UNIVERSITAS INDONESIA

RANCANG BANGUN DAN IMPLEMENTASI IDS *(INTRUSION DETECTION SYSTEM) SERVER* DAN SISTEM MONITORING BERBASIS *WEB* PADA *NETWORK ADMISSION CONTROL (NAC)* UNTUK MENINGKATKAN KEAMANAN JARINGAN

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik

TAUFIK WICAKSONO
0706199994

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK ELEKTRO
UNIVERSITAS INDONESIA
DEPOK
JUNI 2009

HALAMAN PERNYATAAN ORISINALITAS

Skripsi ini adalah hasil karya saya sendiri,

dan semua sumber baik yang dikutip maupun dirujuk

telah saya nyatakan dengan benar.

Nama : Taufik Wicaksono

NPM : 07 06 19 99 94

Tanda Tangan :

Tanggal : 17 Juni 2009

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh :
Nama : Taufik Wicaksono
NPM : 0706199994
Program Studi : Teknik Elektro
Judul Skripsi : Rancang bangun dan Implementasi *IDS (Intrusion Detection System) Server* dan Sistem Monitoring berbasis *Web* pada *Network Admission Control (NAC)* untuk Meningkatkan Keamanan Jaringan.

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik pada Program Studi Teknik Elektro, Fakultas Teknik, Universitas Indonesia

DEWAN PENGUJI

Pembimbing : Muhammad Salman ST, MIT ( )

Penguji : Dr. Ir. Anak Agung Putri Ratna M.Eng ( )

Penguji : Ir. Endang Sriningsih MT., Si ( )

Ditetapkan di : Depok

Tanggal : 17 Juni 2009

KATA PENGANTAR

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya, saya dapat menyelesaikan skripsi ini. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Teknik Jurusan Teknik Elektro pada Fakultas Teknik Universitas Indonesia. Saya menyadari bahwa, tanpa bantuan dan bimbingan dari berbagai pihak, dari masa perkuliahan sampai pada penyusunan skripsi ini, sangatlah sulit bagi saya untuk menyelesaikan skripsi ini. Oleh karena itu, saya mengucapkan terima kasih kepada:

1. Muhammad Salman ST, MIT selaku dosen pembimbing yang telah menyediakan waktu, tenaga, dan pikiran untuk mengarahkan saya dalam penyusunan skripsi ini.
2. Orang tua dan keluarga saya yang telah memberikan bantuan dukungan secara moral,
3. Sahabat yang telah banyak membantu saya dalam menyelesaikan skripsi ini.

Akhir kata, saya berharap Tuhan Yang Maha Esa berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga skripsi ini membawa manfaat bagi pengembangan ilmu.

Depok, 17 Juni 2009

Penulis

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini:

Nama	: Taufik Wicaksono
NPM	: 0706199994
Program Studi	: Teknik Elektro
Departemen	: Teknik Elektro
Fakultas	: Teknik
Jenis karya	: Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*)** atas karya ilmiah saya yang berjudul :

RANCANG BANGUN DAN IMPLEMENTASI IDS *(INTRUSION DETECTION SYSTEM) SERVER* DAN SISTEM MONITORING BERBASIS *WEB* PADA *NETWORK ADMISSION CONTROL (NAC)* UNTUK MENINGKATKAN KEAMANAN JARINGAN

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/ formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan skripsi saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok

Pada tanggal : 17 Juni 1009

Yang menyatakan

(…….)

ABSTRAK

Nama : Taufik Wicaksono
Program Studi : Teknik Elektro
Judul : Rancang bangun dan Implementasi IDS (Intrusion Detection System) Server dan Sistem Monitoring berbasis Web pada Network Admission Control (NAC) untuk Meningkatkan Keamanan Jaringan.

Kebutuhan akan akses internet dewasa ini sangat tinggi, hal ini mengakibatkan peningkatan permintaan akses ke jaringan yang aman semakin tinggi. Keadaan ini menuntut admin jaringan agar lebih selektif dalam memperbolehkan user melakukan akses ke jaringan. Setelah proses seleksi awal pada user admin jaringan juga bertugas untuk memproteksi user dari gangguan yang dilakukan user lain atau dari akses luar jaringan. Konsep jaringan seperti ini menjadi dasar munculnya konsep jaringan NAC.

Network Admission Control (NAC) adalah teknologi keamanan jaringan komputer dimana *client* komputer harus melakukan autentifikasi sebelum diperbolehkan mengakses jaringan. Salah satu teknologi NAC yang terkenal adalah Cisco NAC (C-NAC). Terdapat dua fasilitas utama yang dimiliki oleh NAC server yaitu policy server dan IDS server. *Policy server* bertugas untuk melakukan authentifikasi terhadap *user* yang akan mengakses ke *network devices* jaringan. IDS *server* bertugas untuk melakukan deteksi terhadap serangan yang terjadi terhadap *server*, sehingga *server* dapat memberikan peringatan dan kemudian dapat menghentikan serangan. IDS server juga memiliki kemampuan untuk memberikan peringatan melalui SMS dan memiliki fasilitas monitoring serangan melalui web.

IDS *Server* dibuat menggunakan *operating system* Linux. Sistem ini dibagi menjadi beberapa modul yaitu IDS *software* yaitu snort, report modul yaitu BASE, dan *client – server* modul yang bertugas mengirimkan *alerting* kepada *policy server* . Sementara *network devices* yang digunakan pada arsitektur jaringan ini adalah sebuah switch dan router.

