MANAJEMEN FIREWALL SECARA TERPUSAT UNTUK SERVER BERBASIS WEB

SKRIPSI

Untuk memenuhi sebagian persyaratan untuk mencapai gelar Sarjana Komputer

`

Disusun oleh :

BAGUS AHMAD MAULIDA

NIM. 0910683024

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN

PROGRAM STUDI INFORMATIKA/ILMU KOMPUTER

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2014

LEMBAR PERSETUJUAN

MANAJEMEN FIREWALL SECARA TERPUSAT UNTUK SERVER BERBASIS WEB

SKRIPSI

Untuk memenuhi sebagian persyaratan untuk mencapai gelar Sarjana Komputer

Disusun oleh :

BAGUS AHMAD MAULIDA

NIM. 0910683024

Telah diperiksa dan disetujui:

Dosen Pembimbing I	Dosen Pembimbing II
Achmad Basuki, ST., M.MG., Ph.D	**Sabriansyah Rizqika Akbar, ST., M.Eng**
NIP. 19741118 2003 121 002	**NIK. 820809 06 1 1 0084**

LEMBAR PENGESAHAN

MANAJEMEN FIREWALL SECARA TERPUSAT UNTUK SERVER BERBASIS WEB

SKRIPSI

Untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh :

BAGUS AHMAD MAULIDA

NIM. 0910683024

Skripsi ini telah diuji dan dinyatakan lulus pada

7 Juli 2014

Penguji I	Penguji II
Aswin Suharsono, ST.,MT	**Kasyful Amron, ST.,M.Sc**
NIK. 840919 06 1 1 0251	**NIP.197508032003121003**

Penguji III

Aryo Pinandito, S.T., M.MT.
NIK. 83051916110374

Mengetahui

Ketua Program Studi Teknik Informatika/Ilmu Komputer

Drs. Marji, M.T.
NIP. 19670801 199203 1 001

PERNYATAAN ORISINALITAS SKRIPSI

Saya yang bertanda tangan dibawah ini menyatakan bahwa sepanjang pengetahuan saya, dalam naskah SKRIPSI ini tidak terdapat karya yang pernah diajukan oleh orang lain atau kelompok lain untuk memperoleh gelar akademis di suatu Institusi Pendidikan, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata di dalam naskah SKRIPSI ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia SKRIPSI ini digugurkan dan gelar akademik yang telah saya peroleh (SARJANA) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku. (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70)

Malang, 20 Juli 2014

Mahasiswa,

Bagus Ahmad Maulida

NIM. 0910683102

KATA PENGANTAR

Puji dan syukur Penulis panjatkan kehadirat Allah SWT, karena hanya dengan rahmat dan bimbingannya Penulis dapat menyelesaikan tugas akhir dengan judul "MANAJEMEN FIREWALL SECARA TERPUSAT UNTUK SERVER BERBASIS WEB" dengan baik. Tanpa rahmat dan bimbingan dari Tuhan Yang Maha Esa, maka niscaya Penulis tidak akan dapat menyelesaikan tugas akhir ini dengan baik dan tepat waktu.

Terima kasih pula penulis sampaikan kepada pihak-pihak yang telah membantu penulis dalam penyelesaian tugas akhir ini. Pihak-pihak tersebut antara lain:

1. Kedua orang tua tercinta Ibu Siti Ro'fatun dan Bapak Ainul Yaqin yang tidak lelah untuk mengajarkan, mengingatkan dan mendo'akan segala langkah yang diambil penulis.
2. Seluruh saudari-saudari tercinta Mbak Riyadinuna, Mbak Qurroti A'yunina dan Mbak Salisa Umi Fatih yang mendo'akan dan memotivasi penulis.
3. KH. Marzuki Mustamar, KH. Murtadho Amin, dan KH. Aziz Husein, yang telah turut mendo'akan dan memotivasi penulis.
4. Bapak Achmad Basuki, ST., M.MG., Ph.D dan Bapak Sabriansyah Rizqika Akbar, S.T., M.Eng. selaku dosen pembimbing yang dengan sabar telah membimbing penulis dalam pengerjaan penulisan skripsi.
5. Segenap dosen Program Studi Teknik Informatika/Ilmu Komputer Program Teknologi Informasi dan Ilmu Komputer yang telah mewariskan ilmunya.
6. Segenap staff dan pegawai Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya atas segala bantuan yang bersifat administratif.
7. Sahabat-sahabatku, Jayyid Fuadi Mubarrok, Ryan Nanda Perdana, Fengky Arga Pratama dan Muhammad Nurwiseso Wibisono yang telah memberikan masukan dan dorongan motivasi.
8. Seluruh teman-teman Ma'had Sabilurosyad, KOLAM, POROS, TPL 08, TPL 09, TIF 10 dan TIF 11 yang telah memberikan dukungan, dorongan semangat dan motivasi.

Akhirnya atas segala bantuan semua pihak semoga mendapat balasan yang setimpal dari Allah SWT. Harapan penulis semoga skripsi ini dapat memberikan manfaat kepada semua pihak, terutama kepada penulis dan para pengembang yang nantinya akan mengembangkan penelitian dari penulis.

Malang, 20 Juni 2014

Penulis

ABSTRAK

Bagus Ahmad Maulida. 2014. MANAJEMEN FIREWALL SECARA TERPUSAT UNTUK SERVER BERBASIS WEB. Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya. Dosen Pembimbing : Achmad Basuki, ST., MMG., Ph.D dan Sabriansyah Rizqika Akbar, ST., M.Eng.

Firewall merupakan keamanan mendasar yang ada di jaringan. Fungsi dari *firewall* yaitu untuk mencegah serangan dari pihak yang tidak memiliki otentikasi akses terhadap jaringan internal. Dalam *firewall,* pada dasarnya manajemen dilakukan secara mandiri atau melakukan manajemen secara langsung di setiap *server*. Permasalahan muncul ketika *server* yang dimanajemen lebih dari satu dan tersebar di berbagai tempat sehingga menyulitkan *administrator* di dalam melakukan manajemen *firewall*. Penelitian ini menawarkan sebuah purwarupa aplikasi manajemen *firewall* berbasis web yang dapat melakukan konfigurasi terhadap beberapa *module firewall* yang terdapat pada beberapa *server* secara langsung. Aplikasi manajemen *firewall* berbasis web yang dibangun, diinstall di dalam *server* pusat kontrol yang memanfaatkan protokol SSH untuk melakukan *remote node* dan mengirimkan *rule* konfigurasi menuju *server* yang dikonfigurasi. Berdasarkan hasil pengujian aplikasi di dalam penelitian ini, aplikasi yang dibangun dapat melakukan konfigurasi terhadap beberapa *module firewall* yang terdapat pada beberapa *server* secara langsung dan dapat mengurangi waktu yang dibutuhkan *administrator* di dalam melakukan konfigurasi *firewall*.

Kata Kunci: SSH, HTTPS, Basis Data, Terpusat, Aplikasi Berbasis Web

ABSTRACT

Bagus Ahmad Maulida. 2014. MANAJEMEN FIREWALL SECARA TERPUSAT UNTUK SERVER BERBASIS WEB. *Minor Thesis Informatics Engineering Program, Informatic Technology and Computer Science Program, University of Brawijaya*. *Advisor*: Achmad Basuki, ST., MMG., Ph.D and Sabriansyah Rizqika Akbar, ST., M.Eng.

Firewall is a basic security in the network. The function of firewall is to prevent attacks from those who do not have authentication to the internal network. Firewall management basically conducted independently or perform management directly on each server. Problems appear when the server has managed more than one and scattered in various places, making it difficult for administrators to manage firewall. This study offers a prototype of web-based firewall management application that can perform some configuration on the firewall module on many server directly. A web-based management application firewall installed on the central server and utilizes SSH protocol to perform remote and sends the rule to the configuration server. The results of application testing in this research showed that application can perform configuration to some firewall modules on many server directly and it can reduce time for administrator to configure firewall.

Keyword: SSH, HTTPS, Database, Centralized, Web-Based Application

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Firewall merupakan sistem keamanan mendasar jaringan yang berfungsi untuk mencegah serangan dari luar jaringan atau dari pihak yang tidak memiliki otentikasi akses [SSZ-05]. Di dalam *firewall* mekanisme perlindungan yang diterapkan mengasumsikan daerah yang dilindungi sebagai *safe zone* dan untuk daerah yang tidak dilindungi sebagai *unsafe zone*. Dengan melakukan pembagian tersebut, maka *firewall* mengasumsikan daerah *safe zone* merupakan daerah yang memiliki *otentikasi* akses terhadap jaringan *internal* sedangkan daerah *unsafe zone* merupakan daerah yang tidak memiliki otentikasi akses terhadap jaringan *internal*.

Dalam *firewall*, manajemen *firewall* yang dilakukan dapat melalui *console* atau menggunakan aplikasi manajemen *firewall*. Untuk melakukan manajemen *firewall* melalui *console*, *administrator* dapat langsung menuliskan sintak rule *firewall* melalui *console*. Sedangkan untuk melakukan manajemen *firewall* dengan menggunakan aplikasi manajemen, *administrator* hanya perlu memasukkan *source* dan *destination address*, *source* dan *destination port* dan protokol (icmp, udp dan, tcp) melalui *form* yang ada di dalam aplikasi.

Manajemen *firewall* pada dasarnya dilakukan secara mandiri atau melakukan manajemen secara langsung di setiap *server*. Tetapi, hal tersebut sulit dilakukan apabila topologi jaringannya sangat besar dan memiliki banyak *server* sehingga sulit bagi *administrator* dalam melakukan konfigurasi. Sedangkan, aplikasi manajemen *firewall* berbasis web yang ada masih belum dapat digunakan untuk melakukan konfigurasi *firewall* secara terpusat dan langsung mengirimkan *rule* ke seluruh *server*. Karena pada dasarnya aplikasi berbasis web yang ada saat ini hanya mendukung untuk konfigurasi *firewall* secara lokal *end device* seperti WEBMIN [WBM-13].

Penelitian terkait yang membahas tentang manajemen *firewall* adalah penelitian yang berjudul "menajemen *firewall* berbasis web" [MFW- 10]. Di dalam penelitian tersebut dijelaskan tentang perancangan dan implementasi purwarupa aplikasi manajemen *firewall* berbasis web yang memanfaatkan *module firewall* IPTABLES pada sistem operasi linux di dalam implementasi dan perancangannya. Tetapi, aplikasi purwarupa tersebut masih belum dapat digunakan untuk melakukan

konfigurasi terhadap beberapa *module firewall* lain seperti IPFW pada sistem operasi BSD dan ADVANCED WINDOWS FIREWALL pada sistem operasi windows yang terdapat pada beberapa *server* secara langsung. Oleh karena itu, penelitian ini menawarkan sebuah purwarupa aplikasi manajemen *firewall* berbasis web yang dapat melakukan konfigurasi terhadap beberapa *module firewall* yang terdapat pada beberapa *server* secara langsung. Aplikasi manajemen *firewall* berbasis web yang dibangun, diinstall di dalam server pusat kontrol yang memanfaatkan protokol SSH untuk melakukan *remote node* dan mengirimkan *rule* konfigurasi menuju *server* yang dikonfigurasi.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijabarkan, maka dirumuskan beberapa permasalahan:

1. Bagaimana desain dan implementasi manajemen *firewall* secara terpusat.
2. Bagaimana pengendalian pada *remote node* yang dilakukan untuk mengontrol beberapa *module firewall* secara terpusat dengan *system* yang *heterogen.*
3. Bagaimana kinerja sistem manajemen *firewall* secara terpusat.

1.3 Batasan Masalah

Berdasarkan uraian permasalahan diatas maka batasan masalahnya adalah sebagai berikut:

1. *Module firewall* yang digunakan adalah IPTABLES pada sistem operasi Fedora versi 18, IPFW pada sistem operasi FreeBSD versi 9.1, dan ADVANCED WINDOWS FIREWALL pada sistem operasi Windows Server 2012.
2. Input *rule* hanya dibatasi *source dan destination addrres, source* dan *destination port, interface, protocol* dan *target.*
3. Lingkungan penelitian dilakukan pada lingkungan *Local Area Network* (LAN).
4. Menggunakan *database* untuk menyimpan data *host, username, password yang* dienkripsi dan jenis *module firewall dari server* yang di-remote.
5. Jalur komunikasi di dalam melakukan kontrol firewall dilakukan via SSH.
6. Aplikasi manajemen firewall berbasis web.

1.4 Tujuan

Membuat sebuah purwarupa aplikasi untuk mengontrol *firewall* secara terpusat, sehingga memudahkan seorang *administrator* dalam mengelola *server* yang jumlahnya lebih dari satu.

1.5 Manfaat

1. Menyediakan aplikasi manajemen *firewall* secara terpusat yang berbasis web sehingga *administrator* tidak perlu melakukan *input rule* untuk masing-masing *server* via *console*.
2. Mengurangi waktu konfigurasi yang dibutuhkan *administrator* dalam mengontrol *server* yang lebih dari satu.

1.6 Sistematika Penulisan

Sistematika penulisan dalam skripsi ini sebagai berikut:

BAB I PENDAHULUAN

Memuat latar belakang permasalahan, identifikasi dan pembatasan masalah, rumusan masalah, tujuan, manfaat, dan sistematika penulisan.

BAB II KAJIAN PUSTAKA DAN DASAR TEORI

Menguraikan semua teori dasar dan teori penunjang yang berkaitan dengan penelitian.

BAB III METODOLOGI PENELITIAN DAN PERANCANGAN

Membahas langkah-langkah yang dilakukan dalam menyelesaikan penelitian dan pejabaran perancangan dari sistem yang akan dibangun, langkah-langkah yang dilakukan terdiri dari studi literatur, perancangan, implementasi, pengujian, dan analisis, serta pengambilan kesimpulan dan saran.

BAB IV IMPLEMENTASI

Membahas langkah-langkah implementasi yang dilakukan dalam pembuatan aplikasi. Dalam bab ini juga disertakan gambar dan diagram blok yang menggambarkan tentang implementasi yang dilakukan

BAB V PENGUJIAN DAN ANALISIS

Memuat hasil pengujian dan analisis dari hasil pengujian terhadap sistem yang telah dibangun.

BAB VI PENUTUP

Memuat kesimpulan serta saran yang diperoleh dari pembuatan dan pengujian metode untuk pengembangan lebih lanjut.

BAB II
KAJIAN PUSTAKA DAN DASAR TEORI

2.1 Penelitian Terkait

Penelitian terkait tentang manajemen *firewall* yang dijadikan acuan adalah penelitian yang dilakukan oleh Rosyadi A Ilmawan [MFW-10] dengan judul "Manajemen Firewall Berbasis Web". Di dalam penelitian tersebut dijelaskan tentang perancangan dan implementasi purwarupa aplikasi manajemen *firewall* berbasis web yang memanfaatkan *module firewall* IPTABLES di dalam implementasi dan perancangannya. Penelitian lain yang dijadikan acuan adalah penelitian yang dilakukan oleh Steve M Bellovin [IDF-00] dengan judul " *Distributed Firewall* ". Di dalam penelitian tersebut dijelaskan *distributed firewall* adalah sebuah konsep tentang distribusi kebijakan *firewall* untuk setiap *node* yang tersambung ke dalam kelompok *distributed firewall. Distributed firewall* dapat diimplementasikan apabila *node* yang tersambung di dalam kelompok *distributed firewall* dipercaya oleh pusat kontrol dari *distributed firewall*, Sehingga perlu adanya *keynote trust management* untuk menjamin *node* yang tersambung dipercaya di dalam kelompok *distributed firewall*. Setelah node yang tersambung dipercaya maka *policy* akan didistribusikan ke setiap *endpoint* [IDF-00].

2.2 Firewall

Firewall adalah sistem keamanan mendasar yang ada di *network*, di dalam prinsip kerjanya *firewall* membagi daerah yang dilindunginya menjadi dua bagian yaitu daerah yang dipercaya dan daerah yang tidak dipercaya. Pembagian tersebut dilakukan untuk memberikan otentikasi terhadap paket data yang melalui *firewall* sehingga *firewall* dapat melakukan pecegahan terhadap paket data yang tidak dipercaya dan memberikan otentikasi terhadap paket data yang dipercaya [SSZ-05]. Secara umum fungsi dasar dari *firewall* yaitu paket filtering, paket filtering yaitu melakukan pemeriksaan terhadap seluruh paket yang melalui *firewall* dan melakukan penyaringan berdasarkan *rule* yang telah didefinisikan.

2.2.1 Filtering

Filtering adalah sebuah tindakan yang dilakukan oleh *firewall* untuk menyaring paket berdasarkan *rule* atau aturan yang telah dideskripsikan di dalam

firewall rule. Deskripsi rule di dalam *firewall rule* akan berbeda pada setiap *module firewall*, untuk *module firewall* IPTABLES disetiap *table* yang ada mendefinisikan banyak *chain*, disetiap *chain* memiliki *list rule* untuk tujuan tertentu [SSZ-05], untuk *module firewall* IPFW di dalam mendifinisikan *rule* langsung didefinisikan berdasarkan *rule number* yang memiliki *list rule* untuk tujuan tertentu [BSD-13] dan untuk *module firewall* Advanced Windows Firewall di dalam mendifinisikan *rule* dibagi dalam dua kelompok yaitu *rule* untuk komunikasi ke dalam (*inbound*) dan kemunikasi keluar (*outbound*), di dalam kelompok tersebut terdapat *list rule* untuk tujuan tertentu [NSH-10].

2.3 Manajemen Secara Terpusat di Jaringan

Manajemen secara terpusat adalah sebuah konsep yang menerapkan tentang pemusatan kontrol terhadap sebuah *end device* kontrol. Di dalam manajemen terpusat, *end device* yang menjadi pusat kontrol dapat melakukan kontrol terhadap *end device* yang terhubung dengan pusat, sehingga memudahkan *administrator* di dalam melakukan konfigurasi [CNN-13]. Manajemen terpusat sering diimplementasikan terhadap *network* yang memiliki topologi besar atau *company* yang memiliki banyak *department*, hal tersebut dilakukan untuk mengurangi biaya, waktu dan tenaga di dalam melakukan kontrol.

Gambar 2.1 Topologi Manajemen Terpusat

2.4 Remote Node

Remote node merupakan tindakan yang dilakukan untuk mengontrol dan mengakses *resource* lain secara tidak langsung. Salah satu *tools* yang digunakan untuk melakukan *remote node* adalah openssh, openssh merupakan *tools* yang digunakan untuk melakukan komunikasi terhadap *resource* lain secara melalui protokol SSH.

Dengan memanfaatkan SSH maka setiap data yang melalui jalur komunikasi akan dienkripsi, sehingga mencegah terjadinya pencurian data selama proses komunikasi berlangsung [SSH-13].

