SKRIPSI

OTOMATISASI DETEKSI GANGGUAN KEAMANAN JARINGAN KOMPUTER DENGAN NOTIFIKASI SMS (SHORT MESSAGING SERVICE)

Disusun Oleh:

EGAL DERIEKSEN

NIM : 0512633

JURUSAN TEKNIK ELEKTRO S-1

KONSENTRASI TEKNIK KOMPUTER & INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2012

LEMBAR PERSETUJUAN

OTOMATISASI DETEKSI GANGGUAN KEAMANAN JARINGAN KOMPUTER DENGAN NOTIFIKASI SMS (SHORT MESSAGING SERVICE)

SKRIPSI

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

Disusun oleh :

EGAL DERIEKSEN

NIM : 05.12.633

Diperiksa dan Disetujui

Mengetahui,
Ketua Program Studi Teknik Elektro S-1

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y 1018800189

Dosen Pembimbing I	Dosen Pembimbing II
Sotyohadi,ST	**Sonny Prasetio,ST.MT**
NIP.P 1039700309	**NIP.P 1031000433**

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2012

SURAT PERNYATAAN ORISINALITAS

Yang bertanda tangan dibawah ini :

Nama : Egal Derieksen

NIM :05.12.633

Program Studi :Teknik Elektro S-1

Konsentrasi :Teknik komputer dan Informatika

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

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

Malang 28 Agustus 2012

Yang membuat pernyataan,

EGAL DERIEKSEN

NIM:05.12.633

ABSTRAK

Keamanan jaringan merupakan suatu permasalahan yang harus diperhatikan dalam perancangan suatu jaringan komputer atau server. Gangguan atau penyusupan jaringan seringkali terjadi ketika sistem keamanan yang dimiliki lemah atau bahkan tidak mampu untuk mendeteksi adanya gangguan. Akibatnya sistem jaringan akan tidak dapat mengirim data-data yang dibutuhkan oleh klien.

Sistem AIDS (Automatic Intrusion Detection System) dengan notifikasi SMS ini dapat mengirimkan notifikasi melalui pesan singkat (Short Messaging Service) atau SMS kepada network administrator serta menampilkan nomor IP intruder dalam web, sehingga gangguan yang terjadi akan segera diketahui. Pesan singkat ini akan berisikan alert atau jenis gangguan yang diperoleh dari sistem IDS (Intrusion Detection System) yang dipakai.

Kata Kunci : Snort , gammu , DoS , DDoS , MySQL , PHP

ABSTRACT

Network security is a problem which need to be concerned in making a computer network or server. Disturbance or intruder of network happened frequently when the network security is down or can not be able to detect the obstruction. Consequently, the network system can not send data which is needed by client.

AIDS (Automatic Intrusion Detection System) by using SMS' notification send the notification through short message (SMS) to the network administrator and also deliver intruder IP number in the web, so the obstruction will detect immediately. His short message contains alert or sort of disturbance gained from IDS system which is used.

Keyword : Snort , gammu , DoS , DDoS , MySQL , PHP

KATA PENGANTAR

Dengan mengucap syukur kehadirat Tuhan Yang Maha Esa yang dengan segala Kasih dan Anugerah – Nya, telah memberikan kekuatan, kesabaran, bimbingan dan perlindungan sehingga penulis dapat menyelesaikan laporan skripsi dengan judul :

OTOMATISASI DETEKSI GANGGUAN KEAMANAN JARINGAN KOMPUTER DENGAN NOTIFIKASI SMS (SHORT MESSAGING SERVICE)

Pembuatan skripsi ini disusun guna memenuhi syarat akhir kelulusan pendidikan jenjang Strata–1 di Institut Teknologi Nasional Malang. Dalam penyusunan skripsi ini penulis banyak mendapat bantuan baik moril maupun materiil, saran dan dorongan semangat dari berbagai pihak, untuk itu penulis mengucapkan terima kasih kepada :

1. Bapak Ir. Soeparno Djiwo, MT., selaku Rektor ITN Malang.
2. Bapak Ir. Sidik Noertjahjono, MT., selaku Dekan Fakultas Teknologi Industri.
3. Bapak Ir. Yusuf Ismail Nahkoda, MT., selaku Ketua Jurusan Teknik Elektro S–1 ITN Malang.
4. Bapak Dr. Eng. Aryuanto S, ST, MT., selaku Sekertaris Jurusan Teknik Elektro S–1 ITN Malang.
5. Bapak Sotyohadi, ST., selaku Dosen Pembimbing I.
6. Bapak Sonny Prasetio, ST.MT., selaku Dosen Pembimbing II.
7. Kedua orang tua yang telah Ikhlas memberikan doa dan dukungan.
8. Sahabat-sahabati dan sumua pihak yang telah membantu penulis dalam menyelesaikan skripsi ini yang tidak bisa penulis sebutkan satu persatu.

Penulis menyadari bahwa laporan ini masih banyak yang perlu disempurnakan. Oleh sebab itu kritik dan saran yang membangun sangat diharapkan.

Akhir kata, penulis mohon maaf kepada semua pihak bilamana selama penyusunan skripsi ini penyusun membuat kesalahan secara tidak sengaja dan semoga skripsi ini dapat bermanfaat bagi kita semua. Amin.

Malang, Februari 2012

Penulis

DAFTAR ISI

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Keamanan jaringan komputer sebagai bagian dari sebuah sistem sangat penting untuk menjaga *validitas* dan *integritas* data serta menjamin ketersediaan layanan bagi penggunanya. Sistem keamanan jaringan komputer harus dilindungi dari segala macam serangan dan usaha-usaha penyusupan atau pemindaian oleh pihak yang tidak berhak.

Sistem deteksi penyusup jaringan yang ada saat ini umumnya mampu mendeteksi jenis serangan tetapi tidak memiliki *interaktifitas* dengan *administrator* pada saat *administrator* tidak sedang mengaministrasi sistemnya. ini merupakan suatu hal yang tidak efektif terutama pada saat sistem berada dalam kondisi kritis.

Selain itu sistem pertahanan terhadap aktivitas gangguan saat ini umumnya dilakukan secara manual oleh a*dministrator*. hal ini mengakibatkan *integritas* sistem bergantung pada ketersediaan dan kecepatan *administrator* dalam merespon gangguan. apabila terjadi malfungsi *administrator* tidak dapat lagi mengakses sistem secara *remote* sehingga tidak akan dapat melakukan pemulihan sistem dengam cepat.

Oleh karena itu dibutuhkan suatu sistem yang dapat menanggulangi ancaman yang mungkin terjadi secara optimal dalam waktu yang cepat dan secara otomatis sehingga memungkinkan *administrator* mengakses sistem walaupun terjadi malfungsi jaringan.hal ini akan mempercepat proses penanggulangan gangguan serta pemulihan sistem atau layanan.

1.2 Rumusan Masalah

Untuk menyelesaikan masalah yang telah dijabarkan pada latar belakang, maka dibuat perumusan masalah sebagai berikut :

1. Bagaimana membuat sebuah Sistem yang secara otomatis mengirimankan SMS *(Short Message Servive)* ketika terjadi serangan terhadap jaringan dari pihak luar yang tidak berwenang, sehingga *administrator* bisa dengan cepat mengetahui adanya gangguan?
2. Bagaimana *memonitoring* lalu lintas didalam sebuah jaringan komputer?

1.3 Tujuan

Tujuan dari Skripsi ini adalah membuat sebuah sistem untuk *otomatisasi* pengiriman pesan jika terjadi gangguan pada sebuah jaringan komputer.

1.4 Batasan Masalah

Untuk mengarahkan pokok bahasan agar lebih fokus, maka dalam penulisan Skripsi ini dilakukan pembatasan pada pokok bahasan sebagai berikut :

a. Mendeteksi adanya gangguan jaringan, dan setelah terdeteksi kemudian sistem akan otomatis mengirimkan SMS ke *Administrator.*
b. *Intrusi* atau gangguan yang dideteksi adalah DoS *(Denial of Service)*, DDoS *(Distributed Denial of Service)* dan *Backdor.*
c. Menggunakan Sistem operasi Linux Ubuntu 10.04.
d. Menggunakan Modem GSM *(Global System for Mobile Communications).*
e. Menggunakan *Gammu* sebagai penghubung perangkat *Mobile.*

1.5 Manfaat

Manfaat dari Skripsi ini adalah menjadikan waktu yang diperlukan untuk mengatasi adanya gangguan lebih cepat dan efektif.

1.6 SISTEMATIKA PENULISAN

Pada penulisan skripsi ini penulis menggunakan sistematika penulisan sebagai berikut :

Bab I PENDAHULUAN

Menjelaskan latar belakang, rumusan masalah, batasan masalah, tujuan,manfaat, dan sistematika penulisan.

Bab II DASAR TEORI

Memberikan dasar teori yang mendukung dalam perancangan dan pembuatan otomatisasi IDS (*Intrusion Detection Sistem*) dengan *notifikasi* SMS. Teori yang dibahas meliputi : dasar keamanan jaringan komputer, sumber lubang keamanan, *monitoring sistem*, *flooding*, dan SMS *Gateway*.

Bab III PERANCANGAN DAN PEMBUATAN SISTEM

Bab ini membahas tentang proses pembuatan sistem secara keseluruhan, baik *konfigurasi* paket yang digunakan, maupun peembuatan sistem *notifikasi.*

Bab IV PENGUJIAN SISTEM

Bab ini akan menjelaskan tentang pengujian terhadap sistem dan hasil analisa dari pengujian yang dilakukan.

Bab V PENUTUP

Bab ini berisi tentang kesimpulan dari pembahasan bab–bab sebelumnya dan saran-saran serta beberapa kemungkinan pengembangan dan penyempurnaan Skripsi ini.

BAB II
TINJAUAN PUSTAKA

2.1. *Network Security* secara umum

Hos atau komputer yang terhubung ke *network*, mempunyai ancaman keamanan lebih besar daripada *host* yang tidak terhubung kemana-mana. Dengan mengendalikan *network security*, risiko tersebut dapat dikurangi. Namun *network security* biasanya bertentangan dengan *network access*, yaitu bila *network access* semakin mudah, maka *network security* makin rawan, dan bila *network security* makin baik, *network access* makin tidak nyaman. Suatu *network* didesain sebagai komunikasi data *highway* dengan tujuan meningkatkan akses ke sistem komputer, sementara *security* didesain untuk mengontrol akses. Penyediaan *network security* adalah sebagai aksi penyeimbang antara *open access* dengan *security*.

Disini *network* dikatakan sebagai *highway*, karena menyediakan akses yang sama untuk semua, baik pengguna normal ataupun tamu yang tidak diundang. Sebagai analogi, keamanan di rumah dilakukan dengan cara memberi kunci di pintu rumah, tidak dengan cara memblokir jalan di depan rumah. Hal seperti ini juga diterapkan pada *network security*. Keamanan dijaga untuk (setiap) *host-host* tertentu, tidak langsung pada *network*nya.

Keamanan untuk daerah dimana orang saling mengenal, pintu biasanya dibiarkan terbuka. Sedangkan di kota besar, pintu rumah biasanya menggunakan mekanisme keamanan tambahan. Begitu pula yang dilakukan pada *network*. Untuk jaringan yang menghubungkan *host-host* yang aman dan dikenal, tingkat keamanan *host* bisa tidak dijaga terlalu ketat. Bila jaringan terhubung ke jaringan lain yang lebih terbuka, dan membuka peluang akses oleh *host* yang tidak aman atau tidak dikenal, maka tidak bisa tidak, *hosthost* di jaringan tersebut membutuhkan pengamanan lebih. Ini bukan berarti keterbukaan hanya membawa akibat buruk, sebab banyaknya fasilitas yang ditawarkan dengan keterbukaan jaringan ini merupakan nilai lebih yang sangat membantu kemajuan *network*. Jadi *network security* merupakan harga yang harus dibayar dari kemajuan jaringan komputer.

2.2. Sumber Lubang Keamanan

Lubang keamanan (*security hole*) dapat terjadi karena beberapa hal; salah disain (*design flaw*), salah implementasi, salah konfigurasi, dan salah penggunaan.

2.2.1. Salah Desain

Lubang keamanan yang ditimbulkan oleh salah desain umumnya jarang terjadi. Akan tetapi apabila terjadi sangat sulit untuk diperbaiki. Akibat disain yang salah, maka biarpun dia diimplementasikan dengan baik, kelemahan dari sistem akan tetap ada.

lubang keamanan yang dapat dikategorikan kedalam kesalahan disain adalah disain urutan nomor (*sequence numbering*) dari paket TCP/IP. Kesalahan ini dapat dieksploitasi sehingga timbul masalah yang dikenal dengan nama *"IP spoofing"*, yaitu sebuah *host* memalsukan diri seolah-olah menjadi *host* lain dengan membuat paket palsu setelah mengamati urutan paket dari *host* yang hendak diserang. Bahkan dengan mengamati cara mengurutkan nomor *packet* bisa dikenali sistem yang digunakan.

2.2.2. Implementasi kurang baik

Lubang keamanan yang disebabkan oleh kesalahan implementasi sering terjadi. Banyak program yang diimplementasikan secara terburu-buru sehingga kurang cermat dalam pengkodean. Akibatnya cek atau *testing* yang harus dilakukan menjadi tidak dilakukan. Sebagai contoh, seringkali batas (*bound*) dari sebuah *array* tidak dicek sehingga terjadi yang disebut *out-of-bound array* atau *buffer overflow* yang dapat dieksploitasi (misalnya *overwrite* ke *variable* berikutnya). Lubang keamanan yang terjadi karena masalah ini sudah sangat banyak, dan yang mengherankan terus terjadi, seolah-olah para programmer tidak belajar dari pengalaman.

2.2.3. **Salah konfigurasi**

Meskipun program sudah diimplementasikan dengan baik, masih dapat terjadi lubang keamanan karena salah konfigurasi. Contoh masalah yang disebabkan oleh salah konfigurasi adalah berkas yang semestinya tidak dapat diubah oleh pemakai secara tidak sengaja menjadi "*writeable*". Apabila berkas tersebut merupakan berkas yang penting, seperti berkas yang digunakan untuk menyimpan *password*, maka efeknya menjadi lubang keamanan. Kadangkala sebuah komputer dijual dengan konfigurasi yang sangat lemah. Ada masanya *workstation Unix* di perguruan tinggi didistribusikan dengan berkas */etc/aliases* (berguna untuk mengarahkan *e-mail*), */etc/utmp* (berguna untuk mencatat siapa saja yang sedang menggunakan sistem) yang dapat diubah oleh siapa saja. Contoh lain dari salah konfigurasi adalah adanya program yang secara tidak sengaja diset menjadi *"setuid root"* sehingga ketika dijalankan pemakai memiliki akses seperti *super user (root)* yang dapat melakukan apa saja.

