PERANCANGAN DAN PEMBUATAN EMULASI ROUTER BERBASIS SISTEM OPERASI LINUX

TUGAS AKHIR

Oleh :

ARY NUR SUBAGIA

NRP. 2693100013

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
1999

PERANCANGAN DAN PEMBUATAN EMULASI ROUTER BERBASIS SISTEM OPERASI LINUX

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer
Pada
Jurusan Teknik Informatika
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya

Mengetahui / Menyetujui,

Dosen Pembimbing I	Dosen Pembimbing II
	23/8 99
Ir. ESTER HANAYA, M.Sc.	RULLY SOELAIMAN, S.Kom
NIP. 130 816 212	NIP. 132 085 802

S U R A B A Y A
Agustus, 1999

...dipersembahkan untuk

ibu, bapak, adek ku tercinta

dan

orang-orang yang untuk mereka tidak pernah dipersembahkan apa-apa...

ABSTRAK

Router merupakan salah satu peralatan jaringan komputer yang memegang peranan penting. Router dapat diibaratkan sebagai penujuk jalan bagi paket-paket data yang dikirim antar jaringan komputer. Router dapat juga diibaratkan sebagai polisi yang mengatur lalu-lintas data antar jaringan komputer.

Mengingat pentingnya fungsi router dan mahalnya harga router maka dalam tugas akhir ini akan dibuat sebuah emulasi router dengan mempergunakan *personal computer* (PC) dan sistem operasi yang bersifat *freeware* yaitu Linux.

Emulasi router yang dibuat didasarkan pada sistem operasi Linux dengan mempergunakan kernel versi 2.0.36. Untuk mendapatkan emulasi router yang diinginkan kernel Linux diubah dengan menggunakan metode *patching* dan *shell scripting*. Disamping melakukan perubahan kernel, dilakukan juga perubahan sistem file dan pemaketan ulang program-program yang ada pada Linux. Karena router beroperasi dengan memperlakukan media penyimpanan permanen secara khusus, berbeda dengan sistem operasi pada umumnya, maka emulasi router juga memperlakukan media penyimpanan permanen secara khusus. Hal ini didapat dengan jalan mengubah cara pengoperasian sistem operasi Linux dan mengubah struktur sektor dan track dari media penyimpanan permanen (disket).

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah SWT atas segala rahmat dan karunia-Nya sehingga penulis bisa menyelesaikan Tugas Akhir yang berjudul:

PERANCANGAN DAN PEMBUATAN

EMULASI ROUTER BERBASIS

SISTEM OPERASI

LINUX

Tugas Akhir ini merupakan syarat akademis bagi para mahasiswa dalam rangka menyelesaikan studi kesarjanaan pada Jurusan Teknik Informatika, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember, Surabaya.

Semoga Tugas Akhir ini dapat memberikan manfaat yang sebesar-besarnya bagi kita semua. Sebagai manusia biasa yang memiliki keterbatasan maka penulis menyadari bahwa dalam Tugas Akhir ini masih banyak kekurangan dan jauh dari sempurna. Oleh karena itu kritik dan saran yang bersifat membangun sangat penulis harapkan demi kesempurnaan.

Surabaya, Agustus 1999

Penulis

UCAPAN TERIMA KASIH

Dengan terselesaikannya Tugas Akhir ini, penulis menyampaikan penghargaan dan ucapan terima kasih yang setulus-tulusnya kepada mereka yang telah membantu penulis dalam mengatasi kendala-kendala baik teknis maupun psikologis, terutama kepada:

1. Bapak Daljuri dan Ibu Sri Mulat yang telah memberika perhatian, dorongan serta kesabarannya sehingga penulis mampu menyelesaikan Tugas Akhir ini.
2. Ir. Esther Hanaya, M.Sc., selaku dosen pembimbing yang telah memberikan dorongan selama pembuatan Tugas Akhir.
3. Rully Soelaiman, S.Kom., selaku dosen pembimbing, yang telah memberikan dorongan semangat dan bantuan yang sedemikan banyaknya selama pembuatan Tugas Akhir.
4. Febriliyan Samopa, S.Kom, terimakasih atas masukan dan bantuannya selama pengujian sistem.
5. Rekan-rekan senasib TA-wan Isbat, Ilham, Rosyid, Andri terima kasih atas dorongan semangatnya, tetaplah bersemangat.
6. Rahmad Tri Wibowo, terimakasih banyak atas hiburan dan dukungannya.
7. Roy dan Pram, terimakasih atas dukungan logistiknya.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1. Latar Belakang

Router berfungsi untuk melewatkan data melalui lintasan jalur data tertentu antar segmen jaringan yang berbeda, biasanya berwujud perangkat keras yang dirancang khusus, sebagai contoh produk router dari Cisco, BayNetwork, 3Com dan lain-lain. Disamping berfungsi melewatkan data seperti tersebut di atas, router seringkali berfungsi multiguna. Fungsi multiguna ini dapat dilihat dari fungsinya sebagai *firewall*, *network address translator* (NAT) dan kadang kala berfungsi sebagai *remote access service* (bertindak sebagai perangkat jaringan yang menerima *dial-in* modem).

Router semakin dibutuhkan pada saat ini sejalan dengan meningkatnya kebutuhan koneksi terhadap Internet. Hal ini dikarenakan Internet merupakan kumpulan dari berbagai jaringan komputer, dimana satu dengan lainnya dipisahkan oleh sebuah router. Pembagian jaringan menjadi bagian yang lebih kecil dalam komunitas Internet mempunyai tujuan agar supaya pemeliharaan jaringan dapat didelegasikan pada tiap-tiap kelompok jaringan komputer dan pertumbuhan komunitas Internet dapat selalu terjaga[1].

Router tidak hanya dibutuhkan oleh jaringan komputer besar yang terhubung langsung dengan Internet. Suatu lingkungan jaringan komputer atau *local area network* (LAN) yang terdiri dari beberapa sub jaringan yang lebih kecil juga memerlukan router. Router dibutuhkan pada skala ini disamping untuk fungsi tersebut di atas juga digunakan sebagai pembatas ruang *broadcast* jaringan. Sehingga dengan adanya pembatasan ruang *broadcast*, *traffic* (lalu-lintas dan kepadatan paket data) jaringan yang ada dapat lebih diperkecil. Router dalam komunitas ini juga seringkali digunakan untuk manajemen *traffic*. Manajemen *traffic* yang dilakukan berupa pembatasan akses terhadap suatu layanan (*service*) tertentu.

Dengan melihat hal tersebut di atas, tidak dapat disangkal bahwa kebutuhan akan router sangat tinggi. Namun disisi lain pemenuhan kebutuhan router ini tidak dapat dipenuhi dengan mudah, mengingat harga sebuah perangkat keras router sangatlah mahal. Sehingga akhir-akhir ini berkembang suatu kegiatan yang memanfaatkan sistem operasi pada PC untuk kepentingan router. Dengan mempergunakan sistem operasi yang berjalan pada PC, harga dapat ditekan. Bahkan harga ini dapat semakin ditekan dengan mempergunakan sistem operasi yang bersifat gratis (*free*) seperti FreeBSD, OpenBSD dan Linux.

Dengan mempergunakan sistem seperti tersebut di atas harga dapat ditekan, namun muncul permasalahan baru, yaitu permasalahan operasional

---

[1] Bassam Halabi, "Internet Routing Architectures", Cisco Press, 56, 1997

router dengan mempergunakan sistem tersebut. Sistem operasi yang dipergunakan untuk router adalah sistem operasi yang bersifat *multiuser*. Salah satu kelemahan yang ada pada sistem operasi *multiuser* adalah operasional sistem, yaitu adanya suatu prosedur khusus dalam hal proses *boot* dan *shutdown* sistem operasi. Pada router proses *boot* dan *shutdown* tidak terlalu prosedural. Router dapat dihidupkan dan dimatikan seketika tanpa mempengaruhi sistem operasi yang tersimpan didalamnya. Hal ini tidak dapat berlaku pada sistem operasi *multiuser* pada umumnya, terutama pada proses *shutdown* sistem. Apabila proses *shutdown* sistem tidak dilakukan dengan lengkap, maka sistem operasi tersebut akan mengalami suatu kehilangan salah satu bagian dari sistemnya. Walaupun masalah ini dapat ditanggulangi pada proses *boot* berikutnya, tetapi penanggulangan ini hanya bersifat sementara dan periodik. Apabila proses *shutdown* sistem tidak dilakukan dengan benar berulang kali, sistem operasi akan kehilangan beberapa bagian sistemnya, terutama bagian sistem filenya.

Dengan melihat hal tersebut perlu dibuat sebuah sistem yang dapat digunakan sebagai router dengan memanfaatkan sistem operasi yang dapat memperkecil kerumitan operasionalnya.

1.2. Permasalahan

Untuk mewujudkan sebuah sistem yang berfungsi sebagai router dan berjalan diatas *personal computer* dengan memanfaatkan sebuah sistem operasi, terdapat beberapa permasalahan diantaranya :

- Emulasi Router yang dibuat hendaknya mampu dioperasikan selayaknya perangkat keras Router. Terutama dalam hal proses menghidupkan dan mematikannya. Proses yang biasa terjadi di sistem operasi terutama yang berbasis UNIX, yaitu *mounting* dan *dismounting* haruslah dapat diatasi.
- Program-program yang ada di sistem operasi baik itu dasar sistemnya atau tambahanya tidak semuanya diperlukan untuk keperluan Emulasi Router ini. Untuk itu perlu dilakukan proses pemaketan ulang program-program yang ada pada sistem operasi sehingga keamanan dan keringkasan Emulasi Router dapat dicapai.
- Proses pemaketan ulang tidak dapat terlepas dari struktur sistem file yang digunakan. Untuk itulah perlu dibuat struktur sistem file yang cocok bagi router yang akan dibuat.
- Protokol routing merupakan kemampuan utama yang disediakan oleh router. Untuk itulah router yang akan dibangun haruslah menyediakan protokol routing yang biasa digunakan dalam lingkungan jaringan TCP/IP, seperti RIP, OSPF dan BGP.
- Untuk mengetahui beban yang dipikul oleh router, perlu disediakan sebuah *network monitoring* atau *network management* untuk mengetahui beban kerja router.

1.3. Tujuan

Tujuan pokok dari Tugas Akhir ini adalah membangun sebuah Emulasi Router pada PC dengan memanfaatkan sistem operasi yang bersifat *freeware* yaitu Linux sehingga dapat menggantikan fungsi perangkat keras Router dengan biaya yang jauh lebih murah.

1.4. Batasan Masalah

Dalam Tugas Akhir ini dilakukan pembatasan masalah sebagai berikut:

- Protokol jaringan komputer yang digunakan adalah TCP/IP.
- Emulasi Router mempunyai kemampuan dasar *firewall*.
- Emulasi Router mampu menjalankan protokol routing RIP (*Router Information Protocol*), BGP (*Border Gateway Protocol*) dan OSPF (*Open Shortest Path First*).
- Emulasi Router mampu bertidak sebagai NAT (*Network Address Translator*).
- Emulasi Router mampu menanggani *dial-in asynchronous port* sebanyak 2 buah.
- Keluaran untuk LAN (*Local Area Network*) berupa Ethernet 10 Mbps atau 100 Mbps.

1.5. Metodologi Penelitian

Metodologi yang digunakan dalam Tugas Akhir ini adalah sebagai berikut:

- Studi Literatur

Pada tahap ini dilakuakan pengumpulan data dan studi literatur, terutama yang berkaitan degan jaringan TCP/IP, arsitektur routing dan sistem operasi Linux.

- Perencanaan Sistem

Pada tahap ini dilakukan perencanaan sistem berupa struktur data, algoritma dan diagram alur.

- Pembuatan Sistem

Pada tahap ini dilakukan implementasi dari tahapan sebelumnya yang telah direncanakan.

- Evaluasi dan Revisi Sistem

Pada tahap ini dilakukan evaluasi dan perbaikan dari sistem yang telah diimplementasikan. Proses pengujian dilakukan dengan cara melakukan ujicoba sistem pada LAN yang terhubung ke Internet dan melakukan simulasi jaringan yang mempunyai beberapa segmen jaringan.

- Penulisan Naskah

Semua proses yang telah dilakukan didokumentasikan. Dokumentasi atau penulisan naskah ini meliputi penjelasan dasar terori, proses pembuatan sistem, evaluasi dan hasil yang didapat.

BAB II

TEORI PENUNJANG

2.1. Protokol Routing

Internet merupakan kumpulan dari *autonomous system*[2] yang mendefinisikan wewenang administratif dan aturan routing dari beberapa organisasi yang berbeda. *Autonomous system* menjalankan *Interior Gateway Protocols* (IGPs), seperti RIP, IGRP, EIGRP, OSPF, dan ISIS, dalam batasannya masing-masing dan berhubungan dengan *Exterior Gateway Protocol* (EGP) yang disebut sebagai *Border Gateway Protocol* (BGP).

Router adalah peralatan yang menghubungan *traffic* data secara langsung antar *host*. Router membuat tabel *routing* yang berisi sekumpulan informasi semua jalur terbaik untuk semua tujuan yang diketahui.

2.1.1. Pengenalan Router dan Routing

Router membuat suatu mekanisme hop demi hop untuk menjamin informasi hop selanjutnya (*next hop*) dapat dipakai oleh paket data untuk menemukan tempat tujuan melalui jaringan. Router yang tidak mempunyai hubungan fisik secara langsung ke tempat tujuan melakukan pemeriksaan tabel *routing* dan meneruskan paket data ke router di hop berikutnya yang lebih dekat

---

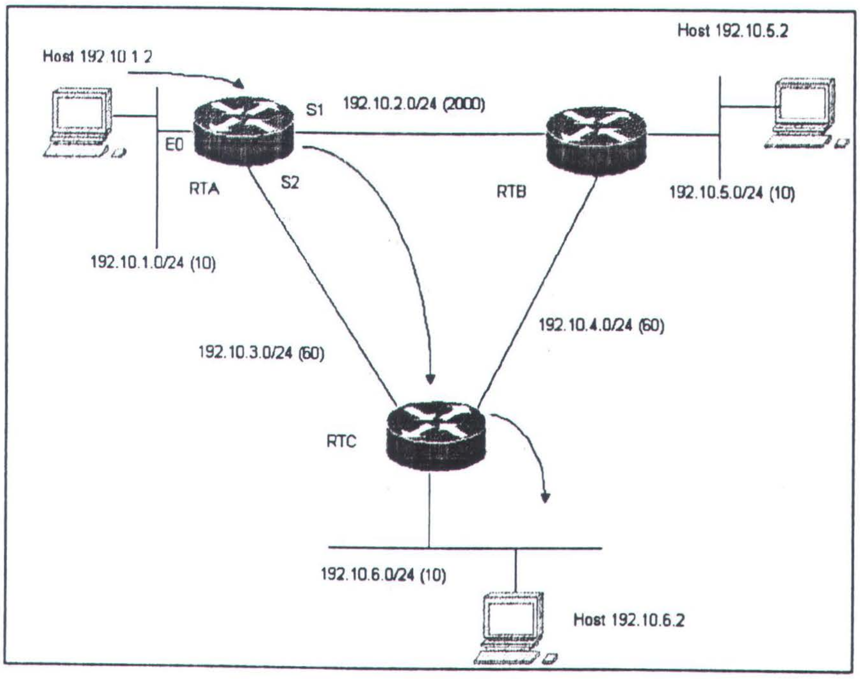
[2] Bassam Halabi, "Internet Routing Architectures", Cisco Press, 87, 1997

ke tempat tujuan. Proses ini diulang-ulang hingga menemukan tujuan akhir di jaringan.

EGP diperkenalkan dikarenakan IGP tidak mendukung jaringan dengan skala besar. IGP tidak didisain untuk *global internetworking* karena IGP tidak memisah-misahkan jaringan besar (*enterprises*) dalam administrasi network masing-masing. Administrasi yang dimaksud adalah administrasi network secara teknik dan politis.

Pada gambar 2.1 dibawah ini digambarkan 3 router RTA, RTB, dan RTC terhubung dengan 3 *local area network* (LAN), 192.10.1.0, 192.10.5.0, dan 192.10.6.0, melalui hubungan serial. Setiap hubungan serial ditujukkan dengan nomer network masing-masing, 192.10.2.0, 192.10.3.0, dan 192.10.4.0. Setiap network mempunyai harga *metric* yang menunjukan *cost* dari setiap hubungan antar network yang ada. Sebagai contoh, hubungan RTA dan RTB mempunyai *cost* 2000, *cost* yang lebih tinggi dipunyai oleh hubungan RTA dan RTC. Pada prakteknya hubungan antara RTA dan RTB adalah 56 Kbps yang mempunyai waktu tunggu (*delay timel*) yang lebih besar daripada koneksi T1 antara RTA dan RTC serta koneksi T1 antara RTC dan RTB.

Router RTA, RTB dan RTC saling bertukar informasi jaringan melalui *interior gateway protocol* dan membuat suatu tabel *routing* IP. Tabel 2.1 dan 2.2 merupakan contoh dari tabel *routing* IP pada RTA pada kondisi yang berbeda, yaitu router bertukar informasi dengan mempergunakan RIP pada satu kondisi dan OSPF pada kondisi yang lain.



Gambar 2.1. Perilaku dasar routing.

Tabel 2.1. Tabel *Routing* IP pada RTA (RIP)

Destination	Next Hop	Hop Count
192.10.1.0	Connected (E0)	-
192.10.2.0	Connected (S1)	-
192.10.3.0	Connected (S2)	-
192.10.4.0	192.10.2.2 (S1)	1
	192.10.3.2 (S2)	1
192.10.5.0	192.10.2.2 (S1)	1
192.10.6.0	192.10.3.2 (S2)	1

Tabel 2.2. Tabel *Routing* IP pada RTA (OSPF)

Destination	Next Hop	Hop Count
192.10.1.0	Connected (E0)	-
192.10.2.0	Connected (S1)	-
192.10.3.0	Connected (S2)	-
192.10.4.0	192.10.3.2 (S2)	120
192.10.5.0	192.10.3.2 (S2)	130
192.10.6.0	192.10.3.2 (S2)	70

Tujuan dibuatnya jaringan adalah untuk menyampaikan data dari satu komputer ke komputer lain. Jika jaringan tidak dapat menyampaikan data ke tujuan yang seharusnya, maka jaringan tersebut tidak berguna. Pada bab sebelum ini kita sudah mengetahui bahwa protokol TCP/IP memiliki kemampuan internetworking dan routing. Dalam jaringan TCP/IP setiap host memiliki *IP address* dan untuk berhubungan dengan host tersebut kita harus memasukkan *IP address* host pada bagian tujuan dari datagram IP yang dikirim. Pertanyaan berikutnya yang harus dapat dijawab adalah bagaimana host mengetahui cara mencapai host yang dituju datagram tersebut ?

2.1.2. Konsep Dasar Routing

Konsep routing adalah hal yang utama pada lapisan internet di jaringan TCP/IP. Hal ini karena pada lapisan Internet terjadi pengalamatan (*addressing*). Kita coba perhatikan kembali aliran data pada arsitektur TCP/IP. Data dari lapisan aplikasi disampaikan ke transport dengan diberi header TCP atau UDP tergantung jenis aplikasinya. Setelah itu segmen TCP atau UDP disampaikan ke lapisan IP dan diberi header, termasuk alamat asal dan tujuan datagram. Pada saat ini host harus melakukan routing dengan melihat tabel routing. Setelah melihat tabel routing, datagram diteruskan ke lapisan network interface dan diberi header dengan alamat tujuan yang sesuai.

Untuk lebih jelasnya, kita perhatikan jaringan TCP/IP yang menggunakan teknologi Ethernet. Setiap frame Ethemet (Ethernet II) mengandung alamat tujuan dan alamat asal, tipe protokol, dan data. Alamat tujuan dan alamat asal adalah sebuah bilangan 48 bit. Setiap kartu Ethernet memiliki sebuah alamat Ethernet yang unik. Agar datagram dapat diterima oleh sebuah host tujuan. Datagram tersebut harus dimasukkan dalam frame dengan alamat Ethernet tujuan yang sama dengan alamat kartu Ethernet host tujuan. Proses ini juga bagian dari routing, yaitu pada saat mengirimkan datagram IP bagaimana menentukan alamat Ethemet host tujuan datagram tersebut ?

