Implementasi Firewall Generator dengan Linux Iptables di Jaringan SCS Universitas Brawijaya

SKRIPSI

Diajukan untuk memenuhi sebagian prasyarat
memperoleh gelar Sarjana Teknik
KONSENTRASI TEKNIK INFORMATIKA DAN KOMPUTER

Disusun oleh:

ANGGRAINI PUSPITASARI

NIM. 0310630018

KEMENTERIAN PENDIDIKAN NASIONAL
UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK
JURUSAN TEKNIK ELEKTRO
2010

KATA PENGANTAR

Alhamdulillah, segala puji bagi Allah SWT semata, sholawat serta salam semoga terlimpahkan untuk Nabi Muhammad SAW, beserta para sahabat yang setia mengikuti jejak beliau. Berkat hidayah dan inayah Allah SWT, skripsi berjudul "Implementasi Firewall Generator dengan Linux Iptables pada Jaringan SCS Brawijaya" dapat diselesaikan.

Ucapan terima kasih yang mendalam penulis sampaikan kepada pihak-pihak yang telah membantu secara langsung maupun tidak langsung, dan telah memberikan dukungan sehingga penyusunan skripsi ini dapat selesai pada waktunya. Atas dukungan dan bimbingannya, pada kesempatan yang berbahagia ini, penulis mengucapkan banyak terima kasih kepada :

1. Bapak Rudy Yuwono, ST, Msc., selaku Ketua Jurusan Teknik Elektro.
2. Bapak M. Azis Muslim, ST., MT.,Ph.D, selaku Sekretaris Jurusan Teknik Elektro.
3. Bapak Ali Mustofa ST., MT. selaku Dosen Wali yang telah membimbing penulis.
4. Seluruh Dosen dan Staf Administrasi Jurusan Teknik Elektro.
5. Keluarga, sahabat dan teman-teman atas bantuan, doa, dan semangat yang diberikan.
6. Semua teman Silvergen dan Teknik Elektro.
7. Semua pihak yang tidak dapat penulis sebutkan satu persatu, yang telah membantu dalam penyelesaian skripsi ini.

Penulis menyadari bahwa skripsi ini masih jauh dari sempurna. Oleh karenanya penulis sangat mengharapkan masukan berupa kritik dan saran yang membangun. Semoga skripsi ini memberikan manfaat bagi semua pihak sesuai dengan fungsinya.

Malang, 4 Agustus 2010

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Disusun Oleh : ANGGRAINI PUSPITASARI

Jurusan Teknik Elektro,Fakultas Teknik Universitas Brawijaya Malang, Juli 2010.

Dosen Pembimbing : Ir. Heru Nurwasito, MT. dan Suprapto, ST., MT.

Sistem paket filtering adalah sistem yang melakukan paket routing antara jaringan internal dengan jaringan eksternal secara selektif. Sistem ini melewatkan atau memblok paket data yang lewat sesuai dengan aturan yang telah ditentukan. Salah satu jenis paket filtering di sistem operasi Linux adalah Iptables. Firewall Generator adalah aplikasi firewall yang dapat membuat skrip Iptables secara instan dan memuat sejumlah pilihan (option), tetapi tidak dimaksudkan untuk mengatasi semua kondisi yang mungkin terjadi. Kita dapat mengisi port dan protokol yang digunakan, IP asal dan tujuan, dan jenis chain yang kita butuhkan. Skrip ini berisi perintah-perintah iptables dan akan berjalan di komputer dengan sistem operasi Linux sebagai paket filtering.

Aplikasi Firewall Generator dapat diimplementasikan pada router, server maupun klien. Aturan iptables yang dibuat akan menyeleksi paket dengan meneruskan paket yang diijinkan dan menghentikan paket yang tidak diinginkan, berdasarkan port, protokol, alamat IP asal dan IP tujuan. Pengujian dilakukan pada komputer router, dengan menggunakan nmap, ping dan traceroute. Dari hasil pengujian menunjukkan bahwa aturan iptables yang dibuat dengan aplikasi Firewall Generator ini dapat melewatkan paket dari port sesuai dengan tabel perencanaan jaringan, dan mengijinkan akses menuju alamat yang diperbolehkan

Kata kunci : firewall, paket filtering, iptables

BAB I
PENDAHULUAN

1.1. Latar Belakang

Saat ini, internet telah tumbuh dan berkembang hingga mencapai angka beberapa juta unit komputer yang terkoneksi di berbagai belahan dunia. Dari hari ke hari informasi yang terkandung di dalam jaringan internet semakin lengkap, akurat, dan penting. Informasi telah menjadi suatu aset yang berharga sehingga perlu mendapat perlakuan yang lebih spesifik. Semakin penting dan berharganya informasi tersebut serta semakin majunya pengembangan *software* menarik minat para pembobol *(hacker)* dan penyusup *(intruder)* untuk terus bereksperimen guna menemukan dan mempergunakan setiap kelemahan yang ada dari konfigurasi sistem informasi yang telah ditetapkan. Oleh karena itu, diperlukan *firewall* untuk melindungi jaringan kita dari gangguan yang berasal dari luar sistem.

Firewall (dari Building Internet Firewalls, oleh Chapman dan Zwicky) didefinisikan sebagai sebuah atau kumpulan komponen yang membatasi akses antara sebuah jaringan yang diproteksi dan internet, atau antara kumpulan jaringan lainnya. *Firewall* dapat berupa *hardware* dalam bentuk *router* atau komputer, atau *software* yang menjalankan sistem *gateway*, atau kombinasi keduanya. Ada dua pendekatan utama yang digunakan dalam membuat *firewall*, yaitu paket *filtering* dan *proxy server*.

Sistem paket *filtering* adalah sistem yang melakukan *paket routing* antara jaringan internal dengan jaringan eksternal secara selektif. Sistem ini melewatkan atau menghentikan paket data yang lewat sesuai dengan aturan yang telah ditentukan. *Router* pada sistem ini disebut *screening router*. *Screening router* tidak hanya menentukan apakah *router* dapat melewatkan suatu paket data atau tidak, tetapi juga menerapkan suatu aturan yang akan menentukan apakah paket data tersebut akan dilewatkan atau tidak.

Sedangkan *proxy server* adalah aplikasi khusus atau program *server* yang berjalan pada *host firewall*; baik pada *dual-homed* host yang memiliki sebuah *interface* ke jaringan internal dan *interface* lain ke jaringan eksternal, atau pada *bastion host* yang memiliki akses ke internet dan dapat diakses oleh mesin internal. Program ini menangani *request* (permintaan) untuk servis-servis internet dari user dan melewatkannya ke servis yang sebenarnya. *Proxy* menyediakan koneksi pengganti dan bertindak selaku *gateway* terhadap servis-servis tersebut. Oleh karena itu *proxy* sering juga disebut *gateway* level aplikasi.

Dibandingkan dengan *proxy server*, paket *filtering* memiliki beberapa kelebihan, yaitu :

1. Satu *screening router* dapat melindungi semua jaringan.
   Paket *filtering* cukup ditempatkan di satu *router*, untuk memfilter paket yang melewati seluruh jaringan yang ada di bawahnya.
2. Paket *filtering* tidak membutuhkan pengetahuan dan kerjasama dari user.
   Paket *filtering* tidak membutuhkan *software* khusus atau konfigurasi dari komputer klien.
3. Paket *filtering* tersedia di sebagian besar *router* [NOO-06].

Salah satu jenis paket *filtering* yang terkenal di sistem operasi Linux adalah Iptables. Pada awalnya, paket NAT atau *firewall* yang terkenal di Linux adalah *ipchains*, tetapi karena memiliki beberapa kekurangan, organisasi Netfilter memutuskan untuk menciptakan produk baru bernama Iptables, yang memiliki kelebihan antara lain :

1. Integrasi yang lebih baik dengan Linux *kernel* dengan kapabilitas *loading* iptables-modul spesifik *kernel*, yang didesain untuk menambah kecepatan dan tahan uji.
2. *Stateful packet inspection. Firewall* menyimpan *track* dari setiap koneksi yang melewatinya dan dalam kasus tertentu akan menampilkan *data flow* dengan tujuan untuk mengantisipasi apa tindakan berikutnya dari protokol tertentu. Ini adalah fitur yang penting dalam mendukung FTP *(File Transfer Protocol)* dan DNS *(Domain Name Server)* aktif, seperti servis jaringan lainnya.
3. *Filtering* paket dilakukan berdasarkan *MAC address* dan nilai dari *flag* dalam TCP *header*. Ini sangat membantu dalam mencegah serangan menggunakan paket dan membatasi akses dari *server* lokal terdekat ke jaringan lain menggunakan alamat IP.
4. Sistem *logging* yang memberikan pilihan untuk mengatur level detil *reporting*.
5. *Network Address Translation* yang lebih baik
6. Dukungan untuk integrasi transparan dengan program *web proxy* seperti Squid.
7. Fitur *rate limiting* yang membantu Iptables memblok beberapa serangan *denial of service* (DoS) [RUS-08].

Iptables merupakan sebuah fasilitas tambahan yang tersedia pada setiap perangkat komputer yang memiliki sistem operasi Linux dan keluarganya. Kita harus mengaktifkan fitur ini terlebih dahulu pada saat melakukan kompilasi *kernel* untuk dapat menggunakannya. Mulai dari *kernel* Linux versi 2.0, Linux sudah memberikan fasilitas penjaga keamanan berupa fasilitas bernama *ipfwdm*. Kemudian pada *kernel* 2.2, fasilitas

bernama *ipchain* diimplementasikan di dalamnya dan memberikan perkembangan yang signifikan dalam menjaga keamanan. Jika menggunakan *kernel* Linux versi 2.4 ke atas, maka fasilitas Iptables sudah merupakan *default* di dalamnya. Namun jika kita menggunakan *kernel* di bawah itu, maka kita harus melakukan modifikasi dan *editing* pengaturan *kernel* yang cukup sulit dan memakan waktu lama [FAR-05].

Firewall Generator adalah aplikasi *firewall* yang dapat membuat skrip Iptables secara instan dengan Linux *kernel* 2.4 ke atas. Firewall generator ini memuat sejumlah pilihan/opsi, tetapi tidak dimaksudkan untuk mengatasi semua kondisi yang mungkin terjadi. Halaman web Firewall Generator akan menampilkan teks PHP dan sintaks Iptables hasil dari opsi-opsi yang kita pilih/isi. Skrip ini berisi perintah-perintah iptables dan akan berjalan di komputer yang diinginkan sebagai paket filtering. Tiap-tiap bagian perintah disertai keterangan untuk mempermudah memahami skrip tersebut. Untuk dapat menjalankannya, terlebih dahulu kita harus menyimpan skrip Iptables ini (menggantikan sintaks Iptables yang sebelumnya), kemudian setelah proses *restart* Iptables akan menjalankan perintah sesuai aturan yang baru.

Ketika komputer terhubung dengan internet atau jaringan, paket-paket data yang masuk atau keluar akan di-*filter* sesuai aturan yang kita buat dengan Firewall Generator. Jika Firewall Generator dapat berfungsi dengan benar, maka paket-paket yang diterima dan diteruskan melalui jaringan hanyalah paket yang diijinkan oleh admin, sedangkan paket yang ditolak atau dibuang adalah paket yang diblok atau dilarang melewati jaringan. Untuk mengetahui apakah Iptables sudah dapat menjalankan fungsinya dengan benar, maka perlu dilakukan pengujian terhadap paket data yang dilewatkan. Jenis pengujian yang dilakukan akan dijelaskan dalam Bab II (Tinjauan Pustaka). Firewall Generator ini diharapkan dapat menjadi alternatif aplikasi paket filtering yang dapat diimplementasikan dengan mudah dan cepat, dan dapat dikembangkan lebih lanjut untuk jaringan yang lebih besar.

1.2. Rumusan Masalah

Berdasarkan pada permasalahan yang telah dijelaskan pada bagian latar belakang, maka rumusan masalah difokuskan pada :

1. Pembuatan *rule* (aturan) dalam Iptables berdasarkan analisis kebutuhan jaringan SCS.
2. Perancangan sintaks Iptables yang sesuai dengan kebijakan yang ditetapkan, atau aturan yang dibuat.
3. Perancangan *interface* dari Iptables dengan menggunakan PHP.
4. Implementasi Firewall Generator pada jaringan SCS.
5. Pengujian Firewall Generator pada jaringan SCS dan analisanya terhadap paket-paket yang lolos melewati jaringan.

1.3. Batasan Masalah

Dalam perencanaan dan pembuatan skripsi ini perlu dilakukan pembatasan masalah. Pembatasan masalah yang diajukan dalam penyusunan tugas akhir ini antara lain:

1. Sistem operasi yang digunakan pada jaringan SCS adalah sistem operasi Linux.
2. *Kernel* yang digunakan untuk Iptables adalah *kernel* 2.4 ke atas.
3. Implementasi Firewall Generator yang akan dibuat hanya untuk jaringan SCS di Universitas Brawijaya.

1.4. Tujuan

Tujuan dari implementasi Firewall Generator ini adalah untuk mengamankan jaringan komputer di SCS Universitas Brawijaya Malang dengan melewatkan, membatasi, ataupun menolak suatu koneksi pada jaringan yang dilindunginya dengan jaringan luar lain seperti internet, dan memberikan alternatif aplikasi paket filtering yang dapat dijalankan dengan mudah dan cepat.

1.5. Manfaat

Manfaat yang bisa didapatkan dari tugas akhir ini adalah:

1. Mengamankan data internal SCS di Universitas Brawijaya dari gangguan yang berasal dari jaringan eksternal maupun jaringan internal

2. Memastikan tersedianya CIA[1] (*confidentiality, integrity,* dan *availability*) yang berkaitan dengan jaringan komputer SCS di Universitas Brawijaya

1.6. Sistematika Penulisan

Sistematika penulisan dalam penelitian ini adalah sebagai berikut:

BAB I Pendahuluan

Memuat latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi pembahasan, dan sistematika pembahasan.

BAB II Dasar teori

Membahas teori-teori yang mendukung dalam perancangan dan pembuatan aplikasi.

BAB III Metodologi

Berisi tentang metode penelitian yang digunakan dalam perancangan dan pengujian sistem.

BAB IV Analisis kebutuhan dan Perancangan

Membahas tentang analisa kebutuhan dari sistem dan kemudian merancang hal-hal yang berhubungan dengan analisa tersebut.

BAB V Implementasi

Bagian ini berisi penjelasan tentang implemetasi yang telah dilakukan (*OS*, perangkat keras dan bahasa pemrograman yang digunakan) dan batasan-batasan implementasi.

BAB VI Pengujian dan Analisis

Bagian ini berisi penjelasan tentang strategi pengujian (unit, integrasi dan validasi) dan teknik pengujian yang dilakukan. Dijelaskan juga seluruh kasus uji beserta hasil pengujiannya.

BAB VII Penutup

Bagian ini berisi kesimpulan dan saran. Kesimpulan didasarkan atas pengujian dan analisis yang dilakukan di dalam proses penelitian.

---

[1] CIA (*confidentiality, integrity,* dan *availability*) merupakan prinsip keamanan dasar yang disetujui secara universal, *confidentiality* yaitu informasi hanya dapat diakses oleh orang yang berhak, *integrity* digunakan untuk menguji ketepatan dan ketelitian informasi/ data, *availability* yaitu informasi dapat tersedia dan dapat diakses [GRE-05].

BAB II
TINJAUAN PUSTAKA

2.1. *Firewall*

Firewall merupakan salah satu aplikasi Linux yang dibutuhkan sebuah server Linux untuk menjaga integritas data yang ada di server Linux dari serangan-serangan *hacker* yang tidak bertanggungjawab dengan melakukan filterisasi terhadap paket-paket yang datang kepadanya. Firewall dapat didefinisikan sebagai suatu aturan yang diterapkan baik terhadap hardware, software, ataupun sistem itu sendiri dengan tujuan untuk melindungi, baik dengan melakukan filterisasi, membatasi, ataupun menolak suatu koneksi pada jaringan yang dilindunginya dengan jaringan luar lainnya seperti internet. Oleh karena seringnya *firewall* digunakan untuk melindungi jaringannya, *firewall* tersebut juga dapat berfungsi sebagai pintu keluar jaringan yang dilindunginya dengan jaringan lainnya atau biasa disebut *gateway*. [FAR-05]

Firewall mempunyai beberapa tugas, antara lain :

a. Harus dapat mengimplementasikan kebijakan *security* di jaringan. Jika aksi tertentu tidak diperbolehkan oleh kebijakan ini, maka *firewall* harus meyakinkan bahwa semua usaha yang mewakili operasi tersebut harus gagal atau digagalkan. Dengan demikian, semua akses ilegal antar jaringan (tidak diotorisasikan) akan ditolak.
b. Melakukan *filtering*: mewajibkan semua trafik yang ada untuk dilewatkan melalui *firewall* bagi semua proses pemberian dan pemanfaatan layanan informasi. Dalam konteks ini, aliran paket data dari/menuju *firewall*, diseleksi berdasarkan *IP-address*, nomor *port*, atau arahnya, dan disesuaikan dengan kebijakan keamanan.
c. *Firewall* juga harus dapat merekam/mencatat even-even mencurigakan serta memberitahu administrator terhadap segala usaha-usaha menembus kebijakan keamanan [MUL-09].

Adapun fungsi dasar dari suatu *firewall* adalah :

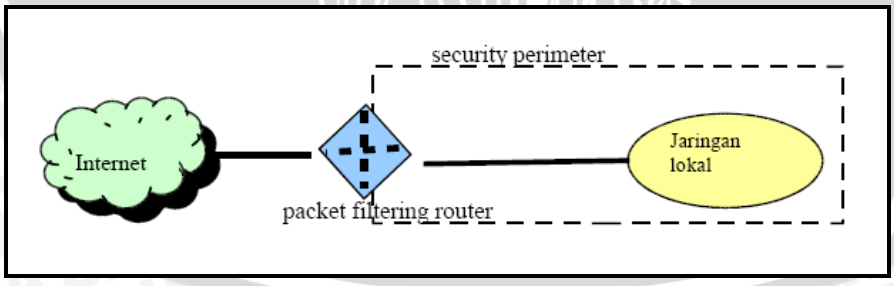
a. Paket Filtering : Seluruh *header* dari paket data yang melewati *firewall* akan diperiksa, di sini *firewall* membuat keputusan yang jelas/tegas untuk mengijinkan atau memblok setiap paket.
b. *Network Address Translation* (NAT): Dunia luar hanya akan melihat satu alamat IP di balik *firewall*, sedangkan komputer-komputer di jaringan internal dapat menggunakan alamat IP apapun yang diperbolehkan di jaringan internal, alamat sumber

dan tujuan dari paket yang melalui jaringan secara otomatis di ubah (diarahkan) ke komputer tujuan (*client* misalnya) yang ada di jaringan internal oleh *firewall*.

c. *Application Proxy*: *Firewall* mampu memeriksa lebih dari sekedar *header* suatu paket data, kemampuan ini menuntut *firewall* untuk mampu mendeteksi protokol aplikasi tertentu yang spesifik.

d. Pemantauan dan pencatatan trafik: Mencatat apa-apa saja yang terjadi di *firewall* sangatlah penting, sehingga bisa membantu kita untuk memperkirakan kemungkinan penjebolan keamanan atau memberikan umpan balik yang berguna tentang kinerja *firewall*.