Berbeda dengan linux yang di dalam sistem operasinya sudah terinstall openssh, di dalam windows dibutuhkan sebuah *tools* untuk melakukan *remote node* secara aman ke dalam sistem windows salah satu *tools* yang digunakan adalah Bitvise SSH. Bitvise SSH merupakan *tools ssh server* maupun *client* yang memanfaatkan protokol SSH dan di desain untuk sistem operasi windows (*Desktop* dan *Server*) [BVS-13].

Di dalam PHP juga mendukung untuk malakukan SSH dengan memanfaatkan *extension ssh2* dimana prinsip kerja dari extension tersebut adalah mengikat libssh2 yang terinstall di dalam linux, sehingga dengan memanfaatkan *extension* tersebut php dapat melakukan *access* secara SSH terhadap *resource* lain [LSH-13].

2.5 Pemrograman Web

Pemrograman web adalah proses penulisan *sourcode* di dalam sebuah *web development*, yang meliputi pembuatan konten web, pemrograman berbasis *client server* dan keamanan jaringan. Bahasa pemrograman yang digunakan untuk pemrograman web yaitu, XML, HTML, Javascript, Perl dan PHP.Pemrograman web berbeda dengan pemrograman biasa, karena di dalam pemrograman web diperlukan pengetahuan tentang aplikasi, pemrograman berbasis client - server dan basis data [WBP-14].

2.6 Web Service

Web service merupakan software yang dapat digunakan untuk membangun aplikasi berbasis web sehingga aplikasi yang dibangun dapat diakses melalui jaringan. *Web service* tidak terikat dengan sistem operasi dan bahasa pemrograman sehingga memungkinkan untuk membangun sebuah aplikasi yang *multiplatform* [WBS-14].

Contoh *web service* adalah *Apache web service, Apache web service* merupakan aplikasi *web service* gratis yang didistribusikan oleh *Apache Software Foundation* yang memiliki fitur diantaranya CGI, SSL dan Vrtual Domain [APC-13].

2.7 Pengolah Data

Pengolahan data merupakan proses pengolahan dan memanipulasi data. Dimana, di dalam pengolahan dan memanipulasi data diperlukan sebuah database untuk menyimpan data. *Database management* yang sering digunakan adalah MySQL. MySQL merupakan database penyimpan data yang ringan dan mudah digunakan serta mendukung untuk penggunaan database yang sedikit ataupun banyak [SQL-13]. Kelebihan yang ada di dalam MySQL yaitu:

1. Mendukung berbagai macam bahasa pemrograman (php, java).
2. Dapat digunakan untuk aplikasi yang berbasis *client - server* atau *embedded system*.
3. *Relieble, scalable,* mudah digunakan dan cepat.

2.8 Keamanan Jaringan

Keamanan jaringan adalah sebuah tindakan yang dilakukan *administrator* di dalam mengamankan sebuah jaringan dari pengguna yang tidak memiliki otentikasi akses ke dalam jaringan [NSC-14]. Bermacam-macam metode dapat digunakan untuk mengamankan jaringan, salah satu metode yang umum digunakan adalah enkripsi sedangkan untuk aplikasi berbasis web metode pengamanan yang umum digunakan adalah menerapkan HTTPS/SSL.

2.8.1 Keamanan Data

Pengamanan data adalah sebuah tindakan yang dilakukan administrator untuk mencegah *hacker* untuk mencuri data dari sebuah sistem. Banyak metode yang diterapkan dalam pengamanan data salah satunya adalah dengan menggunakan enkripsi data. Enkripsi adalah metode yang digunakan untuk pengacakan data yang berfungsi untuk mencegah penggunaan data oleh seorang *hacker*. Jenis enkripsi yang umum digunakan adalah sha1.

Sha1 atau singkatan dari *secure hash algoritm* yaitu sebuah algoritma enkripsi yang digunakan untuk mengacak *password* menjadi data-data acak yang sulit untuk dibaca secara langsung, sehingga perlu melakukan proses dekripsi untuk mengembalikan data yang diacak agar data tersebut bisa dibaca lagi [SHA-10]. Untuk lebih mengamankan sebuah data biasanya selain menggunakan sebuah enkripsi *adminitrator* juga menerapkan metode *salt* pada data yang dienkripsi, *sal*t merupakan sebuah metode *cryptography* yang berfungsi untuk menambahkan *string* kedalam

sebuah data sehingga menyulitkan *hacker* untuk menemukan data yang sesungguhnya.

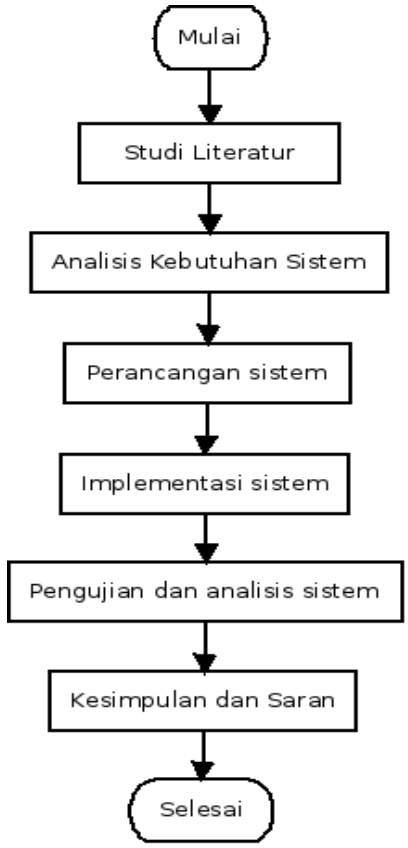
2.8.2 Keamanan Aplikasi Berbasis Web

Aplikasi berbasis web adalah salah satu komponen yang memiliki hubungan erat dengan jaringan, apabila jaringan bekerja di lingkungan back end maka aplikasi berbasis web bekerja di lingkungan front end. Karena berkaitan erat dengan jaringan maka aplikasi berbasis web sangat rentan terhadap *hacking* atau serangan, sehingga dalam implementasi aplikasi berbasis web perlu diterapkan sebuah pengamanan.

Untuk aplikasi berbasis web biasanya menerapkan SSL di dalam pengamananya. SSL atau lebih dikenal dengan HTTPS merupakan akronim dari s*ecure socket layer* yaitu sebuah protokol enkripsi data yang mengamankan jalur komunikasi network [SSL-13].

BAB III
METODE PENELITIAN DAN PERANCANGAN

Bab ini menjelaskan langkah-langkah yang dilakukan dalam menyelesaikan keseluruhan penelitian. Untuk urutan langkah yang akan dilakukan dimulai dengan studi literatur, analisis kebutuhan sistem, perancangan sistem, implementasi sistem, pengujian sistem, analisis sistem dan terakhir penarikan kesimpulan serta pemberian saran sebagai acuan untuk pengembangan sistem selanjutnya. Diagram alir penelitian ditunjukan pada Gambar 3.1 berikut:



Gambar 3.1 Alur Metode Penelitian

3.1 Studi Literatur

Studi literatur dilakukan untuk mendapatkan teori pendukung dari permasalahan yang dikaji, teori pendukung tersebut diperoleh dari dasar teori, kajian pustaka dan informasi yang tersedia di internet baik itu dokumentasi *project* maupun artikel. Studi literatur yang diperlukan meliputi:

1. Firewall
    - Filtering
2. Manajemen secara terpusat di jaringan
3. Remote Node
4. Pemrograman web
5. Web Service
6. Pengolah Data
7. Keamanan Jaringan
    - Keamanan Data
    - Keamanan Aplikasi berbasis web

3.2 Analisis Kebutuhan Sistem

Analisis kebutuhan sistem diperlukan untuk menentukan kebutuhan sistem yang akan diterapkan di dalam aplikasi *manajemen firewall secara terpusat*, sehingga aplikasi ini bisa berjalan dengan baik. Analisa kebutuhan sistem tersebut meliputi:

3.2.1 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional adalah analisis kebutuhan yang mendukung dalam membangun aplikasi manajemen *firewall* secara terpusat. Analisis kebutuhan fungsional meliputi:

1. Sistem dapat melakukan otentikasi terhadap *administrator* yang melakukan *login* di dalam aplikasi.
2. Sistem dapat melakukan *remote* dengan memanfaatkan SSH terhadap *end device*.
3. Sistem dapat melakukan konfigurasi, menampilkan dan menghapus *rule* untuk masing-masing jenis *module firewall*.
4. Sistem dapat menambahkan dan menghapus *host* yang di-remote dan dikonfigurasi.
5. Sistem dapat menjaga kerahasiaan data dari setiap *host* yang terhubung dengan aplikasi.
6. Sistem dapat menyimpan data dari setiap *host* yang terhubung dengan aplikasi.
7. Sistem dapat mengamankan aplikasi dari serangan.

3.2.2 Analisis Lingkungan Sistem

Analisis lingkungan sistem adalah analisis kebutuhan perangkat lunak dan keras yang mendukung dalam membangun aplikasi manajemen *firewall* secara terpusat. Analisis lingkungan sistem meliputi:

1. **Perangkat Keras**

   Perangkat keras yang dibutuhkan dalam melakukan implementasi sistem meliputi:

   a. Satu unit PC/Laptop yang memiliki *browser* untuk mengakses aplikasi.
   b. Satu unit PC yang digunakan untuk menginstall *proxmox* dan memiliki spesifikasi sebagai berikut:

   - CPU: Intel Core i3 522, 3.2 Ghz
   - RAM: 2 GB
   - Harddisk: 500 GB

2. **Perangkat Lunak**

   Agar sistem dapat berjalan dengan baik, maka dibutuhkan beberapa perangkat lunak yang akan diinstal diantaranya:

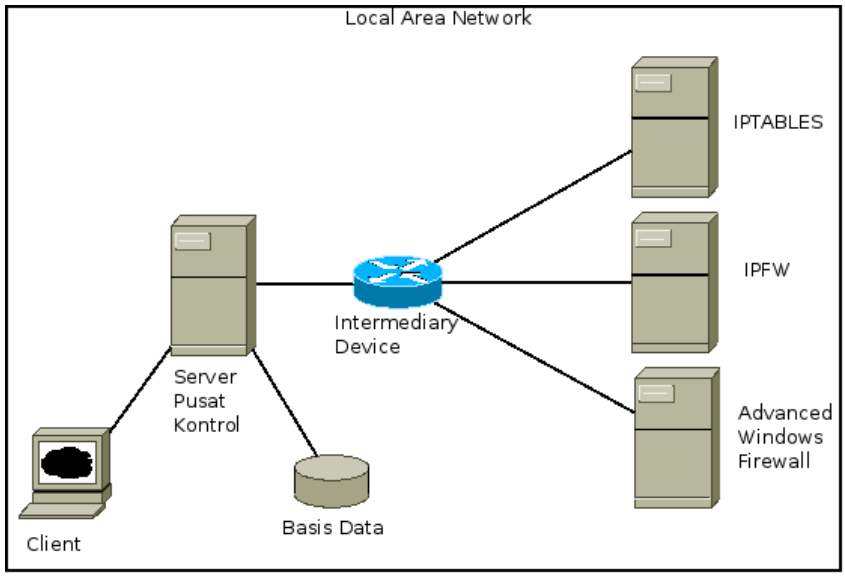
   a. Sistem operasi *proxmox* yang berfungsi untuk *virtualization*
   b. Tiga sistem operasi yang memiliki *module firewall* berbeda terinstall secara *virtual*.
   c. Apache HTTP Server sebagai penyedia *web service*.
   d. Modul bahasa pemrograman PHP versi 5.4.18.
   e. Database MySQL untuk penyimpanan data *host* yang di-remote.
   f. Modul SSL sebagai pengamanan untuk aplikasi.
   g. *Tools* ssh *server* untuk windows *server* yang di-remote.

3.3 Perancangan Sistem

Perancangan dilakukan untuk merancang sistem yang akan dibangun. Perancangan yang dilakukan disesuaikan dengan kebutuhan sistem yang telah didapatkan. Perancangan sistem meliputi:

3.3.1 Perancangan Arsitektur Sistem

Skenario perancangan arsitektur sistem pada Gambar 3.2 menunjukkan bahwa aplikasi manajemen *firewall* yang akan dibangun diimplementasikan di dalam *server* yang menjadi pusat kontrol dan melakukan *remote* terhadap tiga macam sistem operasi yang memiliki *module firewall* berbeda. Masing-masing dari *module firewall* tersebut adalah IPTABLES, ADVANCED WINDOWS FIREWALL dan IPFW.



Gambar 3.2 Arsitektur Sistem

Untuk melakukan konfigurasi *firewall*, *administrator* hanya perlu mengakses aplikasi melalui browser dari PC yang tersambung dengan *server* pusat kontrol dan melakukan *input host* tujuan, *source* dan *destination address*, *source* dan *destination port* dan *protocol* (icmp, udp dan, tcp) di dalam *form* yang disediakan aplikasi. Selanjutnya, aplikasi akan mengirimkan rule yang telah dimasukkan ke seluruh server konfigurasi melalui SSH dengan menggunakan data dari server konfigurasi yang tersimpan di dalam basis data. Sedangkan *user* yang dapat digunakan untuk *log-in* ke dalam aplikasi adalah *user* yang terdapat pada server pusat kontrol.

3.3.2 Perancangan Back End

Pada bagian ini dijelaskan perancangan *back end* yang akan diterapkan disesuaikan dengan kebutuhan fungsional sistem dan juga kebutuhan mendasar yang digunakan untuk membangun sistem manajemen firewall secara terpusat. Perancangan *back end* yang diterapkan yaitu :

3.3.2.1 Perancangan Remote node

Perancangan *remote node* diperlukan untuk merancang jalur komunikasi yang digunakan di dalam melakukan kontrol *firewall*. Jalur komunikasi yang dirancang memanfaatkan SSH sebagai sarana komunikasinya sehingga aplikasi manajemen *firewall* secara terpusat dapat melakukan *remote node* terhadap *server* yang dikontrol. Untuk melakukan SSH terhadap *server* yang dikontrol diperlukan *tools* atau paket *software* yang ditambahkan. *Tools* atau paket *sotfware* tersebut yaitu:

1. LIBSSH2 sebagai *library* yang berfungsi untuk melakukan SSH ke *client*.
2. PHP-SSH2 yang berfungsi untuk mengikat libssh2 ke dalam php sehingga php dapat melakukan *remote node* secara SSH terhadap *server* yang dikontrol.
3. Bitvise SSH Server yang berfungsi sebagai layanan *ssh server* untuk windows sehingga windows dapat di-remote melalui *ssh*.

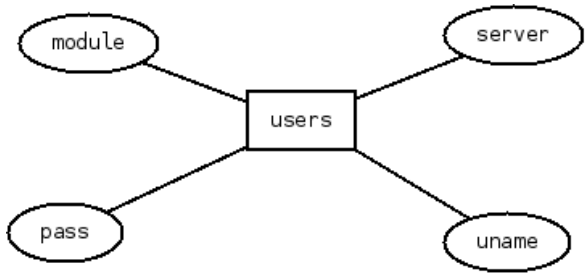
3.3.2.2 Perancangan Keamanan sistem

Perancangan keamanan diperlukan untuk menentukan model pengamanan yang diperlukan untuk mengamankan aplikasi dari *hacking*, perancangan keamanan dilakukan di dua lokasi, yaitu pengamanan untuk aplikasi web dan pengamanan data yang tersimpan di dalam basis data. Perancangan kemanan yang akan diimplementasikan yaitu:

1. Konfigurasi SSL/HTTP yang berfungsi untuk mengamankan aplikasi berbasis web.
2. Enkripsi menggunakan sha1 dan salt yang berfungsi untuk melakukan enkripsi data dari *host* yang tersimpan di dalam basis data.

3.3.2.3 Perancangan Basis data

Perancangan basis data dilakukan untuk merancang model basis data yang digunakan di dalam aplikasi manajemen firewall secara terpusat. Fungsi dari basis data yang dirancang adalah untuk meyimpan data *host, username, password* dan jenis *module firewall* dari *server* yang dikontrol dan model basis data yang akan dibangun tidak memiliki relasi dengan entitas lain. Secara entitas basis data yang dirancang seperti terlihat dalam gambar berikut:



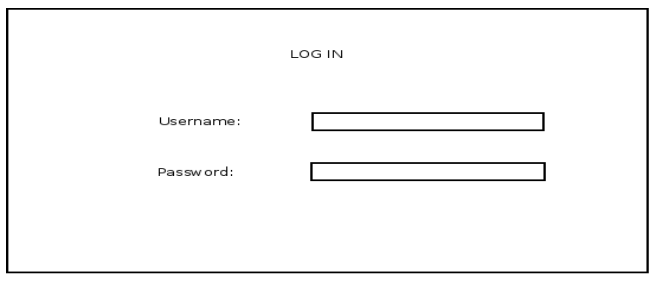
Gambar 3.3 Eentitas Basis Data Aplikasi

Di dalam basis data *users* yang dirancang, terdapat empat macam *attribute* yaitu *server* yang berfungsi untuk menyimpan *host* dari *server* yang dikontrol, *uname* yang berfungsi untuk menyimpan *username* dari *host* yang dikonfigurasi, *pass* yang berfungsi untuk menyimpan enkripsi *password* dari *host* yang dikonfigurasi dan *module* yang berfungsi untuk menyimpan jenis *module* dari *host* yang dikonfigurasi.