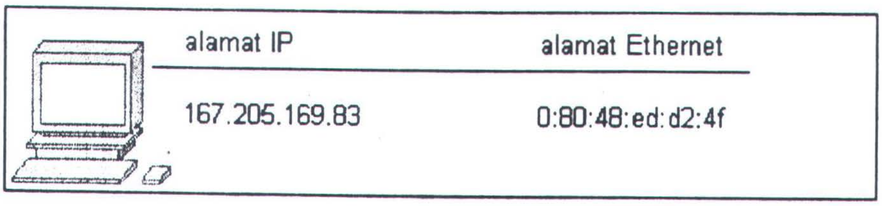
2.1.2.1. ARP

Pertanyaan ini dijawab dengan adanya protokol ARP (*Address Resolution Protokol*). ARP berfungsi untuk menerjemakan *IP address* ke alamat Ethernet. Proses ini dilakukan hanya untuk datagram yang dikirim host karena pada saat inilah host menambahkan *header* Ethemet pada datagram. Penerjemahan dari *IP address* ke alamat Ethernet dilakukan dengan melihat sebuah tabel yang disebut sebagai *cache* ARP, lihat tabel 2.2. Entri cache ARP berisi *IP address* host beserta alamat Ethemet untuk host tersebut. Tabel ini diperlukan karena tidak ada hubungan sama sekali antara *IP address* dengan alamat Ethemet. *IP address* suatu host bergantung pada *IP address* jaringan tempat host tersebut berada, sementara alamat Ethernet sebuah kartu bergantung pada alamat yang diberikan oleh pembuatnya.

Tabel 2.3. Cache ARP

IP address	Alamat Ethernet
132.92.121.1	0:80:48:e3:d2:69
132.92.121.2	0:80:ad:17:96:34
132.92.121.3	0:20:4c:30:29:29

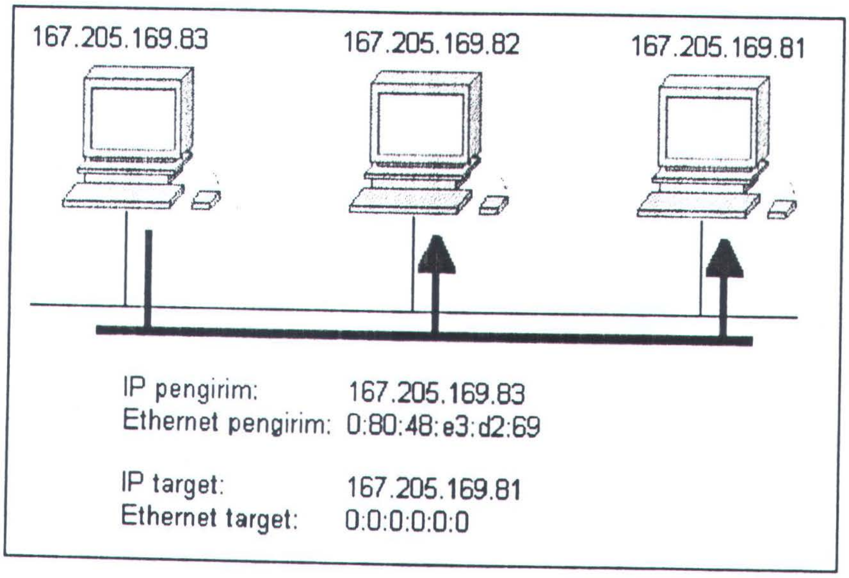
Mekanisme penerjemahan oleh ARP dapat dijelaskan sebagai berikut. Misal suatu host A dengan *IP address* 167.205.169.83 baru dinyalakan, lihat Gambar 2.2. Pada saat awal, host ini hanya mengetahui informasi mengenai interface-nya sendiri, yaitu *IP address*, alamat network, alamat broadcast, dan alamat Ethernet. Dari informasi awal ini, host A tidak mengetahui alamat Ethernet host lain yang terletak satu network dengannya (*cache* ARP hanya berisi satu entri, yaitu host A). Jika host memiliki rute default, maka entri yang pertama kali dicari oleh ARP adalah router default tersebut.



Gambar 2.2. *Cache* ARP awal.

Misalkan terdapat datagram IP dari host A yang ditujukan kepada host B yang memiliki IP 167.205.169.81 (host B ini terletak satu subnet dengan host A). Saat ini yang diketahui oleh host A adalah *IP address* host B tetapi alamat Ethernet host B tidak diketahui. Agar dapat mengirimkan datagram ke host B, host A perlu mengisi *cache* ARP dengan entri host B. Karena *cache* ARP tidak dapat digunakan untuk menerjemahkan *IP address* host B menjadi alamat Ethernet, maka host A harus melakukan dua hal :

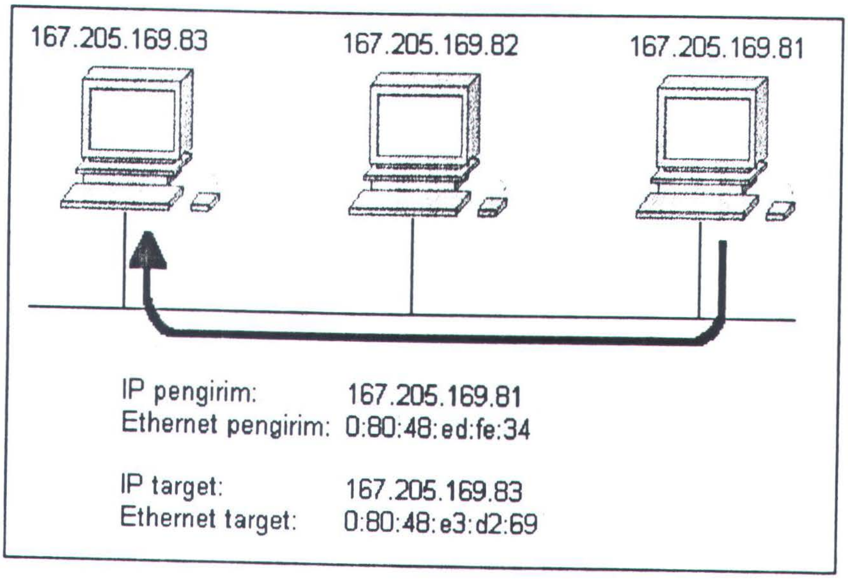
- Mengirimkan paket ARP requet pada seluruh host di network menggunakan alamat broadcast Ethernet (FF:FF:FF:FF:FF:FF) untuk meminta jawaban ARP dari host B, lihat gambar 2.3.
- Menempatkan datagram IP yang hendak dikirim dalam antrian.



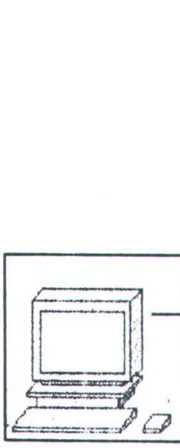
Gambar 2.3. Paket ARP *request*.

Paket ARP *request* yang dikirim host A kira-kira berbunyi "Jika *IP address*-mu adalah 167.205.169.81, mohon beritahu alamat Ethernet-mu".

Karena alamat paket request dikirim ke alamat broadcast Ethernet, setiap interface Ethernet komputer dalam jaringan dapat mendengarnya. Setiap host dalam jaringan tersebut kemudian memeriksa apakah *IP address*-nya sama dengan *IP address* yang diminta oleh host A. Host B yang mengetahui bahwa yang diminta oleh host A adalah *IP address* yang dimilikinya langsung memberikan jawaban dengan mengirimkan paket ARP *response* langsung ke alamat Ethernet pengirim (host A), seperti terrlihat pada gambar 2.4. Paket ARP *response* kira-kira berbunyi "*IP address* 167.205.169.81 adalah milik saya, sekarang saya berikan alamat Ethernet saya". Paket ARP *response* dari host B tersebut diterima oleh host A dan host A kemudian menambahkan entri *IP address* host B beserta alamat Ethernet-nya ke dalam *cache* ARP, lihat gambar 2.5.



Gambar 2.4. Paket ARP *response*.



alamat IP	alamat Ethernet
167.205.169.83	0:80:48:e3:d2:69
167.205.169.81	0:80:48:ed:fe:34

Gambar 2.5. *Cache* ARP setelah penambahan entri host B.

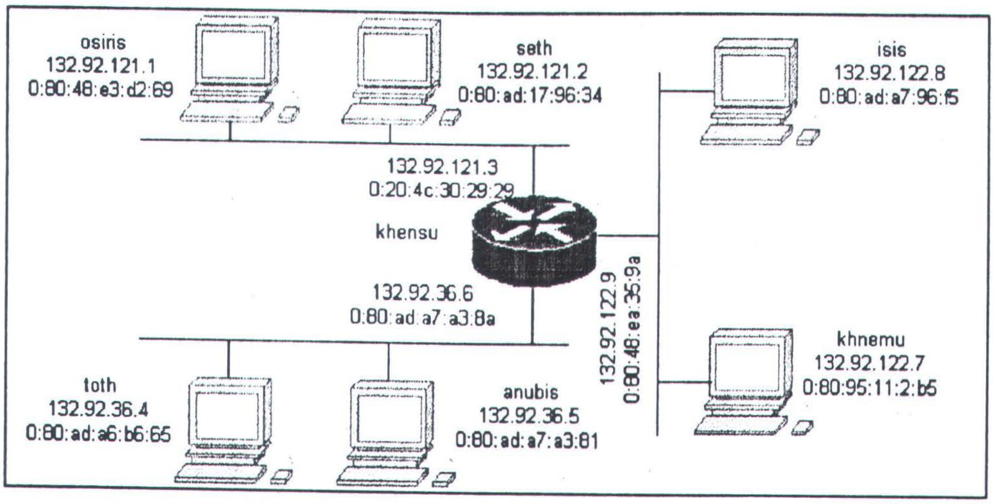
Saat ini host A telah memiliki entri untuk host B di tabel *cache* ARP, dengan demikian datagram IP yang semula dimasukkan ke dalam antrian dapat diberi *header* Ethernet dan dikirim ke host B. Jadi secara ringkas proses ARP adalah :

1. Host mengirimkan paket ARP *request* dengan alamat broadcast Ethernet.
2. Datagram IP yang akan dikirim dimasukkan ke dalam antrian.
3. Paket ARP *response* diterima host dan host mengisi tabel ARP dengan entri baru.
4. Datagram IP yang terletak dalam antrian diberi *header* Ethernet.
5. Host mengirimkan frame Ethernet ke jaringan.

Setiap data ARP yang diperoleh disimpan dalam tabel *cache* ARP dan *cache* ini diberi umur. Setelah umur entri tersebut terlampaui, entri ARP dihapus dari tabel dan untuk mengisi tabel. Jika host akan mengirim datagram ke host yang sudah dihapus dari *cache* ARP, host kembali perlu melakukan langkah-langkah di atas. Dengan cara ini dimungkinkan terjadinya perubahan isi *cache* ARP yang dapat menunjukkan dinamika jaringan. Jika sebuah host di jaringan dimatikan, maka selang beberapa saat kemudian entri ARP untuk host tersebut dihapus karena kadaluwarsa. Jika kartu Ethernet host tersebut diganti, maka beberapa saat kemudian entri ARP host berubah dengan informasi alamat Ethernet yang baru.

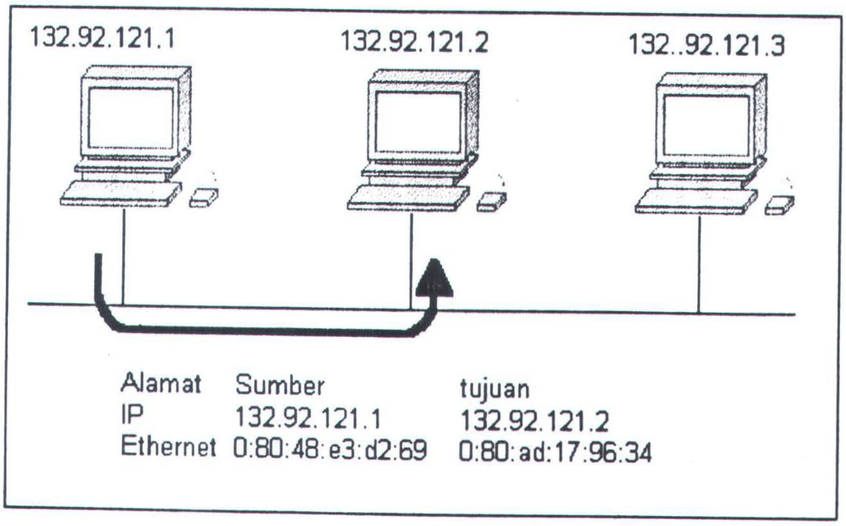
2.1.3. Routing Langsung dan Tidak Langsung

Seperti telah disebutkan diatas, proses pengiriman datagram IP selalu menggunakan tabel routing. Tabel routing berisi informasi yang diperlukan untuk menentukan ke mana datagram harus dikirim. Datagram dapat dikirim langsung ke host tujuan atau harus melalui host lain terlebih dahulu tergantung pada tabel routing.



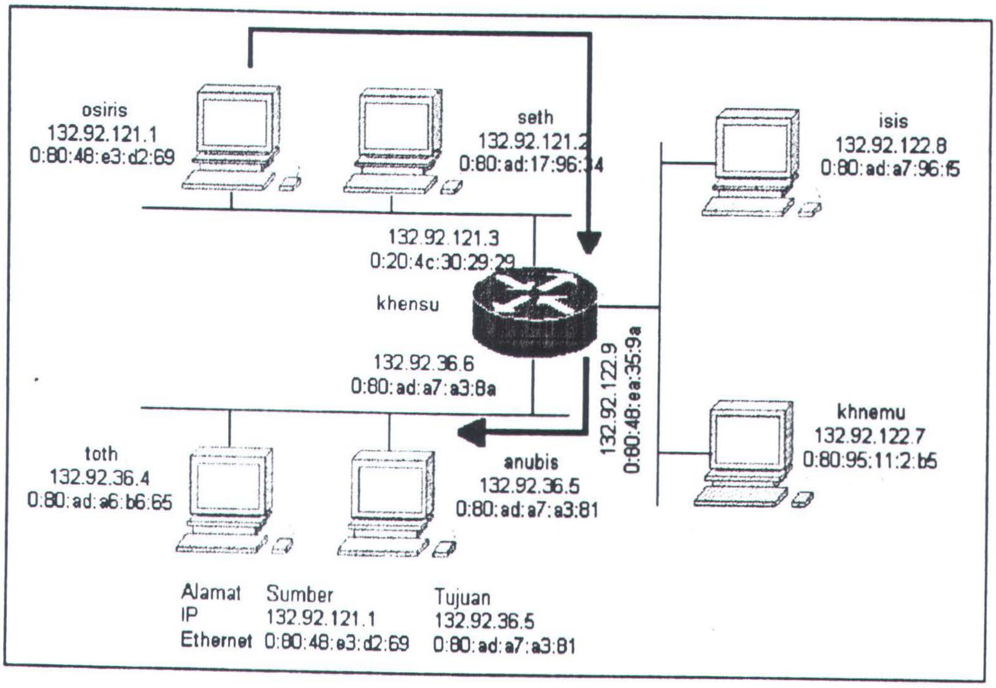
Gambar 2.6. Jaringan TCP/IP.

Gambar 2.6 memperlihatkan jaringan TCP/IP yang menggunakan teknologi Ethernet. Pada jaringan tersebut jika host *osiris* mengirimkan data ke host *seth*, alamat tujuan datagram adalah *seth* dan alamat sumber datagram adalah *osiris*. Frame yang dikirimkan oleh host *osiris* juga memiliki alamat tujuan frame *seth* dan alamat sumbernya adalah *osiris* . Pada saat *osiris* mengirimkan frame, *seth* membaca bahwa frame tersebut ditujukan kepada alamat Ethernetnya. Setelah melepas header frame, *seth* kemudian mengetahui bahwa *IP address* tujuan datagram tersebut juga adalah *IP address*-nya. Dengan demikian *seth* meneruskan datagram ke lapisan transport untuk diproses lebih lanjut. Komunikasi model seperti ini disebut sebagai routing langsung.



Gambar 2.7. Routing langsung.

Pada gambar 2.8 terlihat bahwa osiris dan anubis terletak pada jaringan Ethernet yang berbeda. Kedua jaringan tersebut dihubungkan oleh khensu. Khensu memiliki lebih dari satu interface dan dapat melewatkan datagram dari satu interface ke interface yang lain (bertindak sebagai router). Ketika mengirimkian data ke *anubis*, *osiris* memeriksa tabel routing dan mengetahui bahwa data tersebut harus melewati khensu terlebih dahulu. Dengan kondisi seperti ini datagram yang dikirim *osiris* ke *anubis* memiliki alamat tujuan *anubis* dan alamat sumber *osiris* tetapi frame Ethernet yang dikirimnya diberi alamat tujuan khensu dan alamat sumber *osiris.*



Gambar 2.8. Routing tidak langsung.

Ketika *osiris* mengirimkan frame ke jaringan, *khensu* membaca bahwa alamat Ethernet yang dituju frame tersebut adalah alamat Ethernet-nya. Ketika *khensu* melepas *header* frame, diketahui bahwa host yang dituju oleh datagram adalah host *anubis. Khensu* kemudian memeriksa tabel routing yang dimilikinya untuk meneruskan datagram tersebut. Dari hasil pemeriksaan tabel routing, *khensu* mengetahui bahwa *anubis* terletak dalam satu jaringan Ethernet dengannya. Dengan demikian datagram tersebut dapat langsung disampaikan oleh *khensu* kepada anubis. Pada pengiriman data tersebut, alamat tujuan dan sumber datagram tetap *anubis* dan *osiris* tetapi alamat tujuan dan sumber frame Ethernet menjadi *anubis* dan *khensu*. Komunikasi seperti ini disebut sebagai routing tak langsung karena untuk mencapai host tujuan, datagram harus melewati host lain yang bertindak sebagai router.

Pada dua kasus diatas terlihat proses yang terjadi pada lapisan internet ketika mengirimkan dan menerima datagram. Pada saat mengirimkan datagram, host harus memeriksa apakah alamat tujuandatagram terletak pada jaringan yang sama atau tidak. Jika alamat tujuan datagram terletak pada jaringan yang sama, datagram dapat langsung disampaikan. Jika ternyata alamat tujuan datagram tidak terletak pada jaringan yang sama, datagram harus disampaikan melalui host lain yang bertindak sebagai router. Pada saat menerima datagram host harus memeriksa apakah ia merupakan tujuan dari datagram tersebut. Jika memang demikian maka data diteruskan ke lapisan transport. Jika ia bukan tujuandari datagram tersebut, maka datagram tersebut dibuang. Jika host yang menerima datagram bertindak sebagai router , maka ia meneruskan datagram ke interface yang menuju alamat tujuan datagram.

2.1.4. Tabel Routing

Di atas telah dijelaskan bagaimana cara host menyampaikan datagram. penyampaian datagram tersebut selalu mengunakan tabel routing dalam menjalankan prosesnya. Tabel routing terdiri atas entri-entri rute dan setiap entri rute setidaknya terdiri atas IP address, tanda untuk menunjukkan routing langsung atau tak langsung, alamat router, dan nomor interface. Lihat tabel 2.2 IP address pada entri tabel routing dapat berupa *IP address* jaringan atau IP address host . Perbedaan jenis *IP address* ini dapat diketahui dengan melihat subnet mask bagi tael routing tersebut. Tanda routing langsung/tak langsung menentukan alamat router. Pada routing tak langsung, alamat router adalah IP address host lain yang dapat menyampaikan datagram ke tujuannya. Nomor interface pada entri tabel routing menunjukkan jalur keluar datagram dari host.

Tabel 2.4. Contoh tabel routing

IP address	Langsung/Tidak	Router	No. Interface
132.92.36	Langsung	<kosong>	1
132.92.122	Tidak langsung	132.92.36.6	1

Sekarang kita akan perhatikan jaringan TCP/IP pada gambar 2.6 untuk melihat tabel routing bekerja. Pada gambar terlihat tiga jaringan ethernet yang dihubungkan oleh sebuah router dengan tiga interface. Host-host pada jaringan tersebut dibuatkan tabel routing sehingga dapat menjangkau seluruh host yang lain.

