	0,0089 detik
Client – real server 1	6Mbps	0,126 detik
Client – real server 2	7,8Mbps	0,096 detik
Client – real server 3	7,8Mbps	0,097 detik
Client – pembanding 1	8,8Mbps	0,087 detik
Client – pembanding 2	8,8Mbps	0,087 detik

Pada pengujian yang dilakukan semakin besar ukuran file yang ditransfer maka akan semakin besar pula *throughput*-nya sampai mencapai kondisi saturasi sedangkan *latency* akan semakin cepat jika ukuran paket semakin kecil.

Dari hasil *throughput* dan *latency* dapat dilihat bahwa *cache* memori pada CPU, prosesor, kartu jaringan sangat mempengaruhi kinerja komputer yang digunakan. Mengacu pada hasil pengujian didapatkan bahwa kinerja jaringan *real server 1* < kinerja jaringan *real server* 3 < kinerja jaringan *real server* 2 < kinerja jaringan pembanding 1 < kinerja jaringan pembanding 2 < kinerja jaringan director.

4.2.2 Pengujian dan Analisa Kinerja Jaringan dengan Layanan Virtual Server

Pengujian ini dilakukan dengan mengaktifkan layanan virtual server pada *director* dan akan digunakan untuk menguji sistem cluster yang dirancang. Pengujian dilakukan untuk menguji sistem cluster dengan dua *real server* dan dengan tiga *real server*. Hasil pengujian dengan menggunakan NetPIPE didapatkan hasil sebagai berikut:

Hubungan	*Throughput Max*	*Latency* pada Saturasi
Client 1 – lvs (1 & 2)	5Mbps	0,157 detik
Client 2 – lvs (1 & 2)	5,5Mbps	0,137 detik
Client 1 – lvs (1 & 3)	5Mbps	0,150 detik
Client 2 – lvs (1 & 3)	5,5Mbps	0,130 detik
Client 1 – lvs (2 & 3)	6Mbps	0,123 detik
Client 2 – lvs (2 & 3)	6Mbps	0,124 detik
Client 1 – lvs (1&2&3)	4Mbps	0,185 detik
Client 2 – lvs (1&2&3)	4,5Mbps	0,166 detik
Client 3 – lvs (1&2&3)	4,5Mbps	0,167 detik

Dapat dilihat bahwa jika terdapat lebih dari satu permintaan pada server, maka akan terjadi penurunan *throughput* dan *latency*. Hal ini disebabkan penggunaan jaringan yang hanya 10 Mbps pada hubungan *director – real server*,

pembanding sehingga menyebabkan kolisi pada saat transfer data secara bersamaan.

4.3 Pengujian dan Analisa Kinerja Web Server

4.3.1 WebBench

Hasil dari WebBench untuk pengujian ini dapat dilihat pada tabel sebagai berikut:

Server	*Throughput* (KBytes/sec)	Req/Sec
Real server 1	139	22,872
Real server 2	207	35,108
Real server 3	173	29,952
Pembanding 1	277	47,174
Pembanding 2	372	62,959
Cluster 2 server (1 + 2)	266	45,822
Cluster 2 server (1 + 3)	266	44,97
Cluster 2 server (2 + 3)	314	52,98
Cluster 3 server (1 + 2 +3)	354	59,534

Bandwidth yang tersedia tidak dapat digunakan semua, karenakan *throughput* ini dipengaruhi besarnya data yang ditransfer dari server pada client sedangkan pada WebBench ukuran file yang diminta oleh client berkisar dari 223 bytes sampai 529 Kbytes. File kecil akan memberikan *throughput* yang kecil sedangkan untuk file besar akan memberikan *throughput* yang besar. Rata-rata *throughput* ini lebih rendah dibandingkan *bandwidth* yang tersedia. Selain itu diakibatkan keterbatasan kemampuan CPU dalam mengolah data untuk memenuhi permintaan dari client dan meminta layanan NFS dan NBD pada director dan terjadinya kolisi pada jaringan lokal.

4.3.2 Script Benchmark

Pengujian dengan jumlah *client* 2 dan jumlah *loop* pembacaan halaman web adalah 100 dengan menggunakan browser Internet Explorer 5.0 didapat hasil pengujian rata-rata dari script benchmark dapat dilihat sebagai berikut:

Server	Latency (det)	Throughput (Mbps)
Real server 1	58,84	2,76
Real server 2	24,89	6,2
Real server 3	28,72	5,35
Pembanding 1	21,36	7,16
Pembanding 2	20,76	7,36
Cluster 2 server (1 + 2)	20,91	7,3
Cluster 2 server (1 + 3)	21,7	7,05
Cluster 2 server (2 + 3)	20,23	7,55
Cluster 3 server (1 + 2 +3)	20,23	7,55

Tidak tercapainya *throughput* maksimum seperti yang ditunjukkan pada analisa kinerja jaringan disebabkan keterbatasan pada jaringan yang hanya mempunyai *bandwidth* 10 Mbps, selain disebabkan keterbatasan kemampuan CPU, cache memori, memori dan kartu jaringan, kolisi yang terjadi pada jaringan lokal, dan juga diakibatkan diterapkannya NFS dan NBD yang menyita *bandwidth* jaringan.