2.2.4. **Salah menggunakan program atau sistem**

Salah penggunaan program dapat juga mengakibatkan terjadinya lubang keamanan. Kesalahan menggunakan program yang dijalankan dengan menggunakan *account root (super user)* dapat berakibat fatal. Sering terjadi cerita horor dari sistem *administrator* baru yang teledor dalam menjalankan perintah "rm -rf" di sistem UNIX (yang menghapus berkas atau direktori beserta sub direktori di dalamnya). Akibatnya seluruh berkas di sistem menjadi hilang mengakibatkan *Denial of Service (DoS).* Apabila sistem yang digunakan ini digunakan bersama-sama, maka akibatnya dapat lebih fatal lagi. Untuk itu perlu berhati-hati dalam menjalan program, terutama apabila dilakukan dengan menggunakan *account administrator* seperti *root* tersebut.

Kesalahan yang sama juga sering terjadi di sistem yang berbasis *MS-DOS.* Karena sudah mengantuk, misalnya, ingin melihat daftar berkas di sebuah direktori dengan memberikan perintah *"dir *.*"* ternyata salah

memberikan perintah menjadi *"del *.*"* (yang juga menghapus seluruh file di direktori tersebut).

Insiden-insiden yang mungkin terjadi dalam keamanan jaringan komputer salah satunya adalah *Denial of Service* (DoS).

2.3. Pemantau adanya serangan

Sistem pemantau (*monitoring system*) digunakan untuk mengetahui adanya tamu tak diundang *(intruder)* atau adanya serangan *(attack)*. Nama lain dari sistem ini adalah *"intruder detection system" (IDS)*. Sistem ini dapat memberitahu *administrato*r melalui *email* maupun melalui mekanisme lain seperti melalui *pager*.

Ada berbagai cara untuk memantau adanya *intruder*. Ada yang sifatnya aktif dan pasif. *IDS* cara yang pasif misalnya dengan memonitor l*ogfile*. Contoh *software IDS* antara lain:

- **Autobuse**, mendeteksi *probing* dengan memonitor *logfile*
- **Shadow** dari SANS
- **Snort**, mendeteksi pola (*pattern*) pada paket yang lewat dan mengirimkan *alert* jika pola tersebut terdeteksi. Pola-pola atau *rules* disimpan dalam berkas yang disebut *library* yang dapat dikonfigurasi sesuai dengan kebutuhan.

2.3.1. Snort IDS (Intrusion Detection System)

Snort yang dapat diperoleh di *http://www.snort.org* biasanya di sebut sebagai *Network Intrusion Detection System (NIDS). Snort* sendiri adalah *Open Source* yang tersedia di berbagai variasi *Unix* (termasuk *Linux*) dan juga *Microsoft Windows.*

Sebuah *NIDS* akan memperhatikan seluruh segmen jaringan dimana dia berada, berbeda dengan *host based IDS* yang hanya memperhatikan sebuah mesin dimana *software host based IDS* tersebut di pasang. Secara sederhana, sebuah *NIDS* akan mendeteksi semua serangan yang dapat melalui jaringan komputer *(Internet* maupun *IntraNet).*

a. Fungsi-fungsi Snort

1. NIDS

NIDS (*Network Intrusion Detection System*) adalah sistem pendeteksi *intrusi* yang menagkap paket data yang berjalan pada media jaringan (kabel atau nirkabel), dan menggabungkannya ke sebuah *signature database*. Bergantung pada paket apa yang masuk, *signature* penyusup (*intruder*), *alert* yang dihasilkan akan masuk ke *database* ataupun pada *file log* yang sudah di atur. Fungsi utama *snort* adalah sebagai *NIDS.*

2. HIDS

Host-based Intrusion Detection System dimasukkan sebagai sebuah agen pada sebuah *host*. Sistem-sistem pendeteksi *intrusi* ini dapat dilihat pada sistem dan aplikasi *log* untuk mengetahui aktifitas *intruder*. Sistem ini *reactive*, artinya bahwa sistem-sistem ini memberitahukan ketika hanya terjadi suatu *intrusi.*

3. Signature

Signature adalah pola yang dapat ditemui pada sebuah paket data. *Signature* digunakan untuk mendeteksi satu atau bermacam-macam bentuk serangan. Contohnya kehadiran *"scripts/iisadmin"* dalam sebuah paket data yang akan menuju *web server.*

Signature biasa dihadirkan dalam bagian-bagian yang berbeda dari sebuah paket data, tergantung pada karakter atau sifat serangan. Contohnya pada *IP header*, *Transport Layer header* (*TCP dan UDP header*) dan *header* layer aplikasi.

Biasanya *IDS* bergantung pada *signature* untuk mengetahui aktifitas dari *intruder*. Beberapa vendor *IDS* membutuhkan *update* dari *vendor* untuk menambah *signature* baru dimana tipe serangan ditemukan.

4. Alert

Alert adalah semacam pemberitahuan *user* dari aktifitas *intruder*. Ketika *IDS* mendeteksi adanya *intruder*, IDS harus memberitahukan seorang *network administrator* tentang hal ini

dengan menggunakan *alert*. *Alert* bisa dalam bentuk *logging*, *e-mail notification*, dan sebagainya. *Alert* bsa disimpan dalam sebuah *file* ataupun *database*.

Snort dapat menghasilkan *alerts* dalam berbagai bentuk dan dikontrol oleh *output plug-in*. Misalnya, *alert* dapat ditampilkan kedalam halaman *web*, yang mana nantinya akan dapat di akses oleh seorang *network administrator* untuk kemudian dapat mengetahui serangan yang terjadi.

5. Logs

   Pesan *log* biasanya disimpan dalam *file*. Secara *default*, *snort* akan menyimpan *file* tersebut kedalam *direktori /var/log/snort*. Akan tetapi lokasi dari pesan *log* ini dapat diubah sesuai keinginan *user*. Pesan *log* dapat berupa *file text* atau *binary format*.

6. Sensor

   Sensor merupakan sebuah sistem deteksi *intrusi* yang dijalankan dan digunakan untuk *"sense"* sebuah jaringan. *Sensor* yang dimaksud disini adalah berhubungan dengan alamat *IP* sebuah komputer tujuan lain dimana *snort* dijalankan.

b. Keuntungan dan Kelemahan Snort NIDS

Keuntungan :

1. Memberikan perlindungan yang lebih luas dalam pengamanan sistem
2. Membantu memahami apa yang terjadi di dalam sistem
   Dukungan teknis:
   - Melacak aktivitas pemakai dari awal sampai akhir
   - Mengenal dan melaporkan usaha-usaha modifikasi *file*
   - Mengetahui kelemahan konfigurasi sistem
   - Mengenali bahwa sistem telah atau potensial untuk diserang
3. Memungkinkan operasional pengamanan sistem dilakukan oleh staf tanpa keahlian spesifik

4. Membantu penyusunan kebijakan dan prosedur pengamanan sistem

Kelemahan :

1. Bukan solusi total untuk masalah keamanan sistem
2. Tidak bisa mengkompensasi kelemahan:
   - mekanisme identifikasi dan autentifikasi
   - protokol jaringan
   - *integritas* dan kualitas informasi dalam sistem yang dilindungi
3. Masih memerlukan keterlibatan manusia
4. Banyak berasumsi pada teknologi jaringan konvensional, belum bisa menangani teknologi baru (misalnya: fragmentasi paket pada jaringan ATM).

2.4. Flooding

Seorang *intruder* bisa menguarangi kecepatan *network* dan *host-host* yang berada di dalamnya secara *significant* dengan cara terus melakukan *request* atau permintaan terhadap suatu informasi dari sever yang bias menangani serangan classic *Denial Of Service (DoS),* mengirim *request* ke satu *port* secara berlebihan dinamakan *flooding*, kadang hal ini juga disebut *spraying.* Ketika permintaan *flood* ini dikirim ke semua *station* yang berada dalam *network* serangan ini dinamakan *broadcasting.* Tujuan dari kedua serangan ini adalah sama yaitu membuat *network resource* yang menyediakan informasi menjadi lemah dan akhirnya menyerah.

Serangan dengan cara *flooding* bergantung kepada dua faktor yaitu: ukuran dan atau *volume* (*size and/or volume*). Seorang *intruder* dapat menyebabkan *Denial Of Service* dengan cara melempar *file* berkapasitas besar atau *volume* yang besar dari paket yang kecil kepada sebuah sistem. Dalam keadaan seperti itu *network server* akan menghadapi kemacetan terlalu banyak informasi yang diminta dan tidak cukup *power* untuk mendorong data agar berjalan. Pada dasarnya paket yang besar membutuhkan kapasitas proses yang besar pula, tetapi secara tidak normal paket yang kecil

dan sama dalam volume yang besar akan menghabiskan *resource* secara percuma, dan mengakibatkan kemacetan.

Intruder sering kali menggunakan serangan *flooding* ini untuk mendapatkan akses ke sistem yang digunakan untuk menyerang *network* lainnya dalam satu serangan yang dinamakan *Distributed Denial Of Service (DDOS).* Serangan ini seringkali dipanggil *smurf* jika dikirim melaluli *ICMP* dan disebut *fraggles* ketika serangan ini dijalakan melewati *UDP.*

Suatu *node* (dijadikan *tools*) yang menguatkan *broadcast traffic* sering disebut sebagai *Smurf Amplifiers*, *tools* ini sangat efektif untuk menjalankan serangan *flooding.*

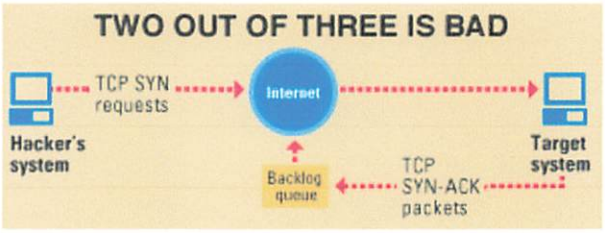
Dengan melakukan *spoofing* terhadap *network* sasaran, seorang *intruder* dapat mengirim sebuah *request* ke *smurf amplifier*, *Network* yang di *amplifiying* (dikuatkan) akan mengirim respon kesetiap *host* di dalam *network* itu sendiri, yang berarti satu *request* yang dilakukan oleh *intruder* akan menghasilkan pekerjaan yang sama dan berulang-ulang pada *network* sasaran, hasil dari serangan ini adalah sebuah *denial of service* yang tidak meninggalkan jejak. Serangan ini dapat diantisipasi dengan cara menolak *broadcast* yang diarahkan pada *router.*

SYN Flooding

Koneksi normal TCP memerlukan tiga paket yang akan dikirim antara *client* dan *server,* yakni :

1. *Client* mengirim paket *SYN.*
2. *Server* mengalokasikan *TCP control block* dan mengirim balik paket *SYN/ACK.*
3. *Server* menunggu *ACK* datang dari *client.*

Cara ini disebut *3-way handshake.* Sepanjang *ACK* ada dari langkah 3 tidak dari *server,* koneksi adalah *state half-open.* Ketika tidak ada *ACK* datang dari *client,* koneksi dalam keadaan *half-open* sampai *TCP time out* setelah beberapa menit. Ketika *TCP time out,* alokasi *control block* menjadi tersedia lagi.

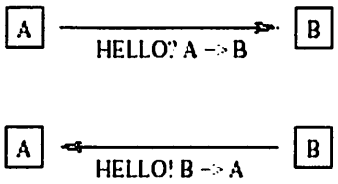


Gambar 2.1 SYN flood

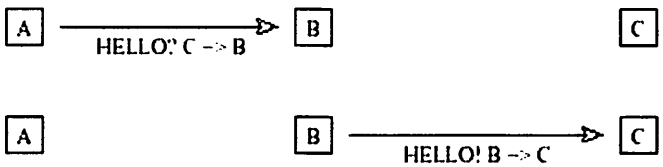
Serangan sederhana (gambar 2.4) adalah A terus mengirimi paket *SYN* dalam *spoof* paket *IP*. Paket *SYN/ACK* akan pergi ke pihak ketiga yang tidak tahu apa-apa (yang akan mendrop paket tersebut), yang membutuhkan paket *ACK* yang tidak dikirim oleh siapa pun. Kasus ini menyebabakan B kehabisan *TCP control block* dengan cepat dengan semua koneksi yang tersedia dalam state *half-open*. Suatu *server* tanpa tersedia *TCP control block* tidak akan mampu untuk menerima koneksi TCP. Beberapa solusi disarankan untuk mengatasi *SYN flood* dengan cara menurunkan waktu *timeout TCP,* meningkatkan jumlah *TCP control block*, *SYN cookies* dapat mengeliminasi kebutuhan penyimpanan informasi pada koneksi *half-open*, dan *firewall* khusus sebagai *buffer* paket *SYN.*

Ping Flooding (Smurf)

Salah satu mekanisme jenis serangan yang baru-baru ini mulai marak digunakan adalah menggunakan "*ping*" ke alamat *broadcast*, ini sering disebut *smurf. Smurf attack* dilakukan dengan menggunakan *ICMP (Internet Control Message Protocol). ICMP* adalah protocol yang digunakan untuk mengontrol *message* yang dikirim. *ECHO REQUEST* dan *ECHO REPLY* adalah dua bagian dari *message.* Program "*ping*" sebagai contoh yang menggunakan *ICMP* untuk mengukur *delay round-trip* antara dua mesin. Gambar 2.1 dapat menerangkan proses ini. A mengirim *message ECHO REQUEST* ke B. B menjawab dengan *message ECHO REPLY.* Notasi "A B" mengartikan *IP source* addres dan *IP destination address* dalam paket *IP* yang membawa *message ICMP.* B mengetahui kemana mengirim *ECHO REPLY* dari melihat *IP source address* dalam paket *IP* yang tiba.