Tabel 2.5. Tabel routing host *isis*

IP address	Langsung/Tidak	Router	No. Interface
132.92.122	Langsung	<kosong>	1
132.92.121	Tidak langsung	132.92.122.9	1
132.92.36	Tidak langsung	132.92.122.9	1

Misal isis hendak mengirim data ke *toth*. Pada lapisan internet, *isis* memeriksa alamat network dari *toth*. Kemudian alamat network tersebut dibandingkan dengan yang terdapat di tabel routing dan diperoleh bahwa alamat network sama dengan baris ketiga tabel routing, lihat tabel 2.3. Agar dapat sampai ke *toth* data harus dikirim melalui *khensu* terlebih dahulu. Host *isis* kemudian menerjemahkan *IP address khensu* menjadi alamat Ethernetnya. Setelah memperoleh hasil proses ARP, datagram kemudian diberi *header* frame dan frame dikirim melalui interface nomor 1.

Tabel 2.6. Tabel routing *khensu*

IP address	Langsung/Tidak	Router	No. Interface
132.92.122	Langsung	<kosong>	1
132.92.121	Lansung	<kosong>	2
132.92.36	Langsung	<kosong>	3

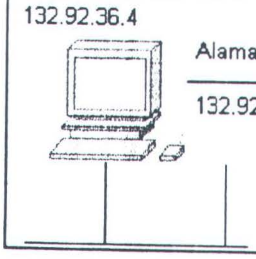
Host *khensu* menerima frame pada interface nomor 1. Frame ini kemudian dilepas dan host *khensu* memeriksa kembali alamat tujuan datagram. Karena alamat tujuan datagram tersebut bukan khensu dan khensu bertindak sebagai router, maka datagram tersebut akan diteruskan ke host tujuan. Proses pembandingan alamat tujuan datagram dengan alamat di tabel routing seperti di *osiris* terulang lagi. Pada proses ini, *khensu* menemukan bahwa host *toth* terletak satu jaringan dengannya (baris ketiga tabel routing, lihat Tabel 2.4). Dengan demikian datagram dapat langsung dibungkus dengan frame yang diberi alamat tujuan host toth dan dikirimkan melalui interface nomor 3. Frame tersebut akhirnya diterima oleh host *toth* dan karena melihat bahwa tujuan datagram adalah host *toth*, maka datagram tersebut diteruskan ke lapisan transport TCP/IP.

Proses pencarian pada tabel routing ini biasanya mengikuti langkah-langkah di bawah ini :

1. Alamat tujuan datagram di-*masking* dengan subnet mask pengirim dan dibandingkan dengan alamat network host pengirim. Jika sama, maka ini adalah routing langsung dan frame langsung dikirimkan ke interface jaringan.
2. Jika tujuan datagram tidak terletak dalam satu jaringan, periksa apakah terdapat entri routing yang berupa host dan dibandingkan dengan *IP address* tujuan datagram. Jika ada entri yang sama, kirim frame ke router menuju host tersebut.
3. Jika tidak terdapat entri host yang cocok pada tabel routing, gunakan alamat tujuan datagram yang telah di-mask pada langkah 1 untuk mencari kesamaan di tabel routing. Periksa apakah ada network/subnetwork di tabel routing yang sama dengan alamat network tujuan datagram. Jika ada entri yang sama, kirim frame ke router menuju network/subnetwork tersebut.
4. Jika tidak terdapat entri host ataupun entri network/subnetwork yang sesuai dengan tujuan datagram, host mengirimkan frame ke router default dan menyerahkan proses selanjutnya kepada router default.
5. Jika tidak terdapat rute default di tabel routing, semua host diasumsikan dalam keadaan terhubung langsung. Dengan demikian host pengirim akan mencari alamat fisik host tujuan menggunakan ARP.

2.1.5. Membentuk Tabel Routing

Ketika sebuah host baru dinyalakan, dia belum memiliki cache ARP yang lengkap. Entri pada cache ARP yang dimilikinya hanya untuk host itu sendiri. Setelah berinteraksi dengan host lain barulah host tersebut memiliki entri-entri tambahan pada cache ARP. Hal yang sama juga terjadi pada tabel routing di host. Pada saat host baru dinyalakan, host tersebut tidak memiliki informasi di tabel routing kecuali entri untuk jaringan lokalnya. Tabel routing seperti ini kadang disebut sebagai tabel routing minimal. Dalam kondisi hanya memiliki tabel routing minimal, host belum siap untuk melakukan internetwork karena hanya dapat berkomunikasi dengan host-host yang terletak pada satu jaringan lokal.



Alamat IP	langsung/ip	router	no. interface
132.92.36	langsung	<kosong>	1

Gambar 2.9. Host dengan tabel routing minimal.

Perhatikan bagaimana jika host ini mengirimkan data ke host dengan alamat network yang berbeda. Jika hal tersebut terjadi, tidak ada entri di tabel routing yang menunjukkan ke mana host harus mengirimkan frame. Akibatnya, host akan mencari alamat fisik host tujuan dan tidak akan menemukannya.

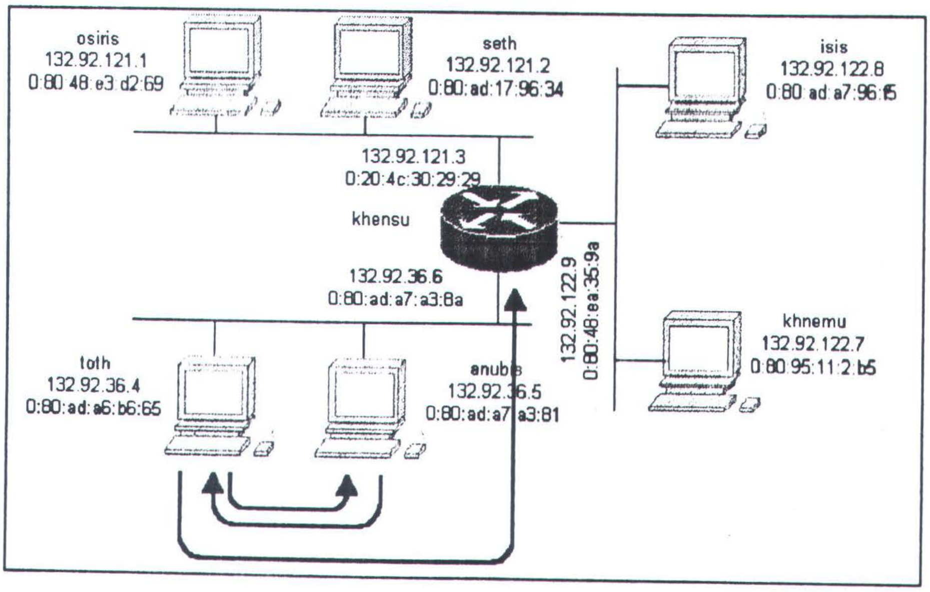
Langkah pertama untuk mempersiapkan host untuk dapat melakukan fungsi internetwork adalah dengan memberikan entri rute default pada tabel routing, lihat Tabel 2.5. Entri rute default adalah 'pembuangan' terakhir jika host tidak memiliki informasi routing lain mengenai alamat tujuan datagram. Host pengirim menyerahkan datagram kepada router default dan berharap router default tersebut memiliki informasi routing untuk mengirimkan datagram sampai ke tujuannya.

Tabel 2.5. Tabel routing dengan rute default

IP address	Langsung/Tidak	Router	No. Interface
132.92.36	Langsung	<kosong>	1
0.0.0.0	Tidak lansung	132.92.36.6	1

Router (dalam hal ini router default) dapat menyatakan bahwa dirinya bukan rute terbaik untuk mencapai host tertentu, melainkan harus melalui router yang lain dalam jaringan lokal berdasarkan tabel routing yang dimilikinya. Jika demikian, maka router tersebut mengirimkan pesan kepada host pengirim datagram menggunakan ICMP *redirect* dan memberitahukan host pengirim tersebut agar datagram menuju host tertentu dialihkan melalui router lain. Selain itu router meneruskan datagram yang diterimanya ke router lain tersebut sehingga host pengirim tidak perlu mengirim ulang datagram tersebut. Host pengirim menerima pesan ICMP redirect itu menambahkan entri host pada tabel routing dengan informasi router yang baru. Proses seperti ini disebut sebagai routing *redirect*. Dengan adanya entri routing yang baru, host pengirim tidak akan mengirim datagram untuk host tadi melalui router default.

Gambar 2.10 memperlihatkan proses routing *redirect*. Misal tabel routing pada host 132.92.36.5 adalah seperti Tabel 5.6. Dari tabel routing kita ketahui bahwa rute default host tersebut adalah 132.92.36.4. Host 132.92.36.5 hendak mengirim data ke 132.92.121.2. Karena pada tabel tidak ada rute yang menuju host 132.92.121.12, maka frame dikirim ke 132.92.36.4.



Gambar 2.10. Routing *redirect*

Tabel 2.8. Tabel routing host 132.92.36.5

IP address	Langsung/Tidak	Router	No. Interface
132.92.36	Langsung	<kosong>	1
0.0.0.0	Tidak lansung	132.92.36.4	1

Sementara itu tabel routing pada host 132.92.36.4 yang juga dapat meneruskan datagram (router) adalah seperti pada tabel 2.7. Pada tabel tersebut terdapat entri rute menuju 132.92.121 yang melalui 132.92.36.6. Ketika router 132.92.36.4 menerima frame dari 132.92.121.1 dan bukan untuk host itu sendiri. Dengan demikian router perlu menyampaikan datagram tersebut lebih lanjut. Dari hasil pemeriksaan terhadap tabel routing, router mengetahui bahwa datagram perlu dilewatkan ke router 132.92.36.6 untuk mencapai 132.92.121.1, Karena frame akan dikeluarkan melalui interface yang sama maka router 132.92.36.4 perlu memberitahu host 132.92.36.5 bahwa ia bukan rute terbaik menuju 132.92.121.1, melainkan rute 132.92.36.6.

Tabel 2.9. Tabel routing router 132.92.36.4

IP address	Langsung/Tidak	Router	No. Interface
132.92.36	Langsung	<kosong>	1
132.92.121	Tidak langsung	132.92.36.6	1
0.0.0.0	Tidak lansung	132.92.36.6	1

Router 132.92.36.4 kemudian melakukan dua hal : memberikan pesan ICMP redirect kepada 132.92.36.5 dan menyampaikan datagram melalui 132.92.36.6. Host 132.92.36.5 menerima pesan ICMP *redirect* tersebut lalu menambahkan entri host 132.92.121.1 di tabel routing dan menyatakan bahwa untuk mencapai host tersebut perlu melalui router 132.92.36.6. Akibat ICMP *redirect* ini routing tabel host 132.92.36.5 menjadi seperti pada tabel 5.8.

Tabel 2.8. Tabel routing host 132.92.36.5 setelah ICMP *redirect*

IP address	Langsung/Tidak	Router	No. Interface
132.92.36	Langsung	<kosong>	1
132.92.121.1	Tidak langsung	132.92.36.6	1
0.0.0.0	Tidak lansung	132.92.36.6	1

Proses routing *redirect* seperti di atas adalah metode paling sederhana dalam membentuk tabel routing sebuah host. Router hanya boleh memberikan pesan ICMP *redirect* untuk host, dan tidak untuk subnet. Karena untuk alasan ini menyebabkan tabel routing hanya terisi dengan entri-entri host yang tidak dihubungi melalui rute default. Metode ini paling sesuai diterapkan pada host-host yang terletak di atau yang dekat dengan titik ujung jaringan TCP/IP. Metode ini tidak sesuai jika diterapkan pada host yang terletak pada jaringan tulang punggung, apalagi jika digunakan pada router-router jaringan TCP/IP.

Metode lain yang dapat digunakan untuk membentuk tabel routing adalah dengan metode routing statik. Pada metode ini entri-entri rute di host dan router diisikan secara manual. Metode ini lebih bagus daripada hanya memanfaatkan ICMP redirect karena tabel routing dapat diisi dengan IP address jaringan dan tidak terbatas pada IP address host saja. Pembuatan tabel routing secara manual umumnya lebih disukai dalam jaringan TCP/IP yang terdiri atas hanya beberapa router karena jumlah entri dan jumlah router yang harus dikonfigurasi hanya sedikit. Semakin besar jaringan TCP/IP (dihitung dari jumlah router) semakin banyak usaha yang diperlukan untuk membuat tabel routing yang lengkap di tiap-tiap router.

Hitungan kasar jumlah total entri tabel routing yang perlu dimasukkan diseluruh router adalah berbanding pangkat dua terhadap jumlah router yang ada di jaringan tersebut. Jumlah yang besar ini dapat menyebabkan ketidakkonsistenan dalam pengisian entri tabel routing. Dari kenyataan di atas kita melihat ada suatu batasan ukuran jaringan (yang besarnya berbeda untuk setiap orang) ketika pengisian tabel routing secara statik menjadi sulit dilakukan karena sedemikian banyaknya entri yang harus dimasukkan. Di samping kelemahan tersebut, pengisian tabel routing secara manual tidak mencerminkan dinamika jaringan.

Selain kedua metode di atas, pembentukan tabel routing dapat dilakukan dengan menggunakan protokol yang digunakan oleh router-router untuk saling bertukar informasi routing. Router-router pada jaringan TCP/IP membentuk tabel routing berdasarkan informasi routing yang dipertukarkan setiap selang waktu tertentu. Pertukaran informasi antar router dapat menunjukkan dinamika jaringan karena jika suatu saat ada router atau jaringan yang putus berarti router-router yang lain tidak akan menerima informasi routing dari router yang putus tersebut. Jika hal ini terjadi, router-router dalam jaringan akan menghapus entri tabel routing yang informasinya tidak diperbarui. Karena penggunaan protokol routing dapat mencerminkan dinamika jaringan maka metode ini disebut juga dengan routing dinamik.

Keunggulan penggunaan protokol routing yang lain adalah fleksibilitas dan konfigurasi yang umumnya relatif sederhana untuk jaringan yang besar. Kita dapat memilih menggunakan protokol routing yang diinginkan. Keunggulan-keunggulan inilah yang menyebabkab mengapa routing dinamik menjadi pilihan untuk membentuk tabel routing pada jaringan TCP/IP.

2.1.6. RIP versi 1

Spesifikasi protokol routing ini ditulis dalam dokumen RFC 1058. RIP merupakan routing verktor-jarak yang dimodifikasi dengan *triggered update* dan *split horizon* dengan *poisonous reverse* untuk meningkatkan kinerjanya. RIP ditujukan agar host dan router dapat bertukar informasi untuk menghitung rute dalam jaringan TCP/IP. Informasi yang dipertukarkan RIP adalah informasi mengenai *destination* yang dapat berupa host, network, subnet atau rute default.

Setiap host yang menjalankan RIP versi memiliki tabel routing yang setidaknya berisi :

- *Destination IP Address* (host, subnet, network, atau rute default).
- *Metric* yang menunjukkan *cost* total untuk mencapai *destination* tersebut dari host.
- *IP address* router yang dilalui.
- Sebuah tanda untuk menunjukkan apakan ruter baru saja berubah.
- Beberapa timer.

Metric yang digunakan RIP untuk menentukan ruter sangat sederhana, yaitu adalah jumlah hop (*hop count*) antara router dengan *destination*. *Metric* ini adalah bilangan bulat 1 sampai dengan 15, 16 dianggap sebagai tak-hingga. Pembatasan *metric* ini adalah konsekuensi penggunaan routing vektor-jarak dan berakibat membatasi diameter jaringan. *Metric* interface jaringan secara default adalah 1 walaupun spesifikasi RIP mensyaratkan bawa *metric* tersebut dapat diubah (menjadi lebih besar). Pemilihan *metric* ini tentu menyebabkan diameter jaringan menjadi lebih kecil.

RIP mengasumsikan bahwa sebuah jaringan dibagi menjadi subnet dengan ukuran yang sama, misal sebuah jaringan kelas C dibagi menjadi 16 subnet dengan netmask 255.255.255.240. Format paket RIP hanya memuat informasi IP address tanpa informasi netmask. Informasi netmask dianggap sama dengan netmask yang digunakan pada interface router atau host. Proses RIP akan mengambil kesimpulan berdasarkan IP address tersebut saja untuk menentukan apakah IP address yang terdapat di paket RIP termasuk IP address network, subnet, host atau default route. RIP membaca IP address sebagai bilangan 32 bit yang terdiri atas bagian alamat network, subnet, dan host. Sebagai contoh, network 132.92 dengan subnet mask 255.255.255.0. Alamat 132.92.0.0 adalah alamat network, 132.92.36.0 adalah alamat subnet, dan 132.92.36.21 adalah alamat host. RIP memeriksa apakah bagian host dari sebuah IP address tidak terdiri atas angka 0. Jika demikian, RIP langsung mengambil kesimpulan bahwa alamat tersebut merupakan alamat host. RIP mencegah informasi routing alamat subnet keluar dari networknya. Jika terdapat router RIP yang berbatasan dengan network lain, RIP hanya membuat satu entri saja yaitu entri alamat network. Sebagai contoh : sebuah router dengan dua interface dengan nomor network yang berbeda 132.92.36.22 subnetmask 255.255.255.0 dan 171.21.13.8 subnetmask 255.255.255.0. RIP hanya mengirimkan satu entri 132.92 ke interface 171.21.13.8 dan juga satu entri 171.21 ke interface 132.92.36.22.

Gambar 2.11 memperlihatkan format paket RIP (versi 1). Pada gambar tersebut hanya diperlihatkan data untuk satu entri saja. Panjang maksimum paket RIP adalah 512 byte, tanpa header UDP. Entri vektor-jarak RIP dimulai dari tipe keluarga alamat sampai metrik. Entri ini dapat muncul sampai 25 kali. Jenis perintah (*command*) pada paket RIP yang digunakan adalah *request* dan *response*. Perintah *request* adalah untuk meminta router mengirimkan tabel routing sementara perintah *response* adalah untuk mengirimkan tabel routing. Versi RIP dalam paket tersebut adalah 1 dan tipe keluarga alamat (*address family identifier*) untuk IP adalah 2. IP address dalam paket adalah alamat biasa dan field *metric* berisi bilangan bulat dari 1 sampai 15 untuk entri yang aktif. *Metric* 16 digunakan untuk entri yang tidak dapat dicapai.

Perintah (1)	Versi (1)	harus nol (2)
Tipe keluarga alamat (1)		harus nol (2)
IP address (4)		
harus nol (4)		
harus nol (4)		
metric (4)		

Gambar 2.11. Format Paket RIP versi 1

Gambar 2.12. Router RIP pada perbatasan jaringan komputer

Pertukaran informasi routing dilakukan RIP setiap 30 detik. Pertukaran informasi routing tersebut pada jaringan lokal dengan beberapa router dapat mengakibatkan kongesti (*congestion*). Kongesti dapat terjadi karena router-router mengirimkan paket RIP dalam waktu yang hampir bersamaan. Untuk menghindari kongesti, router dapat menambah waktur pertukaran itu secara acak menjadi sedikit lebih dari 30 detik. Router-router yang menjalankan RIP bertukar informasi dengan cara melakukan broadcast paket RIP *response* menggunakan UDP pada port 520. Sementara router di jaringan melakukan broadcast paket RIP masing-masing, host-host yang terletak di jaringan yang sama dapat mendengar pesan tersebut. Host-host tersebut juga menjalankan RIP tetapi hanya mendengarkan paket-paket RIP yang disampaikan router dan tidak mengirimkan pesan. Inilah yang disebut sebagai node diam (*silent node*), yaitu node-node yang hanya mendengar paket RIP *response* dan membentuk tabel routing berdasarkan paket-paket tersebut.