3.3.3 Perancangan Front End

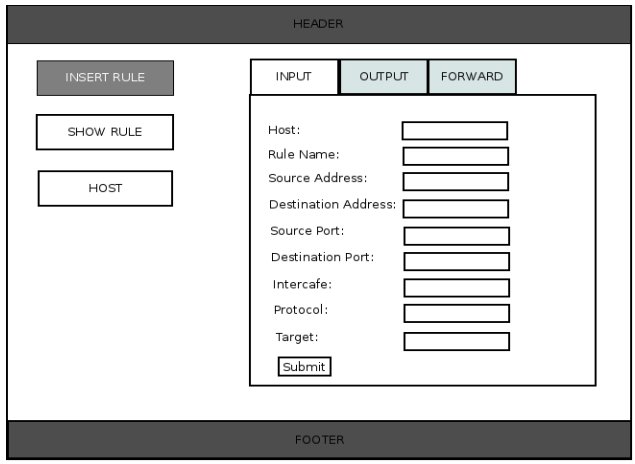
Perancangan *front end* dilakukan untuk merancang antar muka dari aplikasi manajemen *firewall* secara terpusat. Antar muka yang dirancang, berfungsi untuk memudahkan *administrator* di dalam melakukan konfigurasi terhadap *server* yang dikontrol. Berikut perancangan antar muka dari aplikasi manajemen *firewall* secara terpusat:

a) Halaman Login



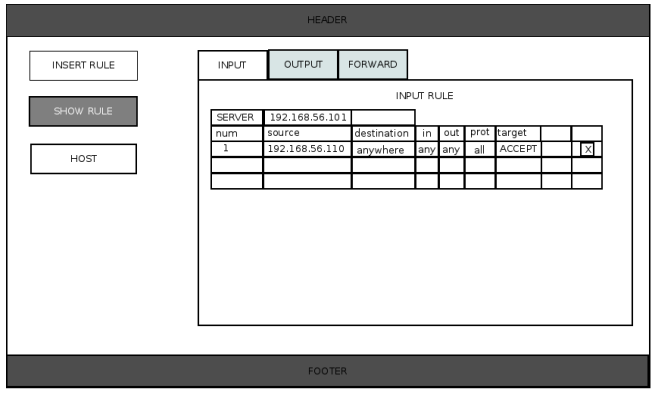
Gambar 3.4 Perancangan antar muka halaman login

b) Halaman Insert Rule

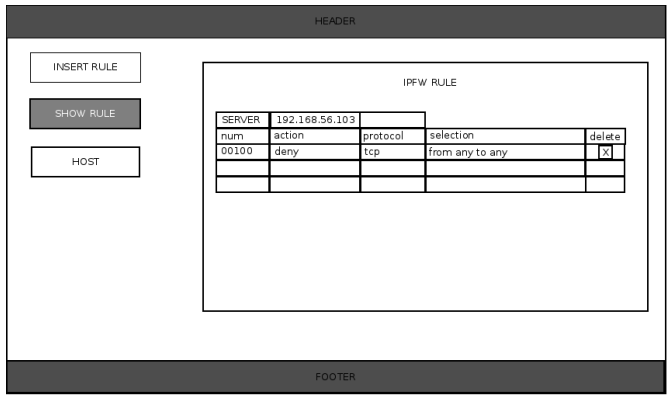


Gambar 3.5 Perancangan antar muka halaman insert rule

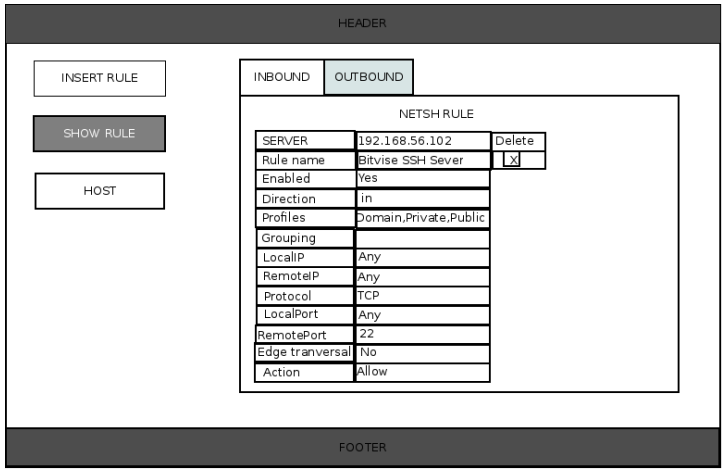
c) Halaman Show Rule



Gambar 3.6 Perancangan antar muka halaman show rule (IPTABLES)

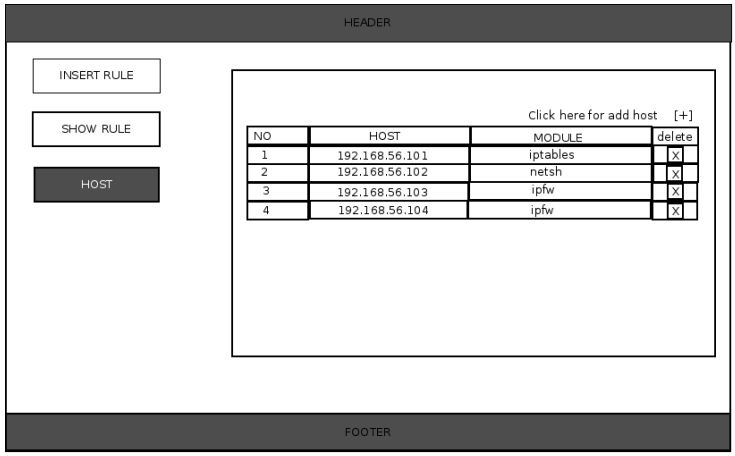


Gambar 3.7 Perancangan antar muka halaman show rule (IPFW)

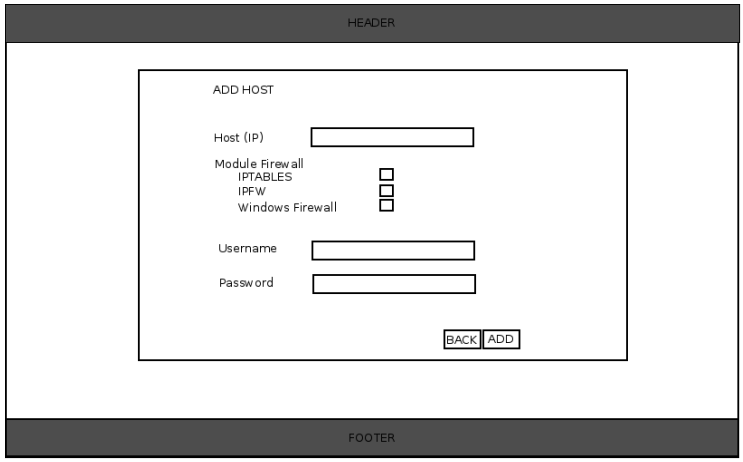


Gambar 3.8 Perancangan antar muka halaman show rule (NETSH)

d) Halaman Host



Gambar 3.9 Perancangan antar muka halaman host

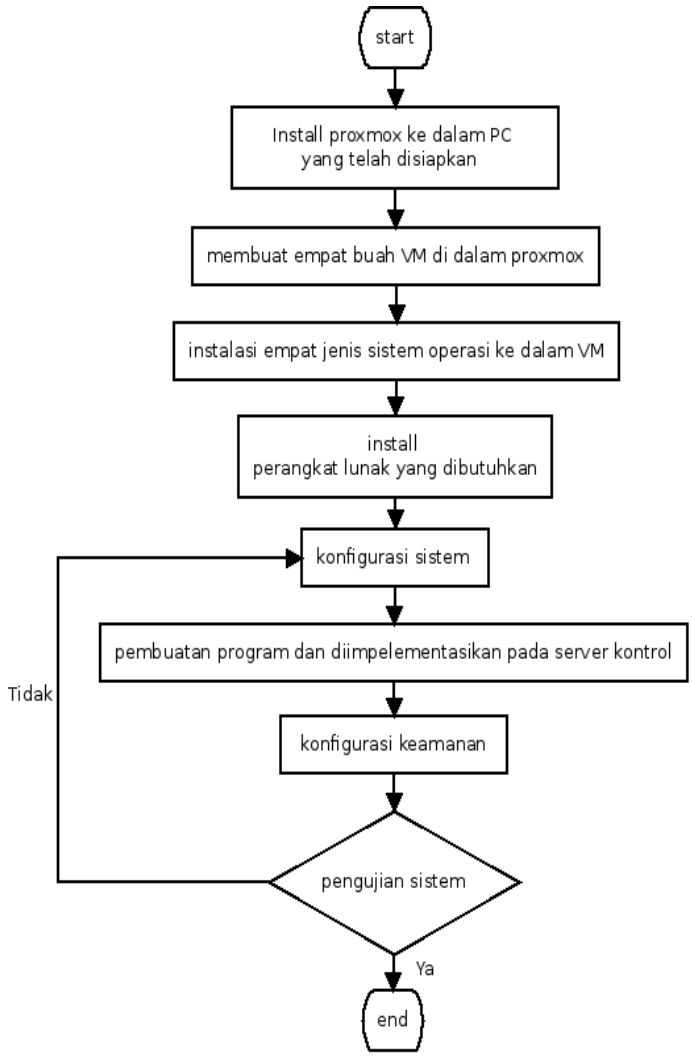


Gambar 3.10 Perancangan antar muka halaman menambahkan host

3.4 Implementasi

Implementasi dilakukan dengan membangun aplikasi yang sebelumnya telah dirancang. Implementasi sistem dibangun dengan menggunakan bahasa pemrograman PHP dan diterapkan pada topologi sistem yang dirancang secara local area network (LAN).

Di dalam topologi LAN yang dibangun, terdapat tiga *server* dan sebuah *server* kontrol yang dibangun secara *virtual* dengan menggunakan *proxmox* sebagai media *virtualization*. Aplikasi yang telah dibangun diinstal ke dalam *server* kontrol dan untuk melakukan konfigurasi *firewall*, *administrator* hanya perlu mengakses aplikasi dari *server* kontrol. Selanjutnya, aplikasi akan melakukan konfigurasi secara *remote node* terhadap ketiga *server* yang memiliki *module firewall* yang berbeda-beda. Langkah-langkah implementasi yang dilakukan yaitu:



Gambar 3.11 Alur implementasi sistem

3.5 Pengujian dan Analisis

Pengujian terhadap purwarupa aplikasi manajemen firewall secara terpusat dilakukan untuk mengetahui sistem yang dibangun telah sesuai dengan spesifikasi sistem yang telah dirancang. Pengujian dilakukan dalam dua tahap:

1. Pengujian kebutuhan fungsional yang dilakukan untuk mengetahui validitas dari masing-masing fitur yang ada di dalam aplikasi.

2. Pengujian kebutuhan non-fungsional yang dilakukan untuk mengetahui perbandingan waktu yang dibutuhkan ketika melakukan *insert rule* menggunakan aplikasi manajemen *firewall* secara terpusat dan konfigurasi secara manual.

Setelah sistem diuji, dilakukan analisis untuk setiap hasil pengujian sehingga dapat diketahui tingkah laku dari sistem yang telah dibangun.

3.6 Kesimpulan dan Saran

Pengambilan kesimpulan dilakukan setelah semua tahapan perancangan, implementasi, dan pengujian sistem aplikasi telah selesai dilakukan. Kesimpulan diambil dari hasil pengujian dan analisis terhadap sistem yang dibangun. Tahap terakhir dari penulisan adalah saran yang dimaksudkan untuk memperbaiki kesalahan-kesalahan yang terjadi dan menyempurnakan penulisan serta memberikan pertimbangan atas pengembangan sistem selanjutnya.

BAB IV
IMPLEMENTASI

Pada bab ini dijelaskan langkah-langkah yang dilakukan dalam implementasi sistem. Berikut adalah langkah-langkah implementasi sistem yang dibangun, terdiri dari lingkungan implementasi, batasan implementasi, implementasi *back end* dan implementasi *front end.*

4.1 Lingkungan Implementasi

Aplikasi manajemen *firewall* secara terpusat di dalam implementasinya dibangun dengan menggunakan bahasa pemrograman PHP versi 5.4.16, untuk penyimpanan data dari *host* yang dikonfigurasi menggunakan *database* MySQL versi 5.0.10 dan untuk keamanan aplikasi menerapkan enkripsi data dan SSL.

Dalam implementasinya, aplikasi manajemen *firewall* secara terpusat membutuhkan tiga buah server untuk di-remote dan sebuah server yang menjadi kontrol untuk menginstall aplikasi. Di dalam masing-masing server tersebut terinstall sistem operasi Fedora versi 18, FreeBSD versi 9.1 dan Windows Server 2012 yang terinstall secara *virtual* di dalam *proxmox*. Dan untuk melakukan *remote* dan konfigurasi terhadap ketiga sistem operasi tersebut digunakan protokol SSH yang terhubung dengan PHP yang diimplementasikan ke dalam baris kode aplikasi.

4.2 Batasan Implementasi

Implementasi aplikasi diterapakan di dalam sistem operasi Fedora 18 yang memanfaatkan protokol SSH dan menghubungkan antara PHP dengan libssh2, untuk implementasi di dalam sistem operasi windows masih belum mendukung karena di dalam windows tidak terdapat libssh2.

4.3 Implementasi Back End

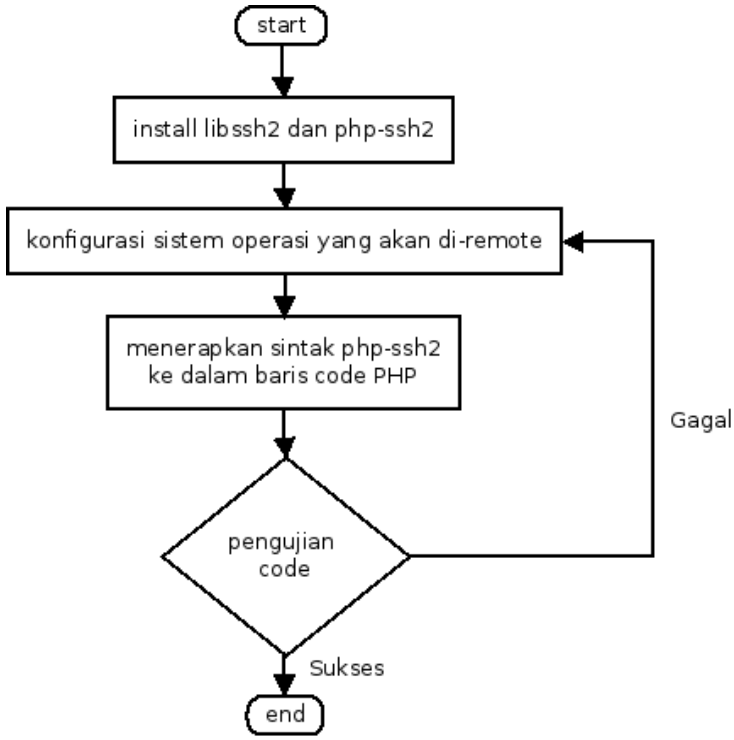
Pada bagian ini dijelaskan implementasi *back end* yang diterapkan disesuaikan dengan kebutuhan fungsional sistem dan juga kebutuhan mendasar yang digunakan untuk membangun sistem manajemen *firewall* secara terpusat. Implementasi *back end* yang diterapkan yaitu:

4.3.1 Implementasi Server Virtual

Langkah awal yang perlu dilakukan di dalam membangun aplikasi manajemen *firewall* secara terpusat yaitu melakukan instalasi sistem operasi *proxmox* ke dalam PC yang digunakan sebagai media *virtualization*. Langkah selanjutya yaitu membuat empat buah *virtual mechine* di dalam *proxmox* yang masing-masing *virtual mechine* tersebut diinstal tiga macam sistem operasi yang memiliki *module firewall* berbeda dan salah satu dari *virtual mechine* tersebut akan digunakan sebagai server kontrol untuk menginstall aplikasi. Untuk mendapatkan IP, *interface network* pada masing-masing *virtual mechine* diatur *bridge*.

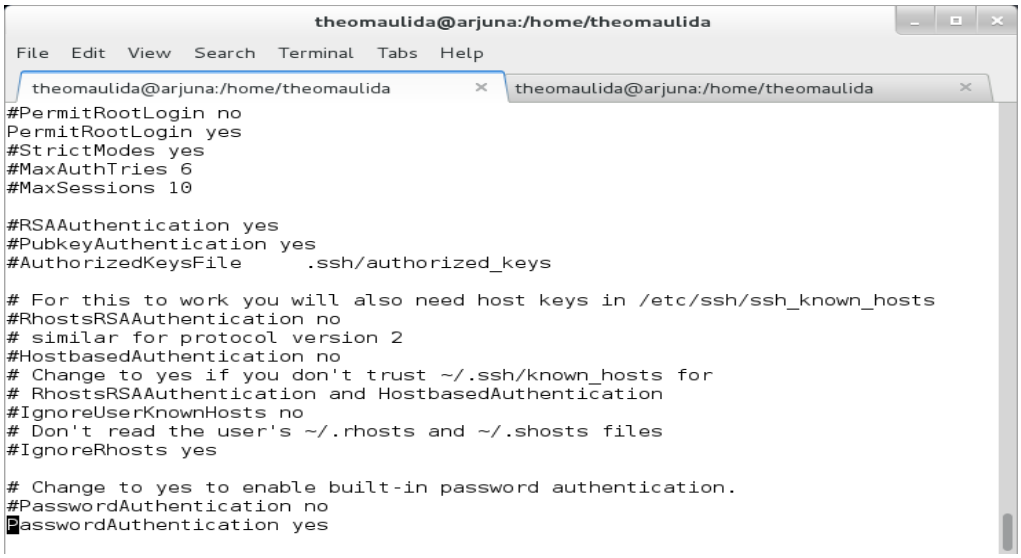
4.3.2 Implementasi Remote Node

Implementasi *remote node* diperlukan untuk memenuhi kebutuhan fungsional aplikasi supaya dapat melakukan *remote* terhadap *server* yang akan dikontrol *rule* firewallnya *remote node* yang diterapkan mengunakan libssh2 yang diikat menggunakan php-ssh2, sehingga PHP dapat melakukan *remote* tehadap *host* yang dikontrol. Secara diagram alir implementasi *remote node* yang dilakukan yaitu:

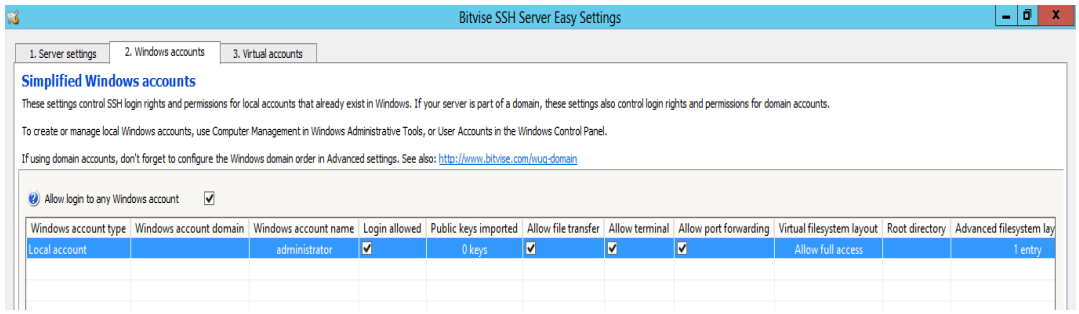


Gambar 4.1 Alur Implementasi Remote Node

Di dalam melakukan *remote node*, aplikasi memerlukan hak akses *root* atau *administrator* untuk melakukan manajemen *firewall* terhadap sistem operasi yang dikontrol. Untuk sistem operasi openBSD perlu adanya konfigurasi sistem yang berada di dalam file /etc/ssh/sshd_config sehingga sistem operasi openBSD dapat di-remote menggunakan akses *root* sedangkan untuk sistem operasi windows perlu adanya *software* tambahan berupa *ssh server*. Berikut konfigurasi yang dilakukan di dalam sistem operasi openBSD dan sistem operasi windows:



Gambar 4.2 Konfigurasi ssh pada openBSD



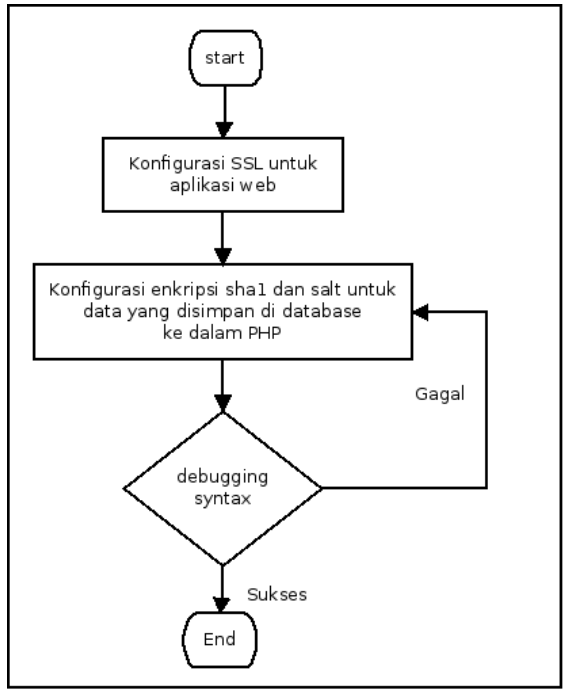
Gambar 4.3 Konfigurasi ssh pada windows

Sedangkan untuk implementasi di dalam baris kode program perlu adanya sintak php-ssh2 yang diimplementasikan. Sehingga, aplikasi yang dibangun dapat melakukan *remote node* terhadap sistem operasi yang dikontrol. Berikut salah satu contoh program yang menerapkan sintak php-ssh2:

```
$con = ssh2_connect( "$i[0]", 22);
ssh2_auth_password($con, "$i[1]", "$decrypted");
IptablesInput($con,$src,$dst,$sport,$dport,$eth,$pro,
$target,$ethWin );
```

4.3.3 Implementasi Keamanan

Implementasi keamanan diperlukan untuk mengamankan aplikasi dari *hacking*, untuk implementasi kemanan diterapkan di dua lokasi, yaitu pengamanan untuk aplikasi web yang menggunakan HTTPS/SSL dan pengamanan data di dalam basis data yang menggunakan enkripsi dengan menggunakan metode *salt* dan sha1. Secara diagram alir implementasi Keamanan yang diterapkan yaitu:

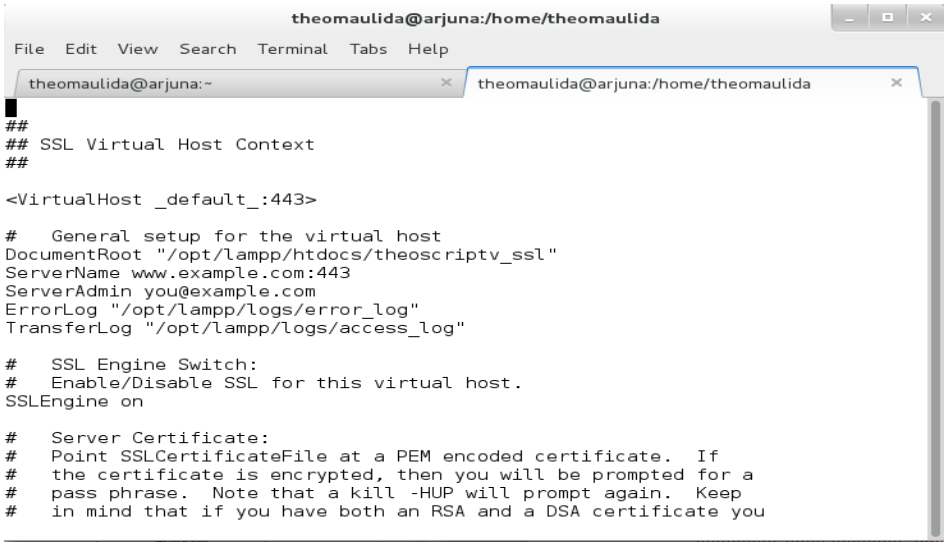


Gambar 4.4 Alur Implementasi Keamanan

Dalam melakukan implementasi HTTPS/SSL dan enkripsi data perlu adanya konfigurasi yang dilakukan di dalam aplikasi yang dibangun. Berikut perintah konfigurasi yang dilakukan untuk implementasi HTTPS/SSL:

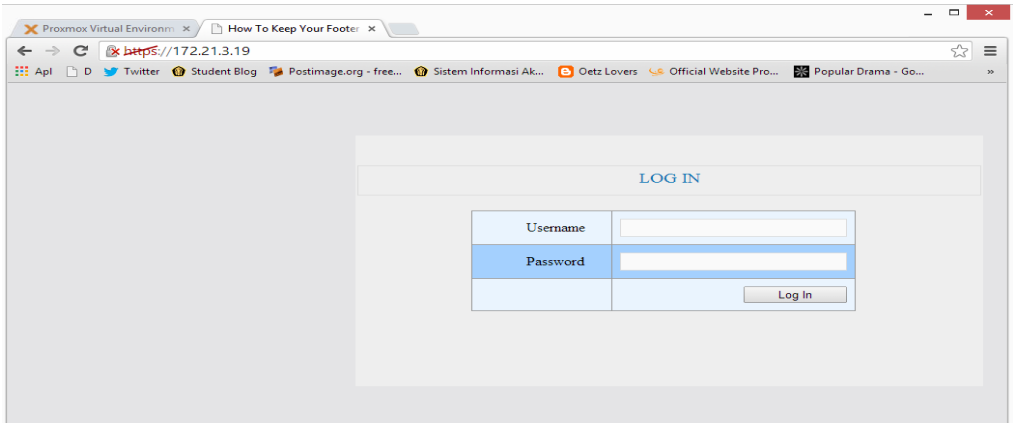
```
$ openssl genrsa -des3 -out server.key 1024
$ openssl req -new -key server.key -out server.csr
$ openssl rsa -in server.key -out server.key
$ openssl rsa -in server.key -out server.pem
$ openssl  x509 -req  -in server.csr -signkey server.pem
-out server.crt
$ sudo cp server.pem server.crt /opt/lampp
$ sudo cp server.pem server.crt /opt/lampp
# chmod 600 /opt/lampp/server.crt server.pem
```

Berikut konfigurasi *virtual host* untuk SSL yang berada di dalam file /opt/lampp/etc/extra/httpd-ssl.conf:



Gambar 4.5 Konfigurasi *virtual host* untuk SSL

Berikut hasil implementasi HTTPS/SSL ketika diakses melalui *web browser*:



Gambar 4.6 Hasil implementasi SSL ketika diakses web browser

Berikut salah satu contoh kode program yang digunakan untuk melakukan enkripsi menggunakan sha1 dan *salt*:

```
$salt = sha1(mt_rand());

$encrytpted = openssl_encrypt( $p, 'aes-256-cbc',
"$salt");
$msg_bundle = "$salt:$encrytpted";

$sql = "INSERT INTO `servers`.`users` (`server`, `uname`,
`pass`, `module`) VALUES ('$h', '$u', '$msg_bundle',
'$module')";
mysql_query($sql);
```

4.3.4 Implementasi Basis Data

Implementasi basis data diperlukan untuk menyimpan data dari *server* yang dikontrol oleh aplikasi. Berikut gambar tabel basis data yang dibangun:

Gambar 4.7 Tabel Basis Data Aplikasi

Berikut sintak PHP yang digunakan untuk menyambungkan aplikasi dengan basis data:

```
<?php

$host     = "localhost";
$user     = "theomaulida";
$pass     = "qwerty12345";
$config   = mysql_connect($host, $user, $pass);
              mysql_select_db("servers");
?>
```

Berikut sintak PHP yang digunakan untuk mengakses data yang berada di dalam basis data:

```
$query = mysql_query("SELECT * FROM `users` WHERE
`server` = '$host' ");
  $i = mysql_fetch_array($query);
  $pass = $i[2];
//// fungsi dekrip
  $pecah_encrypsi = explode(":", $pass);

  $salt    = $pecah_encrypsi[0];
  $pasword = $pecah_encrypsi[1];
/////
  $decrypted = openssl_decrypt($pasword,
'aes-256-cbc', "$salt");

  $con = ssh2_connect( "$i[0]", 22);
  ssh2_auth_password($con, "$i[1]", "$decrypted");


IptablesInput($con,$src,$dst,$sport,$dport,$eth,$pro,
$target,$ethWin);
```

4.3.5 Implementasi Code Program

Setelah proses instalasi dan konfigurasi selesai dilakukan, langkah selanjutnya adalah melakukan *coding* untuk pembuatan program aplikasi manajemen *firewall* secara terpusat. Program tersebut dibuat dengan menggunakan bahasa pemrograman PHP dan disimpan di dalam direktori /opt/lampp/hatdocs/. Berikut daftar nama *file program* PHP yang telah dibuat beserta penjelasan fungsi dari masing-masing *file program*:

No	File Program	Fungsi
1	connect.php	*File program* ini berfungsi untuk menghubungkan aplikasi dengan basis data. Sehingga aplikasi dapat mengambil data username dan password yang tersimpan di dalam basis data.
2	index_login.php	*File program* ini berfungsi untuk melakukan otentikasi terhadap *administrator* yang *log in* ke dalam aplikasi.

3	index_logout.php	*File program* ini berfungsi untuk menghapus *session* dari *log in* data *administrator* yang terjalin selama *administrator log in* ke dalam aplikasi.
4	new_forward_rule.php	*File program* ini berfungsi untuk mengirimkan data yang dimasukkan melalui filter_forward_form.php menuju fungsi forward yang berada di dalam direktori function/. Setiap data masukan yang dikirimkan disesuaikan dengan jenis *module firewall* yang dipilih ketika *administrator* melakukan *insert rule.*
5	new_input_rule.php	*File program* ini berfungsi untuk mengirimkan data yang dimasukkan melalui filter_input_form.php menuju fungsi input yang berada di dalam direktori function/. Setiap data masukan yang dikirimkan disesuaikan dengan jenis *module firewall* yang dipilih ketika *administrator* melakukan *insert rule.*
6	new_output_rule.php	*File program* ini berfungsi untuk mengirimkan data yang dimasukkan melalui filter_output_form.php menuju fungsi output yang berada di dalam direktori function/. Setiap data masukan yang dikirimkan disesuaikan dengan jenis *module firewall* yang dipilih ketika *administrator* melakukan *insert rule.*
7	ethinNotNull.php	*File program* ini berfungsi untuk merubah masukan data yang diterima dari new_forward_rule.php menjadi sintak *firewall* yang akan dikirimkan menuju *host* yang dipilih. File program ini hanya merubah

		masukan yang menyertakan *ethernet* sebagai jalur masuk komunikasi.
8	ethNotNullForward.php	*File program* ini berfungsi untuk merubah masukan data yang diterima dari new_forward_rule.php menjadi sintak *firewall* yang akan dikirimkan menuju *host* yang dipilih. File program ini merubah masukan yang menyertakan *ethernet* sebagai jalur masuk dan keluar komunikasi.
9	ethNullForward.php	*File program* ini berfungsi untuk merubah masukan data yang diterima dari new_forward_rule.php menjadi sintak *firewall* yang akan dikirimkan menuju *host* yang dipilih. File program ini merubah masukan yang tidak menyertakan *ethernet* sebagai jalur komunikasi.
10	ethoutNotNull.php	*File program* ini berfungsi untuk merubah masukan data yang diterima dari new_forward_rule.php menjadi sintak *firewall* yang akan dikirimkan menuju *host* yang dipilih. File program ini hanya merubah masukan yang menyertakan *ethernet* sebagai jalur keluar komunikasi.
11	IpfwInput.php	*File program* ini berfungsi untuk merubah masukan data yang diterima dari new_input_rule.php menjadi sintak *firewall* yang akan dikirimkan menuju *host* yang dipilih. File program ini hanya merubah masukan untuk IPFW INPUT.
12	IpfwOutput.php	*File program* ini berfungsi untuk merubah masukan data yang diterima dari new_output_rule.php menjadi sintak *firewall* yang akan dikirimkan menuju *host* yang

		dipilih. File program ini hanya merubah masukan untuk IPFW OUTPUT.
13	IptablesInput.php	*File program* ini berfungsi untuk merubah masukan data yang diterima dari new_input_rule.php menjadi sintak *firewall* yang akan dikirimkan menuju *host* yang dipilih. File program ini hanya merubah masukan untuk IPTABLES INPUT.
14	IptablesOutput.php	*File program* ini berfungsi untuk merubah masukan data yang diterima dari new_output_rule.php menjadi sintak *firewall* yang akan dikirimkan menuju *host* yang dipilih. File program ini hanya merubah masukan untuk IPTABLES OUTPUT.
15	NetshInput.php	*File program* ini berfungsi untuk merubah masukan data yang diterima dari new_input_rule.php menjadi sintak *firewall* yang akan dikirimkan menuju *host* yang dipilih. File program ini hanya merubah masukan untuk NETSH OUTPUT.
16	NetshOutput.php	*File program* ini berfungsi untuk merubah masukan data yang diterima dari new_output_rule.php menjadi sintak *firewall* yang akan dikirimkan menuju *host* yang dipilih. File program ini hanya merubah masukan untuk NETSH INPUT.
17	rule_show_action.php	*File program* ini berfungsi untuk mengirimkan masukan *host* yang dipilih *administrator* yang selanjutnya dikirimkan menuju *file program* yang berfungsi untuk menampilkan *rule*.
18	rule_filter_input.php	*File program* ini berfungsi untuk menampilkan *rule firewall* IPTABLES

		INPUT sesuai dengan *host* yang dipilih di rule_show_form.php.
19	rule_filter_output.php	*File program* ini berfungsi untuk menampilkan *rule firewall* IPTABLES OUTPUT sesuai dengan *host* yang dipilih di rule_show_form.php.
20	rule_filter_forward.php	*File program* ini berfungsi untuk menampilkan *rule firewall* IPTABLES FORWARD sesuai dengan *host* yang dipilih di rule_show_form.php.
21	rule_ipfw.php	*File program* ini berfungsi untuk menampilkan *rule firewall* IPFW sesuai dengan *host* yang dipilih di rule_show_form.php.
22	rule_netsh.php	*File program* ini berfungsi untuk menampilkan *rule firewall* Adv. Windows Firewall INPUT sesuai dengan *host* yang dipilih di rule_show_form.php.
23	rule_netsh_outbound.php	*File program* ini berfungsi untuk menampilkan *rule firewall* Adv. Windows Firewall OUTPUT sesuai dengan *host* yang dipilih di rule_show_form.php.
24	edit_iptables_input.php	*File program* ini berfungsi untuk melakukan *edit* terhadap *rule* IPTABLES INPUT.
25	edit_iptables_output.php	*File program* ini berfungsi untuk melakukan *edit* terhadap *rule* IPTABLES OUTPUT.
26	edit_iptables_forward.php	*File program* ini berfungsi untuk melakukan *edit* terhadap *rule* IPTABLES FORWARD.
27	edit_ipfw_input.php	*File program* ini berfungsi untuk melakukan *edit* terhadap *rule* IPFW INPUT.
28	edit_ipfw_output.php	*File program* ini berfungsi untuk melakukan *edit* terhadap *rule* IPTABLES OUTPUT.

29	edit_netsh_input.php	*File program* ini berfungsi untuk melakukan *edit* terhadap *rule* Adv. Windows Firewall INPUT
30	edit_netsh_output.php	*File program* ini berfungsi untuk melakukan *edit* terhadap *rule* Adv. Windows Firewall OUTPUT
31	delete_filter_input.php	*File program* ini berfungsi untuk menghapus *rule firewall* IPTABLES INPUT. Aksi ini akan dilakukan apabila *button delete* yang ada di rule_filter_input.php ditekan.
32	delete_filter_output.php	*File program* ini berfungsi untuk menghapus *rule firewall* IPTABLES OUTPUT. Aksi ini akan dilakukan apabila *button delete* yang ada di rule_filter_output.php ditekan.
33	delete_filter_forward.php	*File program* ini berfungsi untuk menghapus *rule firewall* IPTABLES FORWARD. Aksi ini akan dilakukan apabila *button delete* yang ada di rule_filter_forward.php ditekan.
34	delete_ipfw.php	*File program* ini berfungsi untuk menghapus *rule firewall* IPFW. Aksi ini akan dilakukan apabila *button delete* yang ada di rule_ipfw.php ditekan.
35	delete_netsh_input.php	*File program* ini berfungsi untuk menghapus *rule firewall* Adv. Windows Firewall INPUT. Aksi ini akan dilakukan apabila *button delete* yang ada di rule_netsh.php ditekan.
36	delete_netsh_ouput.php	*File program* ini berfungsi untuk menghapus *rule firewall* Adv. Windows Firewall OUTPUT. Aksi ini akan dilakukan apabila *button delete* yang ada di rule_netsh_outbound.php ditekan.
37	setting_add_host.php	*File program* ini berfungsi untuk

		menambahkan *host* baru yang akan dikontrol ke dalam aplikasi.
38	setting_delete_host.php	*File program* ini berfungsi untuk menghapus *host* yang dikontrol di dalam aplikasi.