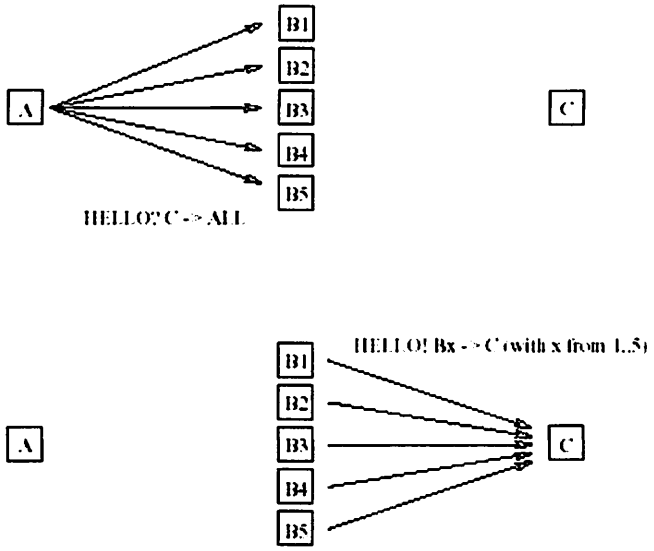


Gambar 2.2. ICMP tanpa IP Spoofing



Gambar 2.3. ICMP dengan IP Spoofing

Jika A *spoof IP source* address, situasi digambarkan pada gambar 2.2. A mengirim sebuah *message ECHO REQUST* ke B, dengan *spoof IP source address* dengan C, bukan A. Sehingga C menerima *ICMP ECHO REPLY* dari B, nampaknya tidak diduga oleh C. Ini belum menjadi suatu ancaman ke C, sebab C dapat dengan mudah membuang *message* tersebut. Situasi ini mejadi berbahaya ketika A mengirim *ECHO REQUEST* ke *broadcast address*. Masing-masing mesin pada jaringan yang menerima mendapatkan *ECHO REQUEST* dan masing-masing mesin meresponnya. Jika A *spoof* ke IP *source address,* suatu mesin yang tidak tahu akan menerima semua *ECHO REPLY* (seperti pada gambar 2.3 dibawah ini). konsekuensinya, *link router* ke C mungkin dapat tersumbat oleh semua *traffic* tersebut.

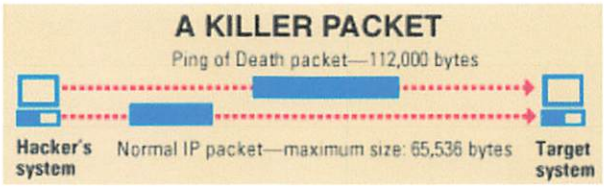


Gambar 2.4 "Smurf" attack

"Smurf" attack bekerja disebabkan *bug* pada banyak implementasi *ICMP.* Sebuah *host* seharusnya tidak boleh mengirim *ECHO REPLY* ke *broadcast* untuk merespon *ECHO REQUEST.* Disayangkan banyak implementasi *ICMP* gagal memeriksa *IP destination address* dari kedatangan *ECHO REQUEST. IP spoofing* adalah suatu kenyataan mutlak berhasilnya suatu serangan. Jika *intruder* gagal untuk spoof *IP source* address, banjir *ECHO REPLY* akan balik ke dirinya sendiri dengan kata lain melumpuhkan (menyerang) diri sendiri.

Ping Of Death

Ping of Death menggunakan program *utility* ping yang ada di sistem operasi komputer. Biasanya ping digunakan untuk men-cek berapa waktu yang dibutuhkan untuk mengirimkan sejumlah data tertentu dari satu komputer ke komputer lain. Panjang maksimum data yang dapat dikirim menurut spesifikasi protokol *IP* adalah 65,536 byte. Pada *Ping of Death* data yang dikirim melebihi maksimum paket yang di ijinkan menurut spesifikasi protokol *IP.* Konsekuensinya, pada sistem yang tidak siap akan menyebabkan sistem tersebut *crash* (tewas), *hang* (bengong) atau *reboot* (*booting* ulang) pada saat sistem tersebut menerima paket yang demikian panjang. Serangan ini sudah tidak baru lagi, semua vendor sistem operasi telah memperbaiki sistem-nya untuk menangani kiriman paket yang *oversize.*



Gambar 2.5. Ping Of Death

Serangan menggunakan Exploits

Serangan menggunakan *exploits*.

Beberapa hal yang harus dipahami sebelum melakukan serangan ini adalah:

a. Serangan membutuhkan Shell Linux (Unix/Comp)

b. Mendapatkan exploits di: *http://packetstormsecurity.nl*

c. Menggunakan/membutuhkan *GCC (Gnu C Compiler)*

1. KOD (Kiss of Death)

Merupakan *tool Denial of Service* yang dapat dugunakan untuk menyerang Ms.Windows pada port 139 (port netbios-ssn). Fungsi utama dari tool ini adalah membuat hang/blue screen of death pada komputer korban. Cara penggunaan:

a. Dapatkan file kod.c

b. Compile dengan Gcc: $ gcc -o kod kod.c

c. Gunakan: $ kod [ip_korban] -p [port] -t [hits]

Kelemahan dari tool ini adalah tidak semua serangan berhasil, bergantung kepada jenis sistem operasi dan konfigurasi server target (misalmya: blocking)

2. BONK/BOINK

Bong adalah dasar dari teardrop (teardrop.c). Boink merupakan Improve dari bonk.c yang dapat membuat crash mesin MS. Windows 9x dan NT

3. Jolt

Jolt sangat ampuh sekali untuk membekukan Windows 9x dan NT. Cara kerja Jolt yaitu mengirimkan serangkaian series of spoofed dan fragmented ICMP Packet yang tinggi sekali kepada korban.

4. **NesTea**

   Tool ini dapat membekukan Linux dengan Versi kernel 2.0. kebawah dan Windows versi awal. Versi improve dari NesTea dikenal dengan NesTea2

5. **NewTear**

   Merupakan varian dari teardrop (teardrop.c) namun berbeda dengan bonk
   (bonk.c)

6. **Syndrop**

   Merupakan 'serangan gabungan' dari TearDrop dan TCP SYN Flooding. Target serangan adalah Linux dan Windows

7. **TearDrop**

   TearDrop mengirimkan paket Fragmented IP ke komputer (Windows) yang terhubung ke jaringan (*network*). Serangan ini memanfaatkan *overlapping ip fragment*, *bug* yang terdapat pada Windowx 9x dan NT. Dampak yang timbul dari serangan ini adalah *Blue Screen of Death.*

2.5. SMS (Short Message Service)

2.5.1. Short Message Service

Short Messaging Service yang lebih dikenal dengan SMS, adalah sebuah teknologi yang memungkinkan untuk menerima maupun mengirim pesan antar telepon bergerak (ponsel). Teknologi baru ini pertama kali diperkenalkan pada tahun 1992 di Eropa oleh ETSI (*Europan Telecommunications Standards Institute*), dan pada awalnya menjadi suatu standar untuk telepon *wireless* yang berbasis GSM (*Global System for Mobile Communication*). Namun, teknologi lain seperti CDMA dan TDMA pun memasukkan SMS ini sebagai fitur standar mereka.

Sebagai mana namanya, SMS yang berarti layanan pesan pendek, maka besar data
yang dapat ditampung oleh SMS ini sangatlah terbatas. Untuk satu SMS yang dikirimkan, hanya dapat menampung paling banyak sebesar 140 bites

atau sekita 1120 bites. Bila diubah kedalam bentuk karakter, maka untuk satu SMS hanya dapat berisi paling banyak 160 karakter untuk karakter latin, dan 70 karakter untuk non-latin seperti karakter CIna maupun Jepang.

Berikut ini adalah jenis data yang dapat dibawa oleh SMS :

- Pesan *text*
- *Ringtone*/Suara
- Gambar, dapat berupa *wallpaper*, logo operator, maupun animasi
- *Bussiness Card* seperti *VCards*.
- Konfigurasi *WAP*

Beberapa alasan menggunakan SMS atau alasan yang menjadikan SMS begitu popular saat ini :

- SMS dapat dibaca maupun dikirimkan, kapanpun dan dimanapun anda berada
- Anda dapat mengirimkan SMS ke nomor tujuan, meskipun nomor ponsel tujuan anda tidak aktif. Hal ini dikarenakan SMS memiliki masa tunggu. Jadi selama masa tunggu SMS tersebut belum habis, SMS akan terus dikirmkan ke nomor tujuan meskipun terlambat.
- SMS tidak memerlukan atau menimbulkan suara, agar lawan dapat mengerti pesan yang dikirimkan. Berbeda dengan panggilan langsung, yang tentunya akan mengganggu orang disekitarnya.
- SMS dianggap lebih murah dan praktis, dibanding berbicara langsung melalui telepon.

2.5.2. Mekanisme Kerja SMS

Ketika anda mengirimkan SMS ke suatu nomor tertentu, SMS yang dikirimkan tidak akan langsung dikirmkan ke nomor tersebut, namun akan masuk terlebih dahulu ke

SMS *Center* (SMSC) operator telepon yang digunakan. SMS *Center* sendiri dapat diartikan sebagai sebuah *server* yang bertanggung jawab pada proses pengiriman SMS dalam suatu operator. SMS yang dikirmkan dari suatu ponsel akan masuk ke SMSC ini, kemudian baru diteruskan ke

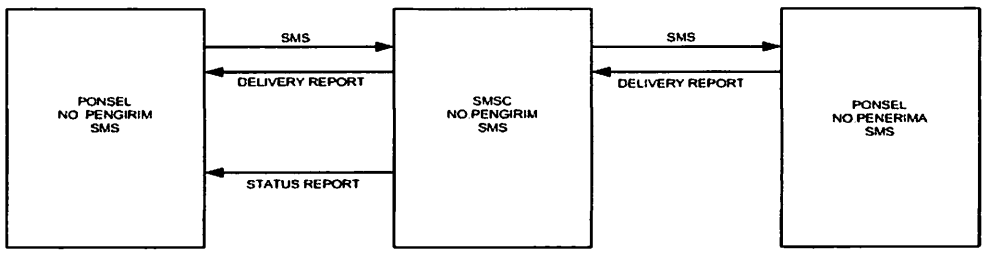
nomor tujuan. Bila nomor yang dituju ternyata sedang mati/*offline*, SMSC ini akan menyimpan SMS tersebut untuk sementara waktu, hingga nomor tujuan itu hidup kembali. Lamanya waktu penyimpanan SMS, sangat tergantung dari lamanya waktu yang telah ditetapkan oleh operator untuk menyimpan SMS tersebut. Nomor yang telah menerima SMS akan mengirimkan laporan ke SMSC bahwa SMS telah diterima. Laporan tersebut kemudian diteruskan kembali ke nomor pengirim SMS.

Secara garis besar, mekanisme kerja pengiriman SMS dapat dibagi tiga macam,

yaitu :

1. Pengiriman SMS dalam satu operator atau sering diistilahkan dengan *Intra-Operator SMS.*

   Gambaran mekanisme pengiriman SMS ini dapat dilihat pada gambar 2.6

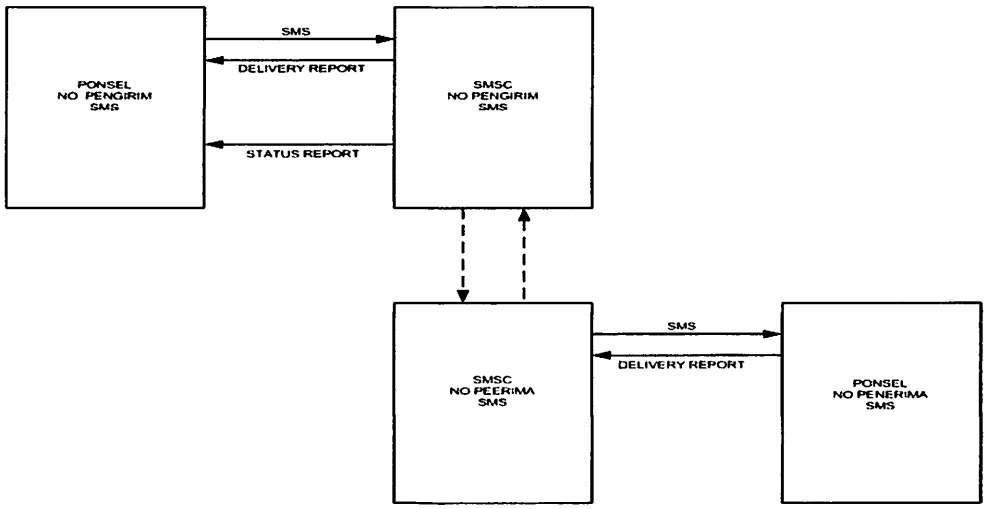


Gambar 2.6 Mekanisme intra-operator SMS

Pada Gambar 2.6 SMS yang dikirmkan oleh nomor pengirim akan dimasukkanke dalam SMSC operator nomor pengirim, kemudian SMSC tersebut akan mengirimkan pesan ke nomor yang dituju secara langsung. Nomor penerima kemudian akan mengirimkan sebuah *delivery report* yang menyatakan bahwa SMS telah diterima ke SMSC. SMSC kemudian meneruskan *report* tersebutke nomor pengirim SMS, disertai status *report* dari proses pengiriman SMS tersebut.

2. Pengiriman SMS antar operator yang berbeda atau *Inter-Operator SMS.*

   Berbeda dengan mekanisme *intra-operator*, pada mekanisme ini, SMS yang dikirimkan akan melalui dua buah SMSC. Perhatikan gambar dibawah :



Gambar 2.7 Mekanisme inter-operator SMS

Pada Gambar 2.7 , selain masuk ke SMSC operator pengirim, SMS yang dikirmkan akan diteruskan oleh SMSC operator pengirim, ke SMSC operator penerima SMS, kemudian baru diteruskan ke nomor tujuan. *Delivery report* yang dihasilkan pun akan melalui jalur tersebut, agar dapat sampai ke nomor pengirim SMS. Dalam mekanisme ini, terlihat ada komunikasi tidak langsung antara dua operator berbeda. Komunikasi tersebut dapat berjalan, setelah terjadi sebuah kesepakatan kerjasama antaroperator gersebut. Tidak adanya kerjasama antaroperator, dapat menyebabkan SMS yang dikirimkan ke nomor tujuan dengan operator berbeda, tidak sampai pada nomor tujuan tersebut.

2.5.3. SMS Gateway

Gateway dalam bahasa inggris berarti pintu gerbang. Namun pada dunia komputer, *gateway* dapat berarti juga sebagai jembatan penghubung antar satu system dengan system lain yang berbeda, sehingga dapat terjadi suatu pertukaran data antar system tersebut. Dengan demikian, SMS *gateway* dapat diartikan sebagai penghubung untuk lalu lintas datadata SMS, baik yang dikirmkan maupun yang diterima.