Firewall dapat kita bedakan berdasarkan mekanisme atau cara *firewall* tersebut bekerja. Salah satu tipe *firewall* adalah *Packet Filtering Router. Packet Filtering Router/Gateway* dapat diartikan sebagai *firewall* yang bertugas melakukan filtrasi terhadap paket-paket yang datang dari luar jaringan yang dilindunginya. *Packet Filtering* diaplikasikan dengan cara mengatur semua paket IP baik yang menuju, melewati atau akan dituju oleh paket tersebut. Pada tipe ini paket tersebut akan diatur apakah akan di terima dan diteruskan, atau ditolak. Penyaringan paket ini dikonfigurasikan untuk menyaring paket yang akan ditransfer secara dua arah (baik dari dan ke jaringan lokal). Aturan penyaringan didasarkan pada *header IP* dan *transport header*, termasuk juga alamat IP awal dan alamat IP tujuan, protokol transport yang di gunakan (UDP,TCP), serta nomor port yang digunakan. Kelebihan dari tipe ini adalah mudah untuk diimplementasikan, transparan untuk pemakai, dan relatif lebih cepat. Adapun kelemahannya adalah cukup rumitnya untuk menyetting paket yang akan difilter secara tepat, serta lemah dalam hal authentikasi [MUA-04].

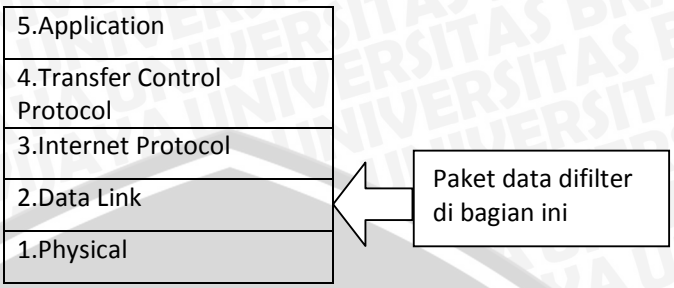


Gambar 2.1. Paket Filtering Router

Sumber : [MUA-04]

Bila kita melihat dari sisi lapisan (*layer*) dari TCP/IP, *firewall* ini akan melakukan filterisasi pada *layer Data Link* pada model OSI. *Firewall* ini biasanya adalah bagian dari

sebuah *router firewall*. Berikut ini gambar Paket Filtering dilihat dari lapisan TCP/IP atau OSI :



Gambar 2.2 *Packet Filtering* dilihat pada lapisan TCP/IP atau OSI

Sumber : [FAR-05]

2.2. Iptables

Iptables atau NetFilter adalah software Linux yang mengimplementasikan sebuah *framework* untuk *firewall* yang bersifat *statefull*. Iptables juga memiliki fitur *Network Address Translation* (NAT). Netfilter hanya bekerja pada kernel versi 2.4 atau 2.6 dan tidak dapat bekerja pada kernel yang lebih rendah dari 2.4. Pada distribusi Linux baru (termasuk yang digunakan penulis), Iptables sudah terinstal saat kita melakukan instalasi Linux di komputer kita. Akan tetapi, kita juga bisa menginstal Iptables dari file biner atau kode sumbernya.

2.2.1. Dasar Koneksi TCP

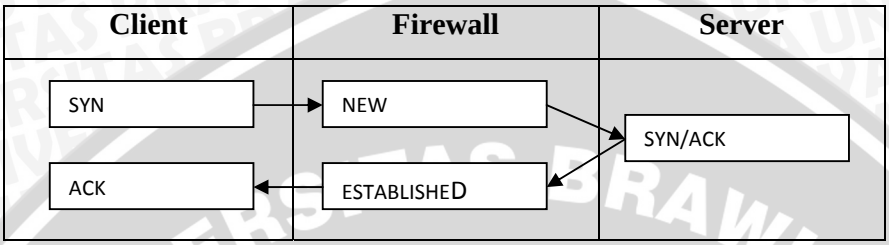
Sebelum memasuki konsep dari Iptables, sebaiknya kita mengenal terlebih dahulu konsep TCP serta hubungannya dengan Iptables sehingga Iptables dapat mengenali jika sebuah koneksi itu adalah koneksi baru (NEW), koneksi yang sudah ada (ESTABLISHED), koneksi yang memiliki relasi dengan koneksi lainnya (RELATED), atau koneksi yang tidak valid (INVALID), keempat macam koneksi itu yang membuat Iptables disebut *stateful protocol* [FAR-05].

2.2.1.1. Koneksi TCP

Sebuah koneksi TCP dikenal sebagai koneksi yang bersifat *connection oriented* yang berarti sebelum melakukan pengiriman data, mesin-mesin tersebut akan melalui 3 langkah cara berhubungan *(3-way handshake).* Pada permulaan koneksi, sebuah *client* akan mengirimkan sinyal ACK ke *server* tujuannya. Sinyal ini akan dikenali sebagai koneksi baru (NEW) oleh sebuah *firewall*. Setelah sinyal ACK diterima oleh *server*, *server* akan mengirimkan kembali sinyal SYN/ACK yang dikenali oleh sebuah *firewall* sebagai koneksi yang sudah terjadi (ESTABLISHED) ke klien tersebut. Setelah sinyal tersebut diterima, pada setiap koneksi yang terjadi klien juga akan menyertakan ACK kepada

server. Pengenalan koneksi oleh *firewall* seperti NEW, ESTABLISHED, dan RELATED dikenal dengan nama *connection tracking*.

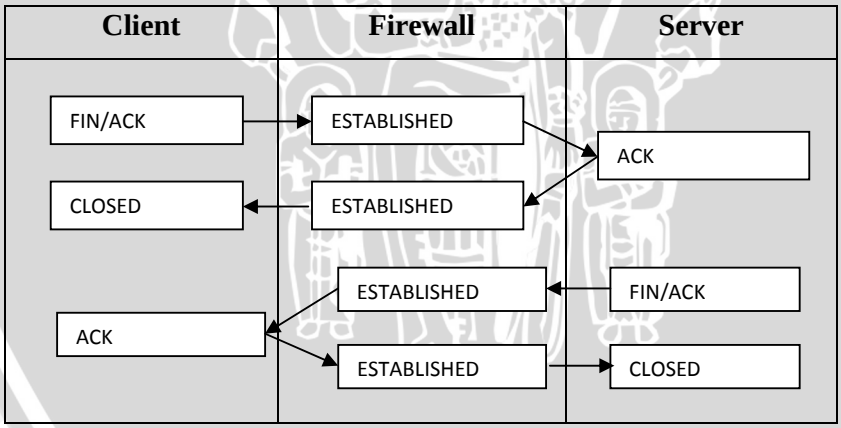
Koneksi TCP juga dikenal sebagai koneksi yang *reliable* dan *byte stream service.* Konsep *reliable* pada koneksi TCP berarti TCP akan mendeteksi *error* pada paket yang dikirim dan bila itu terjadi, paket akan dikirim kembali. Konsep *byte sream service* berarti paket-paket dikirimkan ke tujuan secara urut.



Gambar 2.3. Awal Sebuah Koneksi TCP

Sumber : [FAR-05]

Setelah koneksi TCP selesai dilakukan, klien atau *server* akan mengiriman sinyal FIN/ACK kepada mesin tujuannya. Sinyal ini masih dianggap sebagai koneksi yang sudah terjadi (ESTABLISHED). Setelah mesin tujuan menerima sinyal FIN/ACK, mesin tujuan tersebut akan membalas dengan ACK kepada mesin itu kembali dan koneksi akan terputus [FAR-05].

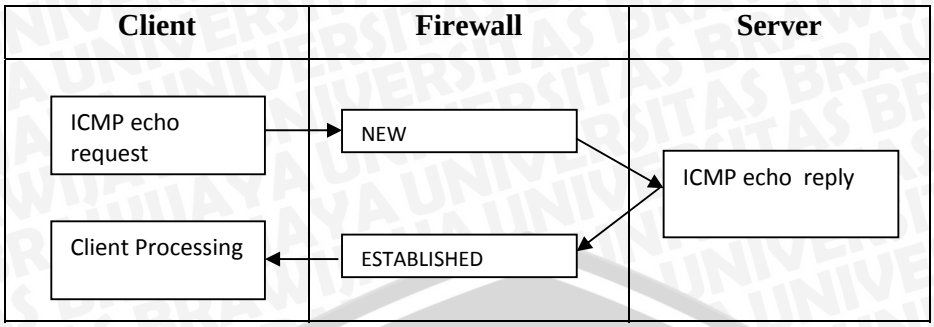


Gambar 2.4. Akhir Sebuah Koneksi TCP

Sumber : [FAR-05]

2.2.1.2. Koneksi ICMP *(Internet Control Message Protokol)*

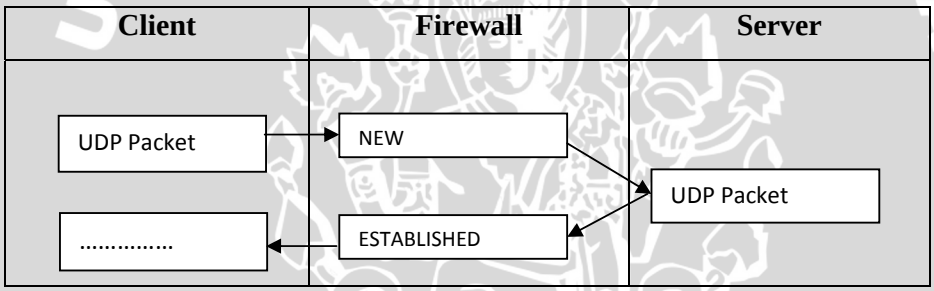
Sebuah koneksi ICMP hanyalah berupa permintaan (*request*) *echo* dan balasannya (*reply*). Ada empat macam tipe *echo* yang akan mendapatkan paket balasan, yaitu *echo request* dan *reply, timestamp request* dan *reply, information request* dan *reply,* serta *address mask request* dan *reply.*



Gambar 2.5 Sebuah Koneksi ICMP
Sumber : [FAR-05]

2.2.1.3. Koneksi UDP *(User Datagram Protocol)*

Berbeda dengan koneksi TCP, koneksi UDP bersifat *connectionless.* Sebuah mesin yang mengirimkan paket UDP tidak akan mendeteksi kesalahan terhadap pengiriman paket tersebut. Paket UDP juga tidak akan mengrimkan kembali paket-paket yang mengalami *error*. Model pengiriman paket ini akan lebih efisien pada koneksi *broadcasting* atau *multicasting.*



Gambar 2.6 Sebuah Koneksi UDP
Sumber : [FAR-05]

2.2.2. *Tables* (tabel) dan *Chain* (rantai) dalam Iptables

Iptables memiliki tiga macam daftar aturan bawaan dalam tabel penyaringan, daftar tersebut dinamakan rantai *firewall* (*firewall chain*) atau sering disebut *chain* saja. Ketiga chain tersebut adalah INPUT, OUTPUT dan FORWARD. Chain tersebut dikelompokkan lagi dalam *tables,* dimana masing-masing *table* mengurusi beberapa proses paket yang berbeda [SAT-05].

Sebuah rantai (*chain*) berisi aturan-aturan (*rules*) yang telah ditentukan, dimana aturan tersebut akan menetukan nasib suatu paket (apakah akan diteruskan, ditolak, dsb). Setiap aturan menyatakan "jika paket memiliki informasi awal (*header*) seperti ini, maka inilah yang harus dilakukan terhadap paket". Jika aturan tersebut tidak sesuai dengan paket, maka aturan berikutnya akan memproses paket tersebut. Apabila sampai aturan terakhir yang ada paket tersebut belum memenuhi salah satu aturan, maka kernel akan melihat kebijakan bawaan *(default)* untuk memutuskan apa yang harus dilakukan kepada

paket tersebut. Ada dua kebijakan bawaan yaitu *default* DROP dan *default* ACCEPT [HDP-07].

Dalam Iptables terdapat tiga macam tabel, dan masing-masing tabel berisi *predefined chain*. Administrator tidak dapat membuat atau menghapus tabel, tetapi dapat membuat atau menghapus *user-defined chains* pada tabel apa saja. Secara *default*, semua *chain* kosong dan memiliki aturan yang mengizinkan semua paket untuk lewat tanpa diblok ataupun dimodifikasi.

Macam-macam tabel tersebut antara lain:

1) Filter Table

Tabel ini bertanggung jawab dalam filtering (menolak atau mengizinkan paket untuk diproses). Tabel ini berisi *predefined chain* seperti di bawah ini:

a) INPUT : Semua paket yang datang ke sistem melewati *chain* ini.
b) OUTPUT : Semua paket yang keluar sistem melewati *chain* ini.
c) FORWARD : Semua paket yang dikirim ke sistem (karena routing) akan melewati *chain* ini.

2) Nat Table

Tabel ini bertanggung jawab dalam penulisan ulang paket-paket atau port-port. Tabel NAT digunakan untuk membuat komputer Linux menjadi *gateway* menuju internet. Tabel ini berisi *chain* seperti di bawah ini:

a) PREROUTING : Paket yang datang akan dilewatkan sebelum *routing table* lokal diperiksa, penggunaan utamanya untuk DNAT (destination-NAT[2]).
b) POSTROUTING : Paket yang datang akan dilewatkan setelah keputusan routing dibuat, penggunaan utamanya untuk SNAT (source-NAT).

SNAT digunakan untuk mengubah alamat IP pengirim (source IP address), misalnya kita menggunakan alamat IP lokal 192.168.0.1. IP tersebut akan diubah oleh SNAT menjadi IP public, misalnya 202.159.121.38. Begitu juga sebaliknya, bila komputer lokal kita bisa diakses dari internet, maka DNAT akan digunakan untuk mengubah IP public menjadi IP lokal.

3) Mangle Table

[2] NAT: *Network Address Translation,* juga dikenal sebagai *Network Masqerading* atau *IP Masquerading* adalah teknik dimana IP paket baik yang akan keluar atau masuk ditulis ulang pada saat melewati *router* atau *firewall*. Biasanya teknik ini digunakan *multiple host* atau *private network* untuk mengakses internet menggunakan IP *Address public* tunggal.

Tabel ini bertanggung jawab dalam pengaturan opsi-opsi paket, seperti *quality of service*. Table ini berisi *chain* seperti di bawah ini:

a) PREROUTING
b) INPUT
c) FORWARD
d) OUTPUT
e) POSTROUTING

Masing-masing *chain* di atas berisi daftar *rules*. Ketika sebuah paket dikirim ke suatu *chain*, paket ini dibandingkan dengan masing-masing *rule* di dalam *chain* dalam urutan atas ke bawah. *Rule* menyebutkan properti apa yang diperlukan paket agar cocok, seperti nomor port dan alamat IP. Jika paket tidak cocok dengan *rule*, maka proses berlanjut ke *rule* (aturan) selanjutnya. Jika tidak ada aturan yang cocok, maka paket ini akan mengikuti *rule target* (dan proses *chain* berikutnya akan menyebabkan paket ini dibatalkan). Target dari suatu *rule* bisa dalam bentuk *user-defined chain* atau salah satu dari *built-in target*. Beberapa contoh dari *target*, yaitu :

1) ACCEPT

Target ini menyebabkan Iptables menerima paket. Paket yang diterima dari *chain* INPUT diizinkan lewat dan diterima oleh host lokal, sedangkan paket yang diterima dari *chain* OUTPUT akan diizinkan meninggalkan host, dan paket yang diterima dari *chain* FORWARD akan diizinkan untuk di-*route* ke host.

2) DROP

Target ini menyebabkan iptables mendrop paket tanpa proses yang lebih lanjut. Paket

hilang tanpa ada indikasi yang diberikan ke host pengirim bahwa paket ini telah didrop.

Yang tampak dari sender biasanya adalah *communication timed-out*.

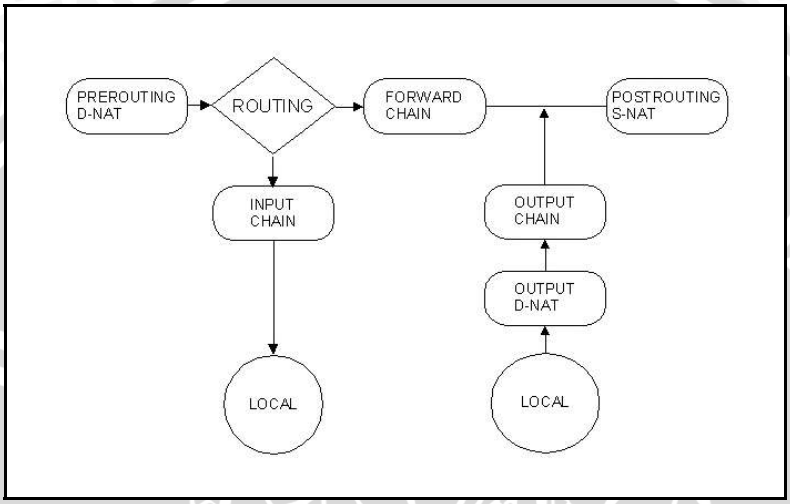
3) MASQUERADING

IPMASQ adalah algoritma perubahan *source-NAT* suatu paket data tanpa menyebutkan asal paket yang sebenarnya. Algoritma ini dapat mencari asal paket sebenarnya tanpa harus menyebutkan secara eksplisit di table routing. Algoritma ini sering dipakai pada Postrouting, dan seringkali digunakan alamat IP network lokal untuk mengakses jaringan internet dengan menggunakan satu *IP live* saja. Masquerading memerlukan proses komputasi yang lebih tinggi daripada pernyataan ACCEPT biasa, namun dalam hal efisiensi *table routing*, Masquerading

sangat bermanfaat. Dengan IPMASQ administrator tidak perlu memasukkan alamat IP baru ke daftar tabel routing ketika suatu alamat IP lokal baru ingin mengakses internet lewat gateway *IP live* [SAT-05].

2.3. Proses Pemfilteran

Proses bagaimana paket-paket data yang lewat dimanipulasi dapat dijelaskan dalam gambar seperti di bawah ini:



Gambar 2.7 Proses Pemfilteran Paket Data
Sumber : [SAT-05]

a. Paket Masuk

Paket data yang masuk akan memasuki Prerouting DNAT terlebih dahulu (di table nat). Kemudian paket akan mengalami keputusan routing di table filter. Keputusan-keputusan routing termasuk *firewalling* dilakukan di sini. Jika paket tersebut menuju ke diri sendiri, maka paket akan masuk chain input untuk diproses lebih lanjut. Jika paket data ternyata hanya lewat saja, paket tersebut akan masuk ke chain forward untuk diproses lebih lanjut. Keputusan untuk menerima (ACCEPT) ataupun menolak (REJECT) atau hanya mendrop saja (DROP) dilakukan di chain ini. Keputusan firewalling juga dilakukan di sini. Jika paket diperbolehkan lewat, maka paket akan memasuki Postrouting SNAT. Manipulasi termasuk pengubahan asal alamat paket (source-NAT) dilakukan di sini.

b. Paket Keluar

Untuk paket data yang keluar, paket akan memasuki Output DNAT terlebih dahulu. Analog dari Output DNAT ini mirip dengan Prerouting DNAT untuk paket masuk. setelah itu paket akan masuk ke chain OUTPUT. Proses filtering termasuk

firewalling akan dilakukan di sini. Jika paket diperbolehkan keluar, paket akan masuk Postouting SNAT. Perubahan SNAT biasanya dilakukan di sini, seperti proses Masquerading IP address agar alamat network lokal bisa mengakses internet dengan satu IP live [SAT-05].

2.4. Bentuk Umum Perintah Iptables

Umumnya perintah iptables memiliki bentuk seperti berikut :

iptables [-t table] -[AD] chain rule-specification [options]

Keterangan:

[-t table] : adalah table yang akan dimasuki rule.

-A atau -D : A adalah menambahkan rule, sedangkan D adalah menghapus rule.