Seluruh *file program* diatas diimplementasikan ke dalam lima fitur dasar aplikasi manajemen firewall secara terpusat. Kelima fitur tersebut yaitu *insert rule, update rule, show rule, delete rule* dan *host*. Berikut penjelasan dari masing-masing fitur tersebut:

1. Fitur *Insert Rule*

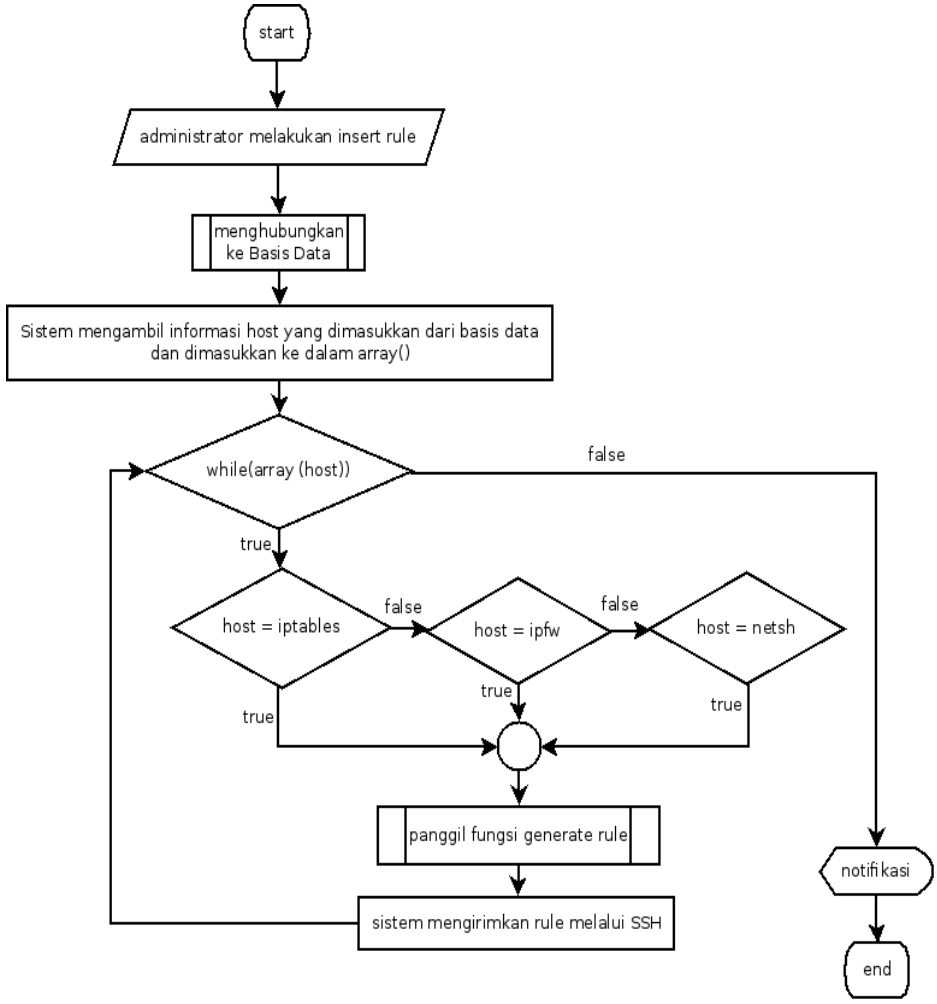
Fitur *insert rule* berfungsi untuk melakukan *insert rule* menuju *server* yang dikonfigurasi. Fitur *insert rule* diimplementasikan di dalam *file program* new_input_rule.php, new_output_rule.php dan new_forward_rule.php. Berikut salah satu potongan kode program yang digunakan untuk melakukan *insert rule*:

```
if ($host == "ALL"){
$query = mysql_query("SELECT * FROM `users` ORDER BY
`server` ASC ");
while ($i = mysql_fetch_array($query)) {
if ($i[3] == iptables ) {
        $pass = $i[2];
        $pecah_encrypsi = explode(":", $pass);
        $salt    = $pecah_encrypsi[0];
        $pasword = $pecah_encrypsi[1];
        $decrypted = openssl_decrypt($pasword,
'aes-256-cbc', "$salt");
        $con = ssh2_connect( "$i[0]", 22);
        ssh2_auth_password($con, "$i[1]",
"$decrypted");

IptablesInput($con,$src,$dst,$sport,$dport,$eth,$pro,$
target,$ethWin );}
```

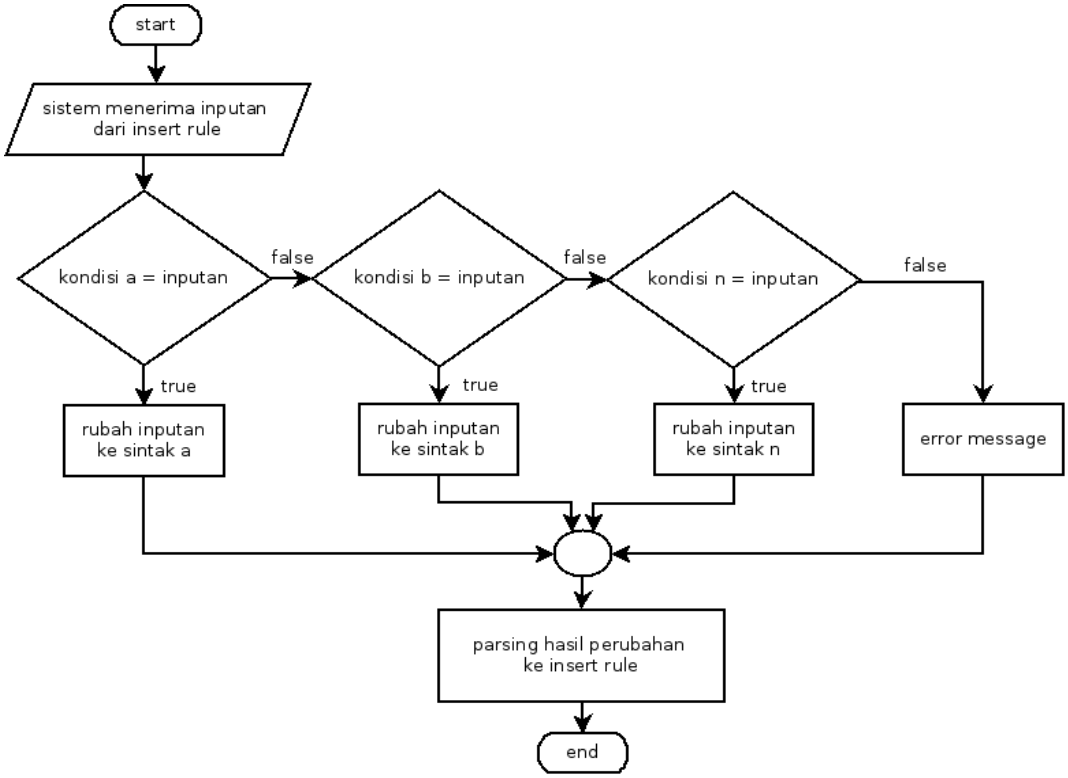
Dari potongan baris kode diatas dapat dilihat bahwa untuk melakukan *insert rule* terhadap seluruh *server* konfigurasi, aplikasi manajemen *firewall* secara terpusat akan akan mengambil informasi berupa *username, password* dan jenis *module firewall* yang sesuai dengan IP (*internet protocol*) *server* konfigurasi di dalam basis data. Selanjutnya aplikasi akan melakukan *generate* dari rule yang dimasukkan *administrator* menjadi sintak *insert rule* yang sesuai dengan *module firewall server*

konfigurasi, selanjutnya aplikasi akan mengirimkan *rule* yang telah dirubah menuju *server* konfigurasi melalui jalur komunikasi SSH. Berikut ditampilkan alur *insert rule*:



Gambar 4.8 Diagram alir *insert rule*

Pada gambar 4.8 dapat dilihat bahwa sebelum aplikasi mengirimkan rule ke server kontrol terlebih dulu aplikasi mengambil informasi server kontrol dari basis data. Hal tersebut dilakukan untuk mendapatkan informasi tentang username, password dan jenis module firewall yang digunakan oleh server kontrol. Setelah mendapatkan informasi, selanjutnya aplikasi melakukan generate terhadap rule yang dimasukkan administrator melalui aplikasi sesuai dengan module firewall yang digunakan oleh server konfigurasi. generate rule dilakukan secara berulang sampai seluruh rule yang dimasukkan dirubah. Selanjutnya aplikasi akan mengirimkan rule firewall yang sudah dirubah ke seluruh server konfigurasi. Apabila pengiriman rule berhasil maka muncul pesan sukses namun apabila gagag maka muncul pesan gagal. Berikut ditampilkan diagram alir dari fungsi *generate rule*:



Gambar 4.9 Diagram alir fungsi *generate rule*

Dari gambar 4.9 dapat dilihat bahwa fungsi generate rule akan melakukan perubahan terhadap rule yang dimasukkan administrator melalui form aplikasi web. Generate dilakukan dengan membandingkan rule yang dimasukkan administrator dengan kondisi yang digunakan untuk melakukan generate rule. Apabila kondisi terpenuhi maka rule yang dimasukkan akan dirubah sesuai dengan sintak rule yang terdapat pada kondisi tersebut namun apabila kondisi tidak terpenuhi maka generate rule akan terus dilakukan sampai kondisi yang terakhir.

2. Fitur Edit Rule

Fitur *edit rule* digunakan untuk melakukan *edit rule* yang terdapat pada *server* konfigurasi. Fungsi *edit rule* diterapkan di dalam *file program* edit_iptables_input.php, edit_iptables_output.php, edit_iptables_forward.php, edit_ipfw.php dan edit_netsh.php. Berikut salah satu potongan kode program yang digunakan untuk melakukan *edit rule*:

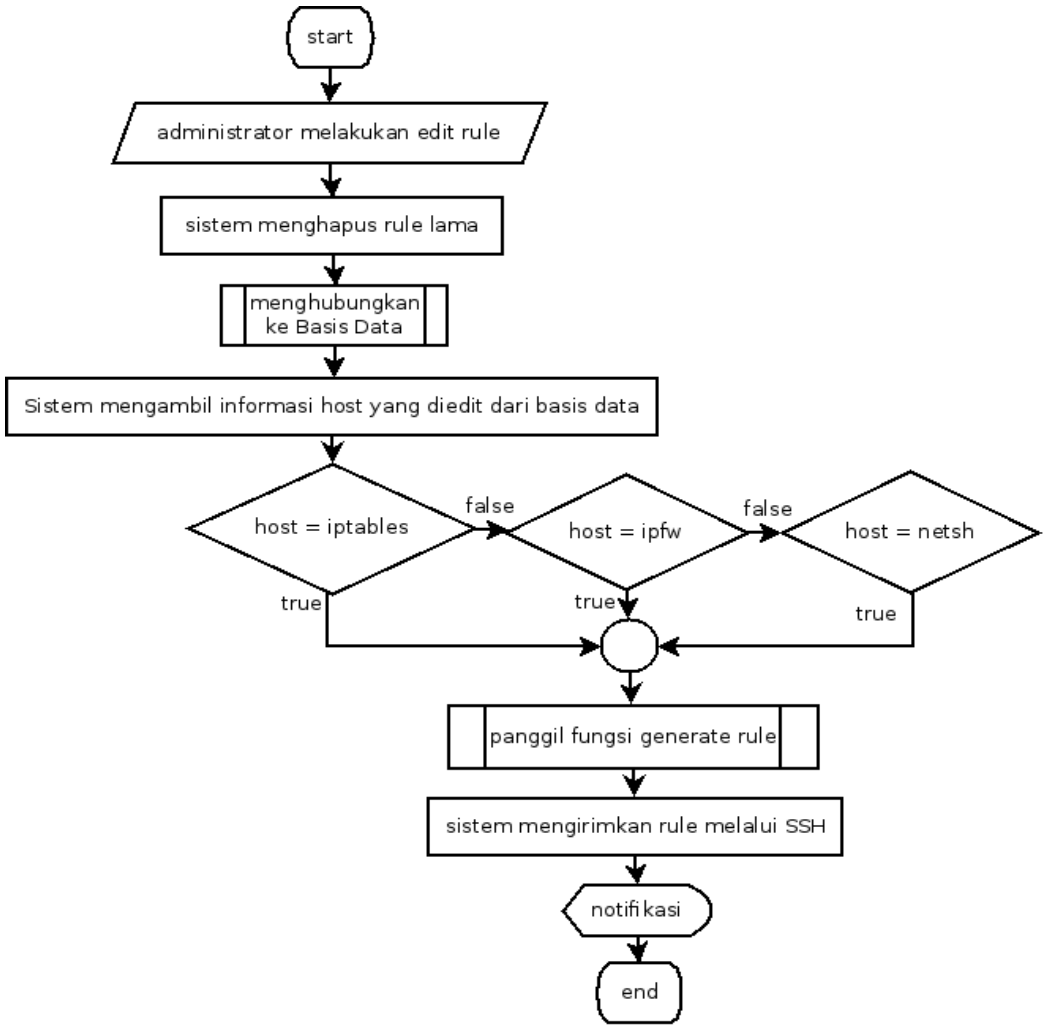
```
include 'connect.php';
include 'Function/IptablesInput.php';

$name=$_POST['rname'];
$src=$_POST['src'];
$dst=$_POST['dst'];
$sport=$_POST['sport'];
$dport=$_POST['dport'];
$eth=$_POST['eth'];
$ethWin=$_POST['menu1'];
$pro=$_POST['menu2'];
$target=$_POST['menu3'];
session_start();
$_edit=$_POST[$i[0]. "|" .$info[$row]['ID']];
$pecah = explode("|", $_edit);
$query = mysql_query("SELECT * FROM `users` WHERE server
   = '$pecah[0]' ");
  $i = mysql_fetch_array($query);
  $pass = $i[2];
  $pecah_encrypsi = explode(":", $pass);
 $salt    = $pecah_encrypsi[0];
  $pasword = $pecah_encrypsi[1];
  $decrypted = openssl_decrypt($pasword,
   'aes-256-cbc', "$salt");
      $con = ssh2_connect( "$i[0]", 22);
      ssh2_auth_password($con, "$i[1]", "$decrypted");
ssh2_exec($con, "iptables -t filter -D INPUT $pecah[1]
   " );
if ($i[3] == iptables ) {
  $query = mysql_query("SELECT * FROM `users` WHERE
   `server` = '$pecah[0]' ");
  $i = mysql_fetch_array($query);
  $pass = $i[2];
  $pecah_encrypsi = explode(":", $pass);
  $salt    = $pecah_encrypsi[0];
  $pasword = $pecah_encrypsi[1];
  $decrypted = openssl_decrypt($pasword,
   'aes-256-cbc', "$salt");
  $con = ssh2_connect( "$i[0]", 22);
  ssh2_auth_password($con, "$i[1]", "$decrypted");

$hasil =
IptablesInput($con,$src,$dst,$sport,$dport,$eth,$pro,
$target,$ethWin);
```

Dari potongan kode program diatas dapat dilihat bahwa untuk melakukan *edit* terhadap rule firewall aplikasi membutuhkan update rule dari administrator, administrator dapat melakukan *edit* dengan menekan tombol *edit* yang terdapat pada tabel *show rule*. Apabila tombol ditekan maka muncul *form edit*, selanjutnya *administrator* memasukkan *update rule* dan menekan tombol *edit* untuk *submit*.

Selanjutnya aplikasi menghapus *rule* lama yang diedit dan mengganti dengan *rule* baru yang dimasukkan *administrator* melalui *form edit*. Apabila *edit rule* sukses maka muncul pesan sukses namun apabila gagal maka muncul pesan gagal. Berikut digram alir untuk fitur *edit rule*:



Gambar 4.10 Diagram alir *edit rule*

Dari gambar 4.10 dapat dilihat bahwa untuk melakukan *edit* terhadap *rule* yang sudah ada terlebih dulu aplikasi harus menghapus *rule* yang akan diedit. Setelah *rule* dihapus, selanjutnya aplikasi melakukan *insert rule* terhadap *rule* dari *module firewall* yang dipilih untuk diedit. Apabila *edit* sukses maka muncul pesan sukses namun apabila gagal muncul pesan gagal.

3. Fitur *Show Rule*

Fitur *show rule* digunakan untuk menampilkan *rule* yang terdapat di dalam masing-masing *server* kontrol. Fitur *show rule* diterapkan di dalam *file program* rule_filter_input.php, rule_filter_output.php, rule_filter_forward.php, rule_ipfw.php, rule_netsh.php dan rule_netsh_outbound.php. Berikut potongan kode program yang digunakan untuk melakukan *show rule*:

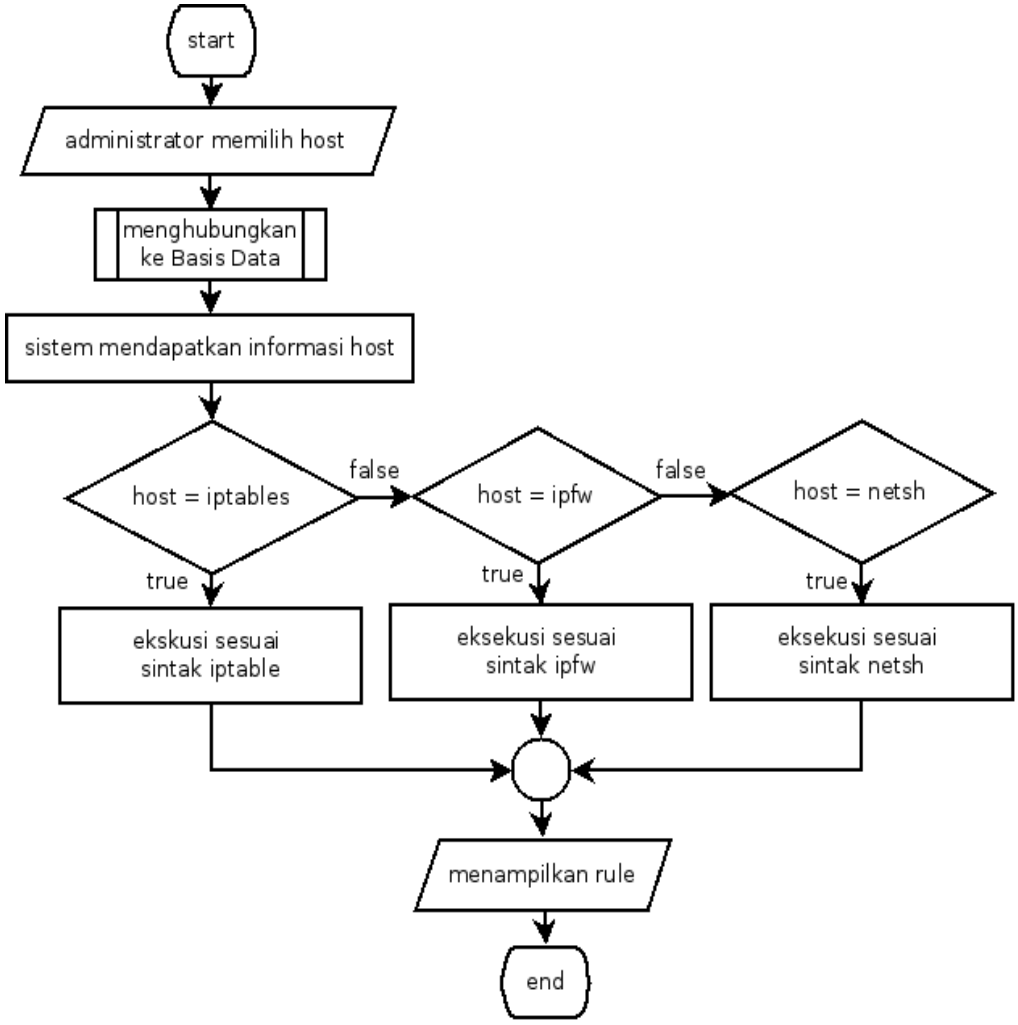
```
if (!($stream = ssh2_exec($coni, "netsh advfirewall
   firewall show rule name = all dir = in" ))) {
         echo "fail: unable to execute command\n";
         } else {
             stream_set_blocking($stream, true);
             $data = "";
             while ($buf = fread($stream,4096)) {
             $data .= $buf;
             $file=fopen("txt/rule_netsh.txt", "w+");
             $isi=$data;
             fwrite($file, $isi);}
             fclose($stream);}

  $txt_file    =
   file_get_contents('txt/rule_netsh.txt');
  $rows        = explode("\n", $txt_file);
  array_shift($rows);

  echo "<table class=\"myTable\">";
  echo "<tr><td> ";
  echo "<b>SERVER : </b>";
  echo "</td>";
  echo "<td>";
  echo '<b>'. "$i[0]" . '</b></td>';
   echo "<td>";
  echo '<b>Delete</b></td>';
  echo "</tr>";
 echo "</table>";}
```

Dari potongan baris kode program diatas dapat dilihat bahwa untuk menampilkan *rule* yang terdapat pada masing-masing *server* konfigurasi *administrator* harus memilih *server kontrol* yang akan ditampilkan *rule* firewallnya melalui *form rule show* lalu menekan tombol *show*. Selanjutnya aplikasi akan mengambil informasi *username* dan *password* dari *server* kontrol yang dipilih *administrator* selanjutnya aplikasi akan mengirimkan sintak *show rule* menuju *server* kontrol yang dipilih

melalui SSH dan akan menampilkan *rule* dari *server* kontrol yang dipilih ke dalam bentuk tabel. Berikut digram alir untuk fitur *show rule*:



Gambar 4.11 Diagram alir *show rule*

Gambar 4.11 menunjukkan bahwa untuk melakukan *show rule* terlebih dulu aplikasi meminta *administrator* untuk memilih *server* konfigurasi mana yang ingin ditampilkan *rule* firewallnya. Apabila *administrator* sudah memilih, selanjutnya aplikasi akan mengambil informasi dari *server* kontrol yang dipilih dan aplikasi akan mengirimkn perintah untuk menampilkan *rule* sesuai dengan *module firewall* dari *server* yang dipilih. Selanjutnya aplikasi akan menampilkan informasi rule dari server yang dipilih di dalam bentuk tabel.