Pengujian sistem dilakukan dengan melakukan beberapa variasi serangan terhadap *server* yaitu *denial of service* (DOS), *port scanning*, dan ICMP *flood*. Dari *server* akan diambil parameter *response time* dan *action time*. Pengujian juga membandingkan nilai *response time* apabila menggunakan 1 client dan 5 client. Apabila penyerangan menggunakan 5 *client* menyebabkan adanya penuruan *response time* sebesar 64.81% apabila dilakukan penyerangan menggunakan DoS dan 92.65% apabila 5 client melakukan penyerangan menggunakan *port scanning.*

Kata Kunci : NAC, IDS *Server*, *Policy server*, Snort, Base, DOS, *Port scan*, ICMP *flood*

ABSTRACT

Name : Taufik Wicaksono
Study Program: Electrical Engineering
Title : Design and Implementation of Intrusion Detection System (IDS) Server and Web based Monitoring System on Network Admission Control (NAC) for Improving the Network Security.

Requirement for accessing internet at the moment is very high, this matter an improvement of request access to secure network. This situation make network administrator to be more selective for give user to access network, so requirement for some system that can perform authentication to user for accessing network is important. This network concept becomes the appearance of base conception of Network Admission Control (NAC).

Network Admission Control (NAC) is computer network security technology where computer client have to establish authentication before allowing to access the network. One of NAC technology most popular is Cisco NAC (C-NAC). There are two main features of NAC server that is policy server and Intrusion Detection System (IDS) server. Policy server undertakes to do authentication to user to access to network devices network. IDS server function to probe attacks or intrusion against the server, so IDS server can give alerting and then be able to stop the intrusion. IDS server also be able to reporting to administrator through SMS and monitoring through web when intrusion detected.

IDS server build use operating system Linux. This system divided becomes three modules that are IDS software use SNORT, report module use Basic Analysis Security Engine (BASE) and client – server module to communicate between IDS server and policy server. NAC network design will be use router and switch.

Examination of system to carry out some variation of attacks against the server. The variation of attack is denial of service (DOS), port scanning, and ICMP flood. Parameters are taken from the server is response time and action time. Examination also use comparison response time if use 1 client and 5 clients. Attack against IDS server show decreasing response time when server attacked by 5 client. IDS sever attacked use denial of service (DoS) response time decreasing 64.81% if attacked by 5 clients and 92.65% when 5 clients attacked use port scanning.

Key Words: NAC, IDS *Server*, *Policy server*, SNORT, BASE, DOS, *Port scan*, ICMP *flood*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR SINGKATAN

ARP	Address Resolution Protocol
BASE	Basic Analysis Security Engine
CNAC	Cisco Network Admission Control
DHCP	Dynamic *Client* Configuration Protocol
DoS	Denial of Service
DDoS	Distribute Denial of Service
NAC	Network Admission Control
TCP/IP	Transmission Control Protocol / Internet Protocol
UDP	User Datagram Protocol
VLAN	Virtual Local Area Network

BAB 1
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 harus dilindungi dari segala macam serangan dan bentuk usaha-usaha penyusupan atau pemindaian oleh pihak yang tidak berhak.

Sistem deteksi penyusup jaringan yang ada saat ini umumnya mampu mendeteksi berbagai jenis serangan tetapi tidak mampu mengambil tidakan lebih lanjut. Selain itu sistem juga tidak memiliki interaktivitas dengan administrator pada saat administrator tidak sedang mengadministrasi sistemnya. Hal ini merupakan suatu hal yang tidak efektif terutatam pada saat sistem berada dalam kondisi kritis.

Untuk mengatasi keadaan diatas, maka dibuatlah arsitektur jaringan berbasiss *Network Access Control (NAC)*. NAC adalah teknologi keamanan jaringan komputer dimana *client* komputer harus melaporkan status "kesehatan" sebelum boleh masuk kedalam jaringan. Teknologi NAC mempunyai 3 variasi yang berbeda yaitu: Cisco NAC (CNAC), NAP dari Microsoft dan *Trusted Network Connect* dari konsorsium TCG. Pada perkembangannyan anggota-anggota NAP dan TNC sedang bekerja sama dalam organisasi IETF untuk menggabung NAP dan TNC.

Cisco Network Admission Control (NAC) atau disingkat C-NAC. Yaitu suatu sistem yang memperbolehkan suatu perangkat *end device* seperti komputer untuk mengakses suatu jaringan utama berdasarkan identitas atau sekuriti ID yang dimiliki oleh user. Ketika sebuah perangkat jaringan seperti (switch, router, access pint, DHCP *server*, dll) dikonfigurasi menggunakkan NAC, maka NAC akan memaksa user untuk melakukan autentifikasi apabila akan mengakses suatu jaringan. Pada kondisi awal *user* tamu akan ditempatkan diarea tertentu yang terpisah dari jaringan utama, sehingga apabila tamu gagal melakukan autentifikasi maka user tersebut tidak dapat mengakses jaringan utama. Sistem NAC dapat melakukan perubahan konfigurasi pada suatu perangkat jaringan, seperti merubah

suatu DHCP class atau merubah konfigurasi VLAN pada switch. Sehingga sistem NAC memiliki kemampuan untuk mengijinkan atau tidak user yang akan mengakses suatu jaringan. NAC juga memiliki kemampuan untuk mengetahui adanya sebuah serangan terhadap sistem, sehingga sistem NAC dapat mengkarantina user yang akan melakukan serangan tersebut.