2.5. Perintah pada Iptables

Ada beberapa opsi pada perintah Iptables yang bisa kita gunakan. Setiap opsi hanya bisa kita gunakan satu kali saja pada setiap baris perintah Iptables tersebut. Opsi-opsi perintah tersebut adalah :

1. **-A, --append** chain rule-specification

   Perintah append akan menambahkan satu aturan baru yang ditempatkan pada posisi paling bawah dari aturan-aturan Iptables yang telah kita buat.

   Contoh penggunan : `iptables –A INPUT ...`

2. -D, --delete chain rule-number/rule-specification

   Perintah ini menghapus suatu aturan pada chain. Dilakukan dengan cara menyebutkan secara lengkap perintah yang ingin dihapus atau dengan menyebutkan nomor baris dimana perintah akan dihapus. Contoh penggunaan :

   ```
   iptables -D INPUT 1
   iptables -D -s 202.159.121.38
   ```

3. -I, --insert chain rule-number/rule-specification

   Memasukkan aturan pada suatu baris di chain. Aturan akan dimasukkan pada baris yang disebutkan, dan aturan awal yang menempati baris tersebut akan digeser ke bawah. Demikian pula baris-baris selanjutnya. Contoh penggunaan :

   ```
   iptables –I INPUT 3 –s 202.159.121.38 –j ACCEPT
   ```

4. –R, --replace chain rule-number/rule-specification

   Penggunaannya sama seperti --delete, tetapi command ini menggantikan entry yang sudah ada dengan entry yang baru. Contoh penggunaan :

   ```
   iptables –R INPUT 2 -s 202.159.121.38 –j DROP
   ```

5. –L, --list chain

Perintah ini menampilkan semua aturan pada sebuah tabel. Apabila tabel tidak disebutkan, maka seluruh aturan pada semua tabel akan ditampilkan, walaupun tidak ada aturan sama sekali pada sebuah tabel. Command ini bisa dikombinasikan dengan option –v (verbose), -n (numeric) dan –x (exact). Contoh penggunaan :

```
iptables –t nat -L
```

6. –F, --flush chain

Perintah ini menghapus semua aturan pada sebuah *chain*. Apabila *chain* tidak disebutkan, maka semua *chain* akan dihapus. Contoh penggunaan :

```
iptables –F OUTPUT
```

7. -N, --new-chain chain

Perintah tersebut akan menambahkan satu kolom tabel baru (*chain*). Contoh penggunaan :

```
iptables –N eth0-IN
```

8. –X, --delete-chain chain

Perintah ini akan menghapus *chain* yang disebutkan. Agar perintah di atas berhasil, tidak boleh ada aturan lain yang mengacu kepada *chain* tersebut. Contoh penggunaan :

```
iptables –X eth0_IN
```

9. –P, --policy chain target

Perintah ini membuat kebijakan *default* pada sebuah *chain*. Sehingga jika ada sebuah paket yang tidak memenuhi aturan pada baris-baris yang telah didefinisikan, maka paket akan diperlakukan sesuai dengan kebijakan *default* ini. Contoh penggunaan :

```
iptables –P INPUT DROP
```

10. –E, --rename-chain oldchain new-chain

Perintah ini akan mengubah nama suatu *chain*. Nama *chain* yang dimaksud adalah nama kolom yang kita buat dengan perintah –N. Contoh penggunaan :

```
iptables –E eth0_IN eth0_masuk
```

2.6. Parameter pada Iptables

Parameter Iptables di bawah ini bertujuan untuk membuat satu baris aturan dari Iptables menjadi lebih spesifik. Sebagai contoh, dengan penggunaan –source kita dapat menentukan paket-paket yang datang dari sumber yang kita ketahui yang akan diberlakukan oleh aturan Iptables yang kita buat.

1. –p, --protocol [!] protocol

Digunakan untuk mengecek tipe protokol tertentu. Contoh protokol yang umum adalah TCP, UDP, ICMP dan ALL. Daftar protokol bisa dilihat pada /etc/protocols.

Tanda seru (!) di atas merupakan negasi (not) yang mengecualikan protokol yang dinyatakan, misal kita menghendaki semua protokol kecuali icmp, maka kita bisa menuliskan --protokol ! icmp yang berarti semua kecuali icmp. Contoh penggunaan :

```
iptables –A INPUT –p tcp …
iptables –A INPUT –p ! tcp …
```

2. –s, --source [!] address [/mask]

Opsi ini digunakan untuk mencocokkan paket berdasarkan alamat IP asal. Alamat di sini bisa berbentuk alamat tunggal seperti 192.168.1.1, atau suatu alamat network menggunakan netmask misal 192.168.1.0/255.255.255.0, atau bisa juga ditulis 192.168.1.0/24 yang artinya semua alamat 192.168.1.x. Kita juga bisa menggunakan inversi (tanda negasi). Contoh penggunaan :

```
iptables –A INPUT –s 202.159.121.38 …
```

3. –d, --destination [!] address [/mask]

Opsi ini digunakan untuk mecocokkan paket berdasarkan alamat tujuan. Bila tujuan dari paket tersebut cocok dengan aturan dari iptables maka aturan iptables akan berlaku pada paket tersebut. Contoh pengunaan :

```
iptables –A INPUT –d 202.159.121.38 …
```

4. –j, --jump target

Jump berguna untuk menentukan nasib paket tersebut, yaitu apakah paket tersebut akan diterima (ACCEPT), ditolak (DROP), dikembalikan ke kolom tabel sebelumnya (RETURN), dll. Contoh penggunaan :

```
iptables –A INPUT –j DROP
```

5. –i, --in-interface [!] name

Di sini setiap paket akan diidentifikasi berdasarkan kartu jaringan (LAN card) yang dimasukinya. Biasanya Linux akan menamai sebuah LAN card dengan nama *ethn*, contohnya eth0, eth1, dst. Contoh penggunaan :

```
iptables –A OUTPUT –o eth1
```

6. –o, --out-interface [!] name

Kebalikan dari –I, out-interface mengidentfikasi berdasarkan di kartu jaringan mana paket itu akan keluar. Contoh penggunaan :

```
iptables –A OUTPUT –o eth1
```

7. –f, --fragment

Opsi ini jarang sekali digunakan pada sebuah aturan iptables, tetapi keberadaannya dibutuhkan. Opsi ini akan berguna pada paket-paket yang berfragmen. Contoh penggunaan:

```
iptables –A INPUT –f
```

2.7. Match pada Iptables

Match di iptables adalah opsi-opsi tambahan agar satu baris perintah iptables atau satu aturan iptables menjadi lebih spesifik. Ada beberapa macam match pada iptables, antara lain :

1. `--mac-source [!] mac-address`

   Opsi ini digunakan untuk membedakan paket berdasarkan alamat MAC dari antarmuka jaringan kita. Setiap MAC address pada antarmka jaringan akan berbeda karena lama ini langsung diberikan oleh pembuat antarmuka jaringan tersebut. Contoh penggunaan :

   ```
   iptables  -A INPUT –m mac –mac-source 44:45:53:54:00:00
   ```

2. `multiport`

   *multiport d*igunakan bila kita ingin mendefinisikan banyak port dalam satu baris perintah iptables. Setiap port harus dpisahkan dengan tanda koma (,). Contoh penggunaan:

   ```
   iptables –A INPUT –mmultiport –source-port 22,25,110,80
   iptables –A INPUT –m multiport --destination-port 25,22,80
   ```

3. `–mark value`

   Mark akan menandai setiap paket yang sesuai dengan “value” di atas dan biasanya tujuan dari penggunaan mark adalah untuk pembatasan traffic atau bandwidth limiting. Contoh penggunaan :

   ```
   iptables –A INPUT –m mark –mark nama_mark
   ```

4. `--state state`

   State berhubungan langsung dengan conncetion tracking. Dengan penggunaan –state, iptables dapat mengetahui apakah koneksi tersebut adalah koneksi yang baru (NEW), telah ada (ESTABLISHED), atau yang memiliki relasi dengan koneksi lainnya (RELATED). Contoh penggunaan :

   ```
   iptables –A INPUT –m state –state NEW, ESTABLISHED
   ```

5. `–tos tos`

   Opsi ini dapat digunakan untuk mendefinisikan paket berdasarkan nilai TOS (Type of Service). Nilai tos terdiri dari 8 bit dan nilai tersebut ada dalam setiap paket TCP. Contoh penggunaan :

   ```
   iptables –A INPUT –m tos –tos 0x16
   ```

6. `–ttl ttl`

Ttl digunakan untuk membedakan paket berdasarkan nilai TTL (*Time to Leave*) yang berada pada header setiap paket. Nilai TTL sebanyak 8 bit ini akan berkurang bila melewati satu buah *router* atau *gateway*. Contoh penggunaan :

```
iptables –A INPUT –m ttl --ttl 60
```

7. Owner

   Opsi ini digunakan untuk membedakan paket berdasarkan identitas pemilik paket tersebut. Kita dapat membaginya menjadi 4 berdasarkan ID usernya (**--uid-owner**), , berdasarkan grupnya (**--gid-owner**), berdasarkan ID prosesnya (**--pid-owner**), atau berdasarkan ID session-nya (**--sid-owner**). Contoh penggunaan :

```
iptables –A INPUT –m owner --uid-owner 500
iptables –A INPUT –m owner --gid-owner 5
iptables –A INPUT –m owner --pid-owner 18
iptables –A INPUT –m owner --sid-owneer 199
```

2.8. Target/Jump

Target atau *jump* adalah perlakuan yang diberikan terhadap paket-paket yang memenuhi kriteria atau *match*. *Jump* memerlukan sebuah *chain* yang lain dalam tabel yang sama. *Chain* tersebut nantinya akan dimasuki oleh paket yang memenuhi kriteria. Analoginya ialah *chain* baru nanti berlaku sebagai prosedur/fungsi dari program utama. Sebagai contoh dibuat sebuah *chain* yang bernama tcp_packets. Setelah ditambahkan aturan-aturan ke dalam *chain* tersebut, kemudian *chain* tersebut akan direferensi dari *chain* input.

1. ACCEPT

   Dengan opsi ini, setiap paket akan langsung diterima oleh *firewall* dan diteruskan kepada tujuan dari paket tersebut. Misalnya paket tersebut menuju ke server kita dengan tujuan port 80, maka paket tersebut akan langsung diteruskan untuk diproses oleh server. Contoh penggunaan :

```
iptables –A INPUT –p tcp –dport 80 –j ACCEPT
```

2. REJECT

   Berbeda dengan ACCEPT, setiap paket yang memiliki target REJECT ini akan ditolak, tetapi *firewall* akan mengirimkan pesan ICMP error kepada pengirim paket. Secara default, *firewall* akan mengirimkan pesan ICMP berupa *port-unreachable.* Kita dapat mengubah pesan lain yang dikirimkan oleh firewall, seperti *icmp-net-unreachable, icmp-host-unreachable, icmp-proto-unreachable, icmp-net-prohibited,* dan *icmp-host-prohibited.* Contoh pengunaan :

```
iptables –A INPUT –p tcp –dport 80 –j REJECT \ --reject-with icmp-
net-unreachable
```

3. DROP

   Bila *firewall* menemukan paket yang di-DROP, firewall akan langsung "membuang" setiap paket yang memiliki target ini tanpa mengirim pesan error kepada pengirim paket tersebut. Contoh penggunaan :

   ```
   iptables –A INPUT –p tcp --dport 80 –j DROP
   ```

4. SNAT

   Target ini bertujuan untuk mengubah sumber pengirim paket (Source Network Address Translation) dan berguna bila kita memiliki beberapa komputer yang ingin berbagi koneksi internet. Seperti terlihat pada Tabel 3.2, SNAT terdapat di tabel NAT kolom POSTROUTING sehingga kita akan menggunakan –A POSROUTING pada perintah iptables. Contoh penggunaan :

   ```
   iptables –t nat –A POSTROUTING –p tcp –o eth0 –j SNAT \ --to-source
   202.159.121.38
   ```

5. DNAT

   Berkebalikan dengan SNAT, DNAT digunakan untuk melakukan translasi field alamat tujuan (Destination Network Address Translation) pada header dari paket-paket yang memenuhi kriteria match. DNAT hanya bekerja untuk tabel nat pada chain PREROUTING dan OUTPUT atau chain buatan yang dipanggil oleh kedua chain tersebut. Contoh penggunaan :

   ```
   iptables –t nat –A PREROUTING –p tcp –d 15.45.23.67 --dport 80 –j
   DNAT --to-destination 192.168.0.2
   ```

6. MASQUERADE

   Secara umum, target MASQUERADE bekerja dengan cara yang hampir sama seperti target SNAT, tetapi target ini tidak memerlukan option --to-source. MASQUERADE memang didesain untuk bekerja pada komputer dengan koneksi yang tidak tetap seperti dial-up atau DHCP yang akan memberi pada kita nomor IP yang berubah-ubah.

   Seperti halnya pada SNAT, target ini hanya bekerja untuk tabel nat pada chain POSTROUTING. Contoh penggunaan :

   ```
   iptables –t nat –A POSTROUTING –o ppp0 –j MASQUERADE
   ```

7. REDIRECT

   Target REDIRECT digunakan untuk mengalihkan jurusan (redirect) paket ke mesin itu sendiri. Target ini umumnya digunakan untuk mengarahkan paket yang menuju

suatu port tertentu untuk memasuki suatu aplikasi proxy. Hal ini sangat berguna untuk membangun sebuah sistem jaringan yang menggunakan transparent proxy, contohnya kita ingin mengalihkan semua koneksi yang menuju port http untuk memasuki aplikasi http proxy (misalnya squid). Target ini hanya bekerja untuk tabel nat pada chain PREROUTING dan OUTPUT atau pada chain buatan yang dipanggil dari kedua chain tersebut. Contoh penggunaan :

```
iptables –t nat –A PREROUTING –i eth1 –p tcp --dport 80 –j REDIRECT
--to-port 3128
```

8. LOG

   LOG berguna bila kita ingin melakukan pencatatan terhadap paket yang diproses oleh firewall, misalnya bila kita ingin melakukan log pada setiap paket yang di-DROP oleh firewall, atau kita ingin menelusuri kesalahan pada perintah iptables kita, atau tujuan lainnya. Ada beberapa parameter yang mengikuti target ini, antara lain :

   a) `--log-level level`

      Parameter ini berhubungan langsung dengan level yang ada pada syslog. Untuk melihat level-level yang ada, kita bisa membuka file /etc/syslog.conf. contoh penggunaan :

      ```
      iptables –A FORWARD –j LOG --log-level debug
      ```

   b) `--log-prefix value`

      Opsi ini berguna untuk memberikan prefix pada log yang akan dicatat. Maksimum karakter yang dapat digunakan adalah 29 karakter. Contoh penggunaan :

      ```
      iptables –A FORWARD –j LOG –log prefix “fwd packet”
      ```

   c) `--log-tcp-options`

      Pada parameter ini, opsi-opsi pada TCP juga akan dicatat pada log Iptables. Contoh penggunaan :

      ```
      iptables –A INPUT –j LOG --log-tcp-options
      ```

2.9. *PHP Hypertext Preprocessor* (PHP)

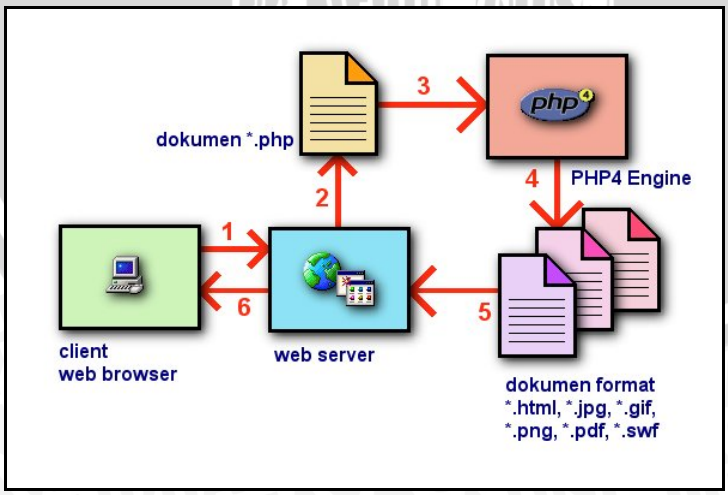
Aplikasi berbasis web dibangun tidak cukup hanya menggunakan HTML. HTML memiliki kelemahan yaitu tidak dapat mengakses dan memanipulasi data di dalam server basis data. Kelemahan HTML dapat diatasi dengan bahasa pemrograman yang mempunyai kemampuan untuk mengakses dan memanipulasi data di dalam server basis data.

Menurut dokumen resmi PHP, PHP singkatan dari *PHP Hypertext* Preprocessor. PHP dikatakan sebagai sebuah server-side embedded script language artinya sintaks-sintaks dan perintah yang kita berikan akan sepenuhnya dijalankan oleh server tetapi disertakan pada halaman HTML biasa. Aplikasi-aplikasi yang dibangun oleh PHP pada umumnya akan memberikan hasil pada web browser, tetapi prosesnya secara keseluruhan dijalankan di server.

PHP dirancang secara khusus untuk membentuk web dinamis. PHP dapat membentuk suatu tampilan berdasarkan permintaan terkini. PHP mempunyai fungsi yang sama dengan script-script seperti ASP (*Active Server Page*), Cold Fusion ataupun Perl. Untuk dapat lebih memahami tentang PHP, maka akan dibahas lebih lanjut mengenai konsep kerja PHP, variabel pada PHP, Script PHP, tag PHP, cookie, session dan keunggulan dari PHP itu sendiri [BAK-04].

2.10. Konsep Kerja PHP

Model kerja PHP diawali dengan permintaan suatu halaman *web* oleh *browser*. Berdasarkan URL (*Uniform Resource Locator*) atau dikenal dengan sebutan alamat internet, *browser* mendapatkan alamat dari *web server*, mengidentifikasi halaman yang dikehendaki dan menyampaikan segala informasi yang dibutuhkan oleh *web server*. Selanjutnya, *web server* akan mencarikan berkas yang diminta dan apabila sudah didapatkan maka *web server* akan mengirimkannya ke mesin PHP. Mesin inilah yang kemudian akan memproses dan memberikan hasilnya (berupa kode HTML) ke *web server*. Selanjutnya *web server* akan menyampaikan hasil tersebut ke *client* [CON-04]. Skema konsep kerja PHP ditunjukkan dalam gambar berikut:



Gambar 2.8. Skema konsep kerja PHP

Sumber : [CON-04]

2.11. Keunggulan PHP

PHP membuat proses pengembangan aplikasi menjadi mudah karena kelebihan-kelebihannya. Kelebihan PHP yaitu :

1. *Script* (kode program) terintegrasi dengan file HTML, sehingga *developer* bisa berkonsentrasi langsung pada penampilan dokumen *web*-nya.
2. Tidak ada proses *compiling* dan *linking*.
3. Mendukung pemrograman berorientasi obyek (*object oriented programming*).
4. Sintaks pemrogramannya mudah dipelajari, sangat menyerupai C dan Perl.
5. Integrasi yang sangat luas ke berbagai *server* basis data. Menulis *web* yang terhubung ke basis data menjadi sangat sederhana. Basis data yang didukung oleh PHP antara lain Oracle, Sybase, mSQL, MySQL, Solid, ODBC, PostgreSQL, dBase, UNIX dbm dan sebagainya [CON-04].

2.12. Sintaks PHP

Kode PHP disimpan sebagai *plain text* dalam format ASCII, sehingga kode PHP dapat ditulis hampir di semua editor teks seperti *windows notepad, windows wordpad*, dll. Kode PHP adalah kode yang disertakan di sebuah halaman HTML dan kode tersebut dijalankan oleh *server* sebelum dikirim ke *browser*.