Pada awalnya, SMS *gateway* dibutuhkan untuk menjembatani antar SMSC. Hal ini dikarenakan SMSC yang dibangun oleh perusahaan yang berbeda memiliki protocol

komunikasi sendiri, dan protokol-protokol itu sendiri bersifat pribadi. Sebagai contoh, Nokia memiliki protocol SMSC yang disebut CIMD, sedangkan CMG memiliki protocol yang disebut EMI. SMS *gateway* ini kemudian ditempatkan di antara kedua SMSC berbeda tersebut, yang berfungsi sebagai *relay* keduanya, yang kemudian akan menterjemahkan data dari protocol SMSC lainnya yang dituju. Berikut adalah ilustrasi dari SMS *gateway* :

Gambar 2.8 Ilustrasi SMS Gateway

Namun, seiring perkembangan teknologi komputer, baik dari sisi *hardware* maupun *software,*dan perkembangan teknologi komunikasi, SMS *gateway* ini tidak lagi dimaksudkan seperti ilustrasi diatas. Dewasa ini, masyarakat lebih mengartikan SMS *gateway* sebagai suatu jembatan komunikasi yang menghubungkan perangkat komunikasi (dalam hal ini ponsel) dengan perangkat komputer mereka, yang menjadikan aktifitas SMS menjadi lebih mudah dan menyenangkan. Pengertian SMS *gateway* kemudian lebih mengarah pada sebuah program yang mengkomunikasikan antara system operasi komputer, dengan perangkat komunikasi yang terpasang untuk mengirim atau menerima SMS. Salah satu komunikasi yang terjadi, dapat dilakukan dengan mengirimkan perintah AT pada perangkat komunikasi tersebut, kemudian hasil operasinya dikirimkan kembali ke komputer. Dibutuhkan suatu *interface* baik dalam bentuk aplikasi maupun halaman *web* untuk membaca SMS yang masuk, atau mengirim SMS tersebut. Dibawah ini adalah gambar ilustrasi SMS *gateway* yang digunakan saat ini :

Gambar 2.9 Aplikasi SMS gateway

BAB III
PERANCANGAN DAN PEMBUATAN SISTEM

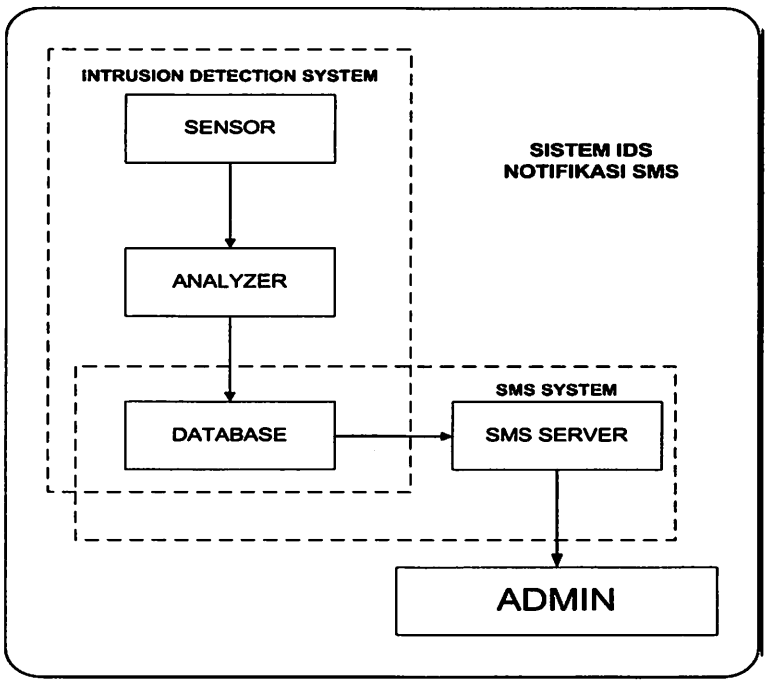
3.1. Perancangan dan Pembuatan NIDS (*Network Intrusion Detection System*)

3.1.1. Arsitektur Sistem

Sistem ini merupakan suatu metode kemanan jaringan yang bertujuan untuk membentuk suatu arsitektur sistem keamanan yang terintegrasi antara Intrusion Detection System (IDS) , Firewall System, Database System dan Monitoring System.. Sistem keamanan ini bertujuan melindungi jaringan dengan kemampuan merespon sesuai dengan kebijakan keamanan.

Untuk mewujudkan metode ini perlu dirancang komponen-komponen system keamanan jaringan berupa :

- Intrusion detection system (IDS)
- Database system
- SMS system



Gambar 3.1. Arsitektur Sistem

Pada gambar 3.1 menunjukkan bahwa keseluruhan sistem akan dihubungkan dengan dengan *database system*, termasuk SMS System yang akan berjalan sesuai dengan keadaan *database system*.

Intrusion Detection System (IDS) pada implementasi tugas akhir ini tediri dari komponen-komponen :

- *Sensor*

  Berfungsi untuk mengambil data dari jaringan. Sensor merupakan bagian dari sistem deteksi dini dari sistem keamanan yang dirancang dan berfungsi untuk mendeteksi nomor IP. Untuk itu digunakan suatu program yang berfungsi sebagai *intrusion detector* dengan kemampuan *packet logging*.

- *Analyzer*

  Berfungsi untuk analisa paket yang lewat pada jaringan. Informasi dari *analyzer* yang akan menjadi input bagi sistem lainnya.

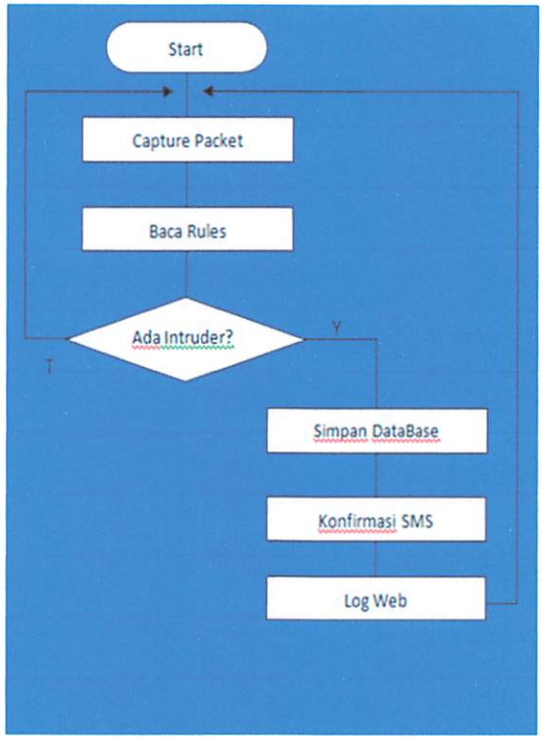
- *Database system*

  Merupakan tempat untuk menyimpan log atau hasil dari monitoring yang dilakukan pada sistem ini.

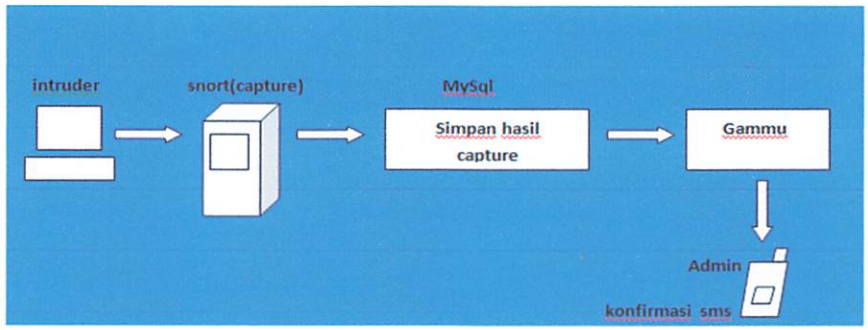
- SMS *System*

  Berfungsi untuk mengirimkan informasi kepada seorang administrator agar intrusi yang di deteksi dapat segera diketahui, sehingga administrator bisa sesegera mungkin mengatasinya.

Flowchart proses program dan interaksinya dengan user ditunjukkan pada Gambar 3.2



Gambar 3.2. Flowchart Sistem



Gambar 3.3. Diagram kerja sistem

3.1.2. Snort

Snort merupakan suatu perangkat lunak untuk mendeteksi penyusup dan mampu menganalisis paket yang melintasi jaringan secara *real time traffic* dan *logging* kedalam database serta mampu mendeteksi berbagai serangan yang berasal dari luar jaringan. Snort tersedia gratis di internet, www.snort.org. Snort bisa digunakan pada *platform* sistem operasi Linux, BSD, solaris, Windows dan sistem operasi lainnya. Secara *default*-nya, snort mempunyai 3 fungsi :

- paket sniffer, seperti tcpdump, iptraf dll
- paket logger, yang berguna untuk paket traffic dll
- NIDS, deteksi intrusi pada network

Setelah semua yang dibutuhkan untuk instalasi snort pada sistem operasi Ubuntu, kemudian instalasi snort itu sendiri. Snort harus terintegrasikan dengan MySQL, hal ini dikarenakan dalam perancangan sistem AIDS (*Automatic Intrusion Detection System*) log yang dicatat oleh snort akan langsung masuk kedalam database. Dengan menambahkan *option* –with –mysql, snort sudah akan terintegrasi dengan database MySQL.

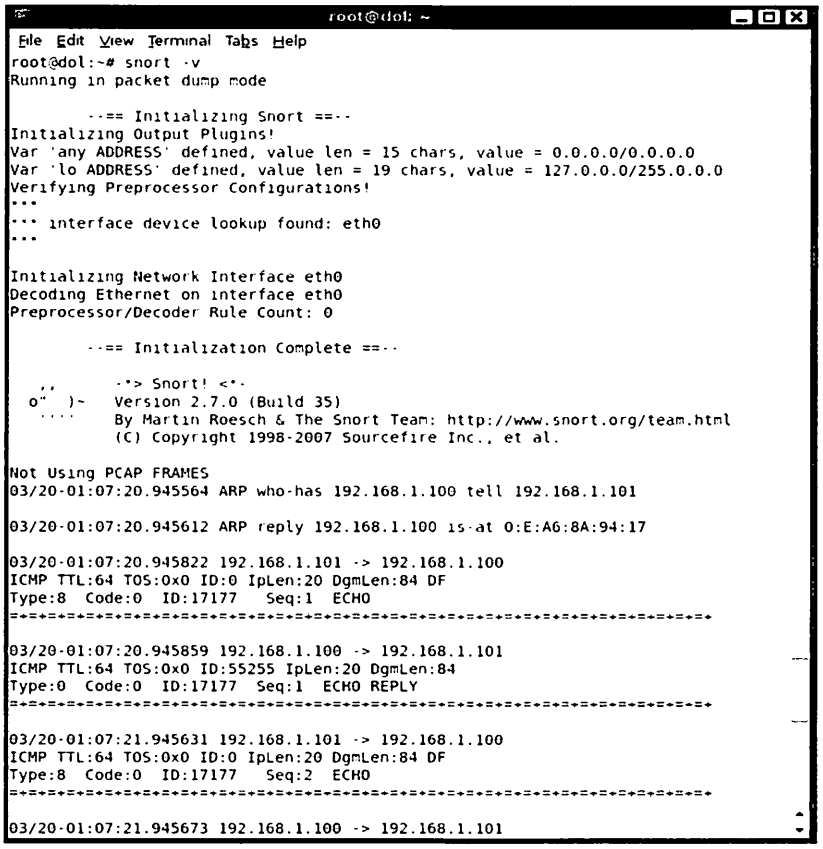
Pembuatan sistem ini dilakukan untuk mendeteksi ketika terdapat suatu intrusi yang masuk kedalam sistem jaringan komputer *server*, oleh karena itu hal yang sangat perlu dilakukan adalah melakukan konfigurasi pada file `/etc/snort/snort.conf`. File tersebut yang menentukan apa saja intrusi yang akan dideteksi, serta membuat folder `rules` didalam `/etc/snort`. Dibawah ini adalah bagian yang penting dalam konfigurasi `/etc/snort/snort.conf` :

```
# variable jaringan yang satu network dengan
server
var HOME_NET 192.168.1.2/24
# variable jaringan luar
var EXTERNAL_NET any
# variabel rule
var RULE_PATH /etc/snort/rules
# konfigurasi output plugin
```

```
# bentuk log adalah syslog
output alert_syslog : LOG_LOCAL7
config logdir: /usr/yaken/log
config alert_with_interface_name
config checksum_mode: none
config show_year
config interface: eth0
output database: log, mysql, user=snort
password=tugasakhir
dbname=snort host=localhost
# path file rule
include $RULE_PATH/dos.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/backdoor.rules
```

Menjalankan Snort dengan tiga mode, yaitu :

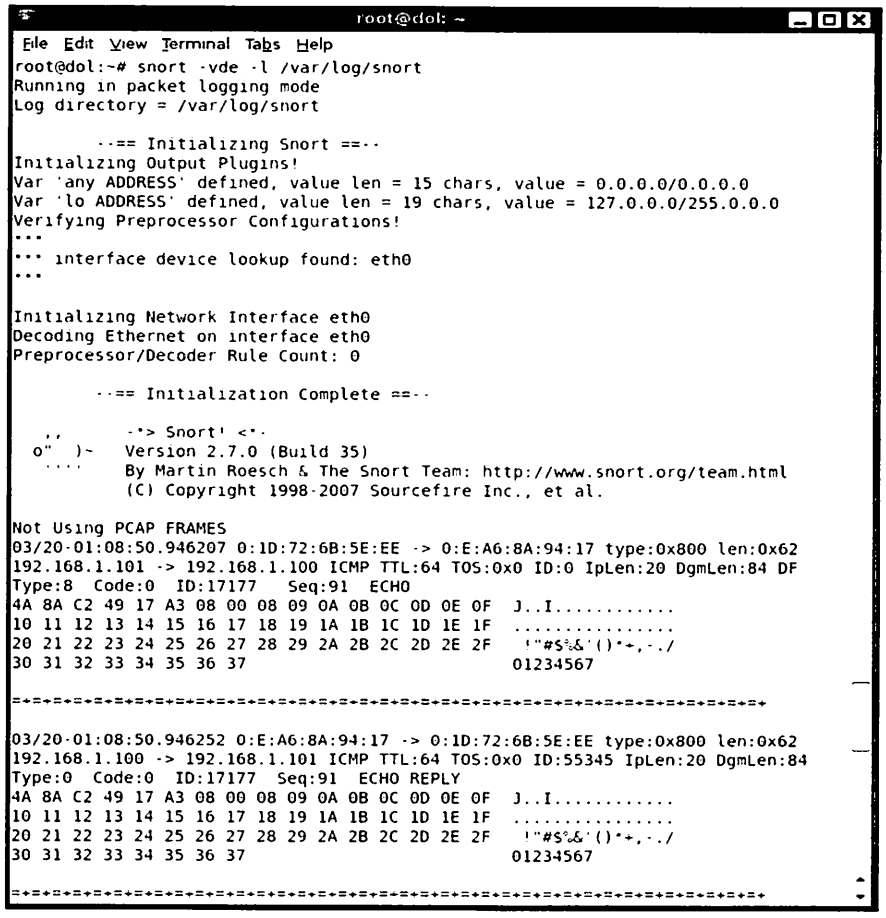
1. Snort dengan paket sniffer :



Gambar 3.4 Tampilan snort paket sniffer

Gambar 3.3 menunjukkan snort berjalan dengan mode *dump*, yang berarti snort mendeteksi semua nomor IP dan *protocol* yang lewat tanpa memasukkan *log* yang tersimpan kedalam *database*.