Network Admission Control (NAC) memiliki dua fungsi utama yaitu sebagai *policy server* dan *intrusion detection system (IDS) server*. *Policy server* bertindak sebagai *server* autentifikasi terhadap user yang akan mengakses kejaringan. IDS *Server* berfungsi sebagai pendeteksi ada tidaknya serangan terhadap sistem, serangan bisa seperti Denial of Service (DOS), Ping of Death (POD), dan *port scan*ner. Semua sitem yang terdapat dalam Network Admission Control (NAC) telah menjadi satu paket yang dikeluarkan oleh Cisco yaitu Cisco NAC. Karena untuk mendapatkan satu paket Cisco NAC membutuhkan biaya yang mahal. Maka pada skripsi ini kami akan membangun sebuah *server* yang memiliki fungsi sebagai *Network Admission Control (NAC)*. *Server* yang akan kami bangun berbasis Linux dan menggunakan *software* yang berlisensi *Genue Packet License (GPL).* Sehingga dapat menekan biaya untuk membangun sebuah *server* NAC.

Pada skripsi ini akan dibahas modul dari NAC *Server* yaitu sebagai *Intrution Detection System (IDS).* Pada modul IDS *server* menggunakan sistem operasi Fedora dan menggunakan Snort sebagai *software Intrution Detection System (IDS)*.

1.2 Tujuan

Tujuan dari penulisan skripsi ini adalah membangun sebuah *Intrusion Detection System (IDS) server* dan membangun suatu *web monitoring system* yang memiliki fungsi yang sama dengan NAC *Server*. Sehingga sistem ini dapat di implementasikan pada jaringan komputer.

1.3 Pembatasan Masalah

Pada skipsi ini akan dilakukan perancangan IDS *server* sebagai bagian dari NAC *server*. Beberapa modul yang digunakan pada disain IDS *server* yang

dibuat, yaitu modul IDS modul, report module dan *client – server* module. Pada IDS modul menggunakan *software* snort, yang memiliki fungsi untuk melakukan deteksi terhadap serangan yang terjadi terhadap *server* dan memberikan alerting. Untuk report module menggunakan *Basic Analysis Security Engine (BASE)* digunakan untuk mengelola data-data *alerting* yang dihasilkan oleh snort dan menampilkanya ke bentuk tampilan web. *Client – server module* bertugas untuk mengirimkan *alerting* ke *policy server*.

Bila terjadi serangan terhadap *server*. Maka snort akan mendetksi serangan tersebut dan memberikan alerting yang kemudian akan dikirimkan ke *policy server* untuk diklasifikasi berdasarkan prioritas serangan yang kemudian dilakukan tindakan untuk menghentikan serangan. Setiap serangan juga disimpan kemudian akan di tampilkan ke bentuk *web monitoring* menggunakan BASE.

Modul – modul *policy server* dibuat menggunakan platfrom Linux. Untuk BASE membutuhkan database MySQL dan program *client – server* menggunakan bahasa pemrograman perl. Sementara *network device* yang digunakan adalah router dan switch CISCO.

1.4 Metodologi Penelitian

Metode yang dilakukan pada tugas akhir ini adalah membuat *policy server* pada jaringan sederhana dimana terdiri dari *server*, network device dan *client*. Pengujian dilakukan dengan melakukan beberapa metode serangan terhadap IDS *server* sehingga bisa didapat *response time* dan *action time* dari IDS *server*.

1.5 Sistematika Penulisan

Sistematika penulisan pada tugas akhir ini dibuat menjadi 5 bab. Bab 1 berisi tentang latar belakang, tujuan dan batasan masalah yang dibahas dalam tugas akhir kali ini. Bab 2 berisi study literature tentang teori – teori yang berhubungan dengan jaringan komputer dan komponen – komponen yang digunakan pada pembuatan *policy server*. Bab 3 merupakan pembahasan tentang design dari modul IDS *server*. Bab 4 berisi tentang analisa dan pengujian modul IDS *server* yang sudah dirancang pada BAB 3. Dan pada bab 5 berisi kesimpulan dari hasil analisa.

BAB 2
KOMPONEN IDS *SERVER*

2.1 UMUM

Pada bab ini akan diberikan teori dasar yang melandasi permasalahan dan penyelesaiannya yang diangkat dalam skripsi ini. Teori dasar yang diberikan meliputi: keamanan jaringan, konsep *intrusion detection system (IDS),* konsep NAC dan komponen – komponen yang digunakan untuk menyelesaikan skripsi ini.

2.2 KEAMANAN JARINGAN

Keamanan jaringan secara umum adalah komputer yang terhubung ke jaringan yang mempunyai ancaman keamanan lebih besar daripada komputer yang tidak terhubung ke mana-mana. Dengan pengendalian yang teliti, resiko tersebut dapat dikurangi. Namun keamanan jaringan akan bertentangan dengan dengan jaringan akses. Dimana bila jaringan akses semakin mudah, maka keamanan jaringan semakin rawan, dan bila keamanan jaringan semakin baik, jaringan akses semakin tidak nyaman. Suatu jaringan di disain sebagai komunikasi data dengan tujuan meningkatkan akses ke sistem komputer, sementara keamanan didesain untuk mengontrol akses. Penyediaan keamanan jaringan adalah sebagian aksi penyeimbang antara jaringan akses dengan keamanan jaringan. [1]

Disini jaringan 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 memberi kunci di depan rumah. Hal seperti ini juga diterapkan pada keamanan jaringan. Keamanan dijaga untuk setiap *client-client* tertentu, tidak langsung pada jaringannya.