4. Fitur *Delete Rule*

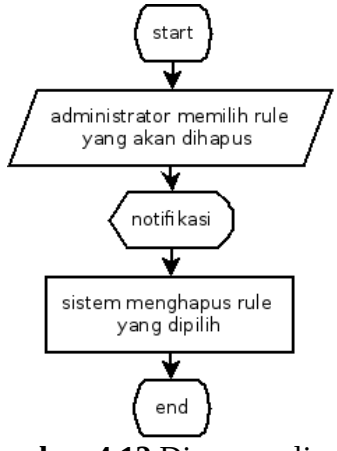
Fitur *delete rule* digunakan untuk menghapus *rule* yang terdapat di dalam masing-masing *server* kontrol. Fitur *delete rule* diterapkan di dalam *file program*

delete_filter_input.php, delete_filter_output.php, delete_filter_forward.php, delete_ipfw.php, delete_netsh_input.php dan delete_netsh_output.php. Berikut salah satu potongan kode program yang digunakan untuk melakukan *delete rule*:

```
$host=$_POST['delete'];
$pecah        = explode("|", $host);

$query = mysql_query("SELECT * FROM `users` WHERE server =
'$pecah[0]' ");
$i = mysql_fetch_array($query);
$pass = $i[2];
$pecah_encrypsi = explode(":", $pass);
$salt    = $pecah_encrypsi[0];
$pasword = $pecah_encrypsi[1];
$decrypted = openssl_decrypt($pasword, 'aes-256-cbc',
"$salt");
$con = ssh2_connect( "$i[0]", 22);
ssh2_auth_password($con, "$i[1]", "$decrypted");
ssh2_exec($con, "ipfw delete $pecah[1] " );
```

Dari potongan baris kode program diatas dapat dilihat bahwa aplikasi meminta masukan nilai dari tombol *delete* yang terdapat pada tabel *show rule*. Apabila tombol *delete* ditekan maka dari tabel *show rule* akan mengirimkan nilai *variable* yang sesuai dengan letak *rule* yang akan dihapus dan selanjutnya aplikasi akan melakukan *delete* terhadap *rule* yang telah dipilih oleh *administrator*. Berikut digram alir untuk fitur *delete rule:*



Gambar 4.12 Diagram alir *delete rule*

Gambar 4.12 menunujukkan bahwa untuk melakukan *delete rule administrator* hanya perlu memilih *rule* yang akan dihapus dengan menekan tombol (X). Apabila

tombol ditekan maka aplikasi akan menampilkan notifikasi, apabila ditekan OK maka *rule* akan dihapus namun apabila ditekan CANCEL maka *rule* tidak dihapus.

5. Fitur *Host*

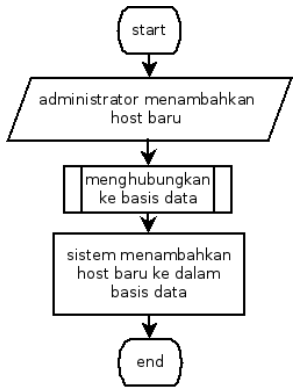
Fitur *host* digunakan untuk menambah dan menghapus informasi dari *server* yang dikontrol oleh aplikasi manajemen *firewall* secara terpusat. Fitur *host* diterapkan di dalam *file program* setting_add_host.php dan setting_delete_host.php. Berikut potongan kode program yang digunakan untuk melakukan tambah *host*:

```
include 'connect.php';
$h = $_POST['host'];
$u = $_POST['uname'];
$p = $_POST['pass'];
$tab = $_POST['iptables'];
$fw = $_POST ['ipfw'];
$net = $_POST ['netsh'];

if ($tab !== null ) {
      $module = $tab;
    }elseif ($fw!== null ) {
      $module = $fw;
    }elseif ($net !== null) {
      $module = $net;
    }
  // enkripsi password
  $salt = sha1(mt_rand());
  $encrytpted = openssl_encrypt( $p, 'aes-256-cbc',
"$salt");
  $msg_bundle = "$salt:$encrytpted";

  $sql = "INSERT INTO `servers`.`users` (`server`,
`uname`, `pass`, `module`) VALUES ('$h', '$u',
'$msg_bundle', '$module')";
  mysql_query($sql);
```

Dari potongan baris kode diatas dapat dilihat bahwa untuk menambahkan *host*, sistem membutuhkan informasi tentang *host* yang akan ditambahkan dari *administrator*. *Administrator* dapat memasukkan informasi *host* yang akan ditambahkan melalui *form* tambah *host* yang berada diatas tabel *host*. Apabila informasi sudah didapatkan maka aplikasi akan melakukan enkripsi *password* dan menambahkan informasi *host* tersebut ke dalam basis data. Berikut digram alir untuk menambah *host*:



Gambar 4.13 Digram alir *add host*

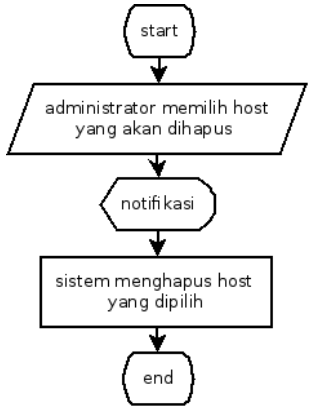
Gambar 4.13 menunujukkan bahwa untuk melakukan *add host administrator* hanya perlu menekan tombol (+) yang terletak diatas *show host*. Selanjutnya administrator hanya perlu mengisi form untuk menambhkan host dan menekan tombol ADD. Apabila ditekan maka host baru akan ditambahkan ke dalam basis data.

Berikut potongan kode program yang digunakan untuk melakukan hapus *host*:

```
include 'connect.php';

$h = $_POST['delete'];
$sqlhapus="DELETE FROM users where server= '$h' ";
mysql_query($sqlhapus);
```

Dari potongan baris kode diatas dapat dilihat bahwa untuk menghapus *host*, sistem membutuhkan masukan nilai dari tombol *delete* yang terdapat pada tabel tambah *host*. Apabila nilai masukan sudah didapatkan maka aplikasi akan menghapus informasi tentang *host* yang dipilih *administrator* untuk dihapus dari basis data. Berikut digram alir untuk menghapus *host*:



Gambar 4.14 Digram alir untuk *delete host*

Gambar 4.14 menunujukkan bahwa untuk melakukan *delete host administrator* hanya perlu memilih *host* yang akan dihapus dengan menekan tombol (X). Apabila tombol ditekan maka aplikasi akan menampilkan notifikasi, apabila ditekan OK maka *host* akan dihapus namun apabila ditekan CANCEL maka *host* tidak dihapus.

4.4 Implementasi Front End

Pada bagian ini dijelaskan implementasi *front end* yang dibangun disesuaikan dengan kebutuhan fungsional sistem dan juga kebutuhan antar muka untuk setiap fitur aplikasi. Implementasi *front end* yang diterapkan, dibangun dengan menggunakan HTML dan CSS yang selanjutnya disimpan di dalam direktori /opt/lampp/htdocs/. Berikut daftar nama *file program* HTML dan CSS yang telah dibuat beserta penjelasan fungsi dari masing-masing *file program*:

No	File Program	Fungsi
1	index.php	*File program* ini berfungsi untuk mengatur tata letak tampilan halaman *log in*. Di dalam *file* ini terdapat *form* yang berfungsi untuk melakukan otentikasi *administrator* yang akan melakukan kontrol *firewall*.
2	filter_forward_form.php	*File program* ini berfungsi untuk menampilkan *form* yang digunakan *administrator* di dalam melakukan konfigurasi IPTABLES FORWARD *rule*.
3	filter_input_form.php	*File program* ini berfungsi untuk menampilkan *form* yang digunakan administrator di dalam melakukan konfigurasi IPTABLES INPUT, IPFW INPUT dan Adv. Windows Firewall INPUT *rule*.
4	filter_output_form.php	*File program* ini berfungsi untuk menampilkan *form* yang digunakan *administrator* di dalam melakukan

		konfigurasi IPTABLES OUTPUT, IPFW OUTPUT dan Adv. Windows Firewall OUTPUT *rule*.
5	rule_show_form.php	*File program* ini berfungsi untuk menampilkan *host* yang tersimpan di dalam basis data secara *drop down menu*. Untuk menampilkan *rule firewall administrator* hanya perlu memilih *host* yang ingin ditampilkan *rule* firewallnya selanjutnya untuk menampilkan *rule firewall* klik *button show*
6	edit_input_form.php	*File program* ini berfungsi untuk menampilkan *form edit* yang digunakan administrator di dalam melakukan *edit rule* IPTABLES INPUT, IPFW INPUT dan Adv. Windows Firewall INPUT *rule*.
7	edit_output_form.php	*File program* ini berfungsi untuk menampilkan *form edit* yang digunakan administrator di dalam melakukan *edit rule* IPTABLES OUTPUT, IPFW OUTPUT dan Adv. Windows Firewall OUTPUT *rule*.
8	edit_forward_form.php	*File program* ini berfungsi untuk menampilkan *form edit* yang digunakan administrator di dalam melakukan *edit rule* IPTABLES FORWARD.
9	setting_addhost.php	*File program* ini berfungsi untuk menampilkan informasi tentang *host* yang tersambung dengan aplikasi. Informasi yang ditampilkan diambil

		dari basis data.
10	setting_addhost_form.php	*File program* ini berfungsi untuk menampilkan *form* yang berfungsi untuk menambahkan *host* yang akan dikontrol *rule firewall*-nya.
11	formjs.js	*File program* ini berfungsi untuk mengatur tampilan *drop down form* yang terdapat di *form insert.*
12	jquery-2.0.3.min.js	*File program* ini berfungsi untuk mengatur tampilan *pop up notification* ketika *administrator* selesai melakukan *input rule*. Notifikasi yang ditampilkan berupa notifikasi eror dan sukses.
13	css.css	*File program* ini berfungsi untuk mengatur tata letak tampilan *form* yang berada di antarmuka *insert rule*.
14	show.css	*File program* ini berfungsi untuk mengatur tata letak konten dari rule_show_form.php.
15	show2.css	*File program* ini berfungsi untuk mengatur tata letak konten dari rule_filter_input.php, rule_filter_output.php, rule_filter_forward.php, rule_ipfw.php, rule_netsh.php dan rule_netsh_outbound.php.
16	style.css	*File program* ini berfungsi untuk mengatur tata letak tampilan konten aplikasi baik itu menu navigasi dan tata letak *form* yang digunakan untuk *input rule*.
17	style_host.css	*File program* ini berfungsi untuk

		untuk mengatur tata letak tampilan konten *host*.
18	tab.css	*File program* ini berfungsi untuk mengatur tata letak dan fungsi *tab menu* yang berada di dalam *insert menu form*.
19	tab2.css	*File program* ini berfungsi untuk mengatur tata letak dan fungsi *tab menu* yang berada di dalam *show rule menu*.

Hasil implementasi antar muka ketika kode program diatas diakses melalui *browser* dilampirkan di dalam lampiran yang disertakan diakhir penulisan laporan ini.

BAB V
PENGUJIAN DAN ANALISIS

Pada bab ini dilakukan pengujian dan analisis terhadap aplikasi manajemen *firewall* secara terpusat. Pengujian dan analisis tersebut dilakukan dalam dua tahap yaitu:

1. Pengujian dan analisis kebutuhan fungsional yang dilakukan dengan menguji dan menganalisis validitas dari seluruh fitur yang ada di dalam aplikasi menajemen firewall secara terpusat. Pengujian ini dilakukan untuk mengetahui validitas dari masing-masing fitur yang ada di dalam aplikasi.
2. Pengujian dan analisis kebutuhan non-fungsional yang dilakukan dengan menguji dan menganalisis kinerja dari aplikasi manajemen *firewall* secara terpusat. Pengujian ini dilakukan untuk mengetahui kinerja dari aplikasi menajemen firewall secara terpusat berdasarkan perbandingan waktu yang dibutuhkan untuk melakukan *insert rule* menggunakan aplikasi manajemen *firewall* secara terpusat dan konfigurasi secara manual.

5.1 Pengujian Tahap Pertama

Pada pengujian tahap pertama ini dilakukan pengujian validitas dari seluruh fitur yang ada di dalam aplikasi manajemen *firewall* secara terpusat. Pengujian dilakukan dengan menguji seluruh fitur yang terdapat di dalam aplikasi manajemen firewall secara terpusat sesuai dengan skenario uji. Skenario uji yang dilakukan di dalam pengujian tahap pertama yaitu:

1. Skenario Uji Login

Kasus uji *login* dilakukan untuk mengetahui fungsi dari fitur login yaitu untuk melakukan otentikasi terhadap *administrator* yang melakukan *login* ke dalam aplikasi. Hal tersebut dilakukan, untuk mengetahui apakah fungsi *login* telah berjalan sesuai kebutuhan.

Tabel 5.1 Kasus uji untuk pengujian validasi fitur *login*

Nama Kasus Uji	Kasus Uji *login*
Objek Uji	Fitur *login*
Tujuan Pengujian	Untuk memastikan fungsi dari fitur *login* telah berjalan sesuai kebutuhan

Prosedur Uji	*Administrator* melakukan *login* ke dalam aplikasi melalui *form* yang ada di halaman *login*.
Hasil yang diharapkan	Sistem dapat melakukan otentikasi terhadap *administrator* yang melakukan *login* di dalam aplikasi.

2. Skenario Uji Insert Rule

Kasus uji *insert rule* dilakukan untuk mengetahui fungsi dari fitur *insert rule* yaitu untuk mengirimkan *rule* dari aplikasi menuju semua *host* yang dikonfigurasi. Hal tersebut dilakukan, untuk mengetahui apakah fungsi *insert rule* telah berjalan sesuai kebutuhan.

Tabel 5.2 Kasus uji untuk pengujian validasi fitur *insert rule*

Nama Kasus Uji	Kasus Uji *Insert Rule*
Objek Uji	Fitur *Insert Rule*
Tujuan Pengujian	Untuk memastikan fungsi dari fitur *insert rule* telah berjalan sesuai kebutuhan
Prosedur Uji	*Administrator* melakukan input *rule firewall* melalui *form*.
Hasil yang diharapkan	Sistem dapat mengirimkan *rule* menuju semua *host* yang dituju dan ketika *rule* sukses dikirimkan aplikasi akan menampilkan notifikasi sukses.

3. Skenario Uji Show Rule

Kasus uji *show rule* dilakukan untuk mengetahui fungsi dari *fitur show rule* yaitu untuk menampilkan *rule* berdasarkan *host*. Hal tersebut dilakukan, untuk mengetahui fungsi *show rule* telah berjalan sesuai kebutuhan.

Tabel 5.3 Kasus uji untuk pengujian validasi fitur *show rule*

Nama Kasus Uji	Kasus Uji *Show Rule*
Objek Uji	Fitur *Show Rule*
Tujuan Pengujian	Untuk memastikan fungsi dari fitur *show rule* telah berjalan sesuai kebutuhan
Prosedur Uji	1. *Administrator* memilih *host* yang ingin ditampilkan *rule firewall*-nya. 2. Klik *show rule*.

Hasil yang diharapkan	Sistem dapat menampilkan *rule* sesuai dengan *host* yang dipilih dan *rule* ditampilkan di dalam bentuk tabel.

4. Skenario Uji Edit Rule

Kasus uji edit *rule* dilakukan untuk mengetahui fungsi dari *fitur edit rule* yaitu untuk melakukan *edit* terhadap *rule firewall*. Hal tersebut dilakukan, untuk mengetahui fungsi edit *rule* telah berjalan sesuai kebutuhan.

Tabel 5.4 Kasus uji untuk pengujian validasi fitur *edit rule*

Nama Kasus Uji	Kasus Uji *Edit Rule*
Objek Uji	Fitur *Edit Rule*
Tujuan Pengujian	Untuk memastikan fungsi dari fitur editd *rule* telah berjalan sesuai kebutuhan
Prosedur Uji	1. *Administrator* memilih *rule* yang ingin diubah. 2. Klik *edit*. 3. Isi *Form edit*. 4. Klik *edit*.
Hasil yang diharapkan	Sistem dapat mengubah *rule* yang sudah dimasukkan ke dalam *firewall*.