2. Snort dengan Paket Logger:



Gambar 3.5 Tampilan snort paket logger

Snort berjalan dengan membuat file *log* yang ditentukan, seperti pada gambar 3.4. *Log* yang masuk berisi paket-paket yang masuk.

3. NIDS, Sistem Deteksi Penyusup di jaringan

Gambar 3.6. Tampilan snort NIDS

3.1.3. Perancangan Database (MySQL)

3.1 MySQL

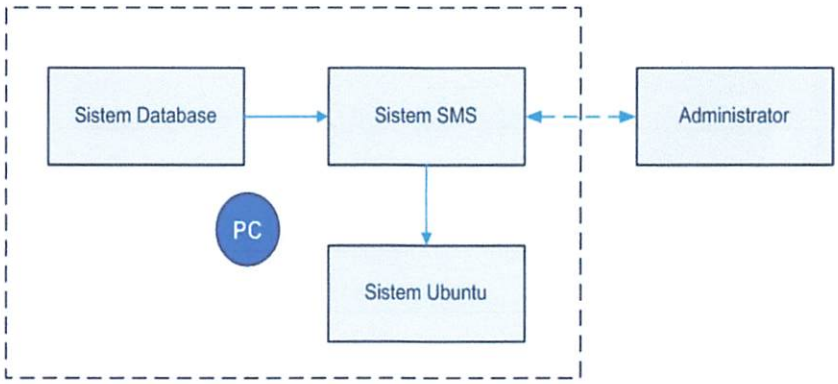
Sistem keamanan ini menggunakan prinsip sentralisasi database untuk menyimpan semua alert yang berasal dari sensor maupun log dari firewall.Informasi yang tersimpan pada data base ini juga merupakan input untuk pengawasan keamanan jaringan yang dilakukan oleh firewall system, monitoring system serta sistem notifikasi SMS.

Pada perancangan system ini, database diinstall menyatu dengan snort untuk *Intrusion Detection System.*

3.2. Perancangan Sistem Notifikasi SMS (Short Messaging Service)

Sistem notifikasi SMS ini dirancang sebagai bagian yang memberikan fungsi interaktif antara sistem dengan administrator. Alasan digunakannya SMS sebagai media interaktif adalah sebagai berikut :

1. Penyampaian pesan yang cepat dan cukup reliable
2. Biaya yang relatif murah
3. Tidak tergantung pada jaringan data host



Gambar 3.7. Diagram Blok Interkoneksi Sistem Notifikasi SMS

Fungsi dasar dari sistem SMS ini sebenarnya hanya memberikan notifikasi atau pemberitahuan kepada administrator sesegera mungkin ketika terjadi suatu event. Pesan yang dikirimkan berupa *alert message* yang di deteksi oleh system dan masuk pada log database.

3.2.1. Konfigurasi koneksi MODEM dengan PC

Instalasi paket yang dibutuhkan

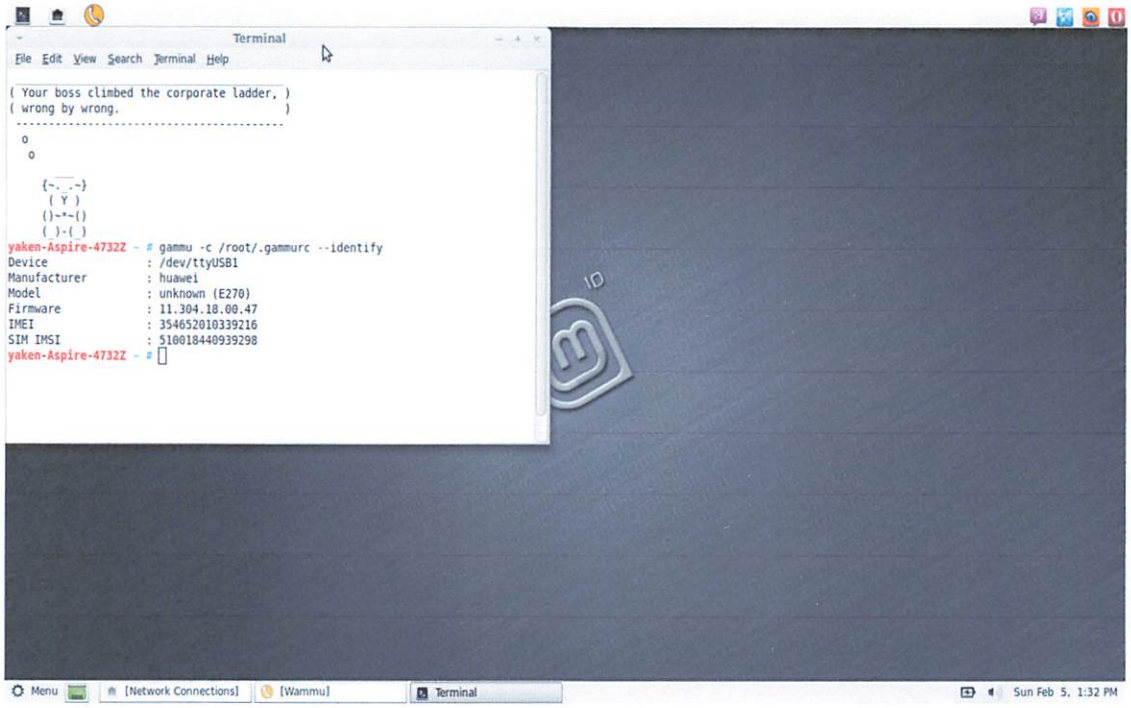
3.2.1.1. Instalasi Gammu

Gammu adalah sebuah aplikasi/daemon yang dikhususkan untuk membangun sebuah SMS Gateway yang dapat menangani ponsel didalam Linux. Dengan lisensi GNU/GPL, menjamin kebebasan menggunakan *tool* ini.

Instalasi gammu pada skripsi ini menggunakan *Synaptic Package Manager* untuk mencari paket yang dibutuhkan,diantaranya *Apache* (web server), MySql (database), PHP (aplikasi untuk sms gateway), selanjutnya menginstal *Gammu* dan library pendukungnya.

Pada system ini koneksi antara PC dengan *Mobile Phone* dilakukan dengan menggukan *Gammu* sebagai perangkat lunak yang menghubungakan,dan Modem Huawei (E270) sebagai perangkat keras dari pembuatan SMS *Gateway*. Konfigurasi Gammu berada pada folder /etc, dan yang dilakukan adalah mengatur port dan koneksi antara PC dan *mobile phone* dengan konfigurasi sebagai berikut :

```
Port = /dev/ttyUSB ##port ttyUSB1 sebagai driver port serial
Model = AT ##model device yang digunakan (Modem Huawei E270)
Connection = serial ##menggunakan koneksi serial
```



Gambar 3.8. Konensi HP dengan Gammu

3.2.2. Sistem untuk Otomatisasi Kirim SMS

Gammu menyediakan *daemon* (sejenis *service*) dalam paket instalasi *gammu*, *service* itu bernama *smsd*. *Smsd* akan menangani SMS yang masuk dan dapat menyimpannya kedalam *database*. Proses yang berlangsung yaitu *smsd* akan mengirimkan sms secara otomatis ketika tabel outbox diberi masukan. Untuk itu perlu membuat sebuah *trigger* pada database, sehingga ketika ada *event* yang masuk pada tabel snort, tabel outbox juga akan di beri masukan, sehingga sms daemon akan bekerja.

Dibawah ini adalah *syntax trigger* yang dibuat :

```
:: Procedure
create procedure cek (out sig int,out tim text,out info text)
SELECT event.signature,event.timestamp,signature.sig_name Into
sig,tim,info FROM `event` ,`signature` WHERE
signature.sig_id=event.sid AND Mid(event.timestamp,1,10) like
current_date group by event.timestamp order by cid desc limit 0,1;

:: Procedure
create procedure get_info(out info text)
SELECT signature.sig_name INTO info FROM `event` , `signature`
WHERE signature.sig_id = event.signature
ORDER BY cid DESC LIMIT 0,1;

:: Trigger
DELIMITER //
CREATE TRIGGER `snort`.`replay` AFTER INSERT ON `snort`.`event`
FOR EACH ROW BEGIN

call cek(@sig,@tim,@info);
call get_info(@isi);

if new.signature <> @sig and new.timestamp<>@tgl then
INSERT INTO `snort`.`outbox` (`number` , `text`)VALUES
('085755959681', concat('Warning Message.Ada intruder : ','"',@isi,'"'));
Else

INSERT INTO `snort`.`outbox` (`number` , `text`)VALUES
('0855755959681', concat('Warning Message.Ada intruder : ','"',@isi,'"'));
end if;
END
```

Gambar 3.9. Syntax trigger

Prosedur dibuat untuk mendefinisikan masukan pada tabel outbox. Berdasarkan prosedur yang telah dibuat diatas, tabel outbox akan diberi masukan dengan mengambil data dari :

1. Tabel event pada kolom signature (id alert)
2. Tabel event pada kolom timestamp (waktu terdeteksi intrusi yang masuk)
3. Tabel signature pada kolo sig_name (pesan alert)

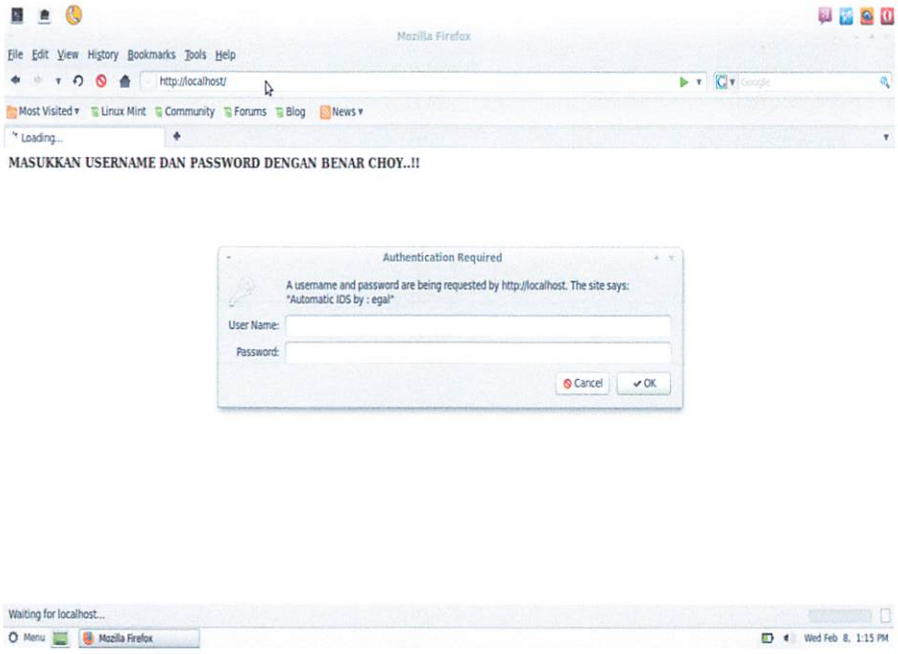
Sedangkan *trigger* diatas berarti bahwa ketika terdapat data yang masuk pada tabel event (db snort), sistem akan mengecek tanggal (timestamp) dan signature pada tabel event
dan signature, jika tidak sama maka tabel outbox akan terisi dengan masukan :

1. Pada kolom *number* berisi nomor tujuan
2. *Text* berisi *sig_name* pada tabel *signature*.

3.3. Monitoring Alert dan Log melalui web

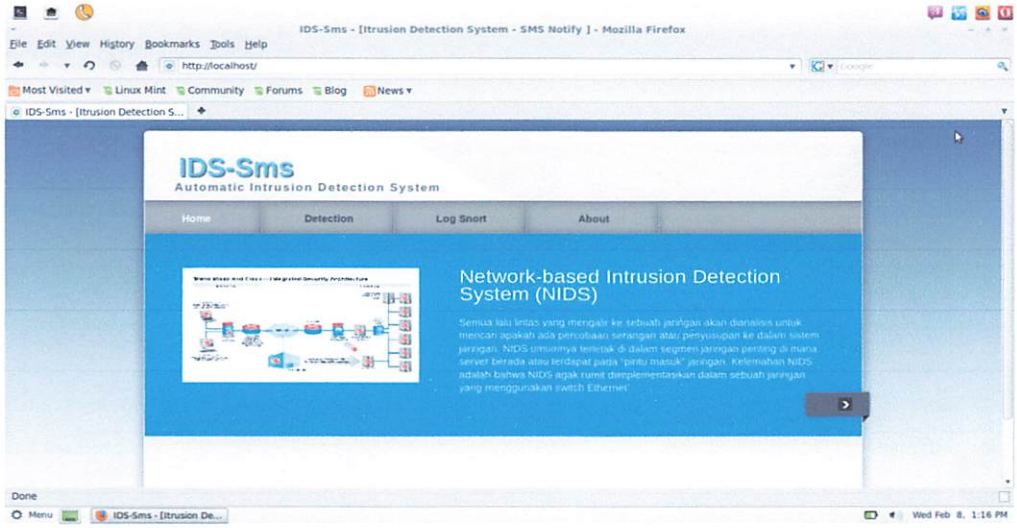
Monitoring ini berfungsi untuk melihat nomor IP *intruder* (penyusup) dan jenis

serangannya dalam bentuk aplikasi web menggunakan php.



Gambar 3.10. Halaman Admin Login

Sebelum masuk pada halaman admin, akan dilakukan autentikasi, sehingga dapat dibatasi yang melakukan akses. Setelah autentikasi berhasil, akan dibuka halaman utama, yaitu memilih untuk melihat log pada database atau log pada *syslog*, ditunjukkan pada gambar 3.11.