Salah satu problem keamanan jaringan yang paling penting adalah menentukan kebijakan keamanan (*security policy*) dalam keamanan jaringan. Perencanaan keamanan yang matang berdasarkan prosedur dan kebijakan dalam

keamanan jaringan akan membantu menentukan apa yang harus dilindungi, berapa besar biaya yang bertanggung jawab untuk menjalankan langkah-langkah yang diperlukan untuk melindungi bagian tersebut.

Langkah awal dalam mengembangkan rencana keamanan jaringan yang efektif adalah dengan mengenali ancaman yang mungkin datang. Dalam RFC 1244, Site Security Handbook, dibedakan tiga tipe ancaman:

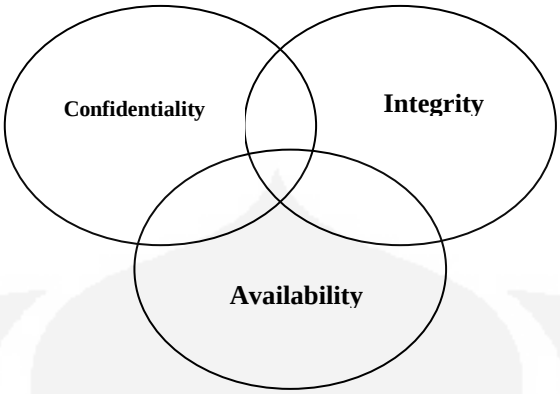
1. Akses tidak sah oleh orang yang tidak mempunyai wewenang.
2. Kesalahan informasi, segala masalah yang dapat menyebabkan diberikanya informasi yang penting atau sensitif kepada orang yang salah, yang seharusnya tidak boleh mendapatkan informasi tersebut.
3. Penolakan terhadap *service*, segala masalah mengenai keamanan yang menyebabkan sistem mengganggu pekerjaan yang produktif.

Disini ditekankan keamanan jaringan dari segi perangkat lunak, namun keamanan jaringan sebenarnya hanyalah sebagian dari rencana keamanan yang lebih besar, termasuk rencana keamanan fisik dan penanggulangan bencana.

2.3 TUJUAN KEAMANAN JARINGAN

Pada dasarnya tujuan dari keamanan komputer yang disingkat dengan CIA, yang merupakan singkatan dari: [1]

- **Confidentiality**: Merupakan usaha untuk menjaga informasi dari orang yang tidak berhak mengakses. Confidentialiy biasanya berhubungan dengan informasi yang diberikan ke pihak lain.
- **Integrity**: Keaslian pesan yang dikirim melalui sebuah jaringan dan dapat dipastikan bahwa informasi yang dikirim tidak dimodifikasi oleh orang yang tidak berhak dalam perjalanan pengiriman informasi tersebut.
- **Availability**: Aspek availability atau ketersediaan berhubungan dengan ketersediaan informasi ketika dibutuhkan. Sistem informasi yang diserang atau yang dijebol dapat menghambat atau meniadakan akses ke informasi.



Gambar 2.1 Tujuan Keamanan Komputer [1]

Tabel 2.2 CIA

	Definition	Tools	Dependencies
Confidentiality	• Concealment of info & resources • Hide existence of info resources	• Encryption • Access Control	• Reliance on sistem • Assumption & trust about reliance
Integrity	• Trustworthinees of info & resources ➢ Authentification • Correctness of data ➢ Data integrity	• Prevention ➢ Block attempts ➢ Unauth action • Detection ➢ Block attempts ➢ Unauth action	• Assumption about source • Trust of source
Availability	• Ability to use info & resouces	• Sistem design • Statistical models use	• Accuracy of statistical models • ID anomalies

Untuk mencapai tujuan keamanan jaringan di atas maka dibutuhkan sebuah metode yang dapat melindungi sistem dari ancaman internal maupun dari eksternal. Salah satu metode yang dapat digunakan *adalah Intrusion Detection Sistem* (IDS) dan *Intrusion Prevention System* (IPS).

2.4 KEBIJAKAN KEAMANAN

Dalam keamanan jaringan, peranan manusia yang memegang tanggungjawab keamanan sangat berperan. Keamanan jaringan tidak akan efektif kecuali orang-orangnya mengetahui tanggung jawabnya masing-masing. Dalam menentukan keamanan jaringan. Kebijakan perlu ditegaskan apa-apa yang diharapkan, dan dari siapa hal tersebut diharapkan. Selain itu, kebijakan ini harus mencakup: [1]

1. Tanggung jawab keamanan user, meliputi antara lain keharusan user untuk mengganti passwordnya dalam periode tertentu, dengan aturan tertentu, atau memeriksa kemungkinan terjadinya pengaksesan oleh orang lain, dll.
2. Tanggung jawab keamanan sistem administrator, misalnya perhitungan keamanan tertentu, memantau prosedur-prosedur yang digunakan pada *client*.
3. Penggunaan yang benar atas resource network, dengan menentukan siapa yang dapat menggunakan resource tersebut, apa yang dapat dan tidak boleh mereka lakukan.

Dalam mendesain keamanan jaringan dapat dipakai 5 tahap dasar berikut ini:

1. Pada aplikasi yang bersangkutan, manakah mekanisme proteksi difokuskan, apakah pada data, apakah pada data, operasi atau pengguna?
2. Pada layer manakah mekanisme keamanan dari sistem komputer akan ditempatkan?
3. Mana yang lebih diinginkan, kesederhanaan dan jaminan tinggi atau sistem yang memiliki fasilitas yang banyak tapi tingkat keamanan rendah?
4. Apakah tugas untuk mendefinisikan dan mengerapkan keamanan harus diberikan pada badan terpusat atau diberikan pada masing-masing individu pada suatu sistem?
5. Bagaimana dapat melindungi dari penyerang yang ingin memperoleh akses pada sistem yang dilindungi mekanisme proteksi?