5. Skenario Uji Delete Rule

Kasus uji *delete rule* dilakukan untuk mengetahui fungsi dari fitur *delete rule* yaitu untuk menghapus salah satu *rule* dari *module firewall* berdasarkan *host* yang dikonfigurasi. Hal tersebut dilakukan, untuk mengetahui fungsi *delete rule* telah berjalan sesuai kebutuhan.

Tabel 5.4 Kasus uji untuk pengujian validasi fitur *delete rule*

Nama Kasus Uji	Kasus Uji *Delete Rule*
Objek Uji	Fitur *Delete Rule*
Tujuan Pengujian	Untuk memastikan fungsi dari fitur *delete rule* telah berjalan sesuai kebutuhan
Prosedur Uji	*Administrator* mengklik *button* (x) yang terdapat di kolom *delete*.
Hasil yang diharapkan	Sistem dapat menghapus salah satu *rule* dari

	masing-masing *module firewall* berdasarkan *host* yang ditampilkan *rule firewall*-nya dan ketika button (x) ditekan muncul notifikasi. Apabila diklik *button* (OK) maka *rule* akan dihapus tetapi apabila diklik *button* (Cancel) *rule* tidak dihapus..

6. Skenario Uji Add Host

Kasus uji *add host* dilakukan untuk mengetahui fungsi dari fitur *add host* yaitu untuk menambahkan *host* baru untuk dikonfigurasi *rule firewall*-nya. Hal tersebut dilakukan, untuk mengetahui fungsi *add host* telah berjalan sesuai kebutuhan.

Tabel 5.5 Kasus uji untuk pengujian validasi fitur *add host*

Nama Kasus Uji	Kasus Uji *Add Host*
Objek Uji	Fitur *Add Host*
Tujuan Pengujian	Untuk memastikan fungsi dari fitur a*dd host* telah berjalan sesuai kebutuhan
Prosedur Uji	1. *Administrator* memilih menu *host* selanjutnya klik simbol (+) yang berada diatas tabel *host.* 2. *Form* menambahkan *host* ditampilkan selanjutnya a*dministrator* memasukan data *host* baru. 3. Klik *add*.
Hasil yang diharapkan	1. Sistem dapat menambahkan *host* ke dalam aplikasi 2. Sistem dapat melakukan enkripsi terhadap data yang disimpan di dalam basis data

7. Skenario Uji Delete Host

Kasus uji delete *host* dilakukan untuk mengetahui fungsi dari fitur *delete host* yaitu untuk menghapus *host* dari dalam aplikasi. Hal tersebut dilakukan, untuk mengetahui fungsi *delete hos*t telah berjalan sesuai kebutuhan.

Tabel 5.6 Kasus uji untuk pengujian validasi fitur *delete host*

Nama Kasus Uji	Kasus Uji *Delete Host*
Objek Uji	Fitur *Delete Host*

Tujuan Pengujian	Untuk memastikan fungsi dari fitur *delete host* telah berjalan sesuai kebutuhan
Prosedur Uji	Klik button (x) yang ada di kolom delete.
Hasil yang diharapkan	Sistem dapat menghapus *host* dari dalam aplikasi dan ketika diklik muncul notifikasi. Apabila diklik *button* (OK) maka *host* akan dihapus tetapi apabila diklik *button* (Cancel) host tidak dihapus.

5.1.1 Hasil Pengujian Tahap Pertama

Hasil pengujian validasi pada setiap skenario uji dijabarkan dalam bentuk tabel sehingga dapat diketahui apakah fungsi dari fitur aplikasi manajemen *firewall* secara terpusat telah bernilai *valid*. Hasil pengujian bernilai valid apabila fitur yang dijalankan telah sesuai dengan sekenario uji yang telah dilakukan pada pengujian tahap pertama. Berikut hasil pengujian validasi yang didapatkan:

Tabel 5.7 Hasil Pengujian Validasi

NO	Nama Pengujian	Hasil yang Diharapkan	Hasil yang Didapatkan	Status Validasi
1	Login	Aplikasi dapat melakukan otentikasi terhadap *administrator* yang melakukan *login* di dalam aplikasi.	Aplikasi dapat melakukan otentikasi terhadap *administrator* yang melakukan *login* di dalam aplikasi.	valid
2	Insert Rule	Aplikasi dapat menambahkan *rule* ke semua *module firewall* sesuai dengan *host*	Aplikasi dapat menambahkan *rule* ke semua *module firewall* sesuai dengan *host*	valid

3	Show Rule	Aplikasi dapat menampilkan *rule* sesuai *host*	Aplikasi dapat menampilkan *rule* sesuai *host*	valid
4	Edit Rule	Aplikasi dapat mengedit rule yang telah dimasukkan	Aplikasi dapat mengedit rule yang telah dimasukkan	valid
5	Delete Rule	Aplikasi dapat menghapus *rule firewall*	Aplikasi dapat menghapus *rule firewall*	valid
6	Add Host	Aplikasi dapat menambahkan *host* baru untuk dikonfigurasi	Aplikasi dapat menambahkan *host* baru untuk dikonfigurasi	valid
7	Delete Host	Aplikasi dapat manghapus *host*	Aplikasi dapat manghapus *host*	valid

5.1.2 Analisis Pengujian Tahap Pertama

Berdasarkan pengujian yang telah dilakukan, seluruh fitur yang terdapat pada aplikasi manajemen *firewall* secara terpusat menunjukkan bahwa fitur dapat berjalan secara fungsional. Berikut dijabarkan analisis untuk masing-masing fitur yang terdapat pada aplikasi manajemen *firewall* secara terpusat:

1. Fitur *login* dapat berjalan dengan baik sesuai skenario uji yaitu melakukan otentikasi terhadap *administrator* yang melakukan *login* ke dalam aplikasi.
2. Fitur *insert rule* dapat berjalan dengan baik sesuai skenario uji yaitu melakukan *insert rule* melalui *form* yang disediakan dan dikirimkan menuju semua host yang dituju secara langsung dengan menekan tombol *submit*.
3. Fitur *show rule* dapat berjalan dengan baik sesuai skenario uji yaitu menampilkan *rule* sesuai *host* yang dipilih.
4. Fitur *edit rule* dapat berjalan dengan baik sesuai skenario uji yaitu mengedit *rule* yang telah dimasukkan ke dalam *firewall*.

5. Fitur *delete rule* dapat berjalan dengan baik sesuai skenario uji yaitu menghapus *rule* dengan menekan *button* (x) yang terletak di setiap baris *rule* ketika ditampilkan.
6. Fitur *add host* dapat berjalan dengan baik sesuai skenario uji yaitu menambahkan *host* dengan menekan simbol (+) yang selanjutnya *administrator* mengisi *form* untuk menambahkan *host* baru. Apabila *button* (ADD) ditekan maka *host* baru akan ditambahkan tetapi apabila *button* (BACK) ditekan maka akan kembali ke menu *host*.
7. Fitur *delete host* dapat berjalan dengan baik sesuai skenario uji yaitu menghapus host dengan menekan button (x) yang terletak di setiap baris *host* ketika ditampilkan.

5.2 Pengujian Tahap Kedua

Pada pengujian tahap kedua pengujian dilakukan untuk mengetahui kinerja dari aplikasi manajemen firewall secara terpusat berdasarkan perbandingan waktu yang dibutuhkan ketika melakukan *insert rule* menggunakan aplikasi manajemen *firewall* secara terpusat dan ketika malakukan *insert rule* secara manual. Pengujian *insert rule* dilakukan dalam tiga kali pengujian, pengujian pertama dilakukan dengan mengirimkan *rule* terhadap tiga *server remote*, pengujian kedua dilakukan dengan mengirimkan *rule* terhadap enam *server remote* dan pengujian ketiga dilakukan dengan mengirimkan *rule* terhadap sembilan *server remote*.

Selanjutnya, untuk mendapatkan data yang dibutuhkan dilakukan *capture* data menggunakan *tools wireshark*. Berikut hasil *capture* data dari perbandingan waktu yang didapatkan selama pengujian:

Tabel 5.8 Hasil Pengujian *Wireshark*

Pengujian ke-	Jumlah Server	Waktu Yang Dibutuhkan (*sec*)	
		Aplikasi	Manual
1	3	1.240	306.2
2	6	1.527	373.9
3	9	2.200	388.5
Rata-rata (*sec*)		1.656	356.2

Dari hasil pengujian dalam tabel 5.8 diatas dapat dilihat bahwa rata-rata waktu yang dibutuhkan di dalam melakukan *insert rule* menggunakan aplikasi manajemen *firewall* secara terpusat menunjukkan waktu sekitar 1.656 *sec* sedangkan rata-rata waktu yang dibutuhkan untuk melakukan *insert rule* secara manual menunjukkan rata-rata sekitar 356.2 *sec*. Waktu yang dibutuhkan untuk melakukan *insert rule* yang cepat pada aplikasi menajemen firewall secara terpusat dibandingkan insert rule secara manual terjadi karena berbeda mekanisme *insert rule* yang dilakukan.

Pada purwarupa aplikasi menajemen firewall secara terpusat, *rule* yang telah dimasukkan oleh *administrator* dirubah sesuai dengan jenis *module firewall* dari *server* yang dikonfigurasi selanjutnya *rule* tersebut langsung dikirimkan oleh aplikasi menuju seluruh server yang dikonfigurasi. Sedangkan pada konfigurasi secara manual *administrator* harus melakukan *remote* satu persatu terhadap *server* yang dikonfigurasi selanjutnya *administrator* melakukan *insert rule* terhadap *server* yang diremote satu persatu via *console* dengan sintak *insert rule* sesuai dengan *madule firewall* yang terdapat pada masing-masing *server* yang dikonfigurasi.

5.2.1 Analisis Pengujian Tahap Kedua

Berdasarkan pengujian yang telah dilakukan, waktu yang dibutuhkan untuk melakukan *insert rule* menggunakan aplikasi manajemen *firewall* secara terpusat lebih cepat dibandingkan waktu yang dibutuhkan untuk melakukan *insert rule* secara manual. Hal ini disebabkan karena di dalam melakukan *insert rule* aplikasi manajemen *firewall* secara terpusat langsung mengirimkan *rule* yang dimasukkan *administrator* menuju seluruh *server* tujuan sedangkan waktu yang lebih lama pada konfigurasi secara manual terjadi karena di dalam melakukan *insert rule administrator* harus melakukan *remote* satu persatu terhadap seluruh *server* yang dikonfigurasi.

BAB VI
PENUTUP

Berdasarkan hasil perancangan, implementasi, dan pengujian yang dilakukan pada aplikasi manajemen *firewall* secara terpusat, maka dapat diambil kesimpulan secara keseluruhan dan saran-saran untuk pengembangan sistem selanjutnya.

6.1 Kesimpulan

1. Purwarupa aplikasi manajemen *firewall* secara terpusat dibangun berbasis web dengan memanfaatkan SSH sebagai jalur komunikasi yang dipakai untuk melakukan *remote* terhadap *server* yang dikonfigurasi. Purwarupa tersebut diimplementasikan ke dalam *server* yang menjadi pusat kontrol untuk seluruh *server* yang dikonfigurasi di dalam topologi yang dibangun secara *Local area Network* (LAN).
2. Untuk melakukan *remote* terhadap sistem operasi yang heterogen, di dalam aplikasi manajemen *firewall* secara terpusat memanfaatkan protokol SSH sebagai jalur komunikasinya. SSH secara *default* terdapat di dalam sistem operasi yang berbasis UNIX berbeda dengan sistem operasi WINDOWS yang di dalamnya belum terdapat SSH, sehingga untuk melakukan *remote* ke dalam sistem operasi windows perlu menambahkan aplikasi SSH server seperti Bitvise SSH server.
3. Hasil kinerja dari purwarupa aplikasi manajemen *firewall* secara terpusat menunjukkan bahwa purwarupa aplikasi berjalan dengan baik. Hal tersebut dapat dilihat dari hasil pengujian tahap kedua yang menunjukkan bahwa perbandingan waktu yang dibutuhkan ketika melakukan *insert rule* menggunakan aplikasi manajemen *firewall* secara terpusat menunjukkan rata-rata waktu yang dibutuhkan sekitar 1.656 *sec* dimana waktu yang dibutuhkan ini lebih cepat dibanding konfigurasi secara manual yang menunjukkan rata-rata waktu yang dibutuhkan sekitar 356.2 *sec.*

6.2 Saran

1. Untuk pengembangan selanjutnya aplikasi manajemen *firewall* secara terpusat dapat mendukung *multipatform* sistem operasi.
2. Untuk pengembangan selanjutnya aplikasi manajemen *firewall* secara terpusat dapat menggunakan seluruh *options* konfigurasi dari IPTABLES, IPFW dan

ADVANCED WINDOWS FIREWALL sehingga tidak hanya terbatas *source dan destination addrres*, *source* dan *destination port*, *interface*, *protocol* dan *target*.

3. Untuk pengembangan selanjutnya aplikasi manajemen *firewall* secara terpusat dapat ditambahkan fitur *log* untuk menyimpan konfigurasi *rule* dari masing-masing *host.* Hal tersebut diperlukan, apabila salah satu *server* yang dikonfigurasi *down* administrator tidak perlu melakukan input secara manual untuk mengembalikan *rule firewall* dari *server* yang *down* tersebut.

DAFTAR PUSTAKA

[SSZ-05] Suehring Steve, Ziegler Robert. 2005, "*Linux Firewall"*, Sams Publishing, United States.

[WBM-13] Webmin. 2013,Webmin Website, http://www.webmin.com/ [diakses 7 Oktober 2013].

[SSH-13] OpenSSH. 2009,OpenBSD,http://www.openssh.com/ [diakses 10 Oktober 2013].

[LSH-13] SSH2. 2013, The PHP Group, http://pecl.php.net/package/ssh2 [diakses 10 Oktober 2013].

[MFW- 10] Ilmawan, Rosyadi A. 2010, "Manajemen Firewall Berbasis Web", Skripsi: Fakultas Teknologi Industri Universitas Pembangunan Nasional Surabaya.

[BVS-13] Bitvise Limited. 2013, Bitvise Website. http://www.bitvise.com/ssh-server [diakses 7 Oktober 2013].

[SHA-10] Valentino febri, Apa itu SHA1. http://integerarea.blogspot.com/2010/10/apa-itu-sha1.html [diakses 5 Oktober 2013].

[SSL-13] Secure Socket Layer (SSL). http://www.pasarhosting.com/informasi/knowledge-base/detail/page/1/back_to/kb-trial/content/secure-socket-layer-ssl/ [diakses 7 Oktober 2013].

[NSH-10] McIllece J, Weston B, Holtzman J, Lindsay G, Plett C, Bishop D, Carncross K. 2010, "Network Shell Technical Reference", Microsoft Corporation, US.

[BSD-13] FreeBSD Foundation. 2013, "The FreeBSD Documentation Project", FreeBSDHandbook.

[CCN-14] Webopedia. 2014, Centralized Network. http://www.webopedia.com/TERM/C/centralized_network.html [diakses 11 Januari 2014].

[WBP-14] Techopedia. 2014, Web Programming. http://www.techopedia.com/definition/23898/web-programming [diakses 11 Januari 2014].

[WBS-14] Webopedia. 2014, Web Services. http://www.webopedia.com/TERM/W/Web_Services.html [diakses 11 Januari 2014].

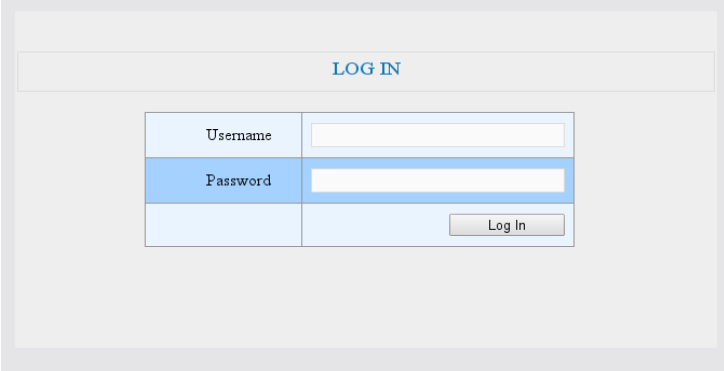
[NSC-14] Webopedia. 2014, network security. http://www.webopedia.com/TERM/N/network_security.html [diakses 11 Januari 2014].