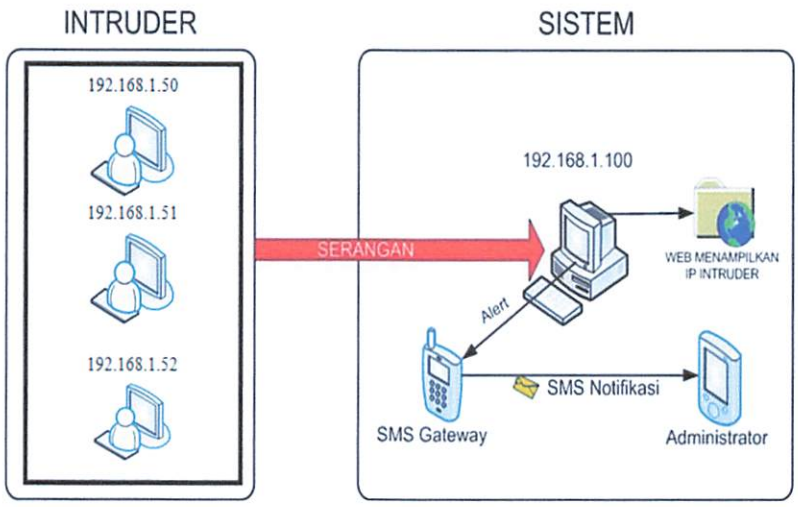


Gambar 3.11. Halaman Utama

BAB IV
PENGUJIAN SISTEM

Pada implementasi AIDS (*Automatic Intrusion Detection System*) ini, sistem yang dipakai merupakan PC linux Ubuntu 10.04 yang mempunyai 1 (satu) *network interface card* (NIC). Keseluruhan sistem, baik IDS, *monitoring*, *database*, maupun notifikasi otomatis, dipasang dalam PC tersebut. Sistem ini dilengkapi dengan paket program Apache Web Server dan MySQL *Database Server*.

4.1. Pengujian AIDS (Sistem Deteksi Gangguan Jaringan Otomatis)



Gambar 4.1. Diagram Pengujian Sistem

Pengujian sistem IDS dilakukan seperti pada gambar 4.1, yaitu PC *intruder* dengan IP 192.168.1.50, 192.168.1.51, 192.168.1.52 ,melakukan 3 jenis *intrusi* yang akan ditujukan pada PC sistem dengan IP 192.168.1.1.2 Sistem IDS akan mendeteksi serangan tersebut, kemudian mengirimkan pesan atau *alert* ke sistem SMS, dan selanjutnya akan mengirimkan *notifikasi* berupa pesan *alert* ke *administrator* serta menampilkan *alert* tersebut pada *web browser*.

Intrusi yang dilakukan untuk pengujian sistem ini adalah program *utility ping* yang ada pada sistem operasi komputer. Biasanya ping

digunakan untuk men-cek berapa waktu yang dibutuhkan untuk mengirim jumlah data tertentu dari satu komputer ke komputer lain. Panjang maksimum data yang dapat dikirim menurut spesifikasi protokol IP adalah 65,536 byte.

Pc system:

Gambar 4.2. Capture Snort

1. **192.168.1.50**

```
Mint- # ping -Rf 192.168.1.2 -s 10000
```

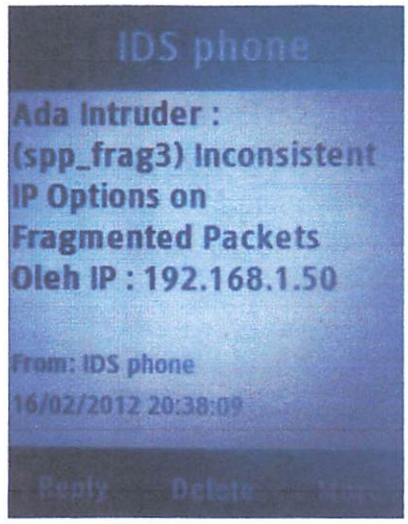
2. **192.168.1.51**

```
Mint- # ping -Fs 65000 192.168.1.2 -s 10000
```

3. **192.168.1.52**

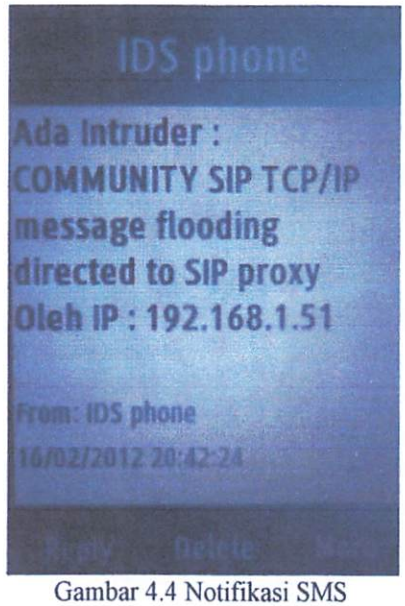
```
Mint- # ping 192.168.1.2 -t -l 65000
```

Dari ketiga uji coba yang dilakukan, dapat dilihat hasil yang ditunjukkan pada HP, *database,* dan halaman web berturut-turut adalah sebagai berikut:



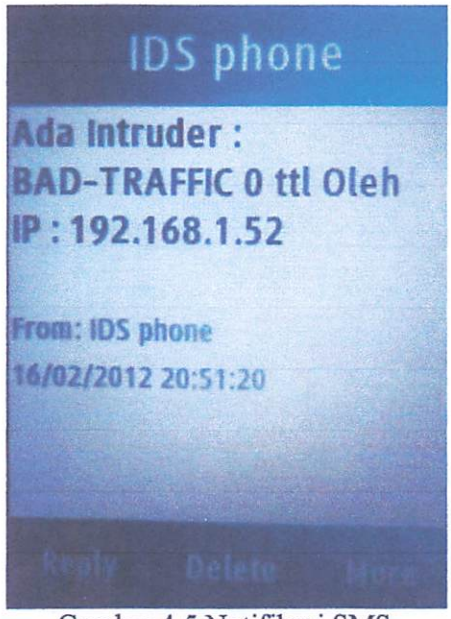
Gambar 4.3 Notifikasi SMS

```
mysql> select * from signature;
+--------+-------------------------------------------------------------
| sig_id | sig_name                                                    |
+--------+-------------------------------------------------------------
|        | (spp_frag3) Inconsistent IP Options on Fragmented Packets|
+--------+-------------------------------------------------------------
1 row in set (0.00 sec)
```



Gambar 4.4 Notifikasi SMS

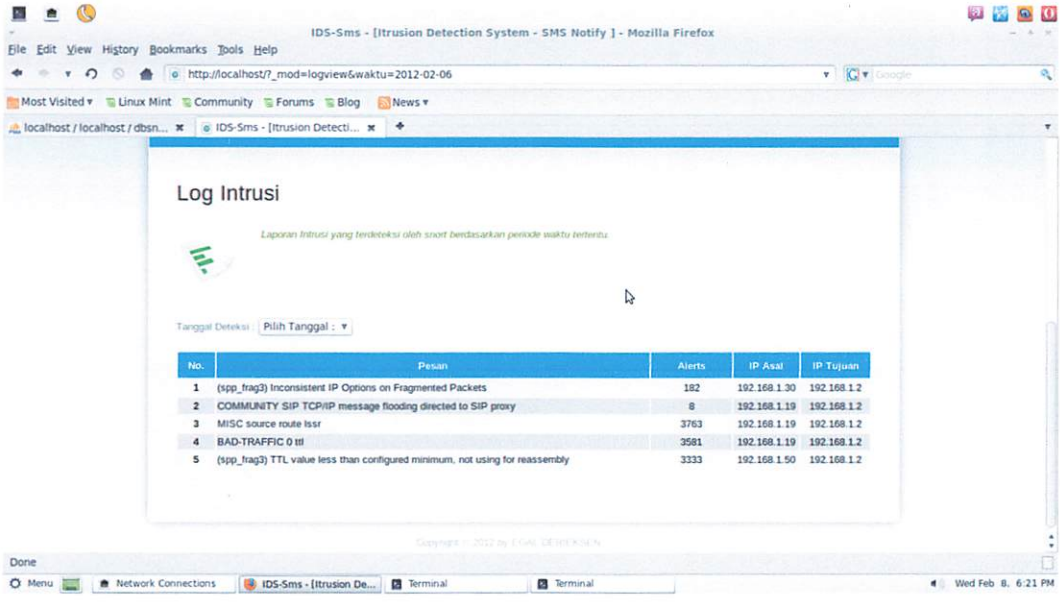
```
mysql> select * from signature;
+--------+-------------------------------------------------------------
| sig_id | sig_name                                                    |
+--------+-------------------------------------------------------------
|        | COMMUNITY SIP TCP/IP message flooding directed to SIP proxy|
+--------+-------------------------------------------------------------
1 row in set (0.00 sec)
```



Gambar 4.5 Notifikasi SMS

```
mysql> select * from signature;
+--------+-----------------------------------------------------------
| sig_id | sig_name
+--------+-----------------------------------------------------------
|        | BAD-TRAFFIC 0 TTL
+--------+-----------------------------------------------------------
1 row in set (0.00 sec)
```

Sedangkan untuk mengetahui No.IP *intruder* dari serangan tersebut dapat dilihat pada halaman web, ditunjukkan pada gambar 4.6



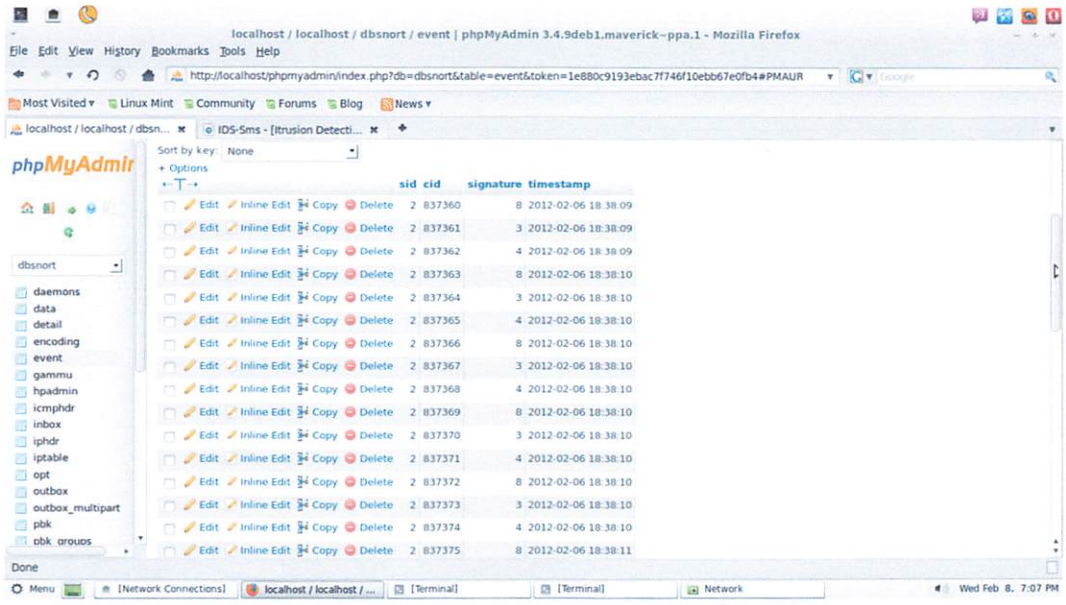
Gambar 4.6. Tampilan No.IP Intruder

4.2. Analisa Sistem

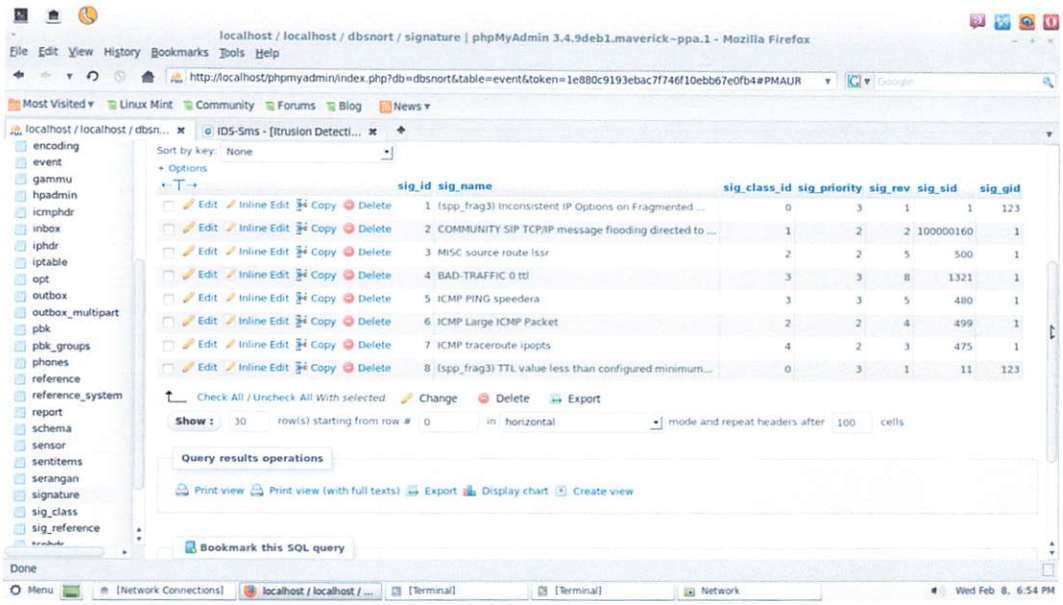
4.2.1. Timestamp

Pengujian yang sudah dilakukan diatas memberikan hasil yang akan tersimpan dalam *database* sistem dan mengirimkan *notifikasi* berupa pesan singkat (SMS) secara otomatis kepada *administrator.*

Analisa ini menjelaskan tentang selisih waktu pada saat sistem menyimpan *database* dan SMS dikirim serta diterima. Gambar 4.7 menunjukkan timestamp atau waktu pada saat terjadinya intrusi dan alert pada tabel *signature*.