Pada saat terjadi pertukaran informasi RIP, karena *triggered update* ataupun yang reguler, timer untuk entri-entri tabel routing yang aktif diperbarui lagi. Jika sebuah entri tabel routing tidak diperbarui dalam 180 detik, entri terseut dianggap kadaluwarsa dan siap dihapus dari tabel. Sebuah entri tabel routing juga dapat dihapus jika *metric* untuk entri tersebut dibuah menjadi 16 (tak-hingga) oleh router yang perlu dilaluinya. *Metric* entri-entri tabel yang hendak dihapus pertama-tama dibuah menjadi 16 dan dibiarkan dahulu di tabel routing 120 detik sebelum benar-benar dihapus. Karena *metric* entri tersebut berubah, maka secara otomatis router perlu mengirimkan paket RIP karena dipicu oleh perubahan tabel routing (*triggered update*). Spesifikasi RIP menyebutkan bahwa paket RIP yang dikirim karena *triggered update* hanya perlu berisi entri-entri yang berubah. Selama 120 detik tersebut, entri-entri yang hendak dihapus itu juga disertakan dalam paket RIP *response* yang dikirim setiap 30 detik. Tujuan menyertakan entri-entri tersebut adalah untuk lebih menjamin agar router-router di jaringan juga akan menghapusnya dari tabel routing.

2.1.7. RIP versi 2

Protokol ini dapat dikatakan sebagai perluasan dari RIP versi 1 dengan menambahkan beberapa kemampuan baru. RIP versi 2 (RIPv2) sama sekali tidak mengubah algoritma routing vektor-jarak yang digunakan RIP (RIPv1). Perubahan yang dilakukan RIPv2 hanya pada format paket RIP yang bersifat menambah informasi yang dikirimkan. Kemampuan-kemampuan informasi baru

RIPv2 adalah : *tag* untuk rute eksternal, subnet mask, alamat hop berikut, dan autentifikasi.

<table>
<tr><td>Perintah (1)</td><td>Versi (1)</td><td>tidak digunakan</td></tr>
<tr><td colspan="2">Tipe keluarga alamat (1)</td><td>Router Tag (2)</td></tr>
<tr><td colspan="3">IP address (4)</td></tr>
<tr><td colspan="3">subnet mask (4)</td></tr>
<tr><td colspan="3">next hop (4)</td></tr>
<tr><td colspan="3">metric (4)</td></tr>
</table>

Gambar 2.13. Format paket RIP versi 2

Perintah tipe keluarga alamat, IP address serta *metric* pada RIPv2 memiliki arti yang sama dengan RIP. Versi yang digunakan pada paket RIPv2 adalah 2. Dari Gambar 2.13 terlihat bahwa RIPv2 mengunakan bagian dari paket yang pada RIP harus diisi nol.

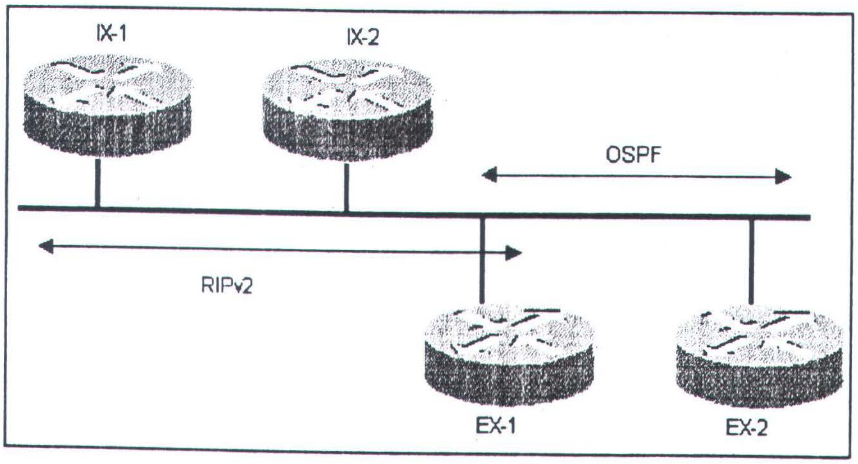
Tag untuk rute eksternal memberikan kemampuan bagi RIPv2 untuk membedakan RIP "internal" (jaringan dalam domain RIP) dari RIP "eksternal". Penggunaan *tag* ini adalah untuk rute-rute yang berasalah dari EGP atau dari protokol routing lain. Informasi subnet mask menjadikan RIPv2 mampu mendukung subnet mask yang berbeda di jaringan atau *variable length subnet mask* (VLSM). Jika field ini kosong RIPv2 menganggap tidak ada informasi subnet mask yang dikirimkan. Interaksi RIPv2 dengan RIP mengenai informasi IP address dan subnet mask mengingat RIP menganggap jaringan subnet mask yang sama. Cara yang digunakan RIPv2 untuk berinteraksi dengan RIP adalah tidak mengirimkan informasi rute dengan subnet mask yang lebih spesifik karena RIP akan menganggapnya sebagai rute menuju host.

Router yang menjalankan RIPv2 dapat memberi tahu agar router lain tidak melewatkan suatu rute melalui dirinya dengan cara mengisi field alamat *next hop* dengan router yang harus dilalui oleh datagram menuju router tersebut. Field yang diisi dengan 0.0.0.0 berarti rute melewati router yang mengeluarkan paket IP *response*. Router yang terletak dalam field *next hop* harus terletak dalam satu jaringan lokal dengan router yang mengirim paket RIP. Field ini berguna untuk mencegah datagram mengambil rute yang tidak efisien. Biasanya field ini digunakan pada perbatasan jaringan yang menggunakan protokol routing selain RIPv2.

Pada gambar 2.14 terlihat empat buah router yang terhubung pada suatu jaringan lokal. Router IX-1, IX-2, dan EX-1 menjalankan RIPv2. EX-1 dan EX-2 menjalankan protokol OSPF. EX-1 dapat memanfaatkan field alamat *next hop* untuk menunjukan rute apa saja yang harus dilewatkan melalui router EX-2. Dengan demikian router EX-2 tidak perlu ikut serta dalam routing RIPv2.

RIPv2 juga dapat menggunakan autentikasi seperti terlihat pada gambar 2.15. Karena paket RIPv2 dengan autentikasi menghabiskan data setara dengan satu entri router, maka jumlah maksimum entri rute dalam satu paket RIPv2 turun menjadi 24. Saat ini autentikasi yang dapat digunakan hanya password tipe 2, yaitu passsword sederhana dalam bentuk teks biasa.

Paket RIP (RIPv1) menggunakan paket broadcast yang membebani seluruh host dalam jaringan lokal. Untuk mengurangi beban host-host, paket RIPv2 dikirim menggunakan paket multicast, dengan IP address 224.0.0.9. Selain itu paket RIPv2 juga dapat dikirim menggunakan paket broadcast untuk kompatibilitas dengan versi 1.



Gambar 2.14. Router EX-1 menggunakan field *next hop*

<table>
<tr><td>Perintah (1)</td><td>Versi (1)</td><td>tidak digunakan</td></tr>
<tr><td colspan="2">0xFFFF</td><td>Tipe Autentikasi (2)</td></tr>
<tr><td colspan="3">IP address (4)</td></tr>
<tr><td colspan="3">Autentikasi (16)</td></tr>
</table>

Gambar 2.15. Format autentikasi RIPv2

2.1.8. OSPF

Protokol routing *Open Shortest Path First* (OSPF) dirancang dan dikembangkan untuk jaringan TCP/IP oleh sebuah kelompok kerja OSPF. Spesifikasi terakhir OSPF versi 2 yang dihasilkan kelompok kerja OSPF tertulis dalam dokumen RFC 2178. OSPF mendukung jaringan *point-to-point, point-to multipoint,* dan jaringan multiakses. Protokol ini termasuk kategori routing *link-state* oleh seba itu OSPF dapat menghitung jalur routing yang pasti terbebas dari *loop.*· Beberapa kelebihan dari OSPF lainnya antara lain dapat dengan cepat mendeteksi perubahan yang terjadi di jaringan dan menjadikan routing kembali kovergen dalam waktu yang singkat dengan sedikit pertukaran data. OSPF mampu menangani routing jaringan TCP/IP yang besar dan membuat hierarki routing dengan membagi jaringan menjadi beberapa area. Paket-paket OSPF dikirim dan diterima dengan memanfaatkan TCP/IP multicast. Setiap paket routing OSPF menggunakan autentikasi untuk mengingkatkan keamanan. Kelemahan protokol OSPF adalah membutuhkan kemampuan CPU dan memori yang relatif besar.

Proses dasar dalam routing OSPF adalah menghidupkan *adjacency*, proses *flooding*, dan penghitungan tabel routing. Router-router mengirimkan paket *hello* ke seluruh jaringan yang terhubung dengannya secara periodik. Jika paket *hello* sebuah router tidak terdengar setelah selang waktu tertentu, router tersebut dianggap mati. Selang waktu ini secara default ditentukan empat kali interval pengiriman paket *hello*.

2.2. *Merit GateD Consortium*

Merit Gate D Consortium membuat dan mendistribusikan perangkat lunak GateD (atau *GateDaemon*), sebuah paket routing protokol yang bersifat modular yang berfungsi menghubungkan antar jaringan komputer. *Merit GateD Consortium* mempunyai tujuan untuk menyediakan dukungan terhadap Internet routing protokol yang paling baru.

GateD merupakan sebuah perangkat lunak yang berjalan di atas sistem operasi Unix atau yang sejenis. GateD menyediakan dukungan terhadap protokol routing RIP, RIPv2, OSPF, ISIS, SLSP dan BGP.

Perangkat lunak GateD digunakan oleh ratusan organisasi di dunia untuk menghubungkan *packet-switched* jaringan komputer. Praktisi dibidang jaringan komputer juga mempergunakan GateD sebagai bahan penelitian dan *prototype platform* protokol routing.

Beberapa alasan penggunaan perangkat GateD oleh pengembang dan industri jaringan komputer di dunia adalah :

1. Merupakan media pengembangan protokol routing.
2. Merupakan referensi *de facto* untuk protokol routing.
3. Merupakan alat untuk mendistribusikan protokol routing baru.
4. Merupakan fasilitas acuan untuk pengembangan produk-produk komersial.
5. Merupakan sumber daya bagi peneliti dan pembuat produk jaringan komputer.

BAB III

PERANCANGAN DAN PEMBUATAN PERANGKAT PERANGKAT LUNAK ROUTER

Bab ini membahas perancangan dan pembuatan sistem perangkat lunak router secara garis besar. Perangkat lunak yang spesifikasinya akan diuraikan dalam sub-sub bab berikut, dibuat dengan memanfaatkan kernel Linux, dan beberapa aplikasi yang bersifat *GNU General Public Lisence*, sedangkan untuk membangun menjadi sebuah aplikasi yang dimaksud, yaitu Linux Router, dipergunakan *GNU C compiler, perl* dan *shell script.*

3.1. Tujuan Sistem

Tujuan dari sistem yang dibuat adalah membuat perangkat komputer menjadi sebuah LAN router.dengan kemampuan:

- Operasional yang mudah.
- Firewall.
- Network Address Translator (NAT)
- Protokol routing RIP, RIPv2, OSPF dan BGP.
- Simple Network Management Protocol (SNMP).
- Radius client dengan 2 buah *dial-in* pada serial port komputer.

3.2. Kebutuhan Sistem

Perangkat lunak yang dibuat dapat dijalankan dengan menggunakan media penyimpanan disket maupun hard disk. Untuk menjalankan perangkat lunak ini dibutuhkan memori fisik minimal sebesar 16 Mbytes dengan prosesor Intel 386 DX atau yang lebih baik. *Network Interface Card* (NIC) yang didukung oleh perangkat lunak ini adalah NE-2000 kompatibel ISA maupun PCI, Intel Express Pro 10, Intel Express Pro 10/100, dan 3Com 3C5X9.

3.3. Entitas Sistem dan Desain Sistem

Secara garis besar perangkat lunak yang dibuat dapat dibagi menjadi tiga bagian:

- Kernel

  Bagian program yang berfungsi sebagai sistem operasi, dalam hal ini dipilih Linux.

- Modul program

  Bagian sistem yang berfungsi pada tingkatan aplikasi untuk alat bantu router.

- Konfigurator

  Bagian program yang berfungsi untuk mengubah file konfigurasi dan melakukan penyimpanan program ke dalam media penyimpanan.

Sistem operasi Linux dipilih sebagai pondasi dari perangkat lunak yang dibagun, dengan alasan:

- Bersifat *GNU General Public Lisence* (GPL) yaitu suatu suatu proyek pembuatan perangkat lunak yang bersifat bebas tanpa adanya jaminan dari pembuat perangkat lunak dan bersifat gratis[3], sehingga dapat digandakan dengan legal tanpa membayar kepada suatu pihak.

- Bersifat *open source*, sehingga dapat diperbaiki dan diubah sesuai dengan keinginan.
- Merupakan salah satu jenis sistem operasi Unix yang telah terpercaya untuk keperluan jaringan komputer.

Metoda desain sistem yang dipilih adalah *patching* dan *scripting*, *Patching* yaitu melakukan perubahan dan penambahan ke dalam program yang telah ada kemudian melakukan kompilasi ulang. Dengan demikian tidak perlu melakukan pembangunan aplikasi program dari nol. Sedangkan *scripting* adalah pemrograman dengan menggunakan bahasa *script*, yang digunakan disini adalah *perl* dan *shell script*.

3.3.1. Gambaran Umum Cara Kerja Perangkat Lunak Router

Perangkat lunak router ini dapat disimpan dan beroperasi dalam satu floppy disket 3.5" atau dalam hard disk, namun lebih diarahkan untuk pemakaian floppy disket. File-file program perangkat lunak router tersimpan dalam bentuk terkompresi. Program akan berjalan dan melakukan proses di dalam memori dan ram disk. Semua proses perubahan dan kerja perangkat lunak berada pada memori. Sistem tidak akan melakukan penyimpanan ke media penyimpanan tetap (disket atau hard disk) selama tidak dilakukan proses backup. Setelah perangkat lunak berjalan, cara pengoperasian dan cara konfigurasi sistem, sama dengan sistem operasi Linux pada umumnya, kecuali pada proses backup.

---

[3] ___,"http://www.gnu.org/gnu/thegnuproject.html", 1998

Perbedaan lain dari sistem operasi Linux pada umumnya adalah, perangkat lunak router yang dibuat tidak memerlukan proses *shutdown* sebagaimana layaknya sistem operasi Linux.

3.3.2. Modul Program

File-file yang terdapat dalam perangkat lunak router dikelompokkan menjadi beberapa modul. Pengelompokkan dilakukan berdasarkan fungsi dari masing-masing file. Masing-masing modul disimpang dalam bentuk satu file terkompresi. Adapun pengelompokkan tersebut adalah sebagai berikut:

- Modul etc (etc.lrp / etc.tgz)

  Modul ini terdiri dari direktori /etc dan file-file konfigurasi sistem operasi Linux. Anggota modul etc lebih detail dapat dilihat pada Lampiran 1.

- Modul log (log.lrp / log.tgz)

  Modul ini terdiri dari direktori /var/log dan file-file log yang dibuat oleh sistem operasi Linux selama sistem operasi berjalan. File-file log dibentuk menjadi modul tersendiri dengan tujuan agar administrator jaringan dapat melakukan dokumentasi atau pembackupan file log sistem saja. Dengan demikian ukuran file backup tidak terlalu besar dan mudah untuk didokumentasikan. Anggota modul log lebih detail dapat dilihat pada Lampiran 2.

- Modul snmp (snmp.lrp / snmp.tgz)

  Modul ini terdiri file-file yang ada hubungannya dengan aplikasi snmp. Aplikasi ini berfungsi untuk menyediakan *agent* snmp. Perangkat lunak *agent* snmp yang digunakan adalah *cmu snmp linux versi 3.2*. Perangkat lunak *agent* snmp ini dibentuk tersendiri menjadi modul dengan tujuan untuk menyediakan kemampuan *network monitoring* dengan menggunakan snmp, yang dapat dipergunakan bila diperlukan. Anggota modul snmp lebih detail dapat dilihat pada Lampiran 3.

- Modul gated (gated.lrp / gated.tgz)

  Modul ini terdiri dari file-file yang ada hubungannya dengan aplikasi gated. Aplikasi ini berfungsi untuk menyediakan protokol routing RIPv1, RIPv2, OSPF dan BGP. Perangkat lunak yang dipergunakan adalah *GateD* dari *Merit Consortium.* Perangkat lunak gated ini dibentuk tersendiri menjadi modul dengan alasan tidak semua router beroperasi membutuhkan protokol routing dinamik. Sehingga apabila sebuah router beropersi dengan hanya mempergunakan *default gateway*, maka modul ini tidak perlu digunakan. Anggota modul gated lebih detail dapat dilihat pada Lampiran 4.

- Modul pslave (pslave.lrp / pslave.tqz)

  Modul ini terdiri dari file-file yang ada hubungannya dengan aplikasi port slave. Aplikasi ini berfungsi untuk menyediakan layanan radius client dan dial-in. Perangkat lunak yang digunakan adalah *Port Slave*. Anggota modul pslave lebih detail dapat dilihat pada Lampiran 5.

- Modul modules (modules.lrp / modules.tgz)

  Modul ini terdiri dari file-file yang dikompile sebagai modul program sistem operasi Linux. Kegunaan utama dari program ini adalah untuk menyediakan *device driver* untuk sistem operasi. Modul ini dipisahkan dengan program lain dengan tujuan agar supaya memudahkan dalam menambahkan ataupun mengurangi program Linux dalam bentuk *module*. Anggota modul modules lebih detail dapat dilihat pada Lampiran 6.

- Modul root (root.lrp / root.tgz)

Modul ini terdiri dari struktur direktori pada sistem operasi Linux, file-file program utama dan file-file konfigurasi yang diperlukan pada saat proses boot. Beberapa file dan program penting yang menjadi anggota modul ini adalah:

- linuxrc

  File linuxrc menyimpan *shell script* yang dipergunakan untuk membentuk file sistem pada saat sistem perangkat lunak router melakukan *boot.* Selain membentuk file sistem file ini juga bertugas melakukan dekompresi dari media *boot* kedalam ram disk.

- ash

  File ini merupakan perangkat lunak minimum unix shell yang mempunyai kegunakan utama untuk menyediakan *user interface* bagi pemakai.

- ifconfig

  File ini merupakan perangkat lunak yang berfungsi untuk melakukan akses ke NIC terutama pemberian alamat IP dan memberikan laporan status suatu NIC.

- route

  File ini merupakan perangkat lunak yang berfungsi untuk memberikan perintah-perintah yang ada hubangan dengan tabel routing.

- ping

  File ini merupakan perangkat lunak yang berguna untuk melakukan pemeriksaan konektifitas jalur jaringan komputer.

- traceroute

  File ini merupakan perangkat lunak yang berguna untuk memeriksa lintasan yang dilalui dari satu titik dalam jaringan komputer ke titik lain.

- insmod

  File ini merupakan perangkat lunak yang berfungsi untuk menjalankan program Linux yang berbentuk *module*.

- rmmod

  File ini merupakan perangkat lunak yang berfungsi untuk menghentikan dan menghapus program Linux dari memori yang berbentuk *module*.

- edit

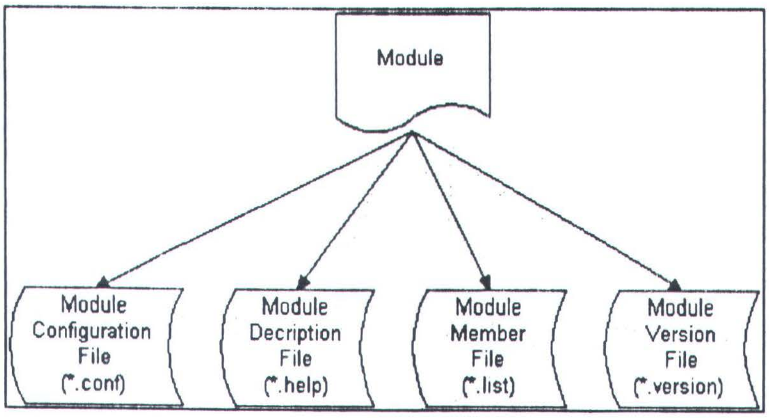
  File ini merupakan perangkat lunak yang berfungsi sebagai *editor* sederhana untuk melakukan perubahan-perubahan terhadap file-file konfigurasi.

Anggota modul root adalah seluruh direktori dalam emulasi router selain anggota modul-modul yang telah disebutkan diatas.