[IDF- 00] Bellovin, Steve M. 2000, "*Distributed Firewall*", http://www.cis.upenn.edu/~dsl/STRONGMAN/Papers/df.pdf [diakses 8 Oktober 2013]

LAMPIRAN

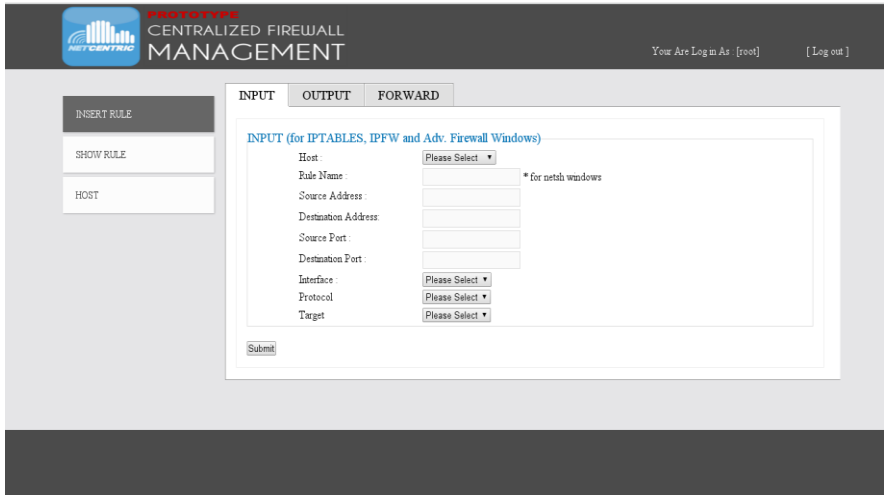
Lampiran 1: Antar Muka Sistem

1. Log in

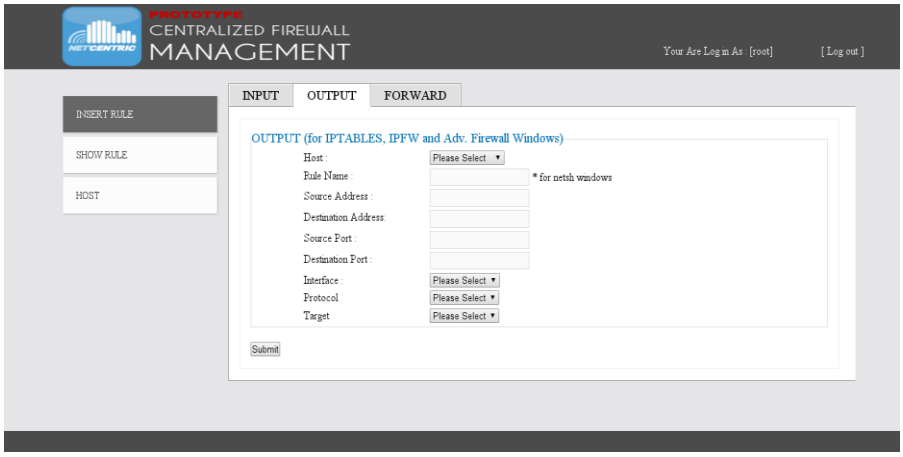


Gambar L1. 1 Antarmuka halaman Log in

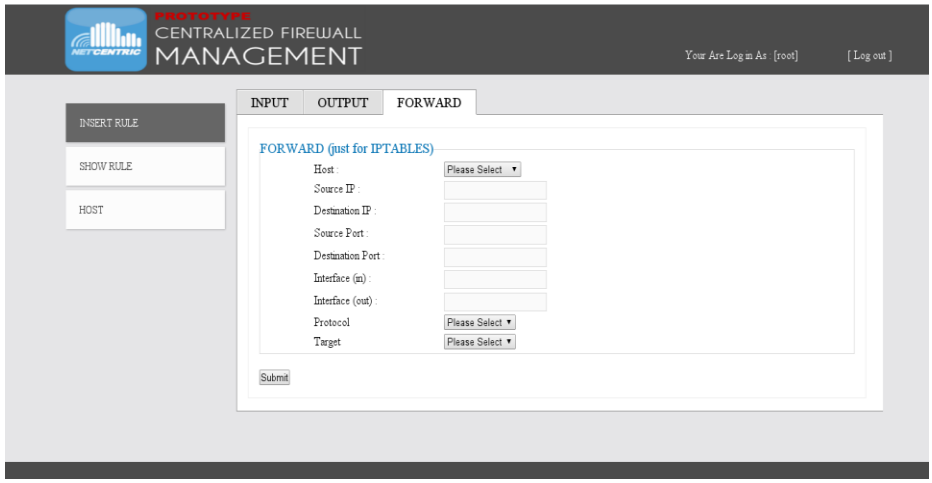
2. INSERT RULE



Gambar L1.2 Antarmuka Halaman Insert Rule (INPUT)

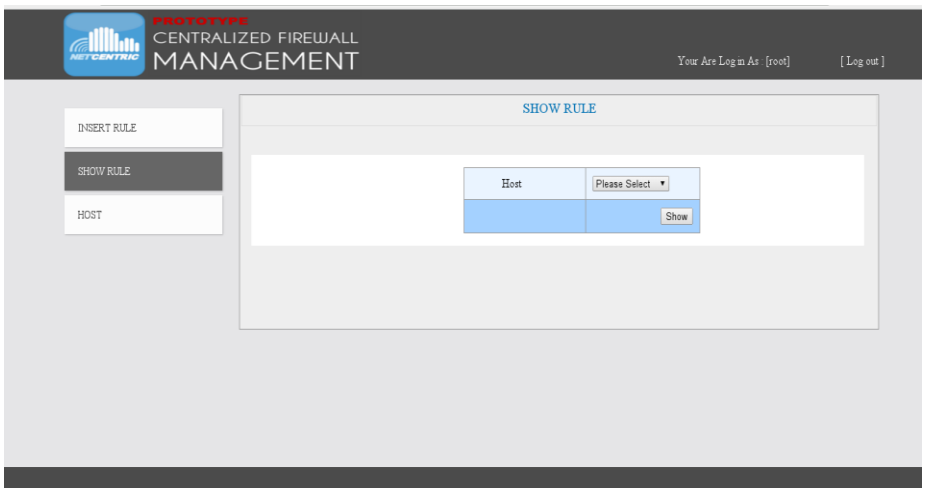


Gambar L1.3 Antarmuka Halaman Insert Rule (OUTPUT)

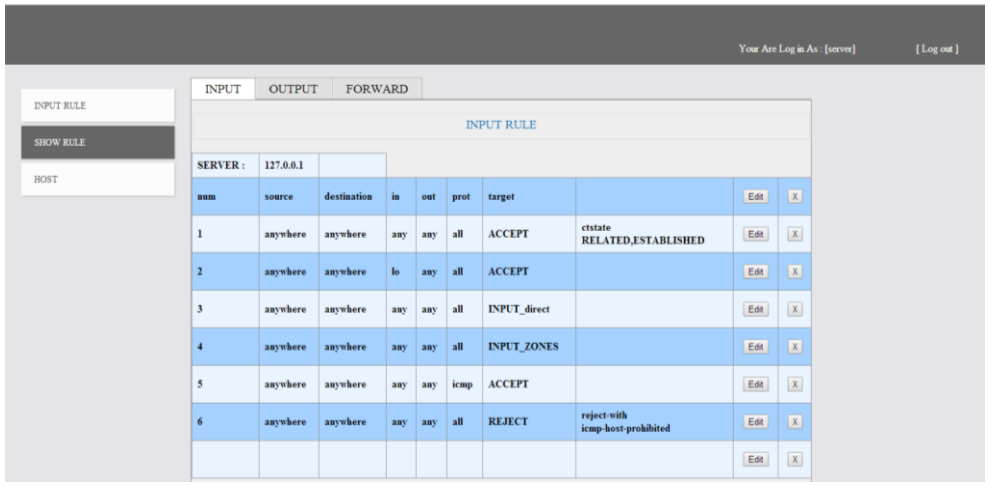


Gambar L1.4 Antarmuka Halaman Insert Rule (FORWARD)

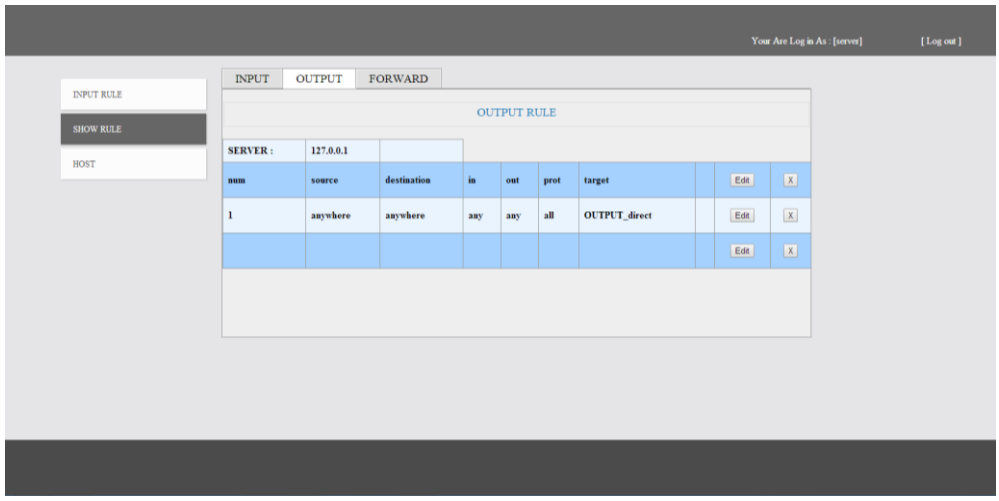
3. SHOW RULE



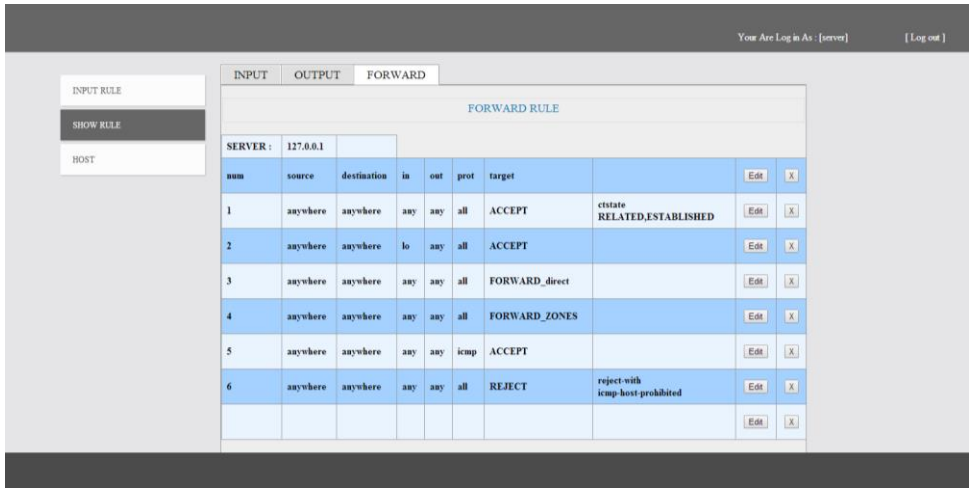
Gambar L1.5 Antarmuka Halaman Utama Show Rule



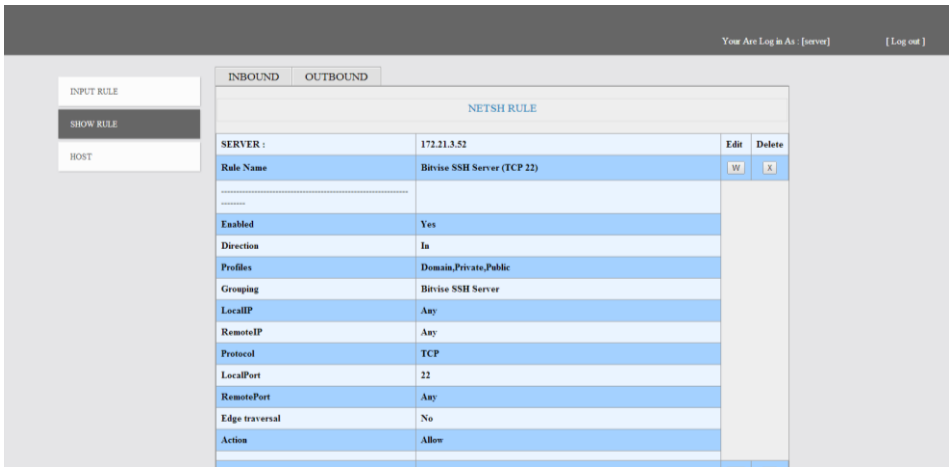
Gambar L1.6 Antarmuka Halaman Show Rule IPTABLES (INPUT)



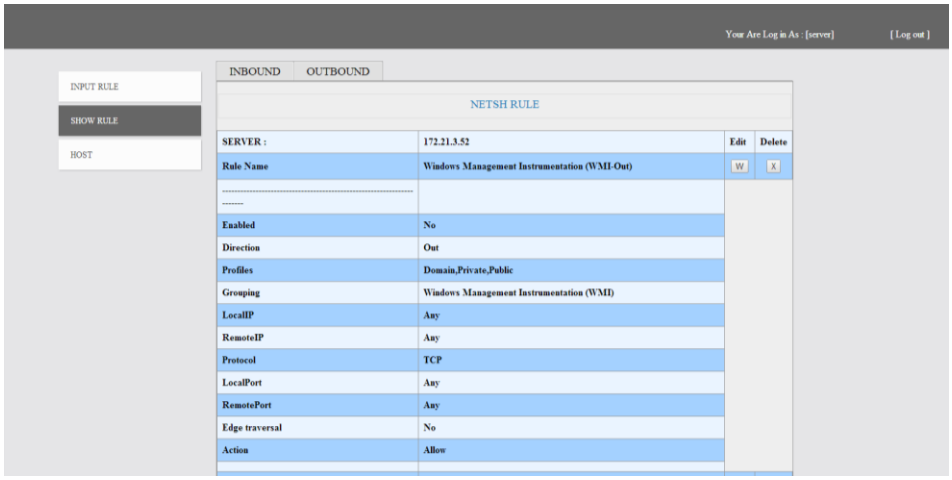
Gambar L1.7 Antarmuka Halaman Show Rule IPTABLES (OUTPUT)



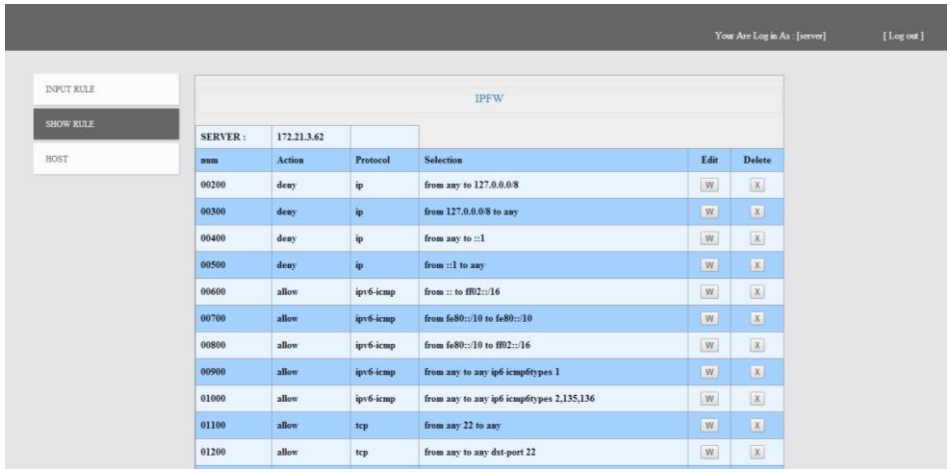
Gambar L1.8 Antarmuka Halaman Show Rule IPTABLES (FORWARD)



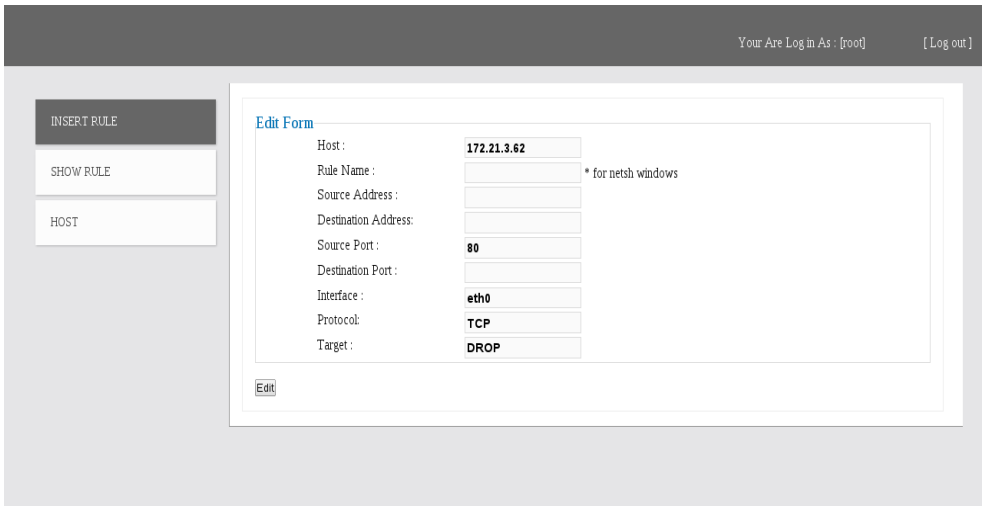
Gambar L1.9 Antarmuka Halaman *Show Rule Windows Firewall* (*INPUT*)



Gambar L1.10 Antarmuka Halaman *Show Rule Windows Firewall* (*OUTPUT*)

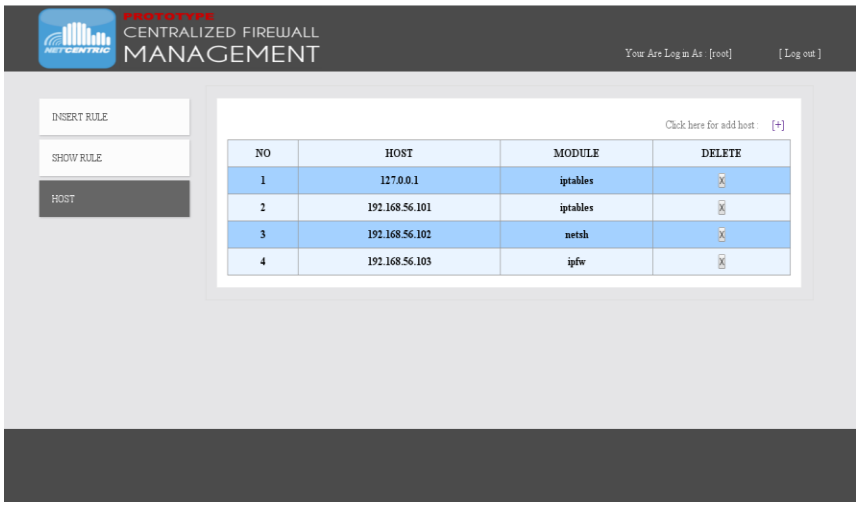


Gambar L1.11 Antarmuka Halaman *Show Rule IPFW*

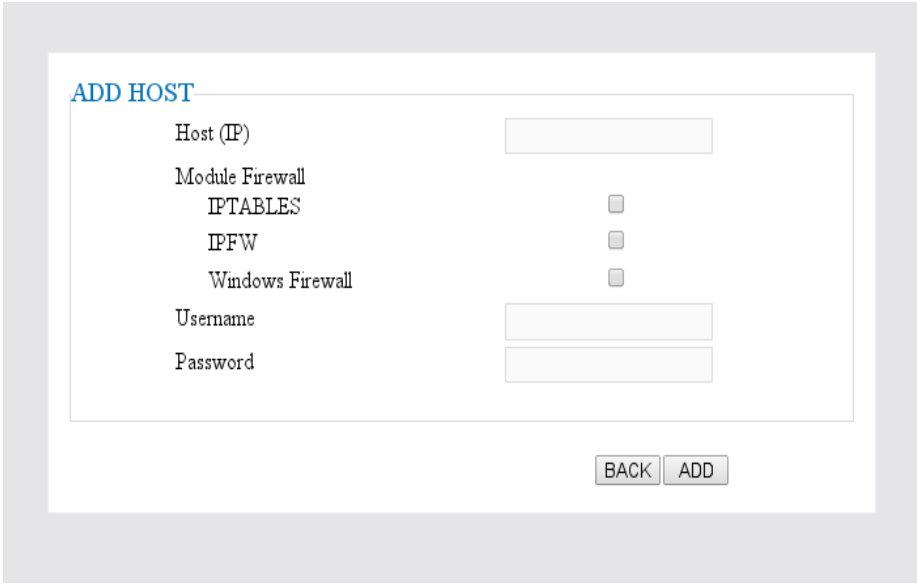


Gambar L1.12 Antarmuka *form Update Rule*

4. HOST



Gambar L1.13 Antarmuka Halaman Utama *HOST*



Gambar L1.14 Antarmuka Halaman *Form* Menambahkan *HOST*