Manusia adalah salah satu faktor yang sangat penting tetapi seringkali dilupakan dalam pengembangan teknologi informasi. Begitu juga dalam mengembangkan sistem keamanan. Sebagai contoh penggunaan *password* yang sulit justru menyebabkan pengguna menuliskannya pada kertas yang ditempel dikomputer. Jadi dalam pennyusunan kebijakan keamanan, factor manusia dan budaya setempat haruslah sangat dipertimbangkan.

Seperti yang diungkapkan oleh Kevin Mitnick (seorang cracker yang terkenal), sebagian besar celah diperoleh melalui rekayasa sosial yang menunjukkan kelemahan user. Saat ini di Indonesia masih banyak praktik dari

pengguna yang sangat mengabaikan faktor keamanan bahkan di bank. Banyak user tergolong sensitif missal bank saling menukar password, bahkan sering menuliskan password dan menempel di dekat monitornya.

Pada dasarnya seorang user memiliki tanggung jawab dalam menggunakan sumber daya komputasinya. Tanggung jawab ini berdasarkan konvensi yang berupa (Ladkin, 1999):

- Legal, sebagai contah, tak mengancam orang lain, tak menyamar sebagai orang lain, jangan merusak pekerjaan orang lain.
- Kontraktual, sebagai contoh tak bermain game, tak menulis email pribadi, menjaga kontrak bisnis tetap bersifat rahasia.
- Sosial, tak menunjukkan gambar porno, atau tak membaca email milik orang lain.

Di atas sudah dibahas tentang bagaimana suatu kebijakan keamanan dibuat dan pertimbangan yang harus dibuat. Semua hal tersebut tergantung dari organisasi yang memerlukannya, keputusan yang diambil merupakan keputusan tentang keamanan computer, masalah biaya dari suatu sistem keamanan juga perlu dipertimbangkan.

Apa yang diinginkan dan keamanan yang bagaimana yang diinginkan perlu dipertimbangkan, yang di antaranya sebagai berikut:

- *Affordability:* berapa banyak uang yang dibutuhkan dari suatu sistem keamanan. Merupakan suatu hal yang sangat penting karena jika ingin membangun suatu sistem computer yang benar-benar aman tentu dibutuhkan biaya yang sangat cukup besar. Sebagai contoh, jika mau memakai firewall yang multilayer, tentu membutuhkan biaya yang besar.
- *Fungsionalitas:* adalah kelangsungan hidurp dari sistem itu sendiri, karena operator yang menjaga suatu sistem harus memiliki skill yang baik.
- *Kompatibilitas budaya:* Budaya juga perlu diperhatikan. Jika suatu organisasi tidak memikirkan hal ini, kemungkinan serangan akan banyak disebabkan oleh factor budaya.
- *Status:* factor hokum perlu juga dipertimbangkan. Meskipun suatu website sudah memenuhi syarat keamanan policy namun tidak memiliki status

yang legal, mungkin web site tersebut akan diblok oleh pihak yang berwenang.

2.5 JENIS SERANGAN

Jenis dan teknik serangan yang mengganggu jaringan komputer beraneka macam, diantaranya: [1]

- **Back Orifice (BO)** adalah sebuah alat bantu *remote* administrasi komputer dari jarak jauh yang dapat digunakan untuk mengontrol sistem operasi Microsoft Windows, yang dikembangkan oleh kelompok peretas profesional *Cult of the Dead Cow*. *Back Orifice* dirilis pertama kali untuk platform Windows NT pada tahun 1997. Namanya merupakan pelesetan dari Microsoft BackOffice *Server*. Pada tahun 1999, grup yang sama merilis versi baru, yang disebut sebagai *Back Orifice* 2000 atau sering disebut BO2K. Meskipun pada dasarnya alat bantu ini merupakan salah satu bentuk dari Trojan horse, yang dapat digunakan untuk mendapatkan akses dan kontrol penuh terhadap mesin target, programini menawarkan banyak fitur, khususnya untuk mengendalikan sistem operasi Windows NT. Tampilan yang digunakannya sangatlah mudah dan sederhana, sehingga para peretas pemula pun dapat menggunakannya.
- **Denial of Service (DOS)** adalah jenis serangan terhadap sebuah komputer atau *server* di dalam jaringan internet dengan cara menghabiskan *resource* yang dimiliki oleh komputer tersebut sampai komputer tersebut tidak dapat menjalankan fungsinya dengan benar sehingga secara tidak langsung mencegah pengguna lain untuk memperoleh akses layanan dari komputer yang diserang tersebut.Dalam sebuah serangan *Denial of Service (DoS)*, penyerang akan mencoba untuk mencegah akses seorang pengguna terhadap sistem atau jaringan dengan menggunakan beberapa cara, yakni sebagai berikut:
    - Membanjiri lalu lintas jaringan dengan banyak data sehingga lalu lintas jaringan yang datang dari pengguna yang terdaftar menjadi

tidak dapat masuk ke dalam sistem jaringan. Teknik ini disebut sebagai *traffic flooding*.

- Membanjiri jaringan dengan banyak request terhadap sebuah layanan jaringan yang disedakan oleh sebuah *client* sehingga request yang datang dari pengguna terdaftar tidak dapat dilayani oleh layanan tersebut. Teknik ini disebut sebagai *request flooding*.
- Mengganggu komunikasi antara sebuah *client* dan kliennya yang terdaftar dengan menggunakan banyak cara, termasuk dengan mengubah informasi konfigurasi sistem atau bahkan perusakan fisik terhadap komponen dan *server*.