Gambar 4.7. Tabel *Snort.event*



Gambar 4.8. Tabel *Snort.signature*

Pada dua gambar diatas diketahui tentang *signature* atau jenis *intrusi* yang terjadi, sebagai contoh pada tabel *event* terdapat kolom *signature* 8, angka tersebut merupakan ID dari tabel *signature* yang berarti kategori jenis serangan.

sid	cid	signature	timestamp
2	837360	8	2012-02-06 18:38:09
2	837361	3	2012-02-06 18:38:09

Gambar 4.9. *Timestamp Snort.event*

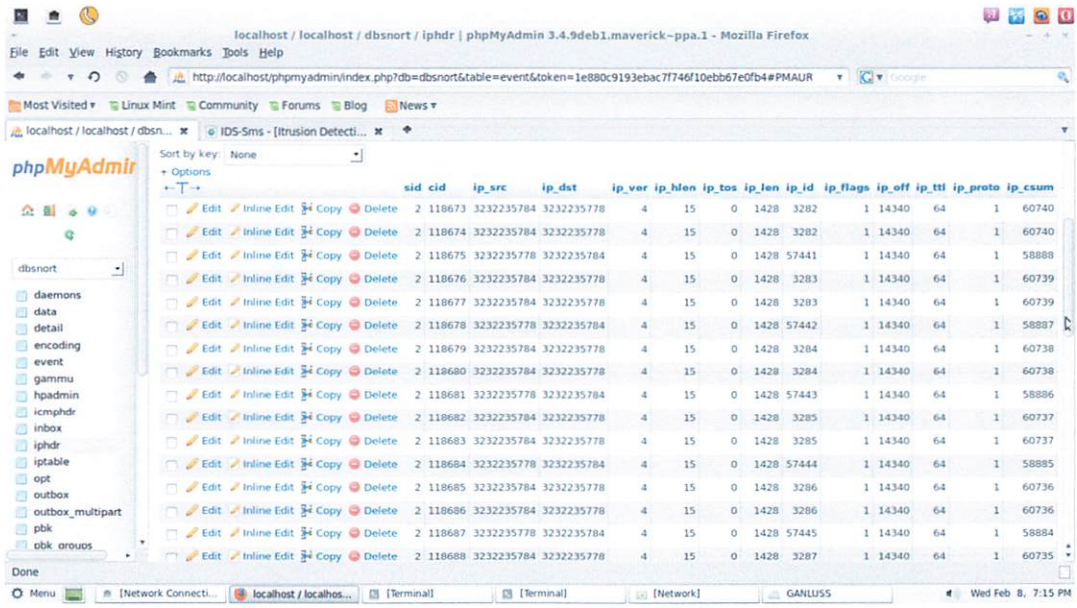
Pada kolom timestamp dapat diketahui terjadi pada pukul 18:38:09 dan SMS yang dikirim yaitu pada tabel *outbox* pada pukul 18:39:09, seperti gambar dibawah:

Gambar 4.10 Tabel *Outbox* (SMS yang dikirim)

Anilasa diatas menunjukkan bahwa waktu antara sistem mendeteksi *intrusi* dan mengirimkan *notifikasi* cukup singkat yaitu kurang lebih satu menit tergantung dari kondisi jaringan provider, sehingga *administrator* akan dengan segera mengetahui adanya *intrusi* yang masuk kedalam sistem.

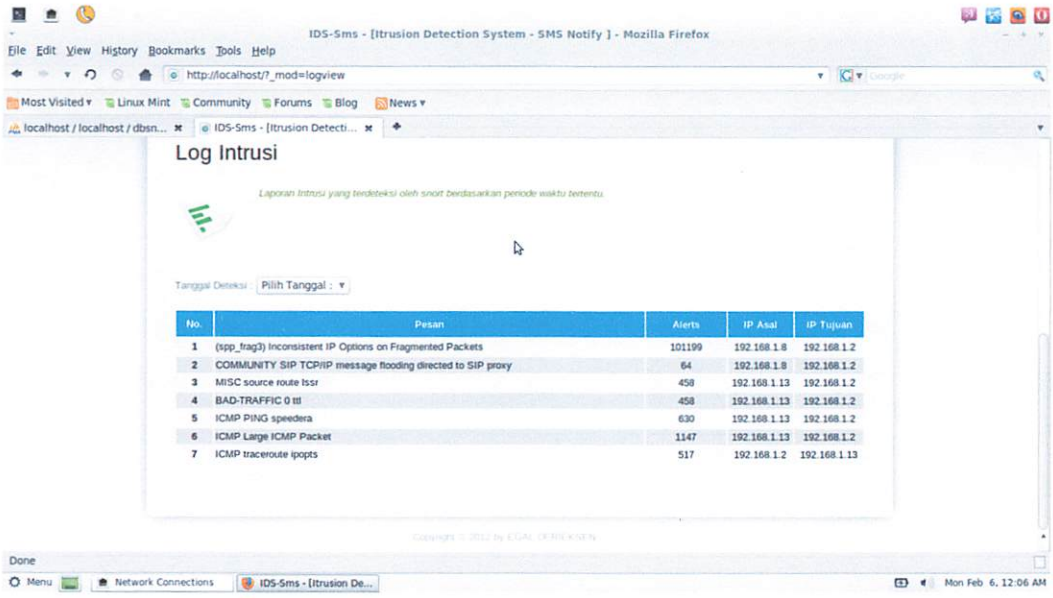
4.2.2. Nomor IP *(Internet Protocol)*

Dalam *database* yang tersimpan, *Snort* mengidentifikasi *sensor* (nomor IP) dalam bentuk desimal, baik IP *Intruder* maupun IP sistem. Oleh karena itu untuk menampilkan IP dalam bentuk format IPv4 pada web diperlukan konfersi, dalam hal ini konversi menggunakan PHP. Gambar 4.11 merupakan IP desimal yang tersimpan dalam *database.*



Gambar 4.11 *Database snort.iphdr*

Dalam halaman web yang ditampilkan adalah IPv4, seperti pada gambar 4.12



Gambar 4.12. tampilan NO.IPv4

BAB V
PENUTUP

Pada Bab ini akan disampaikan mengenai beberapa kesimpulan yang diambil pada skripsi ini,dan saran-saran untuk pengembangan tugas akhir ini selanjutnya.

5.1 Kesimpulan

Setelah menyelesaikan pengembangan aplikasi serta implementasi skripsi ini, maka dapat didapat kesimpulan sebagai berikut

1. Sistem deteksi penyusupan jaringan ini mampu bekerja secara stabil dengan terus mengirim *notifikasi SMS* sampai *administrator* merespon.
2. Sistem *notifikasi* berupa *SMS* dapat memberikan *administrator* informasi terkini tentang adanya *intrusi* yang masuk serta memungkinkan *administrator* merespon secara langsung.
3. *Report SMS* dikirim dengan jeda waktu satu menit sekali.

5.2 Saran

Pada skripsi ini tidak dibuat *firewall* yang bisa menghentikan intruder ketika melakukan serangan. Oleh karena itu akan lebih baik jika sistem dilengkapi *firewall* dengan *input SMS,* sehingga tidak hanya *notifikasi* yang dijalankan tapi eksekusi pun dilakukan oleh *administrator* melalui *SMS.*

DAFTAR PUSTAKA

Amin, Mohamad. 2006. **Kasus-kasus Pengembangan Database**. Andi : Yogyakarta.

Dony Ariyus, M.Kom. 2007. **INTRUSION DETECTION SYSTEM (Sistem Pendeteksi Penyusup Pada Jaringan Komputer)** . Andi : Yogyakarta.

Kadir, Abdul. 2003. **Dasar Pemrograman WEB Dinamis Menggunakan PHP**. Andi : Yogyakarta.

Pribadi , Harijanto. 2008. **Firewall (Melindungi jaringan dari DDoS menggunakan Linux)**. Andi: Yogyakarta.

Yunianto. 2006. **Membangun Aplikasi SMS Gateway di Linux**. PT Dian Rakyat : Jakarta.

CARA INSTAL SNORT DI UBUNTU http://erwinmellowdik.blogspot.com/2012/01/cara-install-snort-di-ubuntu-1004.html (diakses tanggal 25 November 2011)

SNORT UNTUK MENDETEKSI PENYUSUP http://bebas.vlsm.org/v10/onno-ind-2/network/network-security/snort-untuk-mendeteksi-penyusup-4-2002.rtf (diakses tanggal 27 November 2011)

Intrusion Detection System menggunakan Snort, MySQL, BASE http://taujago.web.id/intrusion-detection-system-menggunakan-snort-mysql-base/ (diakses tanggal 5 Desember 2011)

MENGUNGKAP TOOL JARINGAN PING http://lintoherlambang.com/mengungkap-tool-jaringan-ping.html (diakses tanggal 17 Januari 2012)

LAMPIRAN

Tabel Struktur Database Snort

Nama Tabel	Field	Type	Null	Key	Default	Extra
daemons	start	text	NO			
	info	text	NO			
data	sid	int(10) unsigned	NO	PRI	NULL	
	cid	int(10) unsigned	NO	PRI	NULL	
	data_payload	text	YES		NULL	
detail	detail_type	tinyint(3) unsigned	NO	PRI	NULL	
	detail_text	text	NO	PRI	NULL	
encoding	encoding_type	tinyint(3) unsigned	NO	PRI	NULL	
	encoding_text	text	NO		NULL	
event	sid	int(10) unsigned	NO	PRI	NULL	
	cid	int(10) unsigned	NO	PRI	NULL	
	signature	int(10) unsigned	NO	MUL	NULL	
	timestamp	datetime	NO	MUL	NULL	
gammu	version	int(11)	NO			
hpadmin	id	int(11)	NO			AUTO_INCREMENT
	nomer	int(11)	NO			
icmphdr	sid	int(10) unsigned	NO	PRI	NULL	
	cid	int(10) unsigned	NO	PRI	NULL	
	icmp_type	tinyint(3) unsigned	NO	MUL	NULL	
	icmp_code	tinyint(3) unsigned	NO		NULL	
	icmp_csum	smallint(5) unsigned	YES		NULL	
	icmp_id	smallint(5) unsigned	YES		NULL	
	icmp_seq	smallint(5) unsigned	YES		NULL	

inbox	UpdateInDB	Timestamp	NO		Current_timestamp	On update current_timestamp
	ReceivingDateTime	Timestamp	NO	'		
	text	text	NO			
	SenderNumber	Varchar(20)	NO			
	coding	Enum('default_no_compression'.'unicode_no_compres	NO		default_no_compression	
	UDH	text	NO			
	SMSCNumber	Varchar(20)	NO			
	Class	Int(11)	NO		-1	
	Texdecoded	text	NO			
	Id	Int(10)unsigned	NO			Auto_increment
	recepientID	text	NO			
	processed	Enum('false','true)	NO		false	
iphdr	sid	int(10) unsigned	NO	PRI	NULL	
	cid	int(10) unsigned	NO	PRI	NULL	
	ip_src	int(10) unsigned	NO	MUL	NULL	
	ip_dst	int(10) unsigned	NO	MUL	NULL	
	ip_ver	tinyint(3) unsigned	YES		NULL	
	ip_ver	tinyint(3) unsigned	YES		NULL	
	ip_tos	tinyint(3) unsigned	YES		NULL	
	ip_len	smallint(5)	YES		NULL	
		unsigned				
	ip_id	smallint(5) unsigned	YES		NULL	
	ip_flags	tinyint(3) unsigned	YES		NULL	
	ip_off	smallint(5) unsigned	YES		NULL	
	ip_ttl	tinyint(3) unsigned	YES		NULL	
	Ip_proto	tinyint(3) unsigned	NO		NULL	

	Ip_csum	smallint(5) unsigned	YES		NULL	
iptable	Ips	Varchar(20)	NO			
	Status	Varchar(10)	YES		NULL	
opt	sid	int(10) unsigned	NO			
	cid	int(10) unsigned	NO			
	optid	int(10) unsigned	NO			
	opt_proto	tinyint(3) unsigned	NO			
	opt_code	tinyint(3) unsigned	NO			
	opt_len `	smallint(6)	YES			
	opt_data	text	YES			
outbox	UpdatelnDB	Timestamp	NO			
	insertIntoDB	Timestamp	NO			
	sendindDate Time	Timestamp	NO			
	text	text	YES			
	Destination number	Varchar(20)	YES			
	coding	Enum('default_No_compression','unicode_No_compres	NO			
	UDH	text	YES			
	class	Int(11)	YES			
	textdecoded	text	NO			
	ID	Int(10)unsigned	NO			
	multipart	Enum('false','true')	YES			
	Relative Validaty	Int(11)	YES		-1	
	senderID	Varchar(255)	YES		NULL	
	sendingTime Out	timestamp	YES			
	Delivery Report	Enum(default',yes', 'no')	YES		default	
	creatorID	text	YES			

Outbox_ multipart	text	text	YES		null	
	coding	Enum('default_No_compression','unicode_No_compres	NO		default_No_compression	
	UDH	text	YES		NULL	
	class	Int(11)	YES		-1	
	textdecoded	text	YES		NULL	
	id	Int(10)unsigned	NO		0	
	Sequence position	Int(11)	NO		1	
pbk	ID	Int(11)	No			Auto_increment
	Groupid	Int(11)	No		-1	
	Name	Text	No			
	Number	Text	No			
Pbk_groups	Name	Text				
	ID	Tint(11)				Auto_increment
Phones	Id	Text	No			
	updatelnDB	Timestamp	No		Current_timestamp	On updatecurrent_timestamp
	insertintoDB	Timestamp	No			
	Timeout	Timestamp	No			
	Send	Enum('yes','no')	No		No	
	Receive	Enum('yes','no')	No		No	
	Imei	Varchar(35)	No			
	Client	Text	No			
	Battery	Int(11)	No		0	
	signal	Int(11)	No		0	
	sent	Int(11)	no		0	
	received	Int(11)	no		0	
reference	ref_id	int(10) unsigned	NO	PRI	NULL	auto_increment
	ref_system_id	int(10) unsigned	NO			
	ref_tag	text	NO			
	ref_system_id	int(10) unsigned	NO	PRI	NULL	auto_increment

reference_system	ref_system_name	varchar(20)	YES		NULL	
report	sid	Int(10)	yes		NULL	
	waktu	datetime	yes		NULL	
	ips	Varchar(20)	yes		0	
schema	vseq	int(10) unsigned	NO			
	ctime	datetime	NO			
sensor	sid	int(10) unsigned	NO	PRI	NULL	auto_increment
	hostname	text	YES		NULL	
	interface	text	YES		NULL	
	filter	text	YES		NULL	
	detail	tinyint(4)	YES		NULL	
	encoding	tinyint(4)	YES		NULL	
	last_cid	int(10) unsigned	NO		NULL	
sentitems	updatelnDB	timestamp	NO			
	insertintoDB	timestamp	NO			
	Sendingdatetime	timestamp	NO			
	Deliverydatetime	timestamp	yes		NULL	
	Text	Text				
	Destination number	Varchar(20)	NO			
	Coding	Enum('default_No_compression','unicode_No_compres	NO		Default_no_compression	
	UDH	Text	NO			
	SMSCNumber	Varchar(20)	NO			
	Class	Int(11)	NO		-1	
	Textdecoded	Text	NO			
	ID	Int(10)	NO			
	senderID	Varchar(255)	NO			
	Sequenceposition	Int(11)	NO		1	
	Status	Enum('sendingOK'.'	NO		sending	

		sendingOKNOReport','sendingError')			OK	
	Statuserror	Int(11)	NQ		-1	
	TPMR	Int(11)	NO		-1	
	Relativevalidity	Int(11)	NO		-1	
	Cretorid	Text	NO			
serangan	Id_serangan	Int(11)	no	no		
	nama	Varchar(20)	Yes	null		
	Ips	Varchar(20)	yes	null		
	tanggal	date	Yes	null		
	status	varchar	yes	0		
sig_class	sig_class_id	int(10) unsigned	NO	PRI	NULL	auto_increment
	sig_class_name	varchar(60)	NO	MUL	NULL	
sig_reference	sig_id	int(10) unsigned	NO	PRI	NULL	
	ref_seq	int(10) unsigned	NO	PRI	NULL	
	ref_id	int(10) unsigned	NO		NULL	
signature	sig_id	int(10) unsigned	NO	PRI	NULL	auto_increment
	sig_name	varchar(255)	NO	MUL	NULL	
	sig_class_id	int(10) unsigned	NO	MUL	NULL	
	sig_priority	int(10) unsigned	YES		NULL	
	sig_rev	int(10) unsigned	YES		NULL	
	sig_sid	int(10) unsigned	YES		NULL	
	sig_gid	int(10) unsigned	YES		NULL	
	cid	Int(10)unsigned	NO			
	tcp_sport	smallint(5)	NO			
	tcp_dport	smallint(5) unsigned	NO			
	tcp_seq	int(10) unsigned	YES			
	tcp_ack	int(10) unsigned	YES			

	tcp_csum	smallint(5) unsigned	YES		NULL	
	tcp_urp	smallint(5) unsigned	YES		NULL	
udphdr	sid	int(10) unsigned	NO	PRI	NULL	
	cid	int(10) unsigned	NO	PRI	NULL	
	udp_sport	smallint(5) unsigned	NO	MUL	NULL	
	udp_dport	smallint(5) unsigned	NO	MUL	NULL	
	udp_len	smallint(5) unsigned	YES		NULL	
	udp_csum	smallint(5) unsigned	YES		NULL	
tcphdr	sid	int(10) unsigned	NO			
	tcp_off	tinyint(3) unsigned	YES			
	tcp_res	tinyint(3) unsigned	YES			
	tcp_flags	tinyint(3) unsigned	NO			
	tcp_win	smallint(5) unsigned	YES			