Anggota tiap modul didaftarkan pada sebuah file yang tercantum didalam masing-masing modul dengan nama file [*modul*].*list*, sebagai contoh untuk modul snmp nama file daftar anggotanya adalah snmp.list.

Pengelompokan kedalam modul-modul ini ditujukan agar supaya pemakai cukup menyertakan modul yang diperlukan ke dalam media penyimpanan. Hal ini akan menghemat pemakaian ruang penyimpanan. Disamping itu proses backup dapat dilakukan pada bagian modul yang mengalami perubahan saja, sehingga proses backup dapat dijalankan dalam waktu yang relatif lebih singkat.

Untuk membuat pengelompokan modul dalam bentuk terkompresi digunakan *GNU tar archiving utility*.



Gambar 3.1. Anggota khusus sebuah modul.

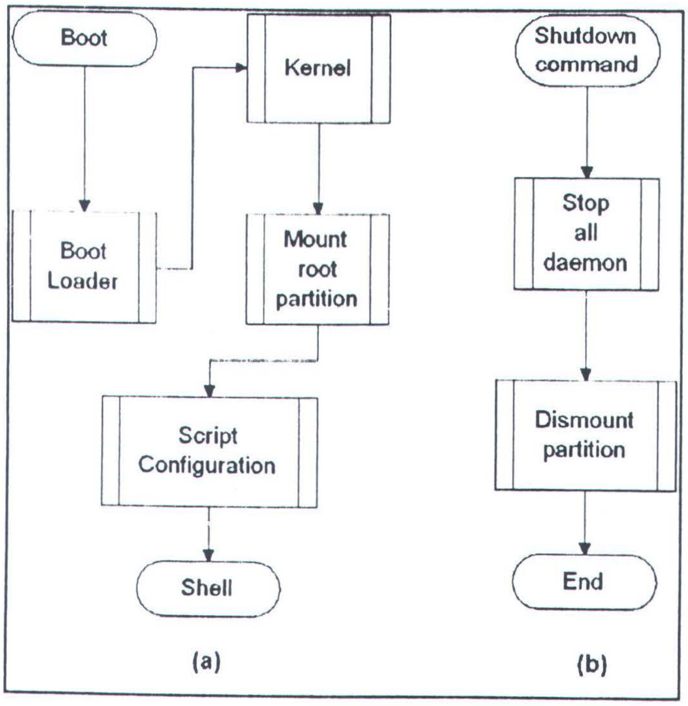
3.3.3. Desain Proses Boot Sistem

Salah satu kelemahan yang ada pada hampir semua sistem multiuser adalah pada proses *boot* dan *shutdown* sistem. Sistem multiuser membuat suatu sistem sedemikian rupa sehingga diperlukan suatu tata cara proses *boot* maupun *shutdown* sistem yang melindungi keseluruhan sistem dari kegagalan ataupun kehilangan proses. Umumnya pada proses *shutdown* dilakukan pengecekkan keseluruhan proses dan melakukan *dismounting* (penutupan file sistem) terlebih dahulu sebelum proses *shutdown* yang sebenarnya dijalankan. Proses ini dilakukan mengingat sistem operasi multiuser memcantumkan suatu tanda tertentu pada setiap proses yang berjalan. Tanda tersebut harus dalam keadaan bersih pada saat sistem pertama kali berjalan. Apabila tanda ini belum dibersihkan atau belum dilakukan penutupan proses, maka pada saat sistem berjalan pertama kali setelah melalui proses *boot* pada saat berikutnya, sistem akan menganggap ada suatu proses yang masih berjalan. Hal ini tidak boleh terjadi mengingat hubungan antara tanda proses dan proses itu sendiri dalam memori haruslah sama. File sistem juga diperlakukan sama dengan proses. File sistem diberikan suatu tanda khusus, mengingat file sistem merupakan tempat penyimpanan file. Pada proses multiuser satu file dapat diakses lebih dari satu proses pada satu saat. Untuk itulah dilakukan proses *mounting* (pembukaan file sistem) yang merupakan persamaan dari proses *boot* pada sistem operasi dan *dismounting* (penutupan file sistem) yang merupakan persamaan dari proses *shutdown* pada sistem operasi.

Pada suatu sistem router, proses-proses tersebut diatas dirasa mengganggu pengoperasian router. Hal ini dikarenakan sebuah router semestinya dapat dioperasikan dengan mudah tanpa harus melalui proses *boot* dan *shutdown* yang berbelit. Disamping itu sistem router umumnya tidak melakukan perubahan atau pengaksesan ke dalam file sistem. Sistem router cenderung melakukan keseluruhan operasi dan melakukan perubahan-perubahan pada memori sistem operasi.

Untuk itulah perlu adanya sistem *boot* dan *shutdown* yang berbeda dengan sistem operasi Linux pada umumnya bagi perangkat lunak router ini. Proses *boot* dan *shutdown* pada sistem operasi Linux ditunjukkan pada Gambar 3.2.

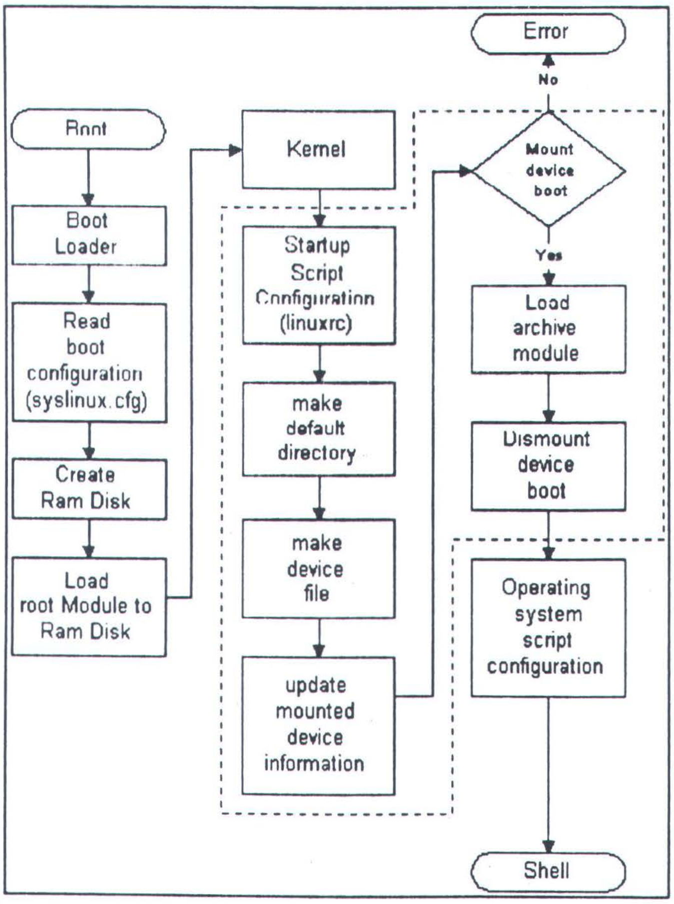
Perubahan proses *boot* dan *shutdown* pada perangkat lunak yang dibuat terutama ditujukan untuk menghindari keharusan proses *dismounting* file sistem. Sehingga sistem router dapat tetap beroperasi sebagaimana mestinya walaupun dimatikan secara mendadak. Untuk mencapai tujuan tersebut dilakukan perubahan pada proses *boot*, seperti terlihat pada Gambar 3.3.



Gambar 3.2. Diagram Alur proses *boot* (a) dan *shutdown* (b) pada sistem operasi Linux.

Diagram alur dari perubahan proses *boot*, seperti terlihat pada Gambar 3.3 dapat dijelaskan sebagai berikut:

- Pertama BIOS membaca boot sector disk dan menjalankan program boot loader yang ada pada boot sector disk. Boot loader ini merupakan standard boot loader dari Linux. Boot loader membaca file konfigurasi boot yaitu file syslinux.cfg.
- Kemudian berdasarkan file konfigurasi boot diatas, sistem akan membuat ram disk dalam memori fisik dan memuat file sistem (*root.lrp/root.tgz*) sekaligus melakukan dekompresi file ke dalam ram disk.



Gambar 3.3. Diagram alur proses boot pada Linux router.

- Proses selanjutnya adalah pemuatan kernel linux ke dalam memori dan kernel dijalankan untuk memulai sistem operasi. Mengingat file sistem berada pada virtual disk, maka dilakukan perubahan pada kernel sehingga mampu mengarahkan proses pengenalan file sistem pada ram disk dan melakukan proses *mounting* ram disk sebagai root file sistem.
- Setelah root file sistem terbentuk, sistem akan menjalankan *startup script configuration* (linuxrc) untuk melakukan inisialisasi sistem operasi. *Startup script configuration* melakukan proses sebagai berikut:
- Membuat *default directory* yaitu struktur file sistem direktori.

- Membuat *device file*. *Device file* dibuat pada saat prose *boot* dan tidak disimpan ke dalam media penyimpanan dengan tujuan agar supaya dapat mengurangi besar file modul.
- Kemudian dilakukan proses pembaharuan (*update*) informasi *device* yang telah di *mount*. Hal ini dibutuhkan oleh sistem operasi Linux.
- Kemudian dilakukan proses *mout* media yang digunakan untuk *booting*.
- Selanjutnya dilakukan dekompresi modul-modul file yang dibutuhkan ke dalam ram disk.
- Setelah modul berhasil di muat dalam ram disk, media *boot* di *dismount*.
- Sampai tahap ini, sistem telah membuat suatu lingkungan sistem operasi Linux sebagaimana sistem operasi Linux pada umumnya. Kemudian proses selanjutnya adalah menjalankan modul-modul program yang dikehendaki pada file konfigurasi sistem operasi, dan menjalankan skrip konfigurasi sistem operasi.
- Langkah terakhir pada proses boot ini adalah, menjalankan program shell, yang berfungsi sebagai user interface.

Dari uraian diatas dapat dilihat bahwa, proses boot tersebut ditujukan agar semua proses yang berhubungan dengan file dilakukan di ram disk. Sehingga media penyimpanan fisik (disk drive atau hard disk) hanya berfungsi sebagai media boot dan backup permanen.

Untuk mengimplementasikan proses ini, dilakukan perubahan pada kernel Linux dengan menggunakan *patching*. Proses *patching* dilakukan pada bagian kernel yang bernama *initrd,* yaitu bagian kernel yang berfungsi untuk membentuk ram disk[4]. Sedangkan agar sistem operasi Linux selalu menjalankan *startup file configuration* (*linuxrc*) dilakukan *patching* pada fungsi *kernel_thread* dalam kernel Linux. Kernel Linux yang digunakan adalah kernel versi 2.0.36.

3.3.4. Konfigurasi Kernel

Seperti telah disebutkan diatas, bahwa kernel yang digunakan dalam pembuatan perangkat lunak router adalah Linux kemel versi 2.0.36. Alasan penggunaan versi ini adalah:

- Merupakan versi kernel paling stabil dan dianjurkan dalam keluarga 2.0.xx.
- Telah memiliki kemampuan dasar *bridging* dan *packet forwarding*.
- Berukuran relatif kecil (< 512 kbytes).

Disamping perubahan kemel dilakukan dengan metode *patching*, kernel juga dikonfigurasi dengan format khusus. Adapun konfigurasi secara detail dijelaskan dibawah ini.

- CONFIG_EXPERIMENTAL=y

  Mengijinkan program-program *driver* yang masih dalam tahap *beta* atau percobaan. Hal ini diperlukan untuk mengijinkan perubahan sebagian kernel maupun *device driver* tanpa harus mencantumkan perubahan pada program utama kernel.

- CONFIG_MODULES=y, CONFIG_MODVERSIONS=y

  Mengijinkan kompilasi sebagian kernel dalam bentuk *modules*, yang dijalankan apabila diperlukan. Hal ini ditujukan agar beberapa program yang

---

[4] ___,"Linux Programming Manual", 1998, INITRD(4)

dapat dipisahkan dari kernel dapat dijalankan pada saat diperlukan, sehingga ukuran kernel menjadi lebih kecil dan lebih cepat dalam proses *boot*.

- CONFIG_NET=y

  Kernel beroperasi dalam mode jaringan (*network*).

- CONFIG_PCI=y

  Dukungan terhadap PCI-bus diaktifkan.

- CONFIG_SYSVIPC=y

  Pengaktifan *Inter Process Communication* (IPC).

- CONFIG_BINFMT_ELF=y

  Pengaktifan mode *Executable and Linkable Format* (ELF) yaitu suatu format *libraries* dan *executable* yang digunakan antar arsitektur dan sistem operasi yang berbeda[5].

- CONFIG_KERNEL_ELF=y

  Kernel berjalan dengan model EFL.

- CONFIG_M386=y

  Kernel dikompilasi dengan 386 *assembly mesin code*. Sehingga perangkat lunak mampun berjalan pada prosesor 386 atau yang lebih baik.

- CONFIG_BLK_DEV_FD=y, CONFIG_BLK_DEV_IDE=y

  Pengaktifan floppy disk *block device* dan IDE *block device*. Perangkat lunak memanfaatkan floppy disk dan IDE interface (hard disk) sebagai media penyimpanan.

- CONFIG_BLK_DEV_IDEFLOPPY=y

  Mengakfitkan mode akses ke floppy disk melalui IDE-bus.

---

[5] EFL Linux HOW-TO

- CONFIG_BLK_DEV_LOOP=y

  Mengijinkan file reguler sebagai *block device* yang berfungsi sebagai enkripsi sistem file.

- CONFIG_BLK_DEV_RAM=y

  Menggunakan sebagaian memori fisik sebagai virtual disk drive (ram disk).

- CONFIG_BLK_DEV_INITRD=y, CONFIG_BLK_DEV_INITRD_ARCHIVE=y, CONFIG_BLK_DEV_INITRD_ARCHIVE_AUTOFS=y

  Mengaktifkan inisialisasi ram disk melalui *boot loader* dan melakukan *mouting* ram disk sebagai partisi root.

- CONFIG_FIREWALL=y, CONFIG_IP_FIREWALL=y, CONFIG_IP_FIREWALL_VERBOSE=y

  Mengijinkan penyaringan paket data TCP/IP.

- CONFIG_NET_ALIAS=y

  Mengaktifkan dukungan IP *aliasing*, yaitu kemampuan mengalamati lebih dari satu alamat IP pada sebuah NIC.

- CONFIG_INET=y

  Pengaktifan protokol TCP/IP.

- CONFIG_IP_FORWARD=y

  Mengakfifkan kemampuan paket *forwading.*

- CONFIG_IP_MULTICAST=y

  Mengaktifkan kemampuan *multicast broadcast.*

- CONFIG_SYN_COOKIES=y

  Menyediakan kemampuan *syn cookies*, yaitu penandaan paket TCP/IP sehingga kernel dapat membedakan paket *flooding* (kiriman paket yang sama dan terus menerus) dan paket dari koneksi normal.

- CONFIG_IP_MASQUERADE=y,

  CONFIG_IP_MASQUERADE_IPAUTOFW=y,

  CONFIG_IP_MASQUERADE_IPPORTFW=y,

  CONFIG_IP_MASQUERADE_ICMP=y

  Pengaktifan mode *masquerade* untuk memberikan layanan NAT.

- CONFIG_IP_TRANSPARENT_PROXY=y

  Pengaktifan kemampuan *transparent proxy* yaitu, kemampuan untuk membelokkan paket data protokol HTTP ke suatu host tertentu yang bertidak sebagai proxy.

- CONFIG_IP_ALWAYS_DEFRAG=y

  Kemampuan membagi paket data kedalam bagian yang kecil-kecil, apabila terdapat paket data dalam ukuran sangat besar.

- CONFIG_IP_ACCT=y

  Kemampuan untuk melakukan perhitungan pada lalu lintas data paket.

- CONFIG_IP_ROUTER=y

  Pengaktifan mode router. Linux kernel dapat beroperasi dalam dua mode, yaitu mode host dan mode router. Kernel menangani berbagai pekerjaan yang menjadi tanggung jawabnya secara simultan. Masing-masing pekerjaan terdapat prioritas dan alokasi waktu yang berbeda. Apabila kernel beroperasi pada mode router, maka prioritas utama kernel adalah melakukan paket *forwarding*.

- CONFIG_IP_ALIAS=m

  Program yang memberi kemampuan IP *aliasing* dikompile dalam bentuk *module*. Sehingga akan dijalankan dan dimuat dimemory bila diperlukan.

- CONFIG_BRIDGE=y

Mengaktifkan kemampuan *bridging* pada kernel.

Sedangkan untuk *device driver* dan protokol PPP, dikompilasi sebagai *module*. Hal ini dipilih agar mengurangi besar file kernel dan untuk mempercepat proses *boot*.

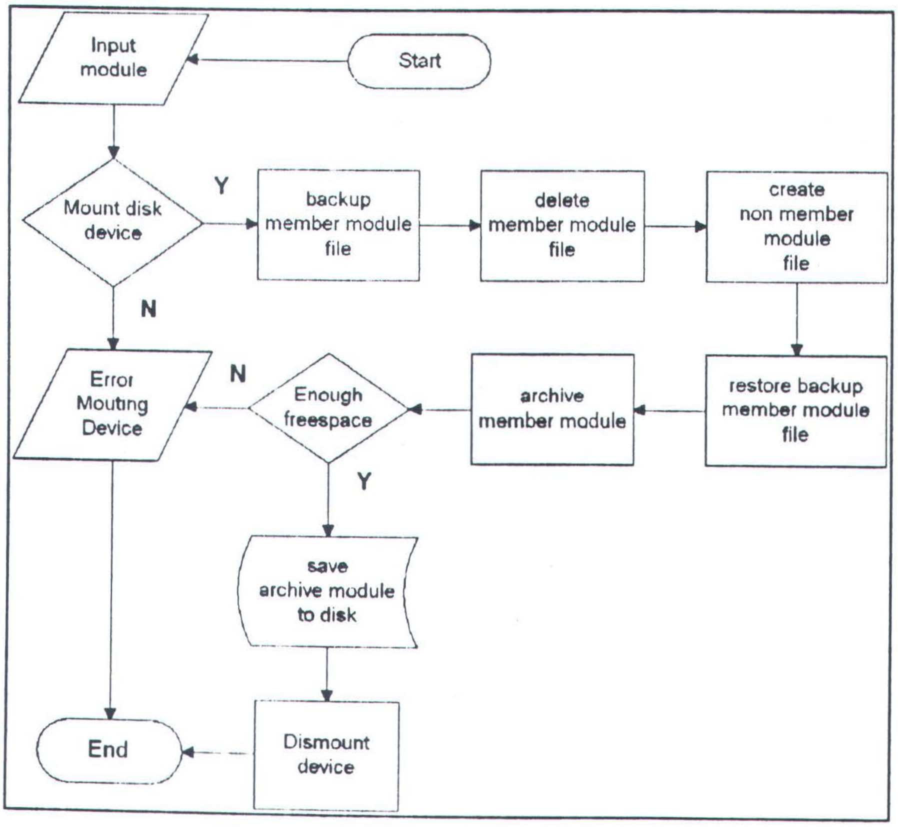
3.3.5. Desain Backup Sistem

Perubahan yang terjadi pada suatu sistem router yang perlu disimpan adalah konfigurasi. Konfigurasi pada perangkat lunak router yang dibuat didesain berupa file text. Sehingga apabila perubahan dilakukan pada konfigurasi router perlu dilakukan backup ke dalam media penyimpanan, mengingat semua proses dilakukan pada ram disk. Karena penyimpanan file-file sistem router pada media penyimpanan dilakukan dalam bentuk terkompresi dan berbentuk modul-modul program, maka alur proses backup sistem dapat dilihat seperti pada Gambar 3.4.

Alur proses backup seperti terlihat pada Gambar 3.4 dapat dijelaskan sebagai berikut:

- Diperlukan masukan berupa jenis modul program yang hendak dibackup.
- *Mouting* media penyimpanan.
- Membuat daftar nama-nama file dari anggota modul program tersebut.
- Membuat daftar nama-nama file yang bukan anggota modul program yang dipilih.
- Melakukan proses kompresi anggota modul program. Proses ini masih dilakukan pada ram disk dan ditempatkan pada direktori /tmp.
- Penyalinan file hasil kompresi yang telah terbentuk, ke dalam media penyimpanan.

- *Dismounting* media penyimpanan.
- Penghapusan file-file yang terbentuk selama proses backup.