Contoh file PHP (contoh.php):

```
<html>
<?
Print ("Contoh text yang menggunakan kode PHP");
?>
</html>
```

Pada file .html, HTTP *server* hanya melewatkan *content* dari *file* menuju ke *browser*. *Server* tidak mencoba untuk mengerti atau memproses *file*, karena itu adalah tugas sebuah *browser*. Pada *file* dengan ekstensi .php akan ditangani secara berbeda. Yang memiliki kode PHP akan diperiksa. *Web server* akan memulai bekerja apabila berada diluar lingkungan kode HTML. Oleh karena itu server akan melewati semua *content* yang berisi kode HTML, CSS, JavaScript, *simple text* di *browser* tanpa diinterpretasikan di server [PRI-08].

Blok *scripting* PHP selalu diawali dengan <?php dan diakhiri dengan ?>. Blok *scripting* PHP dapat ditempatkan dimana saja di dalam dokumen. Pada beberapa server yang mendukung, blok *scripting* PHP dapat diawali dengan <? dan diakhiri dengan ?>. Namun, untuk kompatibilitas maksimum, sebaiknya menggunakan bentuk yang standar (<?php ?>). Setiap baris kode PHP harus diakhiri dengan semikolon (;). Semikolon ini merupakan separator yang digunakan untuk membedakan satu instruksi dengan instruksi

lainnya. PHP menggunakan // untuk membuat komentar baris tunggal atau /* dan */ untuk membuat suatu blok komentar.

2.13. Pengujian Firewall Generator

Jenis pengujian yang dipilih untuk Firewall Generator antara lain nmap, ping dan trace route. Pengujian ini dilakukan untuk mengetahui apakah rules iptables berhasil memfilter/membatasi paket dalam jaringan.

2.13.1. Pengujian dengan nmap

Nmap adalah adalah aplikasi untuk mencari tahu service apa yang aktif pada port tertentu dalam suatu jaringan. Nmap dirancang untuk memungkinkan seorang sistem administrator untuk melakukan *scan* jaringan, melihat mesin yang sedang beroperasi & servis yang mereka berikan. Cukup banyak teknik *scan* yang diberikan oleh nmap seperti UDP, TCP connect(), TCP SYN (half open), ftp proxy (bounce attack), Reverse-ident, ICMP (ping sweep), FIN, ACK sweep, Xmas Tree, SYN sweep, dan Null scan.

2.19.2. Pengujian dengan ping

Ping (singkatan dari Packet Internet Groper) adalah sebuah program utilitas yang digunakan untuk memeriksa konektivitas jaringan berbasis teknologi Transmission Control Protocol/Internet Protocol (TCP/IP). Dengan menggunakan utilitas ini, dapat diuji apakah sebuah komputer terhubung dengan komputer lainnya. Hal ini dilakukan dengan cara mengirim sebuah paket kepada alamat IP yang hendak diujicoba konektivitasnya dan menunggu respons darinya.

Apabila utilitas ping menunjukkan hasil yang positif maka kedua komputer tersebut saling terhubung di dalam sebuah jaringan. Hasil statistik keadaan koneksi ditampilkan dibagian akhir. Kualitas koneksi dapat dilihat dari besarnya waktu pergi-pulang (roundtrip) dan besarnya jumlah paket yang hilang (packet loss). Semakin kecil kedua angka tersebut, semakin bagus kualitas koneksinya.

2.19.3. Pengujian *Traceroute*

Pengujian *traceroute* digunakan untuk mengetahui *node* yang telah dilalui oleh suatu paket IP [NEW-05]. *Traceroute* digunakan untuk mengetahui peta jaringan komputer yang dimiliki orang lain. *Output traceroute* akan menunjukkan *time out* jika *node* tujuan tidak dapat di *trace* yang dikarenakan paket IP aplikasi *traceroute* telah dihentikan oleh *firewall*. Sedangkan o*utput traceroute* standar akan menunjukkan jalur ke alamat IP tujuan. Hal ini menunjukkan alamat IP tersebut dapat di *trace*.

BAB III
METODE PENELITIAN

Metode penelitian menjelaskan langkah-langkah yang dilakukan di dalam perancangan, implementasi dan pengujian dari aplikasi yang akan dibuat. Kesimpulan dan saran disertakan sebagai catatan atas aplikasi dan bagaimana aplikasi dapat dikembangkan lebih lanjut.

3.1. Studi Literatur

Studi literatur yang dilakukan meliputi studi mengenai perancangan firewall, iptables dan bahasa pemrograman PHP.

3.2. Penentuan spesifikasi bahan dan alat

Sebelum melakukan perancangan, maka ditentukan spesifikasi bahan dan alat yang akan digunakan. Adapun spesifikasi bahan dan alat yang akan direalisasikan sebagai berikut:

Bahan:

- Data-data berupa perencanaan awal jaringan komputer SCS di Universitas Brawijaya

Perangkat Keras:

- 1 unit PC dengan spesifikasi minimal Pentium 1 200 Mhz , memori minimal 64 MB, dan Hardisk minimal 4 GB yang akan digunakan sebagai router dan sekaligus sebagai firewall.
- 1 unit PC dengan spesifikasi minimal Pentium III 700 Mhz , memori minimal 128 MB, dan Hardisk minimal 10 GB yang akan digunakan sebagai klien.
- unit PC dengan spesifikasi minimal Pentium 4 1.6 Mhz , memori minimal 512 MB, dan Hardisk minimal 40 GB yang akan digunakan sebagai server.

Sistem operasi yang digunakan untuk router adalah sistem operasi Ubuntu versi 9.04.

Perangkat Lunak untuk Router :

- Iptables
- Apt-get
- Nmap
- Terminal Editor

Perangkat Lunak untuk Server:

- XAMPP server

3.3. SOP (*Standard Operational Procedure*)

Perancangan Firewall Generator ini dimulai dengan pembentukan SOP (*Standard Operational Procedure*). SOP Firewall Generator dengan Linux Iptables di Jaringan SCS Universitas Brawijaya ditunjukkan seperti pada Gambar berikut :

No	Keterangan	User	Firewall
1.	User mengakses firewall generator dari jaringan lokal atau komputer pribadi.	Akses file	
2.	Muncul tampilan yang berisi option-option rule, lalu user memilih option yang dibutuhkannya.	Memilih option	Menampilkan option
3.	Setelah selesai, firewall men-generate skrip iptables, lalu skripnya disimpan dalam komputer.	Skrip iptables	Skrip iptables
4.	Firewall dijalankan untuk mengamankan jaringan sesuai rule yang diinginkan user.	Firewall mengamankan jaringan	Firewall mengamankan jaringan

Gambar 3 SOP (*Standard Operational Procedure*)

Sumber : [Perancangan]

3.4. Pengujian Rancangan

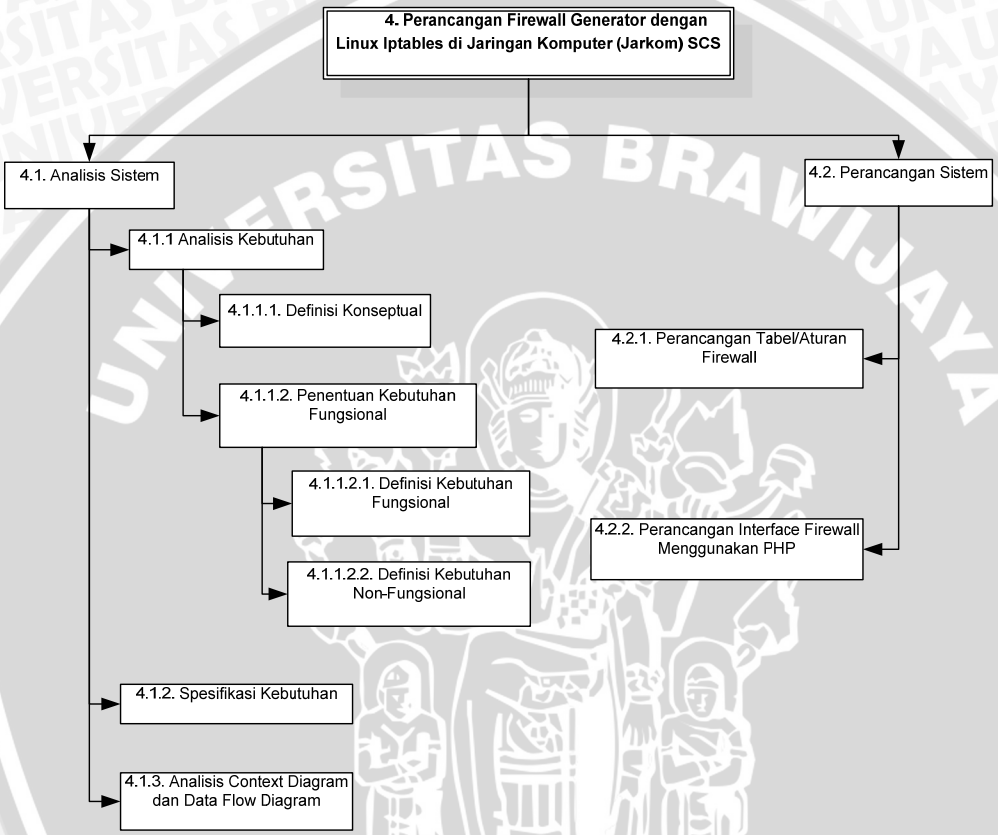
Pengujian rancangan dilakukan untuk mengetahui apakah Firewall Generator yang dibuat dapat berjalan dengan baik, serta sesuai dengan fungsinya sebagai paket filtering. Pengujian yang dapat dilakukan yaitu dengan mencoba menjalankan Firewall Generator pada *router, server*, atau *client,* kemudian menganalisa apakah Firewall Generator sudah dapat menjalankan filtering paket sesuai dengan aturan yang ditentukan. Jenis-jenis pengujian yang akan dilakukan telah dijaleskan pada Bab II.

3.5. Pengambilan Kesimpulan dan Saran

Setelah mendapatkan hasil dan analisis dari pengujian keamanan jaringan tersebut, maka langkah berikutnya adalah penarikan kesimpulan dari hasil pengujian dan memberi saran untuk pengembangan jaringan komputer ini lebih lanjut.

BAB IV
PERANCANGAN

Bab ini menjelaskan tentang perancangan Firewall Generator dengan Linux Iptables di Jaringan Komputer SCS di Universitas Brawijaya. Perancangan yang dilakukan dapat digambarkan dengan diagram pohon seperti dalam Gambar 4.1 berikut:

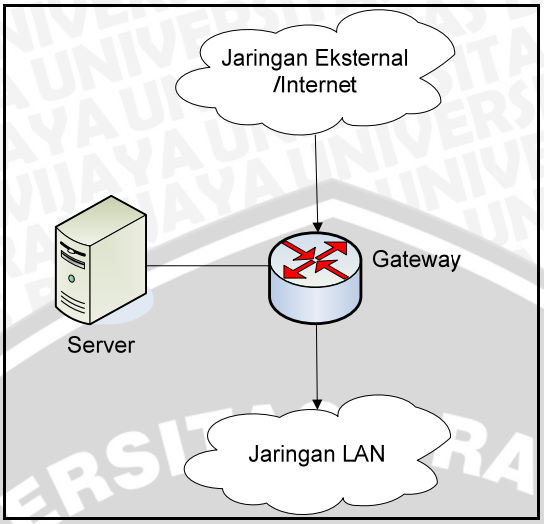


Gambar 4.1. Diagram Pohon Perancangan

Sumber : [Perancangan]

Jaringan SCS (Student Computer Services) adalah jaringan komputer yang menyediakan layanan terminal komputer publik untuk mahasiswa Universitas Brawijaya. Jaringan SCS memiliki satu buah server, satu buah gateway, dan sejumlah klien. Kebijakan keamanan jaringan termasuk konfigurasi aturan pada iptables dan penentuan jenis akses untuk klien diatur pada komputer router oleh seorang admin.

Berikut ini adalah gambar jaringan SCS Brawijaya :



Gambar 4.2. Topologi Jaringan SCS Brawijaya

Sumber : [Perancangan]

4.1. Analisis Sistem

Analisis sistem adalah proses yang menggunakan prinsip-prinsip sistem untuk mengidentifikasi, merekonstruksi, mengoptimalkan, dan mengontrol sebuah sistem [BRE-03]. Proses analisis sistem keamanan jaringan SCS meliputi analisis kebutuhan, spesifikasi kebutuhan, dan. pemodelan analisis sistem. Pemodelan analisis sistem dilakukan dengan membuat *Context Diagram* (CD) dan *Data Flow Diagram* (DFD).

4.1.1. Analisis Kebutuhan

Analisis kebutuhan sistem keamanan jaringan SCS menggunakan pendekatan analisis kebutuhan perangkat lunak yaitu proses yang digunakan untuk mendapatkan, menganalisis, dan memvalidasi kebutuhan-kebutuhan sistem [SOM-04]. Proses analisis kebutuhan sistem keamanan jaringan SCS meliputi definisi konseptual, dan penentuan kebutuhan fungsional.

4.1.1.1. Definisi Konseptual

Definisi konseptual merupakan ruang lingkup proses keamanan atau batasan dari sistem yang harus dilindungi [BRE-05]. Definisi konseptual memiliki peranan yang penting untuk membantu mendefinisikan tangggung jawab dalam proses sekurity. Definsi konseptual keamanan jaringan komputer SCS adalah melindungi *network* internal dan *server* web publik SCS dengan menyeleksi paket-paket yang berasal dari *network* eksternal dan Internet.

4.1.1.2. Definisi Kebutuhan Fungsional

Kebutuhan fungsional sistem yaitu pernyataan layanan sistem yang harus disediakan, bagaimana sistem bereaksi pada input tertentu dan bagaimana perilaku sistem pada situasi tertentu. Kebutuhan fungsional sistem harus dapat menggambarkan layanan sistem secara detail [SOM-04]. Definisi kebutuhan sistem jaringan SCS dihasilkan dari proses wawancara dengan admin untuk mengetahui kebutuhan Jaringan SCS.

4.1.1.2.1. Penentuan Kebutuhan Fungsional

Kebutuhan fungsional Firewall Generator pada jaringan SCS adalah sebagai berikut :

- Paket yang diloloskan oleh Firewall Generator adalah paket yang diperbolehkan oleh admin, sedangkan paket yang tidak dibutuhkan akan dibuang.

4.1.1.3. Definisi Kebutuhan Non-Fungsional

Kebutuhan non-fungsional adalah batasan layanan atau fungsi yang ditawarkan sistem seperti batasan waktu, batasan pengembangan proses, standarisasi dll [SOM-04].

4.1.1.3.1. Penentuan Kebutuhan Non-fungsional

Kebutuhan non-fungsional Firewall Generator pada jaringan SCS adalah sebagai berikut :

a. Firewall Generator merupakan aplikasi yang digunakan oleh admin jaringan, sehingga spesifikasi kebutuhan ditentukan oleh admin jaringan.
b. Firewall Generator berisi aturan iptables yang digunakan untuk mengatur aliran paket dalam jaringan.
c. Pihak yang dikenai kebijakan (policy) adalah server, router dan klien jaringan SCS.
d. Tidak ada pembagian kelompok klien dalam penentuan kebijakan (policy).
e. Sistem operasi yang digunakan adalah Linux Ubuntu versi 9.04.

4.1.2. Spesifikasi Kebutuhan

Spesifikasi kebutuhan terdiri dari definisi dan spesifikasi kebutuhan keamanan jaringan akan menjadi dasar bagi perancangan dan implementasi.

1. Definisi :

   Network SCS terdiri dari 1 buah server, 1 buah router, 1 buah gateway dan sejumlah klien.

1.1. Spesifikasi :

- Arsitektur Firewall Generator menggunakan tipe *Firewall Stateful Packet Filtering*.

- Firewall Generator ditempatkan di komputer router dan digunakan oleh admin.

2. Definisi :

   Paket IP yang tidak dibutuhkan akan dibuang oleh iptables.

2.1. Spesifikasi :

   Aturan iptables dalam Firewall Generator berfungsi menghentikan semua paket IP dan hanya meloloskan paket IP yang dibutuhkan oleh klien SCS.

3. Definisi :

   Aturan iptables meloloskan paket IP yang dibutuhkan oleh klien SCS.

3.1. Spesifikasi :

   Iptables meloloskan paket untuk aplikasi HTTP, FTP, Telnet dan koneksi ICMP dari gateway menuju klien.

4. Definisi :

   Iptables dapat meloloskan paket yang dibutuhkan oleh admin jaringan SCS.

4.1. Spesifikasi :

   Firewall dapat meloloskan akses dari IP admin (server) ke port manapun.

4.1.3. Analisis *Context Diagram* dan *Data Flow Diagram* (DFD)

Context diagram atau diagram konteks merupakan diagram yang menampilkan masukan proses, proses dan keluaran proses dari sistem secara umum. Diagram Konteks adalah DFD yang pertama kali dibuat dikenal dengan DFD level 0. DFD yang dibuat digunakan untuk menjelaskan aliran paket antara klien dan router dalam jaringan berdasarakan aturan iptables.

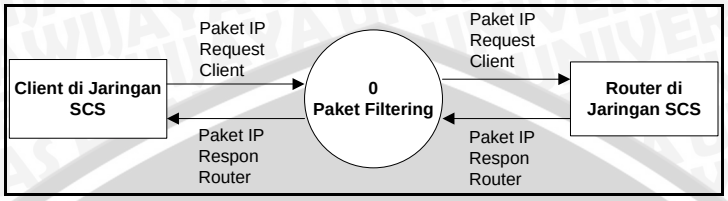
Dalam jaringan SCS ada beberapa tipe aliran data yaitu paket IP request dan paket IP response. DFD level 0 dikelompokkan berdasarkan klien pengirim paket IP request dan router penerima paket IP request. Pengelompokkan tersebut antara lain DFD Klien ke Router dan Router ke Klien.

4.1.3.1. DFD Paket Filtering antara Klien dan Router

DFD Paket Filtering antara Klien dan Router berkaitan dengan spesifikasi kebutuhan nomor 3 dan 4. Sub-bab DFD Paket Filtering antara Klien dan Router menjelaskan DFD level 0 Paket Filtering antara Klien dan Router, dan DFD level 1 Paket Filtering antara Klien dan Router.

4.1.3.1.1. DFD level 0 Paket Filtering antara Klien dan Router

DFD level 0 Paket Filtering antara Klien dan Router merupakan diagram yang menampilkan masukan proses paket filtering secara umum antara Klien dan Router. DFD level 0 Klien ke Router ditunjukkan dalam Gambar 4.3.



Gambar 4.3. DFD level 0 Klien ke Router

Sumber: [Perancangan]

Berdasarkan Gambar 4.3 proses aliran data dalam jaringan mempunyai masukan:

- Paket-IP-Request-Client

  Paket-IP-Request-Client merupakan aliran paket data berisi permintaan aplikasi tertentu dari klien ke router.
- Paket-IP-Respon-Router

  Paket-IP-Respon-Router merupakan aliran paket data berisi respon aplikasi tertentu dari router ke klien.