- ***Port scan*ning:** merupakan suatu proses untuk mencari dan membuka port pada suatu jaringan computer. Dari hasil scanning akan didapat letak kelemahan sistem tersebut. Pada dasarnya sistem *port scanning* mudah untuk dideteksi, tetapi penyerang akan menggunakan berbagai metode untuk menyembunyikan serangan.
- **Teardrop:** Merupakan suatu teknik yang dikembangkan dengan mengekploitasi proses *assembly-reassembly* paket data. Dalam jaringan internet seringkali data harus dipotong kecil-kecil untuk menjamin reliablitas dan proses multiple akses jaringan. Potongan paket data ini kadang harus dipotong ulang menjadi lebih kecil lagi pada saat disalurkan melalui saluran *Wide Area Network (WAN)* agar pada saat melalui saluran WAN yang tidak reliable. Pada proses pemotongan data paket yang normal, setiap potongan diberi informasi offset data yang kira-kira berbunyi “potongan paket ini merupakan potongan 600byte dari total 800 byte paket yang dikirim. Program teardrop akan memanipulasi offset potongan data sehingga akhirnya terjadi overlapping antara paket yang diterima di bagian penerima, setelah potongan paket ini di *reassembly* seringkali *overlapping* ini menimbulkan sistem yang *crass*, *hang*, dan *reboot* di penerima.
- **IP-Spoofing:** adalah suatu serangan teknis yang rumit yang terdiri dari beberapa komponen. Ini adalah eksploitasi keamanan yang bekerja dengan menipu komputer, seolah-olah yang menggunakan computer tersebut

adalah orang lain. Hal ini terjadi karena *design flaw* (salah rancang). Lubang keamanan yang dapat dikategorikan ke dalam kesalah desin adalah desin urutan nomor *sequence numbering* dari paket TCP/IP. Kesalahan ini dapat dieksplotasi sehingga timbul masalah.

- **Smurft Attack:** Serangan jenis ini biasanya dilakukan dengan menggunakan IP *spoofing*, yaitu mengubah nomor IP dari datangnya request. Dengan menggunakan IP spoofing, respons dari ping tadi dialamatkan ke computer yang IP-nya di spoof. Akibatnya, computer tersebut akan menerima banyak paket. Hal ini dapat mengakibatkan pemborosan bandwith jaringan yang terhubung dengan komputer tersebut.
- **UDP *Flood*:** Pada dasarnya mengaitkan dua sistem tanpa disadari. Dengan cara spoofing, *User Datagram Protocol* (UDP) *flood* attck akan menempel pada servis UDP chargen di salah satu mesin yang digunakan untuk keperluan "percobaan" akan mengirimkan sekelompok karakter ke mesin lain, yang diprogram untuk meng-echo setiap kiriman karakter yang diterima melalui *service chargen*. Karena paket UDP tersebut di spoofing di antara ke dua mesin tersebut maka yang tidak berguna di antara ke dua mesin tersebut maka yang terjadi adalah banjir tanpa henti kiriman karakter yang tidak berguna diantara kedua mesin. Untuk menggulangi *UDP flood*, anda dapat mendisable semua *service* UDP di semua mesin dijaringan, atau yang lebih mudah adalah dengan memfilter pada firewall semua *service* UDP yang masuk.
- **ICMP *flood*:** Seorang penyerang melakukan eksploitasi sistem dengan tujuan untuk membuat suatu target *client* menjadi *crash*, yang disebabkan oleh pengiriman sejumlah paket yang besar ke arah target *client*. *Exploting* sistem ini dilakukan dengan mengirimkan suatu perintah ping dengan tujuan *broadcast* atau *multicast* di mana si pengirim dibuat seolah-olah adalah target *client*. Semua pesan balasan dikembalikan ke target *client*. Hal inilah yang membuat target *client* menjadi crash dan menurunkan kinerja jaringan. Bahkan hal ini dapat mengakibatkan *denial of service*.

Dari pembahasan di atas banyaknya teknik dan jenis serangan yang pada dasarnya serangan yang pada dasarnya serangan terhadap sistem dibagi menjadi dua dimensi seperti pada Tabel 2.2.

Tabel 2.2 Point dan Jenis Serangan

Point Serangan		
	Internal User	**External User**
Denial of Service	Mengganggu	Mengganggu
Increase Privilege	Serius mengganggu	Resiko yang serius
Superuser Privilege	Sangat Serius	Bencana

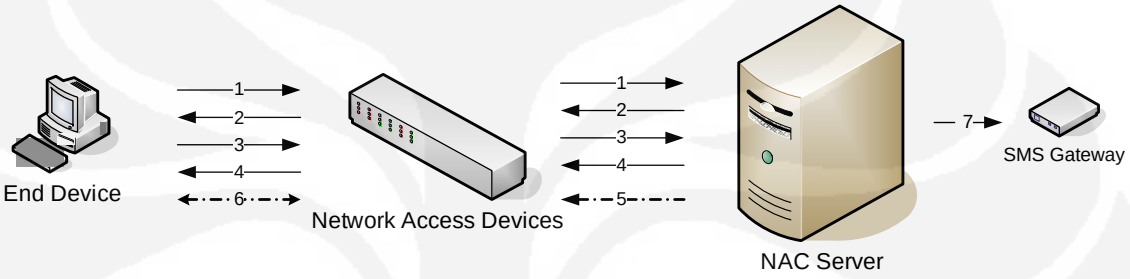
Faktor utama dari bencana ini disebabkan oleh orang yang mempunyai akses dari dalam sistem dengan bantuan orang yang berasal dari external network. Banyak terjadi serangan yang melibatkan orang yang di dalam jaringan. Untuk mendapatkan sistem yang benar-benar aman terlebih dahulu harus membenahi jaringan internal baik dari kebijakan keamanan, dan hal-hal lain yang mungkin.