Gambar 3.4. Diagram alur proses *backup* sistem.

Untuk mengimplementasikan proses ini dilakukan dengan menggunakan *shell script*.

3.3.6. Desain Media Penyimpanan Floppy Disket

Untuk menampung file-file program yang digunakan dalam router dan untuk keperluan boot, dapat digunakan floppy disket 3.5" *High Density* (HD) atau hard disk. Namun mengingat keseluruhan program yang ada pada sistem router hanya berukuran 2,1 Mbytes, maka penggunaan hard disk sebagai media penyimpanan sistem router ini kurang efisien. Untuk itulah dibuat sebuah media penyimpanan pada floppy disket 3.5" dengan format khusus.

Meskipun besar keseluruhan program dalam bentuk terkompresi sebesar 2,1 Mbytes, tidak keseluruhan program atau modul biasa digunakan pada suatu saat. Sebagai contoh sebuah router dengan mengaktifkan suatu protokol routing hampir tidak pernah ada yang sekaligus menjalankan *radius client* dan atau *dial-in host*. Kebutuhan besar media penyimpanan dengan konfigurasi optimum adalah sebesar 1.7 Mbytes. Untuk itulah dibuat sebuah sistem penyimpanan dengan media floppy disket 3.5".

Floppy disket 3.5" *Double Side High Density* (DD/HD) mempunyai kapasitas 1.44 Mbytes pada umumnya. Kapasitas 1.44 Mbytes ini didapat karena floppy disket tersebut di format dengan aturan 18 sektor 80 track. Apabila dilakukan teknik pemformatan floppy disket secara khusus, akan dapat meningkatkan kapasitas penyimpanan. Kapasitas maksimum yang diijinkan untuk melakukan pemformatan khusus pada floppy disket 3.5" DD/HD adalah 23 sektor 83 track (1.9 Mbytes). Namun mengingat keterbatasan BIOS pada komputer, maka kapasitas tersebut tidak dapat dipergunakan semuanya. Kapasitas maksimum yang dapat ditanggani oleh BIOS komputer tanpa kemampuan *booting* adalah 23 sektor 81 track (1.88 Mbytes). Apabila mempertahankan kemampuan *booting* maka kapasitas maksimum yang dapat ditanggani oleh BIOS komputer menjadi turun hingga tinggal 23 sektor 80 track (1.7 MB).

Pembuatan *device file* dilakukan dengan cara memasukan parameter *minor-_device_num* kedalam parameter program *mknod*

mknod /dev/fd0u1743 b 2 76

/dev/fd0u1743 - nama *device file*

b - *block device*

- *major device number*

- *minor device number*

Sehingga untuk mengakses *device file* floppy disk driver dengan 23 sektor 80 track digunakan nama */dev/fd0u1743*.

BAB IV

EVALUASI DAN ANALISA SISTEM

Tujuan utama dibuat emulasi router adalah mendapatkan router dengan harga yang lebih murah dengan fungsionalitas sama dengan router sebenarnya. Oleh karena itu tolak ukur hasil proses perencanaan dan pembuatan emulasi router ini didasarkan pada :

1. sisi operasional, yang meliputi:
   - media boot disk
   - boot dan shutdown
   - backup sistem
2. sisi fungsionalistas, yang meliputi:
   - protokol routing
   - network address translator (NAT)
   - firewall
   - simple network management protocol (snmp)
   - radius client

4.1. Media Boot Disk

Emulasi router yang dibuat dapat memanfaatkan floppy disket maupun hard disk sebagai media boot disk sekaligus sebagai media penyimpanan permanen. Floppy disket yang digunakan adalah floppy disket 3.5" 1.44 MB, yang diformat dengan cara khusus dengan memperhitungkan banyaknya sektor dan track. Hasil uji coba pemformatan disket 1.44 MB dengan berbagai variasi sektor dan track yang diperbolehkan dapat dilihat pada tabel 4.1

Dari tabel 4.1 didapat bahwa dengan melakukan pemformatan khusus didapatkan 3 (tiga) buah kapasitas yang dapat dipergunakan untuk melakukan penyimpanan file-file modul emulasi router. Dari ketiga jenis kapasitas tersebut diambil kapasitas terbesar yaitu 1.74 MB.

Tabel 4.1. Jenis pemformatan pada disket 3.5" 1.44 MB

Jumlah sektor	Jumlah track	Kapasitas	Boot
18	80	1.44 MB	Ya
21	80	1.68 MB	Ya
21	83	1.74 MB	Ya
23	80	1.80 MB	Tidak

Disket boot yang dibutuhkan agar dapat mencapai kapasitas 1.74 MB adalah berupa floppy disket 3.5" 1.44 MB tanpa *bad* sektor. Adapun cara mendapatkan kapasitas 1.74 MB, adalah sebagai berikut:

- Diperlukan disket 3.5" 1.44 MB tanpa *bad* sektor, hal ini dapat diketahui dengan jalan melakukan pemformatan MS-DOS.
- Dengan menggunakan sistem operasi Linux, dan program bantu *superformat* (program bantu yang terdapat dalam distribusi Linux Slackware) dilakukan pemformatan khusus 23 sektor 80 track.

superformat –d /dev/fd0 –s 23 –t 80

- Langkah berikutnya adalah memberikan *boot loader* dan sistem boot kepada disket. Program yang digunakan untuk melakukan ini adalah *syslinux*. Program *syslinux* ini dapat dijumpai pada http://metalab.unc.edu/pub/Linux/system/boot/loaders/

syslinux –s /dev/fd0

- Setelah disket memiliki sistem boot dan *boot loader* dilakukan penyalinan file-file modul utama ke dalam disket. File-file module utama tersebut adalah : root, etc, log, linux, dan syslinux.cfg.

```
# mount /dev/fd0u1743 /mnt
# cp root.lrp /mnt/root.lrp
# cp etc.lrp /mnt/etc.lrp
# cp log.lrp /mnt/log.lrp
# cp linux /mnt/linux
# cp syslinux.cfg /mnt/syslinux.cfg
```

- Setelah file-file modul utama tersalin ke dalam disket, apabila diperlukan dapat dilakukan penyalinan file-file modul lainnya, seperti : snmp, gated dan pslave.
- Setelah seluruh file modul yang diperlukan tersalin ke dalam disket, lakukan pembersihan *buffer disk*, agar supaya seluruh data yang disalin ke dalam disket benar benar telah tertulis ke dalam disket.

```
# sync
```

- Proses diakhiri dengan melakukan *dismounting* disket.

```
# umount /dev/fd0u1743
```

4.2. Boot dan Shutdown

Proses boot emulasi router dimulai dengan menjalankan *boot loader*. Kemudian sistem akan membaca file konfigurasi boot, yaitu *syslinux.cfg*. Adapun file konfigurasi boot secara lebih detailnya dapat dijelaskan sebagai berikut :

- default linux

  Baris ini memberitahukan kepada *boot loader* bahwa nama file kernel yang harus dijalankan adalah *linux*.

- append=load_ramdisk=1 initrd=root.lrp initrd_archive=minix ramdisk_size=8192 root=/dev/ram0 boot=/dev/fd0,msdos LRP=etc,log,modules

  Baris ini memberitahukan kepada *boot loader* bahwa :

- ram disk yang harus dibuat sebanyak 1 buah.
- *boot loader* akan melakukan dekompresi file *root.lrp*.
- mempergunakan jenis sistem file minix.
- besar ram disk yang dibuat adalah 8192 bytes / 8 MB.
- membentuk root file sistem pada *device* /dev/ram0 (ram disk pertama).
- boot disk adalah floppy disk drive pertama (/dev/fd0), jenis format sistem disk adalah MS-DOS
- File modul yang dimuat kedalam emulasi router adalah etc, log dan modules.

Proses *shutdown* pada emulasi router bisa dikatakan tidak diperlukan lagi, dikarenakan semua proses emulasi router dilakukan pada ram disk. Proses *shutdown* yang mempunyai tujuan untuk mengakhiri proses yang ada pada sistem operasi dan memberikan tanda kepada sistem file tidak diperlukan.

Karena router tidak melakukan perubahan sistem file selama proses kecuali perubahan file konfigurasi dan file log. Dua macam perubahan ini dapat disimpan ke dalam media penyimpanan tetap (disket atau hard disk sesuai dengan boot disk yang dipakai) pada saat emulasi router masih berjalan, dengan mempergunakan program bantu backup (*lrcfg*).

4.3. Sistem Backup

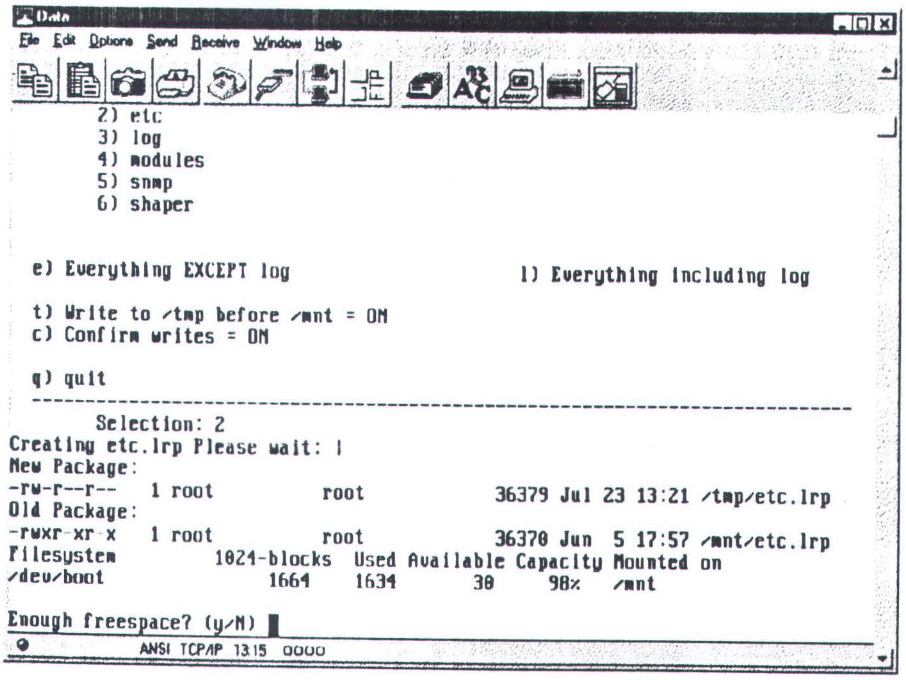
Emulasi router membutuhkan proses *backup* terhadap perubahan file konfigurasi dan file log (bila diperlukan). Proses *backup* ini digunakan untuk menyimpan perubahan tersebut secara tetap di dalam media penyimpanan untuk digunakan proses boot selanjutnya. Proses *backup* ini disediakan oleh program bantu yang bernama *lrcfg*. *Backup* hanya dapat dilakukan ke dalam media penyimpanan yang bersesuain dengan media penyimpanan yang digunakan untuk boot.

Gambar 4.1 menunjukkan contoh tampilan program bantu *backup*.

4.4. Protokol Routing

Emulasi router menggunakan program *gated* dari Merit Konsorsium yang pada umumnya diikut sertakan pada beberapa sistem operasi yang dapat difungsikan sebagai router. Program ini menyediakan protokol routing RIP, RIPv2, OSPF dan BGP. Karena program ini merupakan program jadi yang hanya dilakukan pemaketan ulang dalam bentuk file terkompresi untuk penyimpanannya, maka secara fungsi program ini tidak berubah.

Uji coba protokol routing yang ada tidak dapat dilakukan dalam lingkungan yang nyata, mengingat keterbatasan sumber daya. Uji coba ini dilakukan bukan untuk menganalisa protokol routing namun sekedar untuk mengetahui apakah protokol routing dapat berjalan dalam emulasi router.

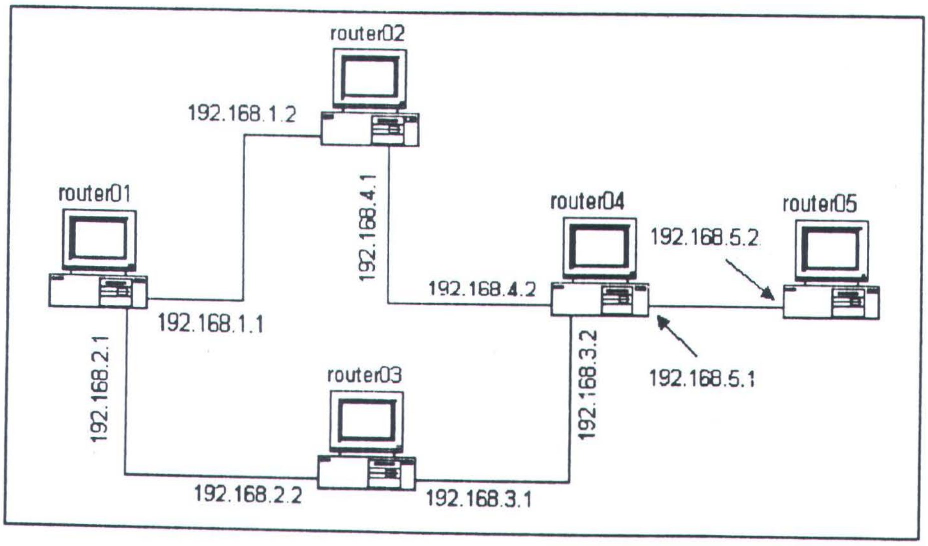


Gambar 4.1. Tampilan program bantu *backup*.

Untuk itu dilakukan uji coba protokol routing dengan cara mensimulasikan pada suatu jaringan komputer dengan mempergunakan topologi jaringan seperti terlihat pada gambar 4.2.

Protokol routing BGP tidak dapat dilakukan uji coba karena protokol ini dikhususkan untuk jaringan komputer besar dengan *autonomous system*. Sehingga protokol jaringan yang dapat dilakukan uji coba adalah RIP, RIPv2 dan OSPF. Analisa protokol BGP lebih lanjut terdapat pada sub ban 4.4.1.

Dari uji coba didapat tabel routing yang sama antara ketiga protokol routing diatas. Kesamaan tabel routing ini tidak akan dibahas mengingat bukan merupakan bagian dari tugas akhir. Tabel routing dapat dilihat pada tabel 4.2, 4.3, 4.4, 4.5 dan 4.6.



Gambar 4.2. Topologi jaringan simulasi routing protokol.

Tabel 4.2. Tabel routing yang terbentuk pada router01.

Destination	Gateway	GenMask	Flags	MSS	Window	Irtt	Iface
192.168.1.0	0.0.0.0	255.255.255.0	U	1500	0	0	eth0
192.168.2.0	0.0.0.0	255.255.255.0	U	1500	0	0	eth1
192.168.4.0	192.168.1.2	255.255.255.0	UG	1500	0	0	eth0
192.168.3.0	192.168.2.2	255.255.255.0	UG	1500	0	0	eth1
192.168.5.0	192.168.1.2	255.255.255.0	UG	1500	0	0	eth0

Tabel 4.3. Tabel routing yang terbentuk pada router02.

Destination	Gateway	GenMask	Flags	MSS	Window	Irtt	Iface
192.168.1.0	0.0.0.0	255.255.255.0	U	1500	0	0	eth0
192.168.4.0	0.0.0.0	255.255.255.0	U	1500	0	0	eth1
192.168.2.0	192.168.1.1	255.255.255.0	UG	1500	0	0	eth0
192.168.3.0	192.168.4.2	255.255.255.0	UG	1500	0	0	eth1
192.168.5.0	192.168.4.2	255.255.255.0	UG	1500	0	0	eth1

Tabel 4.4. Tabel routing yang terbentuk pada router03.

Destination	Gateway	GenMask	Flags	MSS	Window	Irtt	Iface
192.168.2.0	0.0.0.0	255.255.255.0	U	1500	0	0	eth0
192.168.3.0	0.0.0.0	255.255.255.0	U	1500	0	0	eth1
192.168.1.0	192.168.2.1	255.255.255.0	UG	1500	0	0	eth0
192.168.4.0	192.168.3.2	255.255.255.0	UG	1500	0	0	eth1
192.168.5.0	192.168.3.2	255.255.255.0	UG	1500	0	0	eth1

Tabel 4.5. Tabel routing yang terbentuk pada router04.

Destination	Gateway	GenMask	Flags	MSS	Window	Irtt	Iface
192.168.3.0	0.0.0.0	255.255.255.0	U	1500	0	0	eth0
192.168.4.0	0.0.0.0	255.255.255.0	U	1500	0	0	eth1
192.168.5.0	0.0.0.0	255.255.255.0	U	1500	0	0	eth2
192.168.2.0	192.168.3.2	255.255.255.0	UG	1500	0	0	eth0
192.168.1.0	192.168.4.2	255.255.255.0	UG	1500	0	0	eth1

Tabel 4.6. Tabel routing yang terbentuk pada router05.

Destination	Gateway	GenMask	Flags	MSS	Window	Irtt	Iface
192.168.5.0	0.0.0.0	255.255.255.0	U	1500	0	0	eth0
192.168.4.0	192.168.5.1	255.255.255.0	UG	1500	0	0	eth0
192.168.3.0	192.168.5.1	255.255.255.0	UG	1500	0	0	eth0
192.168.1.0	192.168.5.1	255.255.255.0	UG	1500	0	0	eth0
192.168.2.0	192.168.5.1	255.255.255.0	UG	1500	0	0	eth0

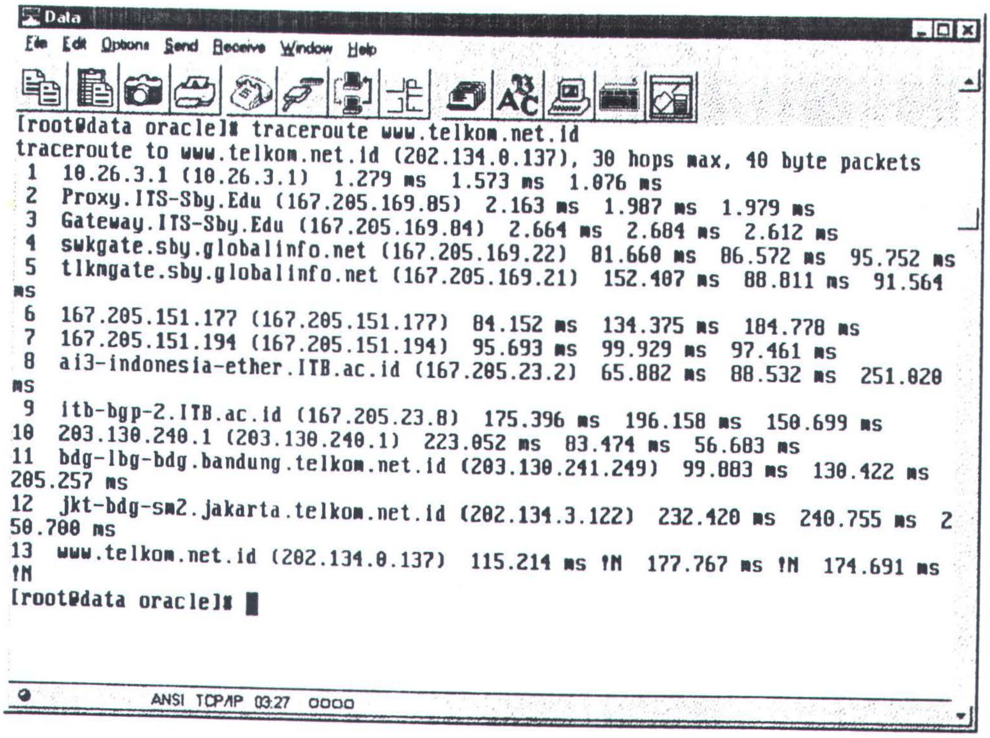
4.4.1. Protokol Routing BGP

Seperti telah dijelaskan pada sub bab sebelumnya bahwa protokol routing BGP ini tidak dapat diujicobakan pada lingkungan jaringan Teknik Informatika ITS. Untuk itulah ditempuh jalan menggunakan referensi pada jaringan komputer yang telah mempergunakan perangkat lunak GateD untuk menangani protokol routing BGP. Jaringan komputer yang dipilih sebagai referensi tersebut adalah jaringan AI-3 Institut Teknologi Bandung (ITB).