Berdasarkan Gambar 4.3 proses aliran data dalam jaringan mempunyai keluaran:

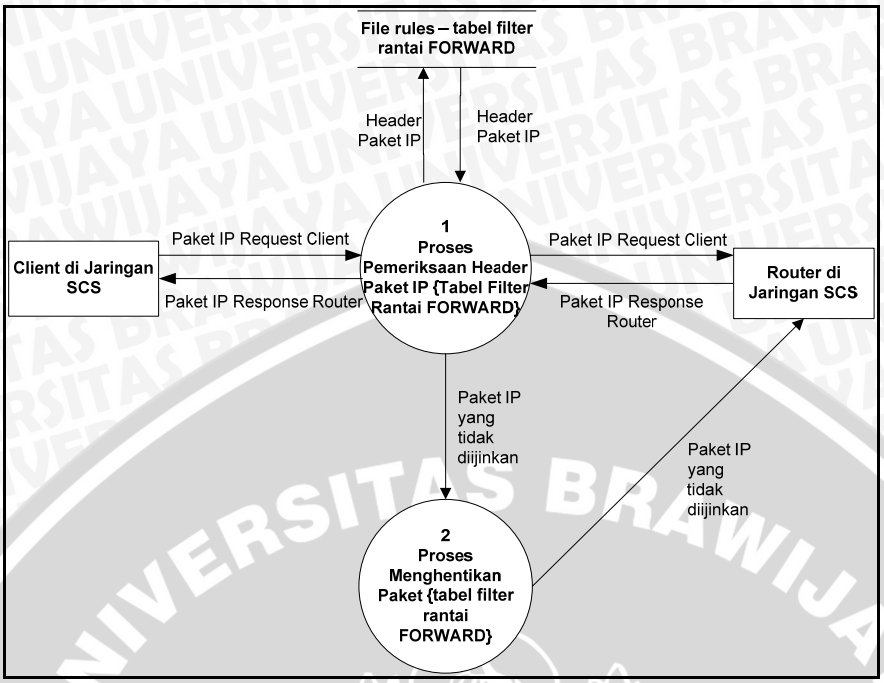
- Paket-IP-Request-Client

  Paket-IP-Request-Client merupakan aliran paket data berisi permintaan aplikasi tertentu dari klien ke router.
- Paket-IP-Respon-Router

  Paket-IP-Respon-Router merupakan aliran paket data berisi respon aplikasi tertentu dari router ke klien.

4.1.3.1.2 DFD level 1 Paket Filtering antara Klien dan Router

DFD level 1 merupakan penjabaran dari DFD level 0. DFD level 1 Paket Filtering antara Klien dan Router ditunjukkan dalam Gambar 4.4.



Gambar 4.4. DFD level 1 Paket Filtering antara Klien dan Router

Sumber: [Perancangan]

DFD level 1 Paket Filtering antara Klien dan Router memiliki 2 proses yaitu proses Pemeriksaan Header Paket IP {Tabel filter Rantai FORWARD}, dan Proses Menghentikan Paket IP{Tabel filter Rantai FORWARD}. Penjabaran proses-proses tersebut antara lain:

1. Proses Pemeriksaan Header Paket IP {Tabel filter Rantai FORWARD}

   Proses ini terletak pada proses program iptables tabel filter rantai FORWARD. Pada proses ini header paket IP akan diperiksa apakah diperbolehkan lewat atau tidak berdasarkan aturan yang dibuat oleh admin.

2. Proses Menghentikan Paket

   Proses ini akan melakukan penghentian paket yang tidak boleh lewat sesuai dengan aturan iptables.

   DFD level 1 *Client* memiliki 2 jenis aliran data yaitu paket *IP request client* dan *paket IP response router*. Urutan paket IP ini :

Paket_IP_Request_Client ► Proses Penyaringan Header Paket IP {Tabel filter rantai FORWARD} ► Paket_IP_Request_Client Paket_IP_Response_Router ► Proses Penyaringan Header Paket IP {Tabel filter rantai FORWARD} ► Paket_IP_Response_Router

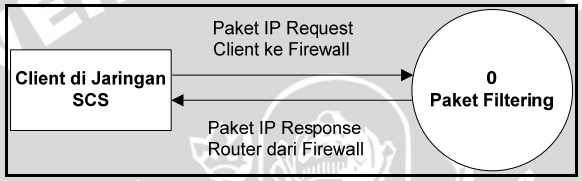
4.1.3.2. DFD Paket Filtering dari Firewall ke Klien

DFD Paket Filtering dari Firewall ke Klien berkaitan dengan spesifikasi kebutuhan nomor 4.1. Sub-bab DFD Paket Filtering dari Firewall ke Klien menjelaskan DFD level 0 dan DFD level 1 Firewall ke Klien.

4.1.3.2.1. DFD level 0 Paket Filtering dari Firewall ke Klien

DFD level 0 Firewall ke Klien merupakan diagram yang menampilkan masukan proses pemfilteran secara umum antara *client* dan Firewall. Proses paket filtering dilakukan oleh firewall yang bertugas melakukan penyaringan header paket IP dan pemblokiran paket IP.

DFD level 0 Paket Filtering dari Client ke Firewall ditunjukkan dalam Gambar 4.5.



Gambar 4.5. DFD Level 0 Client ke Firewall

Sumber : [Perancangan]

Berdasarkan gambar 4.5 proses aliran data dalam jaringan mempunyai masukan:

- Paket-IP-Request-Client-ke-Firewall

  Paket-IP-Request-Client merupakan aliran data paket IP berisi request aplikasi tertentu dari *client* ke Firewall.

Berdasarkan gambar 4.16 proses aliran data dalam jaringan mempunyai keluaran:

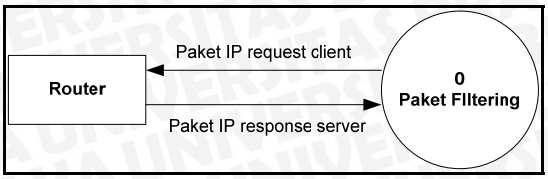
- Paket-IP-Response-Router-dari-Firewall
- Paket-IP-Response-Router-dari-Firewall merupakan aliran data paket IP berisi respons aplikasi tertentu dari router ke *client* jaringan.

4.1.3.3. DFD Paket Filtering di Router

DFD Paket Filtering di Router er berkaitan dengan spesifikasi kebutuhan nomor 4.1. Sub-bab DFD Paket Filtering di Router menjelaskan DFD level 0 dan DFD level 1 Paket Filtering di Router.

4.1.3.3.1. DFD Level 0 Paket Filtering di Router

DFD level 0 Paket Filtering di Router merupakan diagram yang menampilkan proses pemfilteran secara umum dalam router. DFD level 0 Paket Filtering di Router ditunjukkan dalam Gambar 4.7.



Gambar 4.6. DFD Level 0 Firewall ke Router

Sumber: [Perancangan]

Berdasarkan Gambar 4.7 proses aliran data dalam jaringan mempunyai masukan:

- Paket-IP-Response-Router

  Paket-IP-Request-Client merupakan aliran data paket IP berisi response aplikasi tertentu dari router ke *client.*

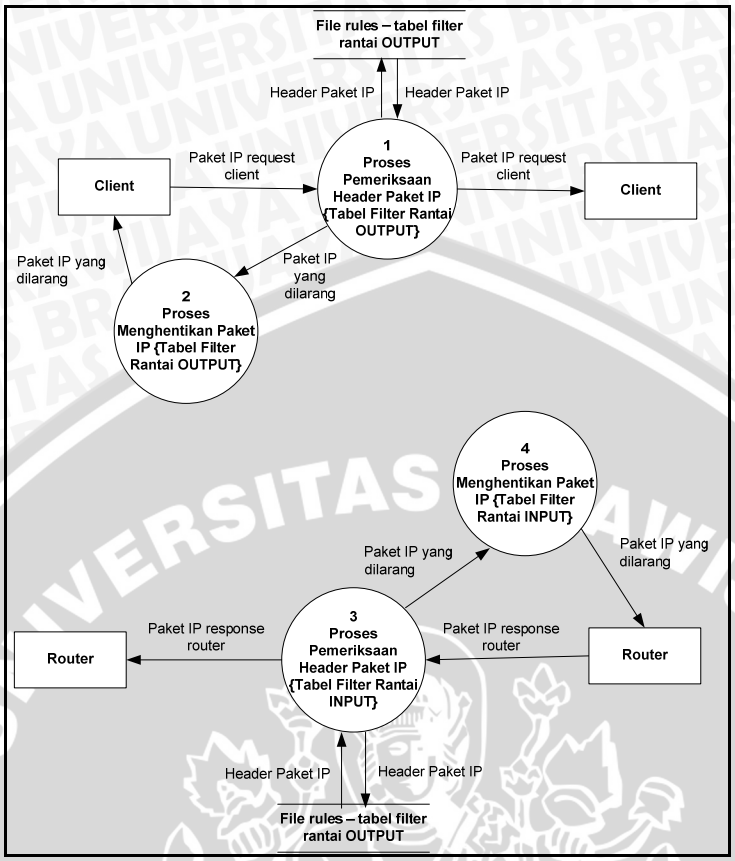
Berdasarkan Gambar 4.20 proses aliran data dalam jaringan mempunyai keluaran:

- Paket-IP-Request-Client

  Paket-IP-Request-Client merupakan aliran data paket IP berisi request aplikasi tertentu dari *client.*

4.1.3.3.2. DFD level 1 Paket Filtering di Router

DFD level 1 merupakan penjabaran dari DFD level 0. DFD level 1 Paket Filtering di Router memiliki 4 proses yaitu Proses Penyaringan Header paket IP {Tabel filter Rantai OUTPUT}, Proses Menghentikan Paket IP {Tabel filter Rantai OUTPUT}, Proses Penyaringan Header Paket IP {Tabel filter Rantai INPUT}, Proses Menghentikan Paket IP {Tabel filter Rantai INPUT}. DFD level 1 Paket Filtering di Router ditunjukkan dalam Gambar 4.8.



Gambar 4.7. DFD Level 1 Firewall ke Router

Sumber: [Perancangan]

Penjabaran proses-proses tersebut antara lain:

1. Proses Penyaringan Header Paket IP {Tabel filter Rantai OUTPUT}

   Pada proses ini header paket IP yang melewati Firewall akan diperiksa apakah diperbolehkan lewat atau tidak berdasarkan aturan iptables. Aturan iptables hanya mengijinkan paket IP yang dibutuhkan oleh admin.

2. Proses Menghentikan Paket IP {Tabel filter Rantai OUTPUT

   Proses ini akan melakukan penghentian paket IP yang tidak boleh lewat ke Firewall sesuai dengan rules iptables tabel filter rantai OUTPUT.

3. Proses Penyaringan Header Paket IP {Tabel filter Rantai INPUT} di Firewall

   Proses ini terletak pada proses program iptables tabel filter rantai INPUT. Pada proses ini header paket IP yang masuk ke komputer lokal Firewall akan diperiksa apakah diperbolehkan masuk atau tidak berdasarkan aturan iptables. Aturan iptables hanya mengijinkan paket IP yang dibutuhkan oleh admin.

4. Proses Menghentikan Paket IP {Tabel filter Rantai INPUT}

Proses ini akan melakukan penghentian paket IP yang tidak boleh masuk ke Firewall sesuai dengan rules iptables tabel filter rantai INPUT.

DFD level 1 Firewall ke Router memiliki 2 jenis aliran data paket IP yaitu paket IP request client dan paket IP response router. Urutan paket IP ini :

Paket_IP_Request_Client ► Proses Penyaringan Header Paket IP {tabel filter rantai OUTPUT} ► Paket IP Request Client

Paket_IP_Response_Router ► Proses Penyaringan Header Paket IP {Tabel filter Rantai INPUT} ► Paket_IP_Response_Router

4.2. Perancangan Aturan Iptables

Model aturan yang dibuat berdasarkan spesifikasi kebutuhan jaringan SCS Brawijaya, sehingga tidak mencakup penggunaan untuk penyaringan *(filter)* paket secara umum. Aturan yang akan dibuat terdiri dari 5 bagian, yaitu :

1. Sintaks awal

   Rule diawali dengan sintaks berikut ini :

   #!/bin/sh

   iptables -P INPUT DROP

   iptables -P OUTPUT DROP

   iptables -P FORWARD DROP

   iptables -A INPUT -m state --state ESTABLIHED,RELATED -j ACCEPT

   iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

   iptables –A INPUT DROP

2. Chain Forward

   Sintaks untuk Chain Forward memiliki bentuk seperti berikut ini :

   iptables -A FORWARD -p [protokol] --dport [port] -s [source] -d [destination] -m state --state NEW -j ACCEPT

3. Chain Input

   Sintaks untuk Chain Input memiliki bentuk seperti berikut ini :

   iptables -A INPUT -p [protokol] --dport [port] -s [source] -m state --state NEW -j ACCEPT

4. Protokol ICMP

   Sintaks untuk Chain Input memiliki bentuk seperti berikut ini :

   iptables -A [chain] -p icmp -s [source] -d [destination] -m limit --limit 1/s -j ACCEPT

5. Open all port

   Open all port yaitu sintaks yang digunakan untuk semua port yang dibuka. Bentuk umum sintaks yang dibuat seperti berikut ini :

   iptables -A [chain] -p all -s [source] -m state --state NEW -j ACCEPT

6. SNAT

   Sintaks untuk SNAT memiliki bentuk berikut ini :

   iptables -t nat -A POSTROUTING -p tcp --dport [port] -s [source] -j SNAT --to-source [To Source]

7. DNAT

   Sintaks untuk DNAT memiliki bentuk berikut ini :

   iptables -t nat -A PREROUTING -p tcp --dport [port] -s [source] -j DNAT --to-destination [To Destination]

8. Log paket (sintaks akhir)

   Pada akhir form, terdapat rule yang digunakan untuk mencatat log paket yang di-drop. Sintaks untuk pencatatan *(log)* paket yaitu :

   iptables -N INPUTLOGDROP

   iptables -A INPUTLOGDROP -j LOG --log-level info --log-prefix "[INPUT_DROP]:"

   iptables -A INPUTLOGDROP -j DROP

   iptables -A INPUT -j INPUTLOGDROP

   iptables -N OUTPUTLOGDROP

   iptables –A OUTPUTLOGDROP -j LOG --log-level info --log-prefix "[OUTPUT_DROP]:"

   iptables -A OUTPUTLOGDROP -j DROP

   iptables -A OUTPUT -j OUTPUTLOGDROP

   iptables -N FORWARDLOGDROP

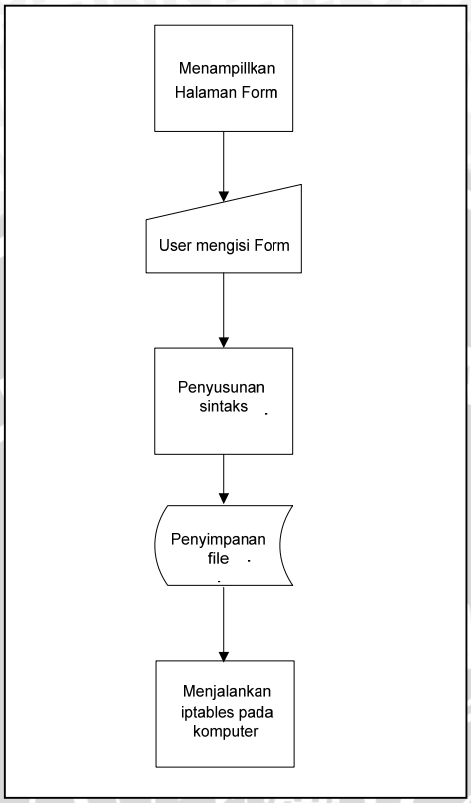
   iptables -A FORWARDLOGDROP -j LOG --log-level info --log-prefix "[FORWARD_DROP]"

   iptables -A FORWARDLOGDROP -j DROP

   iptables -A FORWARD -j FORWARDLOGDROP

4.3. Perancangan Alur Kerja

Urutan atau alur kerja di dalam Firewall Generator ini dapat digambarkan dengan diagram alir berikut ini :



Gambar 4.8. Diagram Alir Sistem Penggunaan Firewall Generator

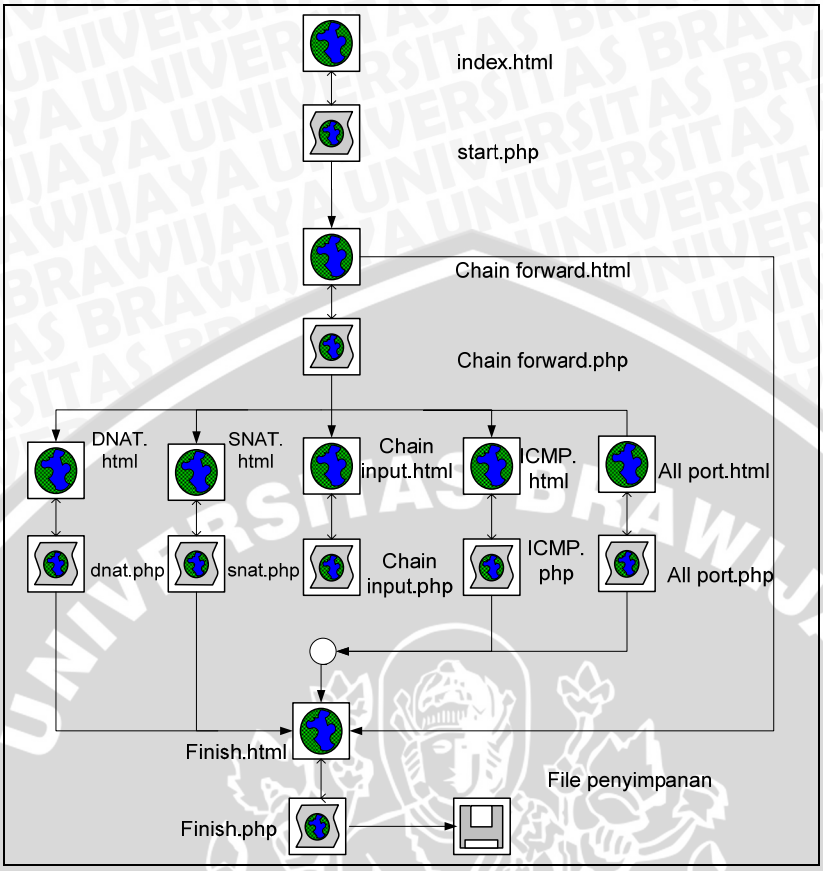
Sumber : [Perancangan]

Penjelasan :

User (dalam hal ini admin) mengakses aplikasi ini dari localhost lalu mengisi form mulai halaman awal sampai dengan halaman akhir. Setiap form membuat satu bagian sintaks. Setelah aturan iptables berhasil disusun, user akan menjalankan file tersebut pada komputer yang diinginkan (dalam hal ini router).

4.3. Perancangan Firewall Generator menggunakan HTML dan PHP

Perancangan interface bertujuan agar informasi dari komputer dapat direpresentasikan pada user. Jenis interface yang digunakan berbentuk web page dan bentuk interaksi user yang digunakan dalam perancangan ini adalah halaman form yang dibuat dengan skrip HTML. Input data dari setiap form HTML diproses oleh satu file PHP. Hubungan setiap halaman HTML dan PHP dapat digambarkan dengan site diagram berikut ini.