2.6 NETWORK ADMISSION CONTROL (NAC)

Network Admission Control (NAC) berasal dari salah satu paket Cisco's yaitu *Cisco Network Admission Control (NAC)* atau disingkat C-NAC. Yaitu suatu sistem yang memperbolehkan suatu perangkat *end device* seperti komputer untuk mengakses suatu jaringan utama berdasarkan identitas atau sekuriti ID yang dimiliki oleh user. Ketika sebuah perangkat jaringan seperti (switch, router, access pint, DHCP *server*, dll) dikonfigurasi menggunakkan NAC, maka NAC akan memaksa user untuk melakukan *autentifikasi* apabila akan mengakses suatu jaringan. Pada kondisi awal user tamu akan ditempatkan diarea tertentu yang terpisah dari jaringan utama, sehingga apabila tamu gagal melakukan *autentifikasi* maka user tersebut tidak dapat mengakses jaringan utama. Sistem NAC dapat melakukan perubahan konfigurasi pada suatu perangkat jaringan, seperti merubah suatu DHCP class atau merubah konfigurasi VLAN pada switch. Sehingga sistem NAC memiliki kemampuan untuk mengijinkan atau tidak user yang akan mengakses suatu jaringan. NAC juga memiliki kemampuan untuk mengetahui adanya sebuah serangan terhadap sistem, sehingga sistem NAC dapat mengkarantina user yang akan melakukan serangan tersebut.

Network Admission Control (NAC) memiliki dua fungsi utama yaitu sebagai *Policy server* dan Intrusion Detection Sistem (IDS) *Server*. *Policy server* bertindak sebagai *server* autentifikasi terhadap user yang akan mengakses kejaringan. IDS *Server* berfungsi sebagai pendeteksi ada tidaknya serangan terhadap sistem, serangan bisa berupa Denial of Service (DOS), Ping of Death (POD), dan virus.

Berikut adalah cara kerja suatu NAC *Server* :



Gambar 2.2 Cara Kerja NAC *Server*

Keterangan :

1. *End user* baru melakukan akses ke jaringan, secara default end user mendapatkan IP secara automatis.
2. NAC *server* mengirimkan autentifikasi menggunakan web browser.
3. *End user* mengirimkan *username* dan *password*.
4. NAC *Server* melakukan cek ke database dan mengirimkan respon ke *end device* (diperbolehkan masuk ke jaringan atau tidak).
5. NAC *Server* melakukan setting VLAN pada *Networks Access Devices (switch)* sesuai dengan *username*.
6. *End Device* dapat masuk ke jaringan den memperoleh *previledge* sesuai dengan *username* yang digunakan.
7. Bila terdeteksi suatu serangan terhadap *server*, NAC akan menutup port VLAN yang terdeteksi melakukan serangan dan mengirimkan pesan kepada administrator jaringan menggunakan sms atau email.

2.7 INTRUSION DETECTION SYSTEM DAN PREVENTION

2.7.1 Intrusion Detection Sistem (IDS)

Intrusion Detection Sistem (IDS) adalah sistem yang dapat melihat pola dari serangan di dalam jaringan komputer, dimana pola tersebut berupa paket yang mencurigakan. Ketika IDS melihat pola terebut di dalam suatu lalu lintas paket di dalam jaringan maka disebut Network Based IDS. Sedangkan *Client* based IDS memiliki dua macam yaitu menganalisa logfile yang berisi pola serangan terhadap suatu *client* dan menganalisa lalu lintas paket yang lewat di dalam suatu jaringan.

2.7.1.1 Network-Based IDS

Network-Based Intrusion Detection Sistems (IDS) menggunakan paket yang terdapat di dalam jaringan sebagai sumber data. IDS menggunakan network adapter dengan *promiscuous mode* sehingga dapat melihat dan menganalisa semua trafik paket yang lewat di dalam jaringan secara *real-time*. Modul untuk mengenali adanya serangan menggunakan empat macam teknik untuk mengetahui pola dari serangan, yaitu: [1]

- Pola, ekspresi atau pencocokan *bytecode*
- Frekuensi atau threshold paket yang lewat di dalam jaringan.
- Hubungan antara setiap event.
- Statistik pendeteksi anomali paket.

Ketika sebuah serangan terdeteksi, IDS akan memberikan respon kepada modul untuk menyediakan beberapa pilihan cara untuk pemberitahuan, alarm dan mengambil respon terhadap serangan tersebut. Respon bermacam-macam tergantung dari setiap produk, tetap selalu ada pemberitahuan terhadap administrator, setiap pemutusan sambungan atau *session* terekam oleh analisa forensik sebagai bukti.

2.7.1.2 *Client* Based IDS

Client_based intrusion detection dimulai pada awal tahun 1980-an sebelum jaringan dikenal secara umum, dan tidak sekompleks dan terkoneksi

secara luas seperti saat ini. Pada situasi yang masih sederhana itu umumnya digunakan *audit log* untuk melihat kembali aktifitas yang mencurigakan. Penyusupan dirasakan masih jarang sampai ditemukan fakta dari analisis yang membuktikan perlunya pencegahan terhadap serangan di masa mendatang.