Tabel struktur *database Gammu*

Nama Tabel	Field	Type	Null	Key	Default	Extra
Outbox	UpdateInDB	Timestamp	no			
	insertIntoDB	Timestamp	no			
	Sending DateTime	Timestamp	no			
	Text	Text	yes			
	Destination Number	Varchar(20)	no			
	Coding	enumdefault	no			
	UDH	Text	yes			
	Class	Int(11)	yes			
	TextDecoded	Relative Validaty	Int(11)	yes		
	ID	Timestamp	yes			
	Multipart	Enum(default' ,yes','no')	yes			
	Relative Validaty	Text	no		-1	
	SenderID	Varchar(225)	yes		Null	
	SendingTime Out	Timestamp	yes			
	Delivery Report	Enum(default' ,yes','no')	yes		default	
	CreatorID	Text	no			

Konfigurasi Snort

```
var HOME_NET 192.168.1.0/24

var EXTERNAL_NET any

# List of DNS servers on your network
var DNS_SERVERS $HOME_NET

# List of SMTP servers on your network
var SMTP_SERVERS $HOME_NET

# List of web servers on your network
var HTTP_SERVERS $HOME_NET

# List of sql servers on your network
# var SQL_SERVERS $HOME_NET

# List of telnet servers on your network
var TELNET_SERVERS $HOME_NET

var HTTP_PORTS 80

# Ports you want to look for SHELLCODE on.
var SHELLCODE_PORTS !80

# Ports you do oracle attacks on
var ORACLE_PORTS 1521

var AIM_SERVERS

[64.12.24.0/23,64.12.28.0/23,64.12.161.0/24,64.12.163.0/24,64.12.200.0

/24,205.18

8.3.0/24,205.188.5.0/24,205.188.7.0/24,205.188.9.0/24,205.188.153.0/24

,205.188.1 79.0/24,205.188.248.0/24]

# Path file tampat menyimpan rule
var RULE_PATH /etc/snort/rules

##################################################
# Step #2: Configure dynamic loaded libraries
##################################################

dynamicpreprocessor directory /usr/lib/snort_dynamicpreprocessor/
dynamicengine /usr/lib/snort_dynamicengine/libsf_engine.so
```

```
##################################################
# Step #3: Configure preprocessors
##################################################

preprocessor flow: stats_interval 0 hash 2
preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy first detect_anomalies
preprocessor ftp_telnet: global encrypted_traffic yes inspection_type
stateful
preprocessor ftp_telnet_protocol: telnet normalize ayt_attack_thresh
200
preprocessor ftp_telnet_protocol: ftp server default def_max_param_len
100
alt_max_param_len 200 { CWD } cmd_validity MODE < char ASBCZ >
cmd_validity MDTM
< [ date nnnnnnnnnnnnnn[.n[n[n]]] ] string > chk_str_fmt { USER PASS
RNFR RNTO
SITE MKD } telnet_cmds yes data_chan
preprocessor ftp_telnet_protocol: ftp client default max_resp_len 256
bounce yes
telnet_cmds yes
preprocessor smtp: ports { 25 } inspection_type stateful normalize
cmds normalize_
cmds { EXPN VRFY RCPT } alt_max_command_line_len 260 { MAIL }
alt_max_command_
line_len 300 { RCPT } alt_max_command_line_len 500 { HELP HELO ETRN }
alt_max_command_line_len 255 { EXPN VRFY }

##################################################################
# Step #3: Setting output plugin
##################################################################

output alert_syslog: LOG_LOCAL7
config logdir: /var/www
config alert_with_interface_name
config umask: 022
config checksum_mode: none
config show_year
config interface: eth0
config detection: search-method ac
config threshold: memcap 131072
config nolog
config order: pass activation dynamic alert log

#database
output database: log, mysql, user=root password=tugasakir dbname=snort
host=localhost
```

```
##################################################################
# Step #4: Setting rule file path
##################################################################
include $RULE_PATH/dos.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/backdoor.rules
include threshold.conf
```

Syntax PHP Konversi Desimal ke IP

```
<?
function dec2IP ($dec) {
      $hex = dec2hex ($dec);
      if (strlen($hex) == 7) $hex = "0".$hex;
      $one = hexdec(substr($hex,0,2));
      $two = hexdec(substr($hex,2,2));
      $three = hexdec(substr($hex,4,2));
      $four = hexdec(substr($hex,6,2));
      $ip = $one.".".$two.".".$three.".".$four;
      return ($ip);
}
function ip2dec ($ip)
{
  $dec = 0.0;
  $val = explode('.', $ip);
  for ($i=0; $i<count($val); $i++)
  {
  $dec = ($dec *256.0);
  $dec+=($val[$i]*1.0);
  }
  return $dec;
 }



function dec2hex($dec) {
      if($dec > 2147483648) {
      $result = dechex($dec - 2147483648);
      $prefix = dechex($dec / 268435456);
      $suffix = substr($result,-7);
      $hex = $prefix.str_pad($suffix, 7, "0000000", STR_PAD_LEFT);
   }
   else {
      $hex = dechex($dec);
   }
   $hex = strtoupper ($hex);
   return($hex);
}
function hex2str($hex) {
```

```
$str = "";



   $i = 0;
while ($i < strlen($hex)) {
      $tmp = hexdec(substr($hex,$i,2));

      if ($tmp < 32) $tmp = 46;
      if ($tmp > 126) $tmp = 46;
      $tmp = chr($tmp);
      if ($tmp == "<") $tmp = "&lt";
      if ($tmp == ">") $tmp = "&gt";
      $str .= $tmp;
      $i += 2;
    }
return $str;

}
function convertTCPFlags($decdata){
      $str = "";
      $binstr = decbin ($decdata);
      $binstr = str_pad($binstr, 8, "0", STR_PAD_LEFT);
      $i = 8;
      while ($i > 2) {
      $tmp = substr($binstr,$i-1,1);
      switch ($i) {
      case 3:
      if ($tmp == 0) $str = "*".$str;
      else $str = "U".$str;
      break;

case 4:
      if ($tmp == 0) $str = "*".$str;
      else $str = "A".$str;
      break;

case 5:
      if ($tmp == 0) $str = "*".$str;
      else $str = "P".$str;
      break;

case 6:
      if ($tmp == 0) $str = "*".$str;
      else $str = "R".$str;
      break;

case 7:
      if ($tmp == 0) $str = "*".$str;
      else $str = "S".$str;
      break;
```

```
case 8:
      if ($tmp == 0) $str = "*".$str;
      else $str = "F".$str;
      break;



      }
      $i--;
   }
   $str = "**".$str;
   return ($str);
}
?>
```

Konfigurasi Gammu

```
$ sudo apt-get install gammu gammu-smsd

$ sudo dmesg

[  16.622702] USB Serial support registered for GSM modem (1-port)
[  16.622727] option 1-3:1.0: GSM modem (1-port) converter detected
[  16.622813] usb 1-3: GSM modem (1-port) converter now attached to
ttyUSB0
[  16.622821] option 1-3:1.1: GSM modem (1-port) converter detected
[  16.622864] usb 1-3: GSM modem (1-port) converter now attached to
ttyUSB0
```

Buat file bernama gammurc di dalam direktori /etc, lalu isikan letal port yang terdeteksi di atas.

```
$ sudo vim /etc/gammurc
[gammu]
port = /dev/ttyUSB1
connection = at #huawei E270
logfile = /etc/gammulog
logformat = textall
use_locking = yes
```

Selanjutnya adalah konfigurasi gammu dengan mysql dan PHP agar dapat berhubungan. edit pada file /etc/gammu-smsdrc. Konfigurasi sama dengan konfigurasi yang ada di MySQL

```
$ sudo vim /etc/gammu-smsdrc
[gammu]
port=/dev/ttyUSB1
connection = at #huawei E270


[smsd]
PIN=”
service=mysql
DeliveryReport = sms
logfile = /etc/smsdlog
debuglevel=255

#sesuaikan dengan konfigurasi MySQL
User = root
Password =tugasakhir
PC = 127.0.0.1
Database = sms

$ sudo gammu -identify

############################################################
# Yaken-Aspire-4732z ~  # gammu -c /root/.gammurc -identify
# Device                : /dev/ttyUSB1
# Manufacturer          : huawei
# Model                 : unknown (E2700)
# Firmware              : 11.304.18.00.47
# IMEI                  : 354652010339216
# SIM IMSI              : 510018440939298
# Yaken-Aspire-4732z ~  #
############################################################
```

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
PT. BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

NAMA : EGAL DERIEKSEN
NIM : 05.12.633
JURUSAN : Teknik Elektro S-1
KONSENTRASI : Teknik Komputer Informatika
MASA BIMBINGAN : 22 November 2011 s/d 4 Juni 2012
JUDUL : **OTOMATISASI DETEKSI GANGGUAN KEAMANAN JARINGAN KOMPUTER DENGAN NOTIFIKASI SMS (SHORT MESSAGING SERVICE)**

Dipertahankan dihadapan Majelis Penguji Skripsi Jenjang Strata Satu (S-1) pada :

Hari : Sabtu
Tanggal : 18 Februari 2012
Dengan Nilai : 76,85 (B+)

PANITIA UJIAN SKRIPSI

Ketua Majelis Penguji,

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y.1018800189

Sekretaris Majelis Penguji,

Dr. Eng. Aryuanto S, ST, MT
NIP.Y.1030800417

ANGGOTA PENGUJI

Dosen Penguji I

Dr. Eng. Aryuanto S, ST, MT
NIP.Y.1030800417

Dosen Penguji II

Michael Ardita, ST, MT
NIP. P. 1031000434

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
PT. BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

FORMULIR PERBAIKAN SKRIPSI

Dalam pelaksanaan ujian skripsi jenjang Strata Satu (S-1) Jurusan Teknik Elektro Konsentrasi Teknik Komputer dan Informatika, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : EGAL DERIEKSEN
NIM : 05.12.633
JURUSAN : Teknik Elektro S-1
KONSENTRASI : Teknik Komputer Informatika
MASA BIMBINGAN : 22 November 2011 s/d 22 Juni 2012
JUDUL : **OTOMATISASI DETEKSI GANGGUAN KEAMANAN JARINGAN KOMPUTER DENGAN NOTIFIKASI SMS (SHORT MESSAGING SERVICE)**

Tanggal	Uraian	Paraf
Penguji II 01- 08 - 2012	Judul diperbaiki	11/9'12
	Bab II: -Rumusan masalah ✓ -Flowcart diperbaiki ✓ -Tabel Database pindah ke Lampiran ✓	
	Bab III: -Tambahkan diagram cara kerja system ✓	
	Bab IV: -Implementasi & pengujian ✓ -Konfogurasi Gammu ✓ -Konfigurasi Snort ✓ -Konfigurasi MySql ✓	
	Daftar Pustaka :Lengkapi dengan Judul dan tangga akses ✓	

Disetujui,

Dosen Penguji I	Dosen Penguji II
Dr. Eng. Aryuanto S, ST, MT	Michael Ardita, ST, MT
NIP.Y.1030800417	NIP. P. 1031000434

Mengetahui,

Dosen Pembimbing I	Dosen Pembimbing II
Sotyohadi,ST	Sonny Prasetio, ST, MT
NIP. P. 1039700309	NIP. P. 1031000433

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura – gura No.2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Egal Derieksen
Nim : 05.12.633
Masa Bimbingan : 22-November-2011 s/d 22-Juni-2012
Judul Skripsi : Otomatisasi Deteksi Gangguan Sistem Jaringan Komputer Dengan Notifikasi SMS(Short Message Service)

No	Tanggal	Uraian	Paraf Pembimbing
1	11-02-2012	Revisi bab 1 : Pendahuluan	
2	12-02-2012	Acc: Bab 1	
3	12-02-2012	Revisi bab 2 dan 3 : Format Penulisan	
4	14-02-2012	Acc: Bab 2	
5	15-02-2012	Acc: Bab 3	
6	15-02-2012	Revisi bab 4 : Diagram Pengujian Sistem	
7	16-02-2012	Acc: Bab 4	
8	16-02-2012	Acc: Bab 5	
9			
10			

Malang,

Dosen pembimbing I

Sotyohadi, ST
NIP. P 1039700309

Form S-4b

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura – gura No.2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Egal Derieksen
Nim : 05.12.633
Masa Bimbingan : 22-November-2011 s/d 22-Juni-2012
Judul Skripsi : Otomatisasi Deteksi Gangguan Sistem Jaringan Komputer Dengan Notifikasi SMS(Short Message Service)

No	Tanggal	Uraian	Paraf Pembimbing
1	01-02-2012	Revisi bab 1 : Rumusan Masalah	
2	03-02-2012	Acc: Bab 1	
3	03-02-2012	Revisi Bab 2 : Keterangan Penunjukkan Gambar	
4	06-02-2012	Acc: Bab 2	
5	06-02-2012	Revisi Bab 3 : Instalasi Gammu	
6	07-02-2012	Acc Bab 3	
7	07-02-2012	Revisi Bab 4 : Keterangan Penunjukkan Gambar	
8	08-02-2012	Acc Bab 4	
9	08-02-2012	Acc Bab 5	
10			

Malang,

Dosen pembimbing II

Sonny Prasetio, ST.MT
NIP. P 1031000433

Form S-4b