Gambar 4.9. Perancangan Site Diagram Firewall Generator

Sumber : [Perancangan]

Penjelasan :

Aplikasi Firewall Generator terdiri dari 6 halaman form HTML dan 6 file PHP. Masing-masing form HTML memiliki satu file PHP untuk memproses input dari form dan penulisan sintaks ke dalam file test. Setiap file PHP terdiri dari 2 bagian sintaks yaitu untuk pemrosesan form dan penulisan ke dalam file test. Skrip HTML dan PHP yang akan dibuat antara lain :

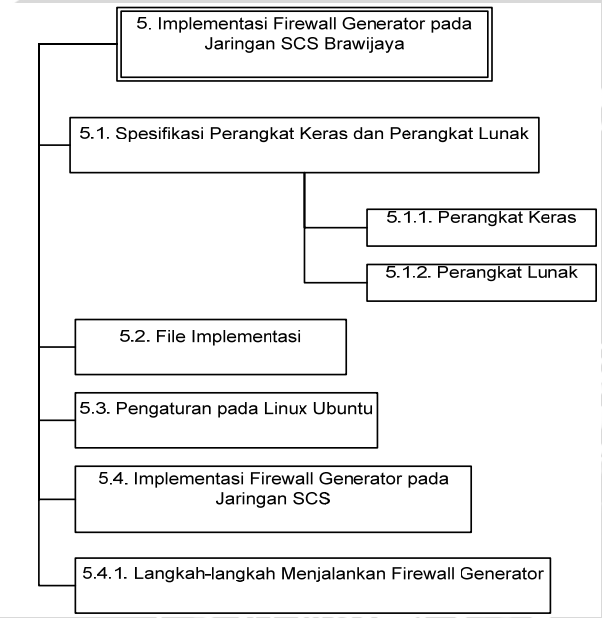
- Index.html, diproses oleh start.php.
- Chain forward.html, diproses oleh chain forward.php
- Chain input.html, diproses oleh chain input.php
- ICMP.html, diproses oleh ICMP.php
- All port.html, diproses oleh all port.php
- SNAT.html diproses oleh snat.php
- DNAT.html diproses oleh dnat.php
- Finish.html, diproses oleh finish.php

Selain skrip HTML dan PHP, akan dibuat pula skrip CSS untuk mengatur layout dan konten HTML, dan skrip javascript untuk menampilkan input form dari setiap halaman.

Penyusunan skrip iptables dimulai dari halaman awal (start.html). Kemudian, user akan menuju halaman Chain forward (Chain forward.html). Pada halaman Chain Forward user dapat memilih untuk menuju halaman Chain Input, ICMP, Open all port atau langsung mengakhiri penyusunan (halaman akhir). User dapat memilih form mana yang akan diisi atau dilewati (kecuali halaman awal dan akhir). Pada halaman akhir terdapat link untuk menampilkan hasil aturan iptables yang telah dibuat.

BAB V
IMPLEMENTASI

Bab ini menjelaskan tentang implementasi Firewall Generator dengan menggunakan iptables pada jaringan SCS Brawijaya. Implementasi yang dilakukan dapat digambarkan dengan diagram pohon seperti gambar 5.1 berikut ini.



Gambar 5.1. Diagram Pohon Implementasi Firewall Generator

Sumber : [Implementasi]

5.1. Spesifikasi Perangkat Keras dan Perangkat Lunak

5.1.1. Perangkat Keras

Perangkat keras yang dibutuhkan untuk implementasi yaitu komputer yang memiliki spesifikasi minimal sebagai berikut :

1. Processor : Intel Pentium 4 - 1.60 GHz
2. Harddisk : Maxtor 40 GB - 7200 RPM
3. Memory : Samsung 256 MB RDRAM PC800
4. LAN Card : - Realtek RTL-8139C sebanyak satu buah
   -D-Link RTL-8139C sebanyak satu buah
5. Graphic Card : Creative 32 MB nVidia GeForce 2 (dedicated memory)

5.1.2. Perangkat Lunak

Perangkat lunak yang harus dipersiapkan untuk implementasi antara lain sebagai berikut.

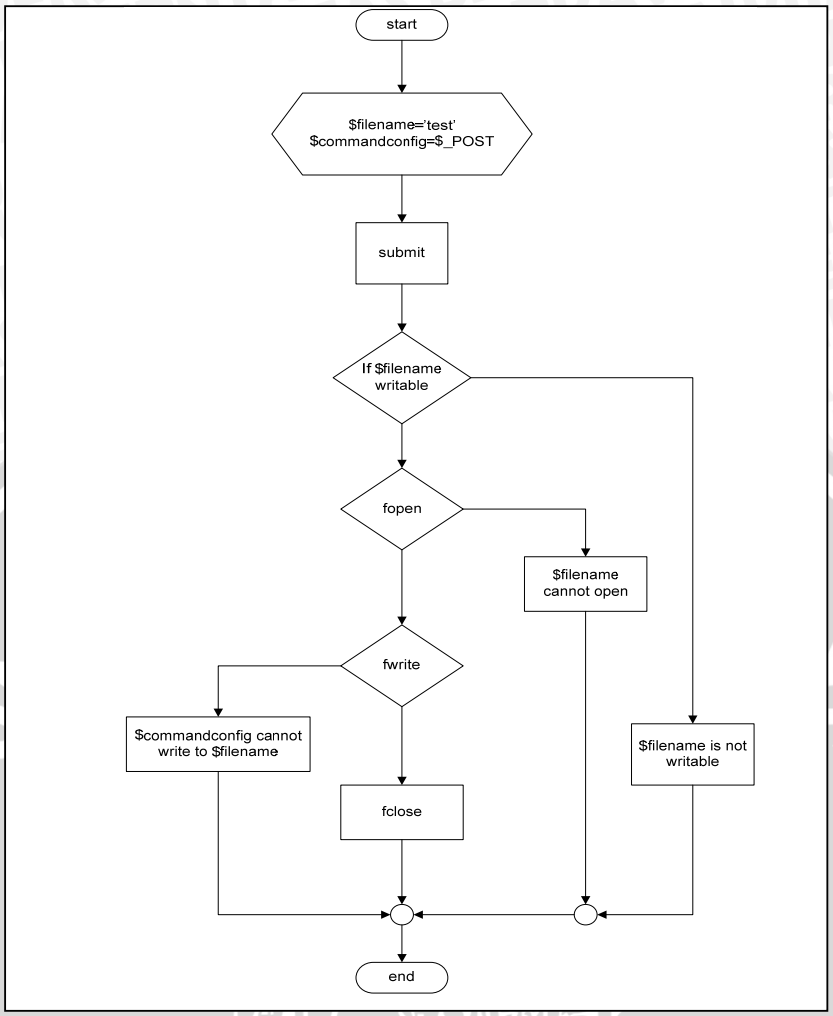
- Sistem Operasi
  a. Linux Ubuntu 9.04 untuk implementasi iptables.
  b. Microsoft Windows XP 2002 untuk pembuatan interface.
- *Software* dan aplikasi
  6. Dreamweaver CS 4 untuk pembuatan file HTML dan PHP.
  7. Mozilla Firefox untuk Windows dan Linux Ubuntu 9.04.
  8. XAMPP 1.7.2 untuk Windows dan Linux Ubuntu 9.04

5.2. File Implementasi

File aplikasi Firewall Generator ini terdiri dari :

1. Folder PHP file, yang terdiri dari file :
   - Start.php
   - Chain forward.php
   - Chain input.php
   - ICMP.php
   - All port.php
   - Finish.php
   - jquery-1.2.3.pack.js (untuk menampilkan input form dalam satu halaman)
   - file bernama *test* (tanpa ekstensi)

   Setiap file PHP memiliki fungsi untuk memproses input data dari form dan menuliskan skrip iptables ke dalam file *test*. Perintah-perintah yang digunakan antara lain : if else, $_POST, fungsi fwrite, fopen, dan fclose. Listing program dapat dilihat dalam lampiran. Algoritma yang digunakan untuk pembuatan file PHP ini dapat digambarkan sebagai berikut :



Gambar 5.2. Algoritma Sintaks PHP

Sumber : [Implementasi]

2. File HTML

File HTML yang terdapat dalam aplikasi ini yaitu :

- Index.html
- Chain forward.html
- Chain input.html
- ICMP.html
- All port.html
- SNAT.html
- DNAT.html
- Finish.html

3. File content.css untuk mengatur konten dan layout halaman html.

Listing program file di atas dapat dilihat pada lampiran di bagian akhir laporan tugas akhir ini.

5.3. Pengaturan Pada Ubuntu 9.04

Pada Ubuntu 9.04, iptables merupakan aplikasi yang sudah ada dan dapat dikonfigurasi ulang dengan akses sebagai root. Beberapa pengaturan yang diperlukan sebelum menjalankan Firewall Generator antara lain sebagai berikut.

1. Menghapus semua aturan yang sudah ada

```
root@ubuntu:~# iptables -F
root@ubuntu:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@ubuntu:~#
```

2. Menginstal XAMPP, dengan mengetikkan perintah pada terminal editor :

```
root@ubuntu:~# tar xvfz /media/MasTer/xampp-linux-1.7.2.tar.gz -C /opt/
```

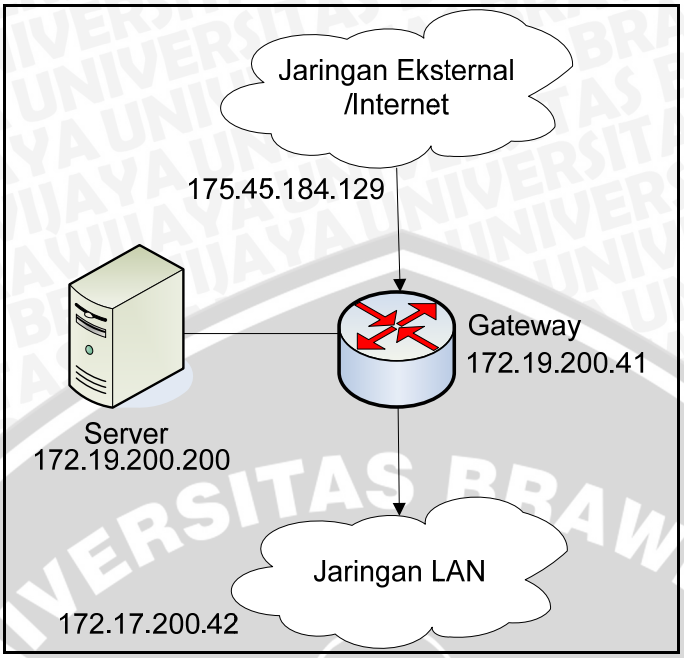
3. Menjalankan XAMPP dengan perintah

```
root@ubuntu:~# ./lampp start
```

4. Memeriksa apakah XAMPP sudah dapat berjalan di browser, dengan membuka aplikasi Mozilla, menon-aktifkan mode work offline dan membuka http://localhost. Jika halaman XAMPP telah muncul, berarti XAMPP telah berhasil dijalankan.

5.4. Implementasi Firewall Generator pada Jaringan SCS

Berdasarkan gambar Jaringan SCS, dibuat tabel perencanaan chain yang digunakan, port apa saja yang dibuka, protokol yang digunakan, network/IP Asal dan network/IP tujuan.



Gambar 5.3. Gambar Jaringan SCS

Sumber : [Implementasi]

Pada Jaringan LAN dalam gambar di atas penulis mengambil satu sampel host untuk sarana pengujian dengan alamat IP 172.19.200.42, sedangkan alamat IP 175.45.184.129 adalah alamat IP jaringan eksternal.

Untuk memudahkan perencanaan akses antara alamat IP asal dan tujuan, dibuat sebuah tabel perencanaan jaringan. Tabel Perencanaan Port dan Network yang akan diijinkan adalah sebagai berikut :

Chain	Port	Protokol	Network/IP Asal (source)	Network/IP Tujuan (destination)
FORWARD	22	TCP	175.45.184.129	172.19.200.42
FORWARD	80	TCP	175.45.184.129	172.19.200.42
FORWARD	111	TCP	175.45.184.129	172.19.200.42
FORWARD	199	TCP	175.45.184.129	172.19.200.42
FORWARD	631	TCP	175.45.184.129	172.19.200.42
FORWARD	3128	TCP	175.45.184.129	172.19.200.42
FORWARD	-	ICMP	172.19.200.42	172.17.200.200
FORWARD	-	ICMP	172.19.200.42	172.17.200.41

Tabel 5.1. Perencanaan Port dan Network

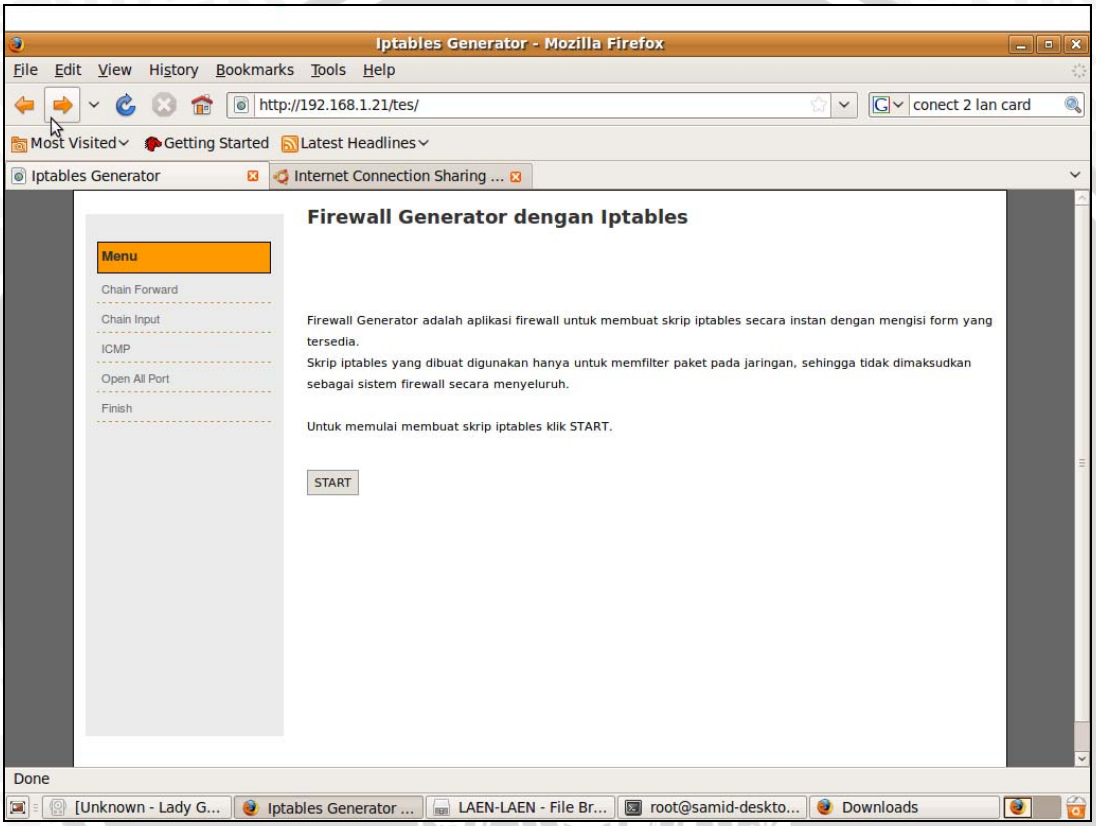
Sumber : [Implementasi]

Firewall Generator ini diimplementasikan pada router, sehingga *chain* yang digunakan oleh penulis hanya *chain forward* untuk mengatur paket yang diteruskan dari jaringan eksternal menuju jaringan internal LAN, dan ICMP untuk mengatur akses ping

dari host ke alamat lokal, tetapi pada pembahasan berikutnya disertakan pula tampilan dari form lain seandainya form tersebut digunakan.

2.1.1. Langkah-langkah Menjalankan Firewall Generator

Jika folder tes diletakkan dalam server jaringan, maka user dapat mengakses folder tersebut melalui alamat FTP lokal melalui Mozilla. Jika folder tes disimpan dalam folder htdocs, maka user dapat mengakses aplikasi ini dengan alamat http://localhost. Aplikasi Firewall Generator dimulai dengan tampilan halaman awal seperti berikut ini.



Gambar 5.4. Tampilan Awal Firewall Generator

Sumber : [Implementasi]

Pembuatan skrip dimulai dengan meng-klik tombol START, kemudian user akan menuju halaman berikutnya yaitu form chain forward. Kemudian isi field pada form berdasarkan tabel perencanaan bagian chain forward.

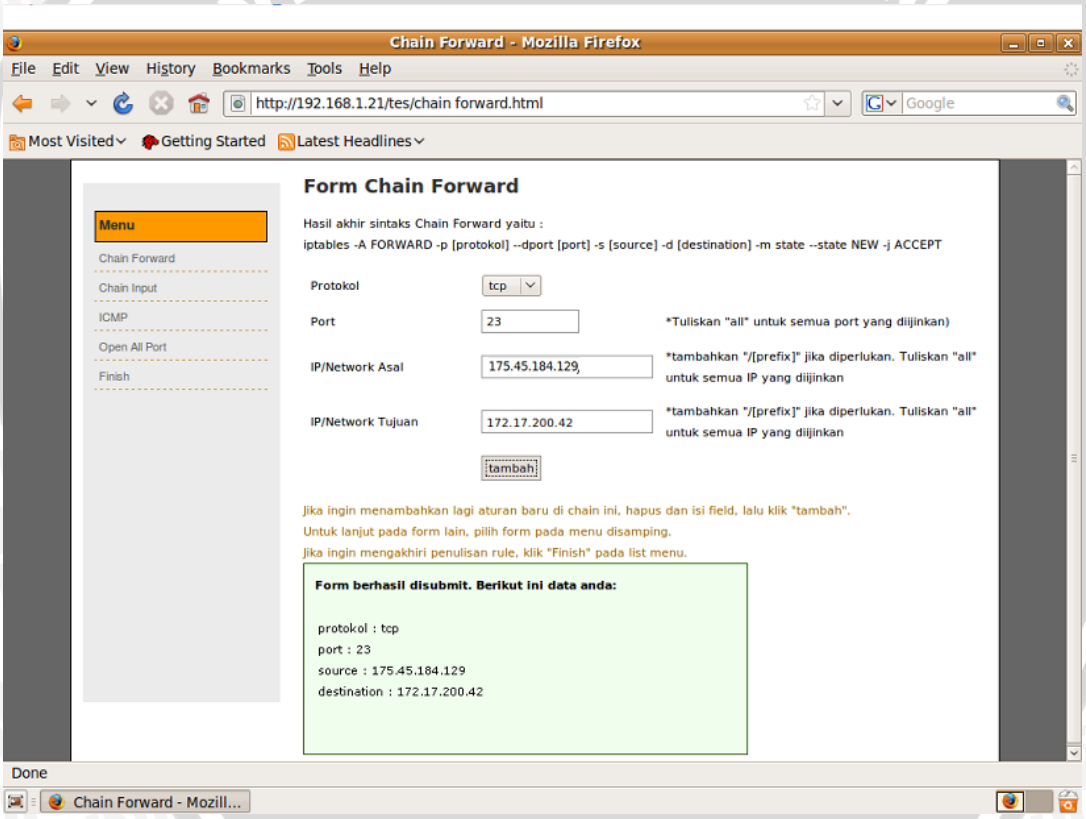
Bagian tabel yang digunakan :

Chain	Port	Protokol	Network/IP Asal (source)	Network/IP Tujuan (destination)
FORWARD	22	TCP	175.45.184.129	172.19.200.42
FORWARD	80	TCP	175.45.184.129	172.19.200.42
FORWARD	111	TCP	175.45.184.129	172.19.200.42
FORWARD	199	TCP	175.45.184.129	172.19.200.42
FORWARD	631	TCP	175.45.184.129	172.19.200.42
FORWARD	3128	TCP	175.45.184.129	172.19.200.42

Tabel 5.2. Form Chain Forward

Sumber : [Implementasi]

Contoh tampilan Form Chain Forward yang sudah diisi :



Gambar 5.5. Form Chain Forward

Sumber : [Implementasi]