Dalam lingkungan jaringan AI3 ITB terdapat beberapa PC router yang telah difungsikan untuk menangani routing baik itu untuk jaringan lokal maupun jaringan Internet. Salah satu host yang berfungsi sebagai PC router dan menjalankan protokol BGP adalah *itb-bgp-2.itb.ac.id* (IP = 167.205.23.8). Host ini menjalankan protokol BGP dapat dilihat dari hasil *traceroute* (gambar 4.3) dan *scanning port* yang aktif pada host tersebut (gambar 4.4).

Gambar 4.3 memperlihatkan hasil *traceroute* dari jaringan Teknik Informatika ke alamat Internet www.telkom.net.id (IP = 202.134.0.137). Dalam gambar tersebut tersebut terlihat bahwa untuk mencapai host www.telkom.net.id melewati sebuah host *itb-bgp-2.itb.ac.id*. Hal ini menunjukkan bahwa host *itb-*

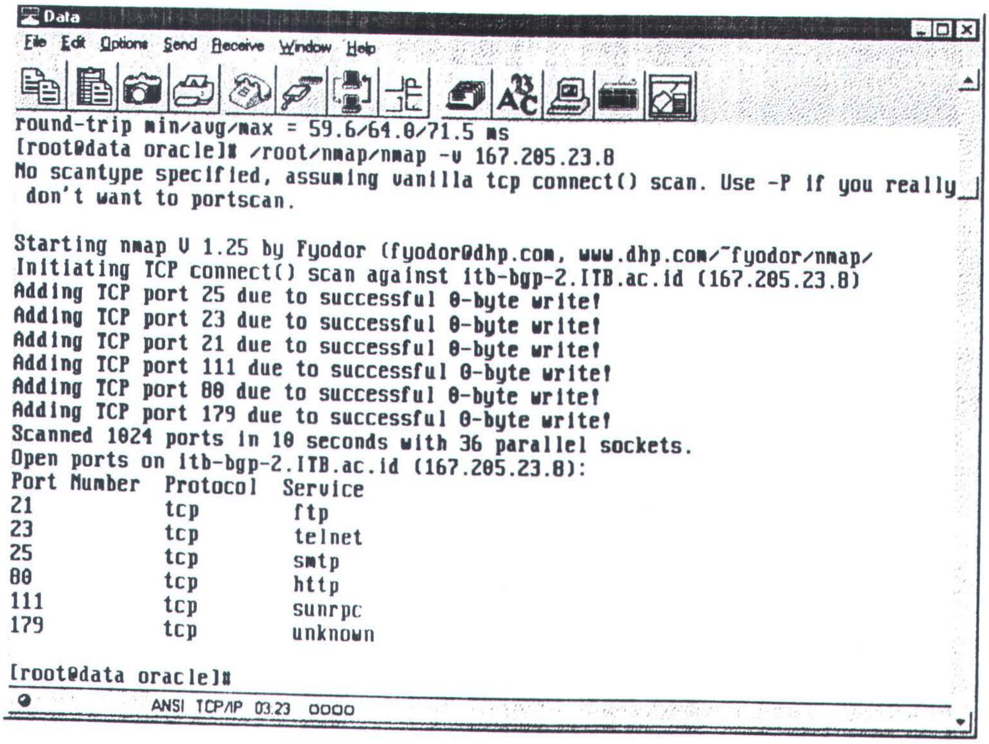
bgp-2.itb.ac.id menjalankan protokol routing BGP. Karena apabila dalam lingkungan jaringan AI3 ITB tidak menjalankan protokol routing BGP ini, lintasan yang semestinya untuk mencapai www.telkom.net.id adalah menuju ke AI3 Jepang. Alasan lain mengapa bukan protokol routing selain BGP yang dijalankan untuk membelokkan *traffic* tersebut adalah jaringan Telkom-Net merupakan jaringan dengan *AS number* yang berbeda dengan AI3. Pertukaran informasi antara *AS number* yang berbeda tidak mungkin dilakukan tanpa mempergunakan protokol routing BGP.



```
[root@data oracle]# traceroute www.telkom.net.id
traceroute to www.telkom.net.id (202.134.0.137), 30 hops max, 40 byte packets
 1  10.26.3.1 (10.26.3.1)  1.279 ms  1.573 ms  1.076 ms
 2  Proxy.ITS-Sby.Edu (167.205.169.85)  2.163 ms  1.987 ms  1.979 ms
 3  Gateway.ITS-Sby.Edu (167.205.169.84)  2.664 ms  2.684 ms  2.612 ms
 4  swkgate.sby.globalinfo.net (167.205.169.22)  81.660 ms  86.572 ms  95.752 ms
 5  tlkmgate.sby.globalinfo.net (167.205.169.21)  152.407 ms  88.811 ms  91.564
ms
 6  167.205.151.177 (167.205.151.177)  84.152 ms  134.375 ms  184.778 ms
 7  167.205.151.194 (167.205.151.194)  95.693 ms  99.929 ms  97.461 ms
 8  ai3-indonesia-ether.ITB.ac.id (167.205.23.2)  65.882 ms  88.532 ms  251.020
ms
 9  itb-bgp-2.ITB.ac.id (167.205.23.8)  175.396 ms  196.158 ms  150.699 ms
10  203.130.240.1 (203.130.240.1)  223.052 ms  83.474 ms  56.683 ms
11  bdg-lbg-bdg.bandung.telkom.net.id (203.130.241.249)  99.883 ms  130.422 ms
205.257 ms
12  jkt-bdg-sm2.jakarta.telkom.net.id (202.134.3.122)  232.420 ms  240.755 ms  2
50.700 ms
13  www.telkom.net.id (202.134.0.137)  115.214 ms !N  177.767 ms !N  174.691 ms
!N
[root@data oracle]#
```


Gambar 4.3. Hasil *traceroute* dari TC-Network ke Telkom-Net

Gambar 4.4 menunjukkan juga bahwa host *itb-bgp-2.itb.ac.id* menjalankan protokol routing BGP. Hal ini terlihat pada hasil *scanning port* yang aktif pada host tersebut. Port 179 adalah TCP port yang digunakan oleh *service* protokol routing BGP.



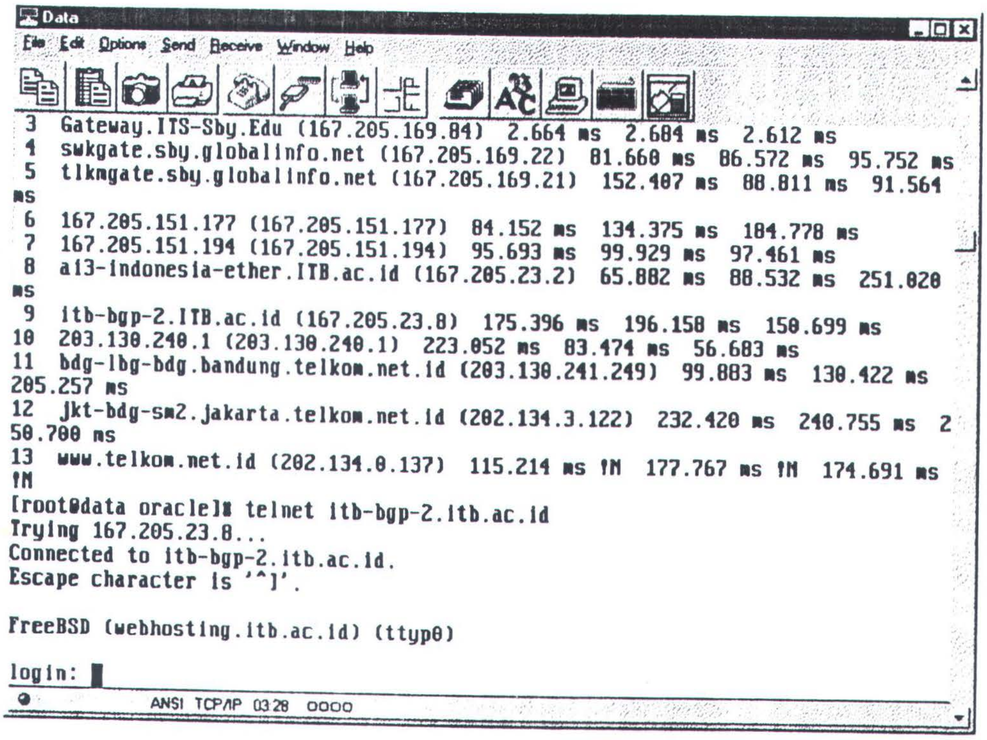

```
round-trip min/avg/max = 59.6/64.0/71.5 ms
[root@data oracle]# /root/nmap/nmap -v 167.205.23.8
No scantype specified, assuming vanilla tcp connect() scan. Use -P if you really
 don't want to portscan.

Starting nmap V 1.25 by Fyodor (fyodor@dhp.com, www.dhp.com/~fyodor/nmap/
Initiating TCP connect() scan against itb-bgp-2.ITB.ac.id (167.205.23.8)
Adding TCP port 25 due to successful 0-byte write!
Adding TCP port 23 due to successful 0-byte write!
Adding TCP port 21 due to successful 0-byte write!
Adding TCP port 111 due to successful 0-byte write!
Adding TCP port 80 due to successful 0-byte write!
Adding TCP port 179 due to successful 0-byte write!
Scanned 1024 ports in 10 seconds with 36 parallel sockets.
Open ports on itb-bgp-2.ITB.ac.id (167.205.23.8):
Port Number  Protocol  Service
21           tcp       ftp
23           tcp       telnet
25           tcp       smtp
80           tcp       http
111          tcp       sunrpc
179          tcp       unknown

[root@data oracle]#
```

Gambar 4.4. Hasil *scanning port* aktif pada host *itb-bgp-2.itb.ac.id.*

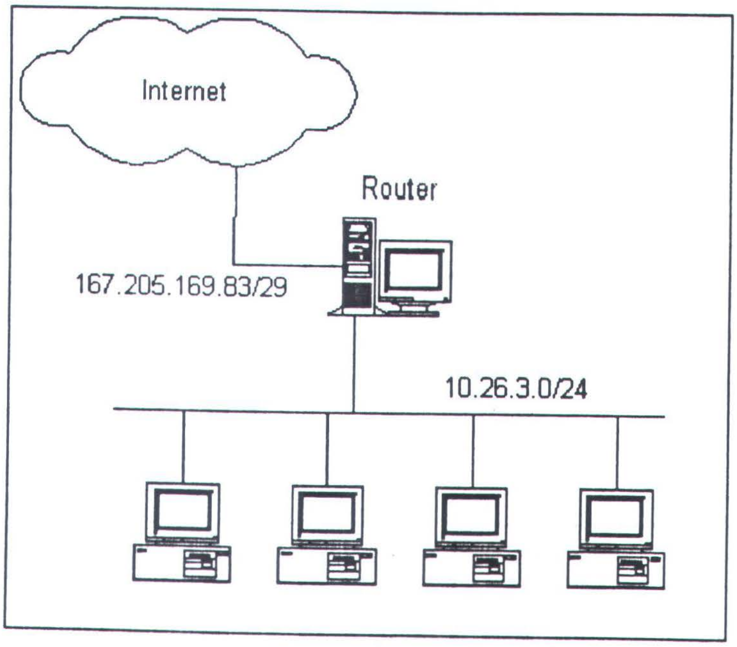
Sedangkan gambar 4.5 menunjukkan bahwa host *itb-bgp-2.itb.ac.id* merupakan sebuah PC Router dengan sistem operasi FreeBSD.

4.5. *Network Address Translator* (NAT)

Emulasi router mampu menyediakan NAT dengan mempergunakan IP Masquerade. NAT pada emulasi router ini bekerja pada lapisan *transport*. NAT yang disediakan berupa ICMP, UDP dan TCP. Dalam uji coba emulasi router dipergunakan sebagai router jaringan Laboratorium KOMISI untuk menghubungan LAN KOMISI yang mempunyai IP Intranet (10.26.3.0) dengan komunitas Internet. Untuk lebih jelasnya topologi jaringan Laboratorium KOMISI dapat dilihat pada gambar 4.6.



Gambar 4.5. Hasil tampilan *telnet* ke host *itb-bgp-2.itb.ac.id.*

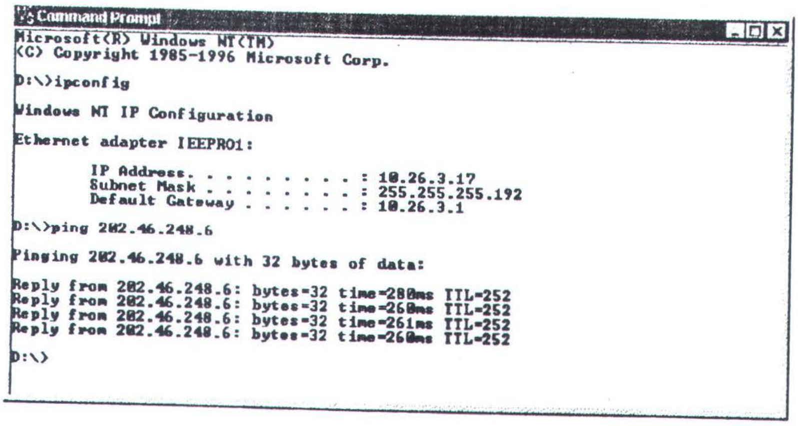


Gambar 4.6. Topologi jaringan Laboratorium KOMISI

Uji coba NAT dilakukan dengan melakukan NAT Intranet IP (10.26.3.0) ke komunitas Internet untuk beberapa keperluan, yaitu :

- ICMP NAT

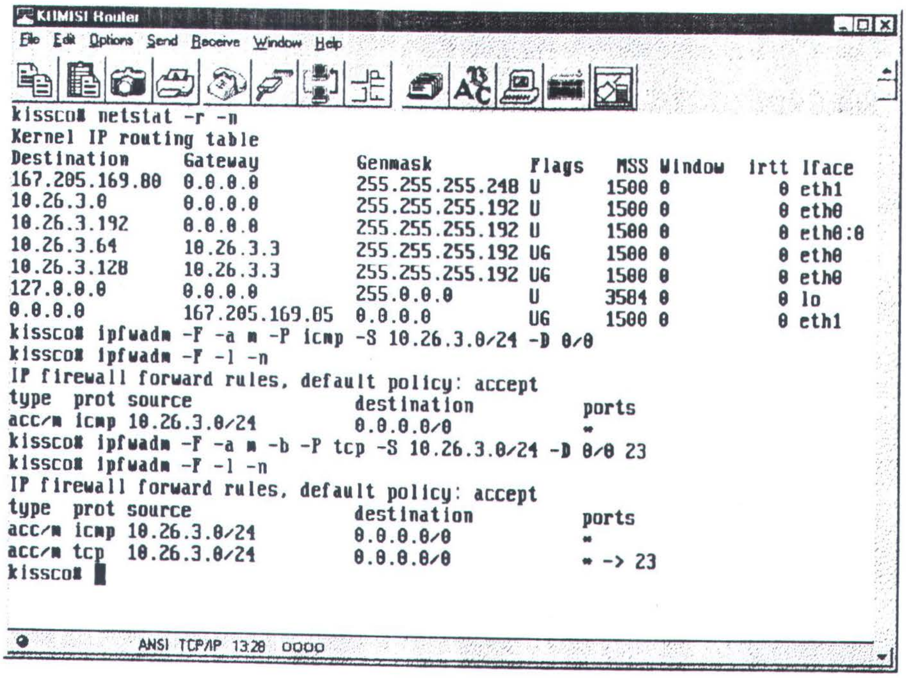
Pada bagian ini dilakukan proses ICMP NAT untuk jaringan lokal Laboratorium KOMISI ke komunitas diluar jaringan Laboratorium KOMISI. Proses NAT dapat dilihat pada gambar 4.7. Gambar 4.8 menunjukan proses ping sebelum dilakukan ICMP NAT, gambar ini menunjukkan bahwa jaringan lokal KOMISI belum dapat melakukan *ping* ke jaringan di luar KOMISI (Internet). Sedangkan hasil dari proses ICMP NAT dapat dilihat pada gambar 4.9 yang menunjukkan jaringan lokal KOMISI telah dapat melakukan ping ke luar jaringan KOMISI.



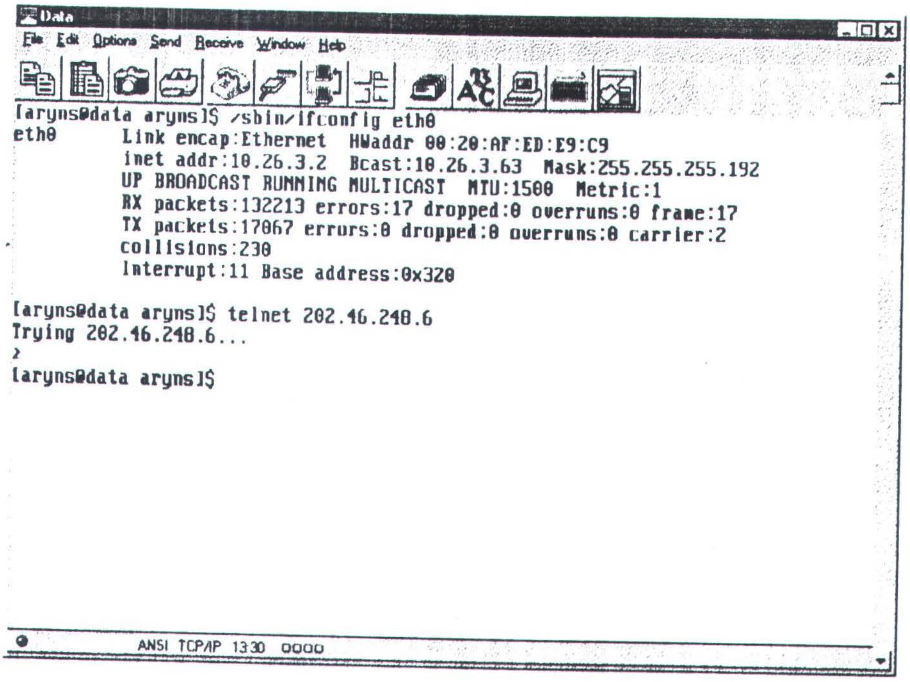
Gambar 4.9. Hasil *ping* setelah proses ICMP NAT.

- TCP port NAT

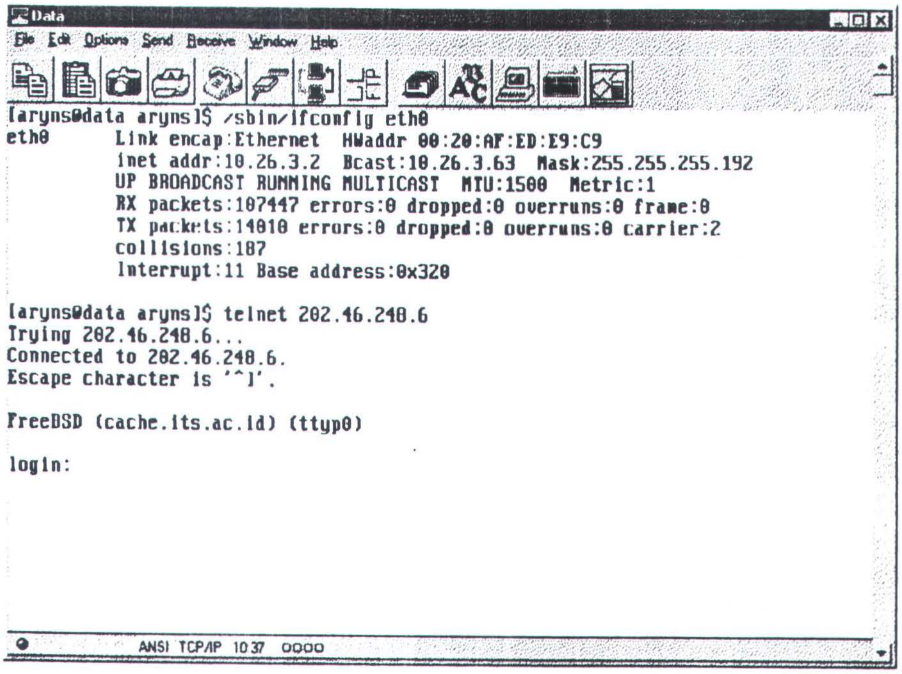
Pada bagian ini dilakukan proses TCP port NAT yang bermanfaat untuk melakukan koneksi layanan (*service*) yang mempergunakan TCP port ke Internet dengan menggunakan IP Intranet. Gambar 4.10 menunjukan proses TCP port NAT untuk keperluan layanan (*service*) *telnet*. Gambar 4.11 menunjukkan hasil proses *telnet* ke Internet sebelum proses TCP port NAT. Sedangkan gambar 4.12 menunjukkan hasil proses *telnet* setelah proses TCP port NAT.