5. PENUTUP

Dari hasil analisa dalam pengujian sistem cluster, dapat ditarik kesimpulan beberapa hal:

1. Pengujian kinerja jaringan diperlukan untuk mengetahui sejauh mana sistem komputer mampu menangani pengolahan data pada jaringan, dengan demikian dapat diperkirakan kemampuan sistem komputer dalam menjalankan aplikasi jaringan. Dari hasil pengujian dengan menggunakan NetPIPE didapatkan hasil berikut: kinerja *real server* 1 < kinerja *real server* 3 < kinerja *real server* 2 < kinerja pembanding 1 < kinerja pembanding 2, sehingga komputer dengan kinerja yang lebih tinggi mampu menjalankan aplikasi jaringan dengan lebih baik.
2. *Throughput* web server cluster dapat lebih besar dibandingkan *throughput* web server tunggal, *throughput* web server cluster dua dan tiga komputer 486 mendekati penjumlahan *throughput* masing-masing komputer yang digunakan.
3. Web server cluster dapat melayani lebih banyak permintaan dibandingkan web server tunggal, permintaan yang dapat dilayani oleh web server cluster dua dan tiga komputer 486 mendekati penjumlahan permintaan yang dapat dilayani oleh masing-masing komputer yang digunakan..
4. Web server cluster dapat meningkatkan kecepatan dalam penyedian layanan halaman web, sehingga waktu yang dibutuhkan untuk melayani permintaan halaman web dari *client* dapat dilakukan dengan lebih cepat.
5. Web server cluster dengan menggunakan dua buah komputer 486 mampu menangani layanan web sebanding dengan Pentium 233 MMX, sedangkan dengan tiga buah komputer 486 mampu menangani layanan web mendekati Pentium II 300 MMX, untuk penggunaan peralatan/kartu jaringan yang sama.

6. DAFTAR PUSTAKA

[1]. Barr, T., Langfeldt, N., and Vidal, S., *Linux NFS-HOWTO*, 2000.

[2]. Desrochers, George R., *Principles of Parallel and Multiprocessing*, International Edition, New York, McGraw-Hill, 1988.

[3]. Dietz, H., *Linux Parallel Processing HOWTO*, v980105, Januari 1998.

[4]. Drake, J., *Linux Networking HOWTO – DocBook Rev .02*, Commandprompt, Inc., 2000.

[5]. Fathi, Eli T., and Krieger, M., *Multiple Microprocessor Systems: What, Why, and When*, IEEE Computer Magazine, Maret 1983.
[6]. Ghelmer(ftp://ftp.scl.ameslab.gov/pub/netpipe/README), *README*, v 1.13, Iowa State University Research Foundation, Inc., Desember 1999.
[7]. Haddad, I., and Pourzandi, M., *Linux on Carrier Grade Web Servers*, Linux Journal, April 2001.
[8]. Hayes, John P., Computer Architecture and Organization, 2nd ed., New York, McGraw-Hill, 1988.
[9]. Kostyrka, A., *NFS-Root Mini-Howto*, Agustus 1997.
[10]. Langfeldt, N., *DNS HOWTO*, Desember 1999.
[11]. LVS project, and Mack, J., *LVS Performance: Initial Test with a single Realserver LVS*, v1.0, Juni 2000.
[12]. Machek, P.(http://atrey.karlin.mff.cuni.cz/~pavel/nbd/nbd.html), Network Block Device (TCP version)
[13]. Mack, J., *LVS-HOWTO*, v1.0, Maret 2001
[14]. Mack, J., *LVS-mini-HOWTO*, v1.1, Desember 2000.
[15]. Maor, O., *NFS-Root-Client Mini-HOWTO*, v4.1, Februari 1999.
[16]. O'Rourke, P., *Performance Evaluation of Linux Virtual Server*, Mission Critical Linux, Inc., April 2001.
[17]. Purbo, Ono W., Maryanto, D., Widodo, W., dan Hubbany, S., *Buku Pintar Internet Membangun Server Internet dengan FreeBSD*, Jakarta, PT. Elex Media Komputindo, 2000.
[18]. Radajewski, J., and Eadline, D., *Beowulf Installation and Administration HOWTO*, v 0.1.2, Juni 1999.
[19]. Sherrill, W., *LVS Cluster Configuration HOWTO*.
[20]. Tanenbaum, Andrew S., *Structured Computer Organization*, New Jersey, Prentice-Hall Inc. , 1999.
[21]. Zhang, W., Jin, S., and Wu, Q., *Creating Linux Virtual Servers*, China, National Laboratory for Parallel & Distributed Processing, 1998.