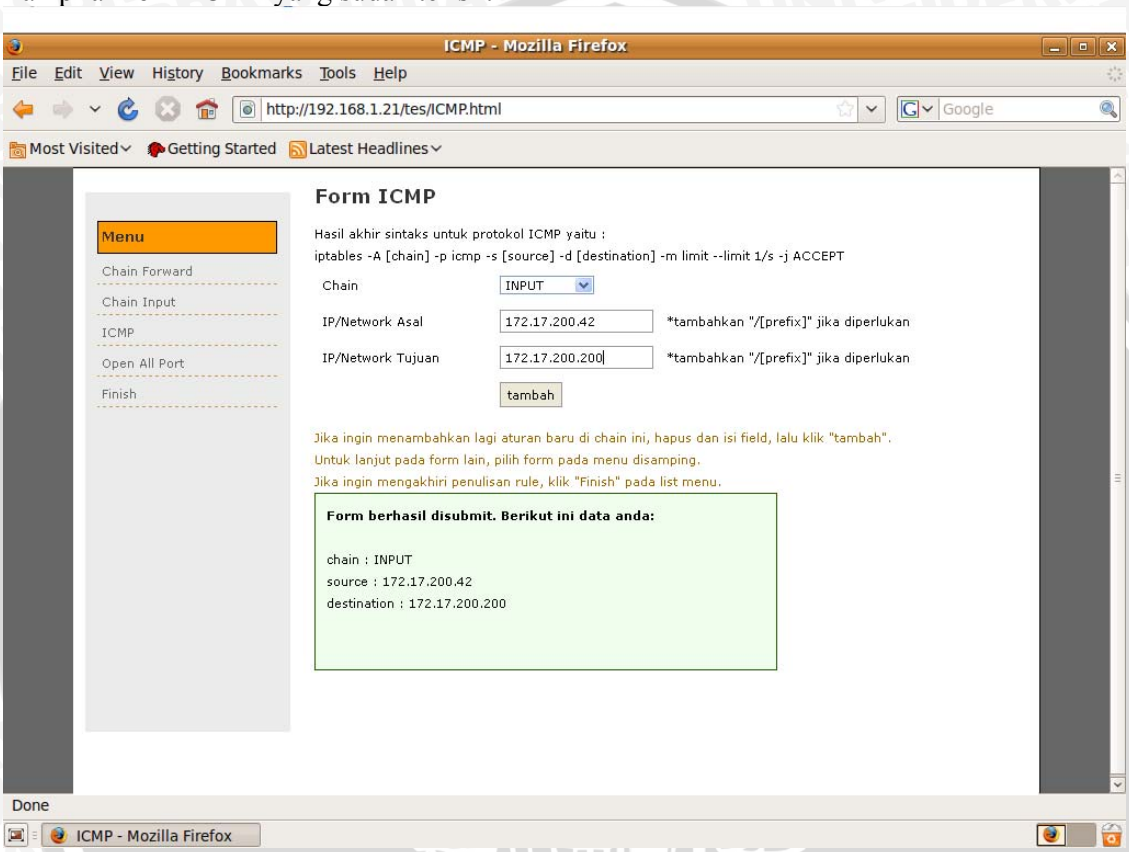
Bagian tabel perencanaan untuk ICMP :

Chain	Port	Protokol	Network/IP Asal (source)	Network/IP Tujuan (destination)
FORWARD	-	ICMP	172.17.200.42	172.17.200.200

Tabel 5.3. ICMP

Sumber : [Implementasi]

Tampilan form ICMP yang sudah terisi :

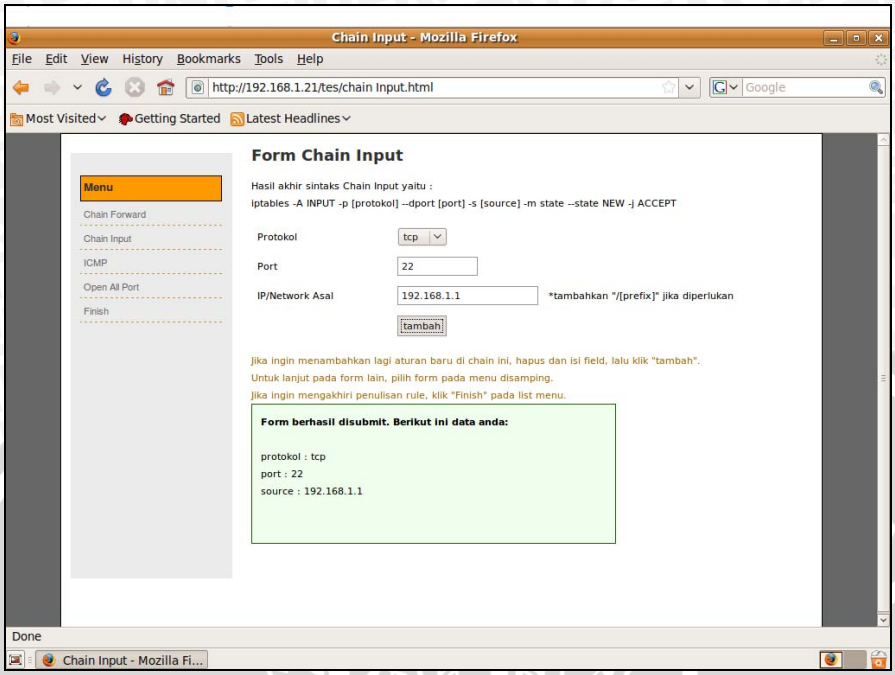


Gambar 5.6. Form Protokol ICMP

Sumber : [Implementasi]

Selain Chain Forward dan ICMP, menu lain yang dapat diisi adalah Chain Input dan Open All Port. Namun misalnya form Chain Input dan Open All Port digunakan, maka tampilan form yang terisi akan tampak seperti berikut ini.

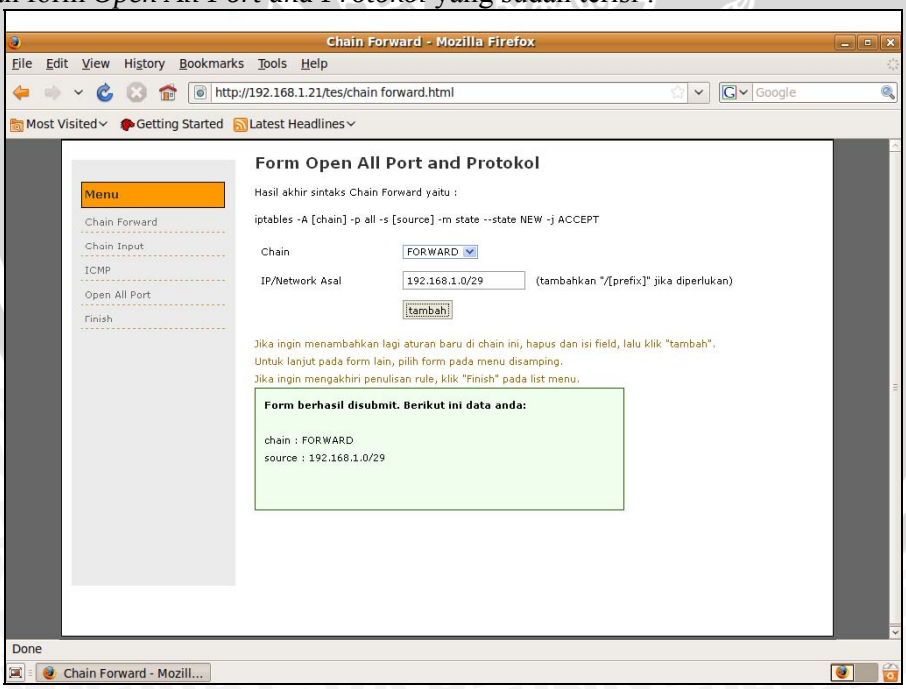
Tampilan Form Chain Input yang sudah terisi :



Gambar 5.7. Form Chain Input

Sumber : [Implementasi]

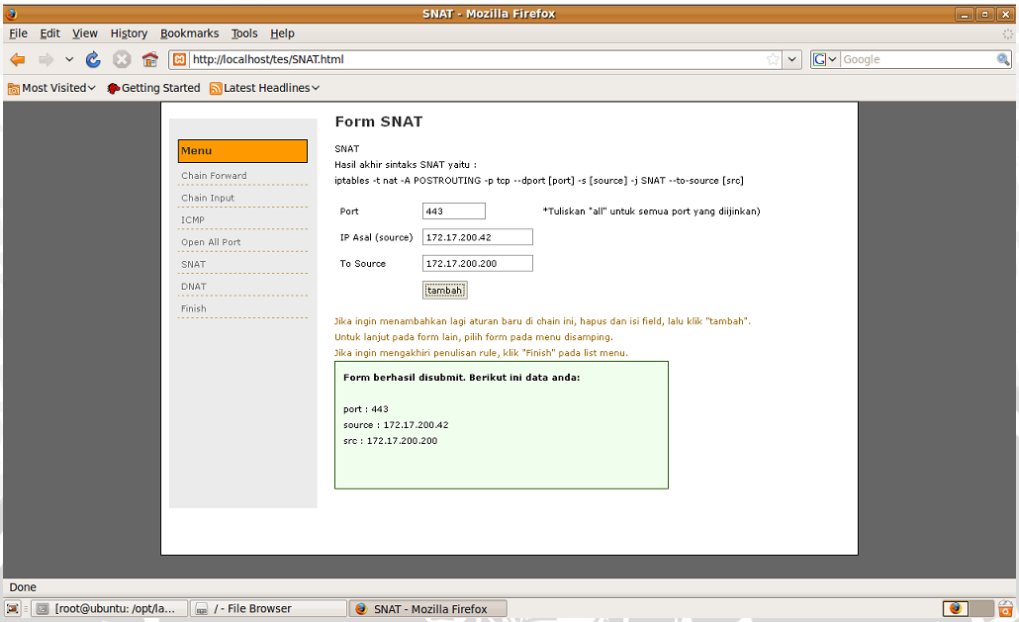
Tampilan form *Open All Port and Protokol* yang sudah terisi :



Gambar 5.8. Form *Open All Port and Protokol*

Sumber : [Implementasi]

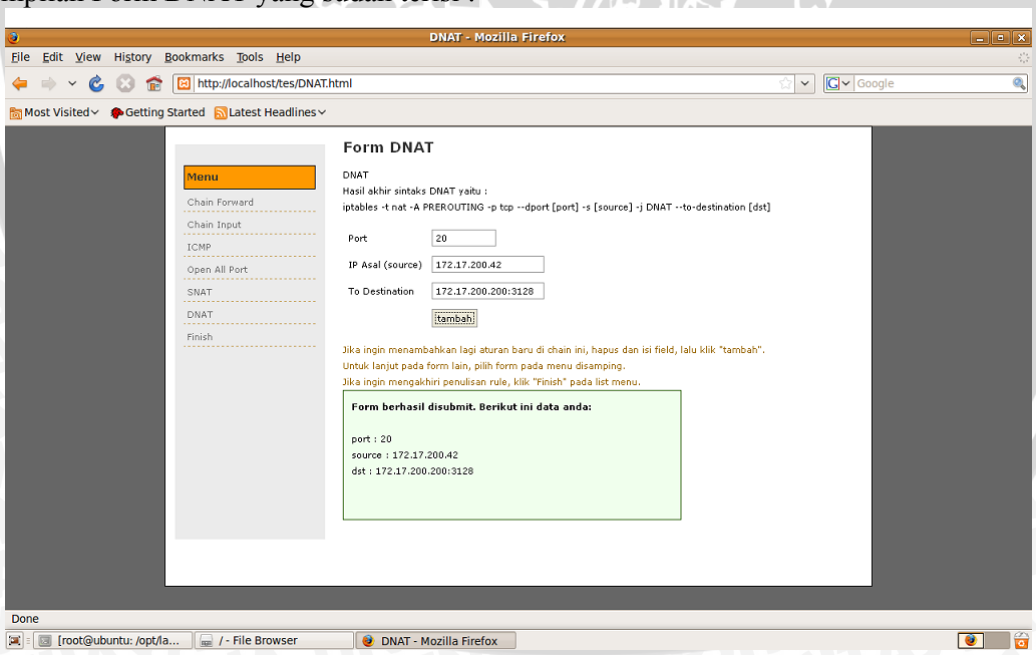
Tampilan Form SNAT yang sudah terisi :



Gambar 5.9. Form SNAT

Sumber : [Implementasi]

Tampilan Form DNAT yang sudah terisi :

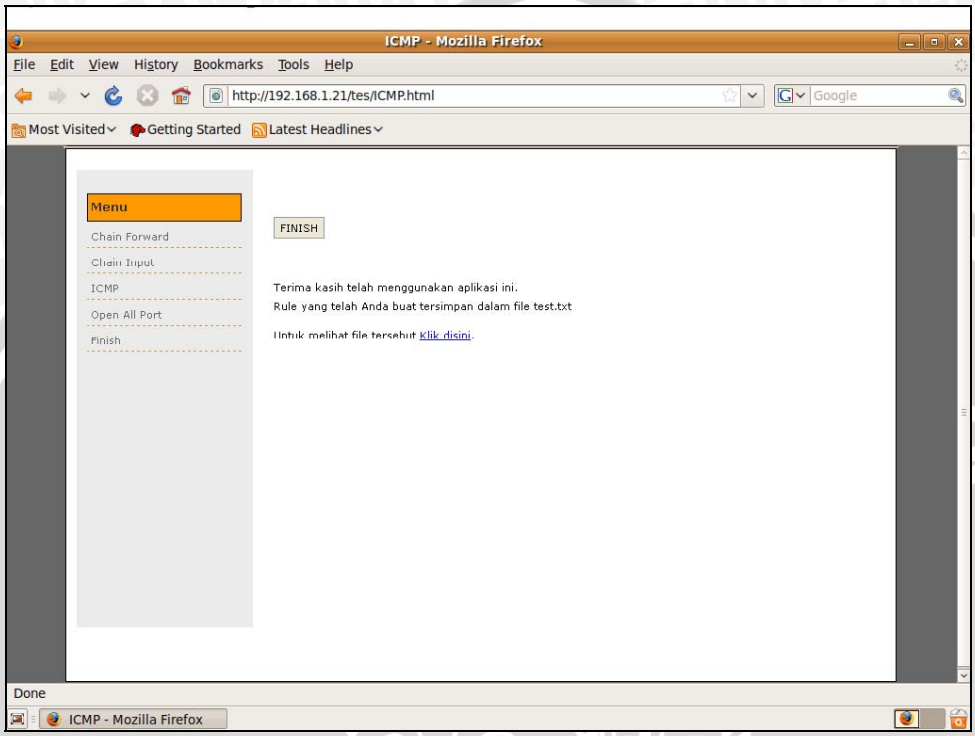


Gambar 5.10. Form DNAT

Sumber : [Implementasi]

Setelah semua form diisi, selanjutnya menu yang dipilih yaitu menu Finish untuk mengakhiri penulisan skrip iptables dan membuat aturan akhir. Aturan akhir berisi aturan yang digunakan untuk pencatatan paket *(log)* yang dibuang. Dengan meng-klik tombol finish, maka penulisan skrip telah selesai.

Tampilan halaman Finish :

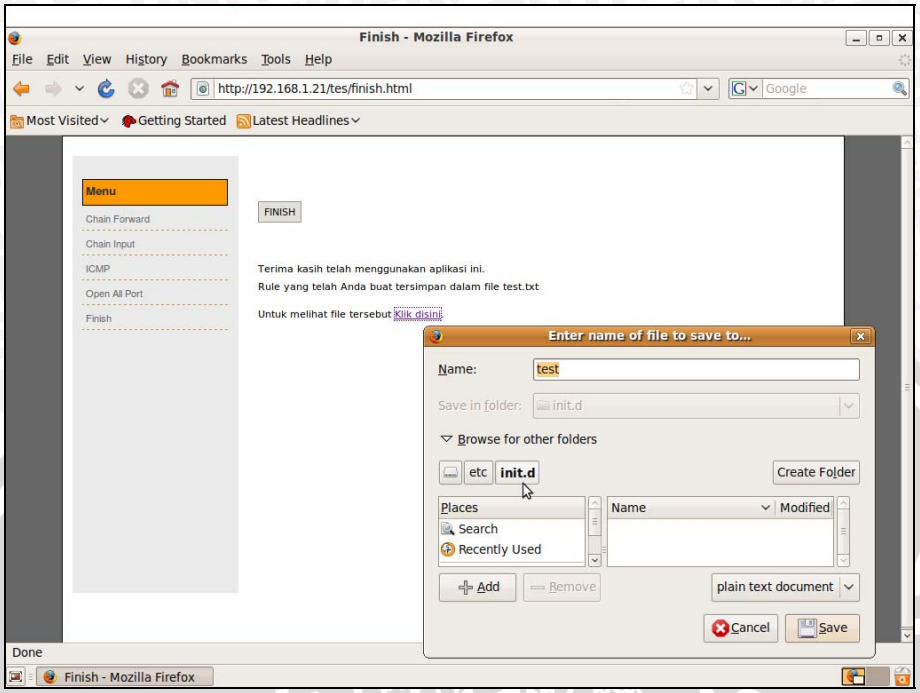


Gambar 5.11. Halaman Form Finish

Sumber : [Implementasi]

Kemudian klik kiri untuk melihat aturan yang telah dibuat dalam jendela browser, atau klik kanan hyperlink “klik disini” untuk memilih target penyimpanan aturan iptables yang telah dibuat.

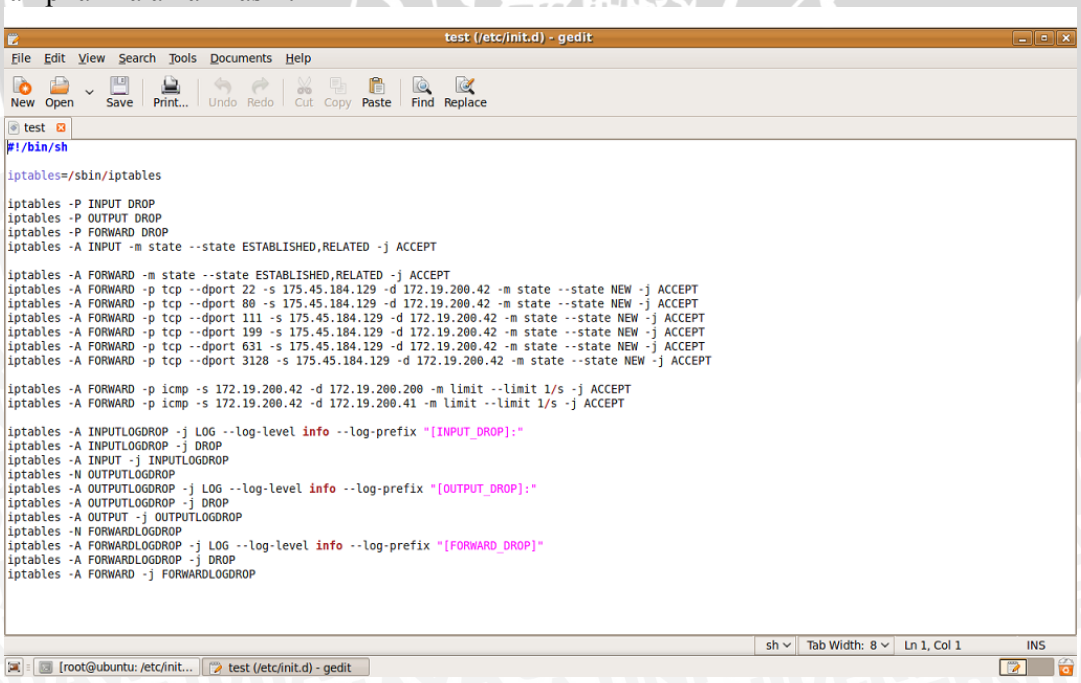
Tampilan pemilihan target penyimpanan file :



Gambar 5.12. Pemilihan target penyimpanan file

Sumber : [Implementasi]

Tampilan halaman hasil :



Gambar 5.13. Tampilan Halaman Hasil

Sumber : [Implementasi]

Langkah selanjutnya adalah meng-copy file ke direktori /etc/init.d, dan membuat file menjadi executable.

```
root@ubuntu:/etc/init.d# chmod +x test
```

Setelah itu mengeksekusi file *test*

```
root@ubuntu:/etc/init.d# ./test
```

Memeriksa apakah file sudah berhasil dieksekusi dan melihat hasil aturan iptables.