Gambar 4.10. Proses TCP port NAT



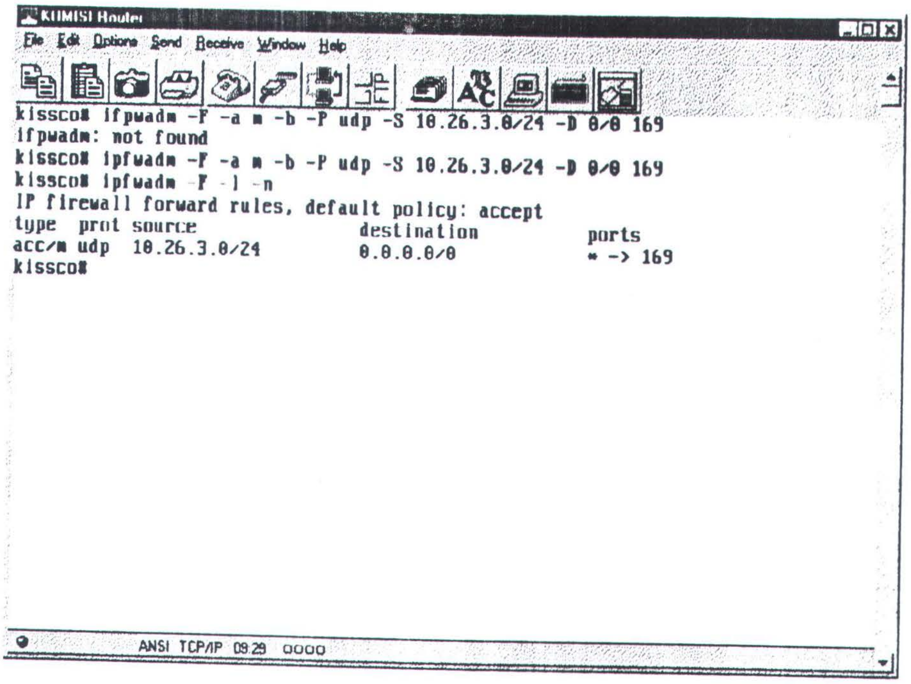
Gambar 4.11. Hasil proses *telnet* ke komunitas Internet sebelum proses TCP port NAT.



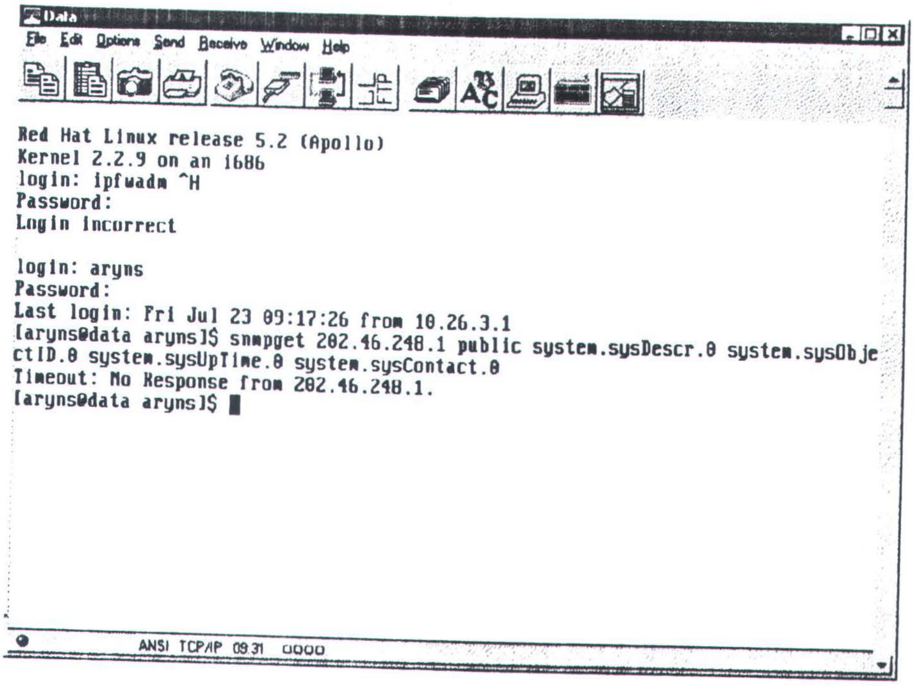
Gambar 4.12. Hasil *telnet* setelah proses TCP port NAT.

- UDP port NAT

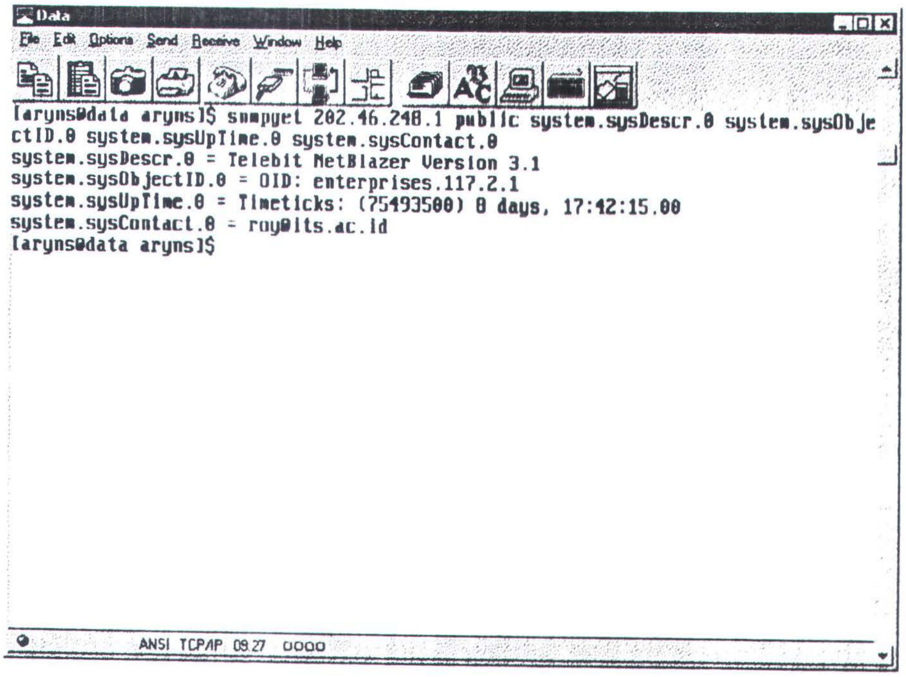
Pada bagian ini dilakukan proses UDP port NAT yang bermanfaat untuk melakukan koneksi layanan (*service*) yang mempergunakan UDP port ke Internet dengan menggunakan IP Intranet. Gambar 4.13 menunjukan proses UDP port NAT untuk keperluan layanan (*service*) snmp. Gambar 4.14 menunjukkan hasil proses snmp ke Internet sebelum proses TCP port NAT. Sedangkan gambar 4.15 menunjukkan hasil proses snmp setelah proses TCP port NAT.



Gambar 4.13. Proses UDP port NAT.



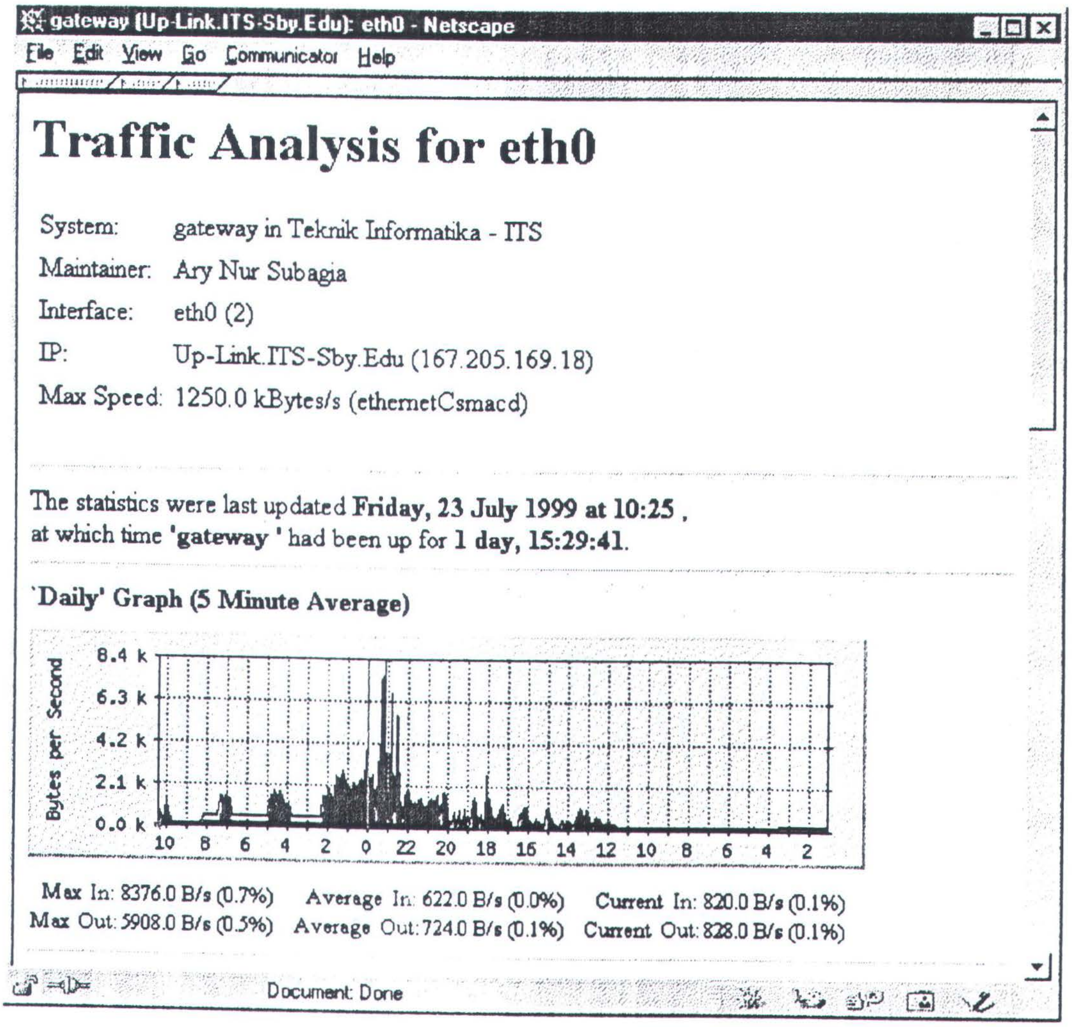
Gambar 4.14. Hasil proses snmpget sebelum proses UDP port NAT.



Gambar 4.15. Hasil proses snmpget setelah proses UDP port NAT.

4.6. *Firewall*

Emulasi Router menyediakan *firewall* dasar yang beroperasi dalam tingkatan paket IP. *Firewall* ini melakukan proses penyaringan paket IP untuk selanjutnya memproses paket tersebut untuk diteruskan atau tidak. Sebagai contoh pada emulasi router yang dipasang pada router Laboratorium KOMISI diterapkan aturan *firewall* untuk mencegah akses *sharing resource* NETBIOS di lingkungan jaringan KOMISI dari luar jaringan KOMISI. Gambar 4.16 menunjukkan implementasi aturan *firewall* tersebut.

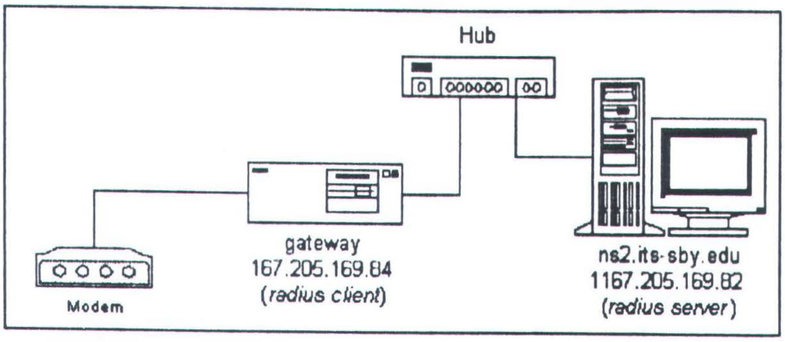


Gambar 4.17. Hasil pembacaan program MRTG terhadap Router Utama

TC-Network

4.8. *Radius Client*

Emulasi router menyediakan *radius client* yang berfungsi untuk menerima *dial-in* yang kemudian melakukan *authentication* user dari *radius server*. *Radius server* berupa host/server terpisah yang menyimpan database user. Dalam implementasi *radius client* ditempatkan pada host *gateway* (IP : 167.205.169.84) sedangkan database user diambil dari host *ns2.its-sby.edu* yang menyimpan database user Internet TC-Network. Gambar 4.14 menunjukkan implementasi *radius client.*



Gambar 4.18. Topologi jaringan implementasi *radius client.*

BAB V

KESIMPULAN DAN SARAN

5.1. Kesimpulan

Router pada umumnya berbentuk perangkat keras dapat diemulasikan mengunakan perangkat lunak yang berjalan di atas komputer pribadi (*personal computer*). Dalam Tugas Akhir ini perangkat lunak yang dipergunakan adalah sistem operasi Linux dengan kernel versi 2.0.36. Emulasi router berjalan dengan baik setelah mengalami uji coba dilingkungan jaringan komputer Jurusan Teknik Informatika ITS.

Emulasi router mempunyai kemampuan sebagai berikut :

- Emulasi router dengan dasar sistem operasi Linux ini dapat dimatikan tanpa melalui proses *shutdown* sebagaimana sistem operasi Linux pada umumnya. Hal ini dikarenakan telah dilakukan suatu perubahan mekanisme boot dan cara kerja sistem operasi.
- Pemaketan ulang program-program yang telah ada pada sistem operasi Linux dalam bentuk modul-modul dilakukan berdasarkan fungsinya. Sehingga hanya modul program yang dibutuhkan dalam operasional emulasi router saja yang diikutsertakan.
- Emulasi router mampu menjalankan protokol routing RIP, RIPv2, OSPF dan BGP.

- *Traffic* yang terjadi pada emulasi router dapat diketahui dengan adanya modul SNMP.
- Emulasi router dapat melakukan backup pada modul yang dikehendaki saja.
- Emulasi router dapat beroperasi hanya dengan sebuah floppy disket 3.5" 1.44 MB.

5.2. Saran

Dalam rangka pengembangan emulasi router ini lebih jauh perlu diperhatikan beberapa hal diantaranya :

- Perlu studi lebih lanjut mengenai protokol routing, khususnya mengenai protokol routing *multicast*. Sehingga dapat menambah kemampuan emulasi router untuk menangani routing *multicast*.
- Dalam hal media penyimpanan permanen dan media *boot* perlu dikembangkan ke dalam bentuk *Flash ROM*. Sehingga emulasi router menjadi lebih ringkas dan tahan lama.
- Perlu studi lebih lanjut mengenai protokol IP versi 6, mengingat protokol IP versi 6 merupakan standard baru yang akan diberlakukan beberapa tahun mendatang untuk komunikasi jaringan komputer berdasarkan protokol IP.
- Peru dipikirkan mengenai perangkat lunak yang dapat membaca keluaran yang dihasilkan oleh *SNMP agent* pada emulasi router, sehingga pemanfaatan SNMP pada emulasi router dapat lebih dioptimalkan.

DAFTAR PUSTAKA

- Halabi, Bassam, "*Internet Routing Architectures*", Cisco Press, 1997
- Purbo, Onno, "*TCP/IP Standar, Desain, dan Implementasi*", Elex Media Komputindo, 1998
- ____________, 4"*Linux Kernel Hacking Guide*", Linux Community, 1997
- ____________, "*Merit GateD Consortium*", http://www.gated.org, 1998
- ____________,"*Linux Kernel 2.0.36 Documentation*", GPL, 1997
- ____________,"*Linux HOW-TO*", GPL, 1997
- ____________,"*Linux Router Project*", http://www.linuxrouter.org, 1997
- ____________,"*Freshmeat*", http://www.freshmeat.net, 1998

Lampiran 1 : Anggota Modul

A. Modul etc

etc
var/lib/lrpkg/etc.help
var/lib/lrpkg/etc.list
var/lib/lrpkg/etc.version

B. Anggota Modul log

var/adm
var/log
var/lib/lrpkg/log.help
var/lib/lrpkg/log.list
var/lib/lrpkg/log.version

C. Anggota Modul snmp

etc/init.d/snmpd
etc/rc0.d/K20snmpd
etc/rc1.d/K20snmpd
etc/rc2.d/S20snmpd
etc/rc3.d/S20snmpd
etc/rc4.d/S20snmpd
etc/rc5.d/S20snmpd
etc/rc6.d/K20snmpd
etc/snmpd.agentinfo
etc/snmpd.conf
usr/bin/authkey
usr/bin/nstat
usr/bin/snmpget
usr/bin/snmpgetnext
usr/bin/snmpnetstat
usr/bin/snmpset
usr/bin/snmptest
usr/bin/snmptrap
usr/bin/snmpwalk
usr/lib/libsnmp.so.3.5
usr/lib/mib.txt
usr/sbin/snmpd
usr/sbin/snmptrapd
var/lib/lrpkg/snmp.conf
var/lib/lrpkg/snmp.help
var/lib/lrpkg/snmp.list
var/lib/lrpkg/snmp.version

D. Anggota Modul gated

etc/cron.daily/gated

etc/init.d/gated
etc/rc0.d/K15gated
etc/rc1.d/K15gated
etc/rc2.d/S15gated
etc/rc3.d/S15gated
etc/rc4.d/S15gated
etc/rc5.d/S15gated
etc/rc6.d/K15gated
etc/gated.conf
usr/bin/gdc
usr/bin/ospf_monitor
usr/bin/ripquery
usr/sbin/gated
var/lib/lrpkg/gated.conf
var/lib/lrpkg/gated.help
var/lib/lrpkg/gated.list
var/lib/lrpkg/gated.version

E. Anggota Modul pslave

etc/portslave
etc/ppp-radius
etc/rc.boot/radinit
usr/bin/rlogin-radius
usr/lib/libpsr.so
usr/sbin/ctlportslave
usr/sbin/in.fingerd
usr/sbin/portslave
usr/sbin/pppd-radius
usr/sbin/radinit
var/lib/lrpkg/pslave.conf
var/lib/lrpkg/pslave.help
var/lib/lrpkg/pslave.list
var/lib/lrpkg/pslave.version

F. Anggota Modul modules

etc/modules
etc/init.d/modutils
etc/rcS.d/S20modutils
lib/modules
sbin/insmod
sbin/rmmod
var/lib/lrpkg/modules.conf
var/lib/lrpkg/modules.help
var/lib/lrpkg/modules.list
var/lib/lrpkg/modules.version

Lampiran 2 : *Firewall Command*

- Forwarding Firewall
  ipfwadm –F command [options]
- Input Firewall
  ipfwadm –I command [options]
- Output Firewall
  ipfwadm –O command [options]

Commands :
- Menyisipkan aturan (*rule*)
  -i [policy]
- Menambahkan aturan (*rule*)
  -a [policy]
- Menghapus aturan (*rule*)
  -d [policy]
- Melihat daftar aturan (*rule*)
  -l
- Menetapkan *default policy*
  -p [policy]

Options :
- Pemilihan protokol
  -P *protocol* ⟶ *protocol* adalah salah satu dari tcp, udp, icmp atau all
- Spesifikasi *source address*
  -S address/[mask] [port]
- Spesifikasi *destination address*
  -D address/[mask] [port]
- Spesifikasi *network interface address*
  -V address
- Spesifikasi *network interface name*
  -W name

- NAT/Masquerade Command

Bentuk umum :
  ipfwadm –F –a m –S address/[mask] –D address/[mask]