File yang dieksekusi memiliki tampilan sebagai berikut ini :

```
root@ubuntu:/etc/init.d# iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTAB
LISHED

Chain FORWARD (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTAB
LISHED
ACCEPT     tcp  --  175.45.184.129       172.19.200.42       tcp dpt:ssh state NEW
ACCEPT     tcp  --  175.45.184.129       172.19.200.42       tcp dpt:www state NEW
ACCEPT     tcp  --  175.45.184.129       172.19.200.42       tcp dpt:sunrpc state NEW
ACCEPT     tcp  --  175.45.184.129       172.19.200.42       tcp dpt:smux state NEW
ACCEPT     tcp  --  175.45.184.129       172.19.200.42       tcp dpt:ipp state NEW
ACCEPT     tcp  --  175.45.184.129       172.19.200.42       tcp dpt:3128 state NEW
ACCEPT     icmp --  172.19.200.42        172.19.200.200      limit: avg 1/sec burst 5
ACCEPT     icmp --  172.19.200.42        172.19.200.41       limit: avg 1/sec burst 5
FORWARDLOGDROP  all  --  anywhere             anywhere

Chain OUTPUT (policy DROP)
target     prot opt source               destination
OUTPUTLOGDROP  all  --  anywhere             anywhere

Chain FORWARDLOGDROP (1 references)
target     prot opt source               destination
LOG        all  --  anywhere             anywhere            LOG level info prefix `[FORWARD_DROP]'
DROP       all  --  anywhere             anywhere

Chain OUTPUTLOGDROP (1 references)
target     prot opt source               destination
LOG        all  --  anywhere             anywhere            LOG level info prefix `[OUTPUT_DROP]:'
DROP       all  --  anywhere             anywhere
root@ubuntu:/etc/init.d#
```

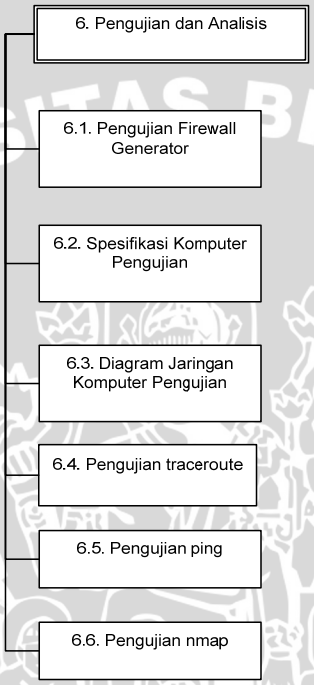
Gambar 5.14. Tampilan File Test yang Telah Dieksekusi

Sumber : [Implementasi]

Dengan demikian, maka Firewall Generator telah berhasil dijalankan. Pembahasan berikutnya adalah mengenai pengujian pada Bab VI.

BAB VI
PENGUJIAN DAN ANALISIS

Bab ini menjelaskan tentang pengujian Firewall Generator pada jaringan SCS. Pengujian dapat digambarkan dengan diagram pohon seperti dalam gambar 6.1 berikut ini.



Gambar 6.1. Diagram pohon pengujian Firewall Generator

Sumber : [Pengujian]

6.1. Pengujian Firewall Generator

Pengujian Firewall Generator terdiri dari pengujian *port scanning* dan ping. Pengujian *port scanning* bertujuan untuk mengetahui port apa saja yang dibuka untuk host dari router jaringan SCS. Pengujian ping bertujuan untuk menunjukkan bahwa paket ICMP dapat dikirimkan dari host ke router. Pengujian Firewall Generator juga bertujuan untuk menunjukkan bahwa aturan iptables telah dapat berjalan dengan baik pada jaringan.

6.2. Spesifikasi komputer pengujian

Perangkat keras yang dibutuhkan untuk implementasi yaitu 2 buah komputer yang digunakan sebagai router dan *gateway*, 1 buah komputer untuk server, 1 buah komputer klien.

Komputer yang akan berfungsi sebagai router dan *gateway* memiliki spesifikasi sebagai berikut :

6. Processor : Intel Pentium 4 - 1.60 GHz
7. Harddisk : Maxtor 40 GB - 7200 RPM
8. Memory : Samsung 256 MB RDRAM PC800
9. LAN Card : - Realtek RTL-8139C sebanyak satu buah
   -D-Link RTL-8139C sebanyak satu buah
10. Graphic Card : Creative 32 MB nVidia GeForce 2 (dedicated memory)

Komputer yang akan berfungsi sebagai server memiliki spesifikasi sebagai berikut :

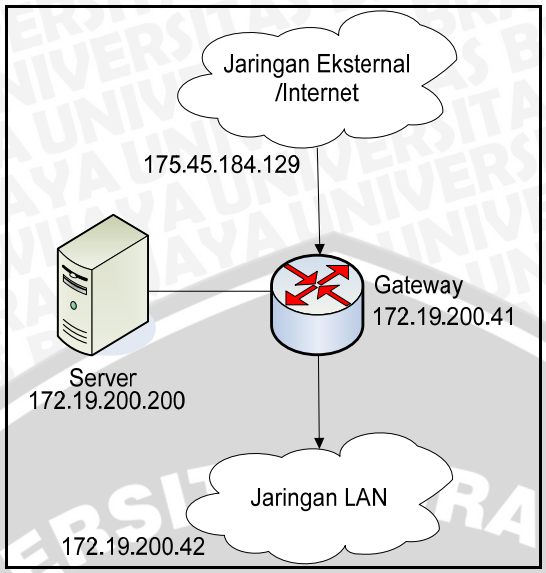
1. Processor : Intel Pentium 4 -1.6 GHz
2. Harddisk : Maxtor 40 GB - 7200 RPM
3. Memory : Samsung 256 MB DDRAM PC 2700
4. LAN Card : Realtek RTL-8139C sebanyak satu buah
5. Graphic Card : Intel 82915G/GV/910GL Chipset (share memory)

Komputer yang akan berfungsi sebagai komputer klien memiliki spesifikasi sebagai berikut :

1. Processor : Intel Pentium 4 – 1.6 GHz
2. Harddisk : Maxtor 40 GB - 7200 RPM
3. Memory : Samsung 256 MB DDRAM PC 2700
4. LAN Card :-Realtek Semiconductor Co., Ltd. RTL-8139C sebanyak satu buah
5. Graphic Card :Intel 82915G/GV/910GL Chipset (share memory)

6.3. Diagram jaringan komputer pengujian

Pengujian Firewall Generator menggunakan 1 buah host sebagai sampel jaringan internal LAN. Alamat IP dari host adalah 172.19.200.42. Sedangkan alamat IP jaringan eksternal yang dihubungkan oleh router adalah 175.45.184.129. Dalam pengujian ini diasumsikan semua port terbuka dari *gateway* menuju host.



Gambar 6.2. Gambar Jaringan Pengujian

Sumber : [Pengujian]

Tabel Perencanaan Jaringan :

Chain	Port	Protokol	Network/IP Asal (source)	Network/IP Tujuan (destination)
FORWARD	22	TCP	175.45.184.129	172.19.200.42
FORWARD	80	TCP	175.45.184.129	172.19.200.42
FORWARD	111	TCP	175.45.184.129	172.19.200.42
FORWARD	199	TCP	175.45.184.129	172.19.200.42
FORWARD	631	TCP	175.45.184.129	172.19.200.42
FORWARD	3128	TCP	175.45.184.129	172.19.200.42
FORWARD	-	ICMP	172.19.200.42	172.17.200.200
FORWARD	-	ICMP	172.19.200.42	172.17.200.41

Tabel 6. Perencanaan Port dan Network

Sumber : [Pengujian]

6.4. Pengujian traceroute

Pengujian traceroute bertujuan untuk mengetahui jalur yang dilewati host menuju server dan router. Pengujian dilakukan dengan perintah traceroute ke alamat IP yang dituju.

6.4.1. Hasil yang diharapkan

Host dapat melakukan traceroute ke server dan router.

6.4.2. Hasil Pengujian

Traceroute ke server :

```
administrator@ubuntu:~$ traceroute 172.19.200.200
traceroute to 172.19.200.200 (172.19.200.200), 30 hops max, 60 byte packets
 1  film.local (172.19.200.200)  0.261 ms  0.238 ms  0.221 ms
```

Traceroute ke router :

```
administrator@ubuntu:~$ traceroute 172.19.200.41
traceroute to 172.19.200.41 (172.19.200.41), 30 hops max, 60 byte packets
 1  172.19.200.41 (172.19.200.41)  0.500 ms  0.475 ms  0.458 ms
```

6.4.3. Analisis dan Kesimpulan

Berdasarkan hasil pengujian dengan traceroute, dapat terlihat bahwa posisi host berada langsung di bawah router dan server, dan tidak membutuhkan router lain untuk menuju alamat 172.17.200.41 dan 172.17.200.200, sehingga membuktikan bahwa gambar jaringan SCS yang dicantumkan pada tugas akhir ini sesuai dengan kondisi sebenarnya.

6.5. Pengujian nmap

Berdasarkan tabel perencanaan, port yang dilewatkan dari router ke klien adalah port 22, 80, 111, 199, 631 dan 3128. Pengujian ini dilakukan untuk mengetahui pada saat aturan iptables dijalankan, paket IP port scanning ke port-port tersebut tidak dihentikan.

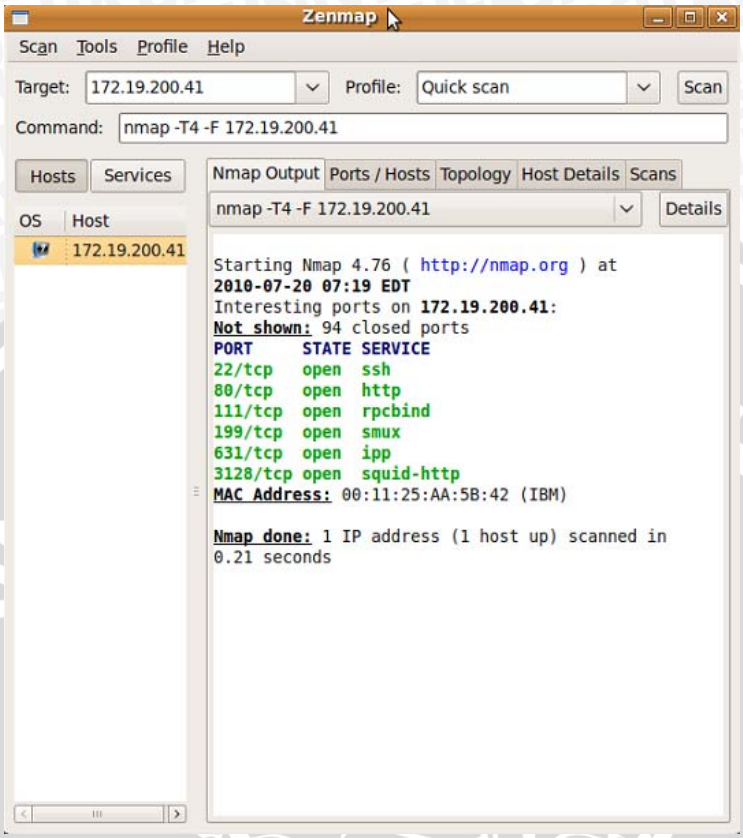
Prosedur pengujian port scanning dilakukan dengan cara mengetikkan perintah nmap pada aplikasi nmap di komputer host. Aplikasi nmap yang digunakan dalam pengujian ini adalah zenmap.

6.5.1. Hasil yang diharapkan

Pengujian nmap dapat menunjukkan port yang terbuka pada router dan klien.

6.5.2. Hasil Pengujian

Port scanning pada router :



Gambar 6.3 Hasil Pengujian NMAP

Sumber : [Pengujian]

6.5.3. Analisis dan Kesimpulan

Berdasarkan hasil pengujian, port yang terbuka dari router ke klien adalah port 22, 80, 111, 199, 631, dan 3128, sedangkan 94 port yang lain dalam status tertutup. Hal ini menunjukkan bahwa aturan iptables yang dibuat dengan aplikasi Firewall Generator telah dapat berjalan di router dan melaksanakan paket filtering dengan melewatkan paket dari port yang diijinkan saja.

6.6. Pengujian ping

Pengujian ping ini dilakukan untuk mengetahui pada saat aturan iptables dijalankan, host (172.17.200.42) dapat mengirimkan paket ICMP dengan ping kepada

router (172.17.200.41) dan server (172.17.200.200), karena berdasarkan Tabel Perencanaan protokol ICMP ke server dan router diijinkan.

6.6.1. Ping ke google.com

Ping dijalankan dengan mengetikkan perintah berikut ini dalam terminal :

```
root@ubuntu:/home/administrator# ping google.com
```

6.6.1.2. Hasil yang diharapkan

Komputer host tidak dapat melaksanakan ping ke alamat google.com

```
root@ubuntu:/home/administrator# ping google.com
ping: unknown host google.com
```

6.6.1.3. Hasil pengujian

Pengujian ping menunjukkan bahwa host tidak dapat melakukan ping menuju google.com

6.6.1.4. Analisis dan Kesimpulan

Dari hasil pengujian dengan ping ini diketahui bahwa akses protokol ICMP selain menuju server dan router tidak diperbolehkan, maka dapat disimpulkan bahwa aturan iptables telah dapat menghentikan paket sesuai dengan tabel perencanaan.

6.6.2. Ping menuju router dan server

Ping menuju router dilakukan dengan perintah : ping 172.17.200.41, sedangkan ping menuju server dilakukan dengan perintah : ping 172.17.200.200.

6.6.2.1. Hasil yang diharapkan

Berdasarkan tabel perencanaan, akses ICMP menuju router dan server diperbolehkan sehingga host dapat melakukan ping ke server dan router.

6.6.2.2. Hasil Pengujian

Ping ke server :

```
administrator@ubuntu:~$ ping 172.19.200.200
PING 172.19.200.200 (172.19.200.200) 56(84) bytes of data.
64 bytes from 172.19.200.200: icmp_seq=1 ttl=64 time=4.54 ms
64 bytes from 172.19.200.200: icmp_seq=2 ttl=64 time=0.220 ms
64 bytes from 172.19.200.200: icmp_seq=3 ttl=64 time=0.212 ms
64 bytes from 172.19.200.200: icmp_seq=4 ttl=64 time=0.206 ms
64 bytes from 172.19.200.200: icmp_seq=5 ttl=64 time=0.212 ms
^Z
[3]+  Stopped                 ping 172.19.200.200
```

Ping ke router :

```
administrator@ubuntu:~$ ping 172.19.200.41
PING 172.19.200.41 (172.19.200.41) 56(84) bytes of data.
64 bytes from 172.19.200.41: icmp_seq=1 ttl=64 time=0.389 ms
64 bytes from 172.19.200.41: icmp_seq=2 ttl=64 time=0.466 ms
64 bytes from 172.19.200.41: icmp_seq=3 ttl=64 time=0.293 ms
64 bytes from 172.19.200.41: icmp_seq=4 ttl=64 time=0.367 ms
64 bytes from 172.19.200.41: icmp_seq=5 ttl=64 time=0.441 ms
64 bytes from 172.19.200.41: icmp_seq=6 ttl=64 time=0.288 ms
64 bytes from 172.19.200.41: icmp_seq=7 ttl=64 time=0.380 ms
64 bytes from 172.19.200.41: icmp_seq=8 ttl=64 time=0.344 ms
64 bytes from 172.19.200.41: icmp_seq=9 ttl=64 time=0.334 ms
64 bytes from 172.19.200.41: icmp_seq=10 ttl=64 time=0.421 ms
^C
--- 172.19.200.41 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 8997ms
rtt min/avg/max/mdev = 0.288/0.372/0.466/0.058 ms
```

6.6.2.3. Analisis dan Kesimpulan

Dari hasil pengujian di atas dapat terlihat bahwa host dapat melakukan ping menuju router dan server, sehingga dapat disimpulkan bahwa aturan iptables dapat melewatkan paket ICMP dari host ke router dan server sesuai dengan tabel perencanaan yang dibuat.

BAB VII
PENUTUP

7.1. Kesimpulan

Berdasarkan hasil pengujian dan analisis yang dilakukan terhadap aplikasi Firewall Generator dapat diambil kesimpulan sebagai berikut :

1. Pembuatan aturan iptables melalui aplikasi Firewall Generator sesuai dengan analisis jaringan SCS
2. Perancangan sintaks iptables sesuai dengan kebijakan yang ditetapkan, yang ditunjukkan dengan hasil pengujian.
3. Perancangan *interface* (tampilan) Firewall Generator berhasil dilakukan dengan bahasa pemrograman PHP.
4. Implementasi Firewall Generator pada jaringan SCS dapat berjalan dengan hasil yang diharapkan, sehingga aplikasi ini juga dapat digunakan untuk jaringan lain dengan kebijakan yang sama.
5. Pengujian berhasil dilakukan dan berdasarkan analisis terhadap paket yang lolos, disimpulkan bahwa akses antara host menuju router, server dan jaringan eksternal sesuai dengan tabel perencanaan jaringan yang dibuat.

7.2. Saran

Saran yang dapat diberikan untuk pengembangan aplikasi Firewall Generator ini yaitu penambahan model aturan iptables yang dibuat sehingga dapat digunakan oleh topologi jaringan yang lain

DAFTAR PUSTAKA

[AND-06] Andreasson, Oskar. *Iptables Tutorial 1.2.2.* http://www.borisoop.com/attachment/493e0e9cca2f9EZ.pdf . Copyright © 2006.

[BAK-04] Bakken, Stig Sather, dkk. *PHP Manual.chm.* PHP Documentation Group. 2004.

[CON-04] Converse, Tim dan Joyce Park, Clark Morgan. *PHP5 and MySQL Bible.* Wiley Publishing, Inc., Indianapolis, Indiana. 2004.

[CRE-04] Cressy, LeRoy D. *Iptables*. http://lrcressy.com/linux/iptables.pdf. 2004.

[FAI-05] GP, Faiz. *Firewall dengan Menggunakan iptables.* http://purwakarta.org/flash/firewall.pdf . 2005.

[FAR-05] Farunuddin, Rakhmat. *Membangun Firewall dengan IPTables di Linux.* PT Elex Media Komputindo, Jakarta. 2005.

[MUA-04] Muammar. W. K, Ahmad. 2004. *Firewall*. http://www.ilmukomputer.com/umum/ammar-firewall.pdf.

[NOO-06] Noonan, Wes, dan Ido Dubrawsky. *Firewall Fundamentals.* Cisco Press. 2006.

[RUS-08] Russel, Rusty. *Linux 2.4 Paket Filtering HOWTO*. ftp://202.108.60.60/file/iptables-HOWTO.pdf. 2004.

[SAT-05] Satriaji, Galih. Domain Name Server dan Iptables. http://118.98.163.253/view.php?file=LINUX/dns_iptables.pdf . 2005.