Saat ini HBIDS tetap sebagai tool yang mampu memahami serangan-serangan yang terjadi sebelumnya dan menentukan metoda yang sesuai untuk mengatasi jika mereka melakukan serangan lagi. HBIDS masih menggunakan audit logs, tetapi menjadi lebih otomatis, dengan peningkatan teknik deteksi yang lebih responsif dan lebih canggih. HBIDS khususnya memonitor sistem, kejadian-kejadian, dan log keamanan pada windows NT dan *syslog* pada lingkungan UNIX. Bilamana beberapa dari file berubah, maka IDS membandingkan masuknya log yang baru dengan *attack signatures* jika terjadi kecocokan, maka sistem akan menanggapi dengan memberikan tanda bahaya kepada administrator dan panggilan yang lain untuk memberikan tindakan segera.

HBIDS menggunakan informasi yang dihasilkan oleh *client* untuk mendeteksi penyalahgunaannya. Sumber informasi berbeda juga dapat digunakan, seperti *event log* dari sistem dan aplikasi, statistik penggunaan *account*, akses data penting, dan modifikasinya.

Kemampuan untuk merespon adanya suatu serangan sangat penting bagi sebuah intrusion detection sistem. Setiap network dan *client* based IDS memiliki beberapa pilihan di dalam merespon suatu serangan. Terdapat tiga kategori respon yaitu *notification, storage* dan aktif respon. Network dan *client*-based IDS juga memiliki tambahan kemampuan berdasarkan orentasi masing-masing. Hal ini dapat dilihat pada Tabel 2.3.

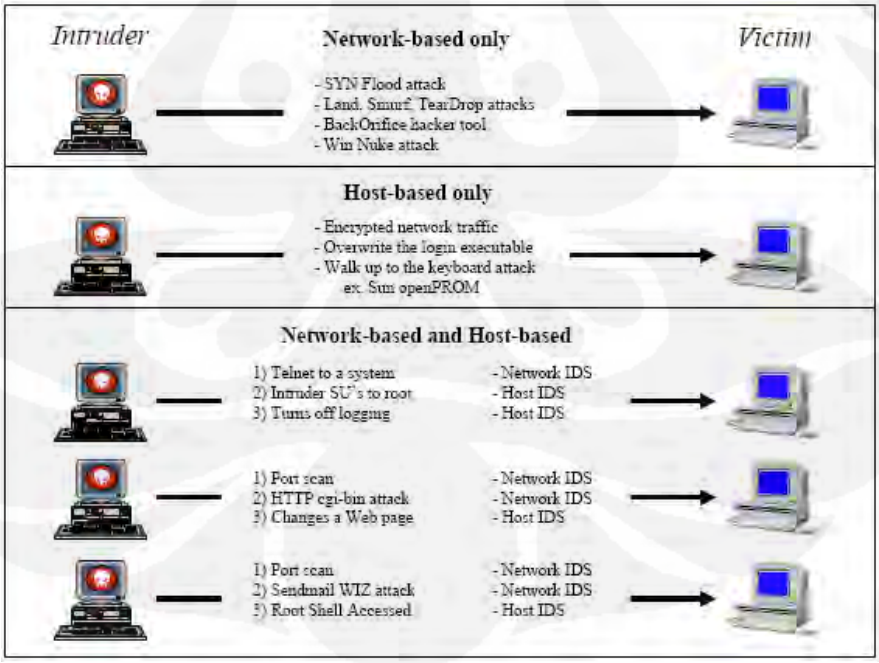
Tabel 2.3 Perbandingan NIDS dan HIDS

***Network-Based* IDS**	**Client-Based IDS**
Ruang lingkup yang luas (mengamati semua aktivitas jaringan)	Ruang lingkup yang terbatas (mengamati hanya aktivitas pada *client* tertentu)
Lebih mudah melakukan setup	Setup yang kompleks
Lebih baik untuk mendeteksi serangan yang berasal dari luar jaringan	Lebih baik mendeteksi serangan yang berasal dari jaringan
Lebih murah untuk diimplementasikan	Lebih mahal untuk diimplementasikan
Pendeteksi berdasarkan pada apa yang direkam dari aktivitas jaringan	Pendeteksian berdasarkan paa single *client* yang diamati semua aktivitasnya

Menguji Packet Headers	Packet Headers tidak diperhatikan
Respons yang real time	Selalu merespons setelah apa yang terjadi
OS-Independent	OS-Spesifik
Mendeteksi serangan terhadap jaringan serta payload untuk dianalisis	Mendeteksi serangan local sebelum mereka memasuki jaringan
Mendeteksi usah dari serangan yang gagal	Menverifikasi sukses atau gagalnya suatu serangan

Baik *network* dan *client-based IDS* masing-masing memiliki kekuatan dan keuntungan yang saling melengkapi. Generasi IDS masa depan harus memiliki komponen yang dimiliki *network* dan *client-based* IDS. Menggabungkan dua teknologi tersebut atau yang disebut dengan *Hibrid IDS* yang akan meningkatkan resistensi jaringan terhadap serangan dan meningkatkan keamanan policy.

Pada Gambar 2.2 dapat diilustrasikan bagaimana teknik yang digunakan network dan *client-based intrution detection* dalam berinteraksi dan menciptakan sebuah pertahanan yang tangguh di dalam jaringan. Beberapa event hanya dapat dideteksi oleh *network-based IDS,* tetapi ada event yang hanya dapat dideteksi oleh *client-based IDS.* Beberapa juga membutuhkan kedua tipe *intrusion detection* agar dapat berfungsi secara optimal.

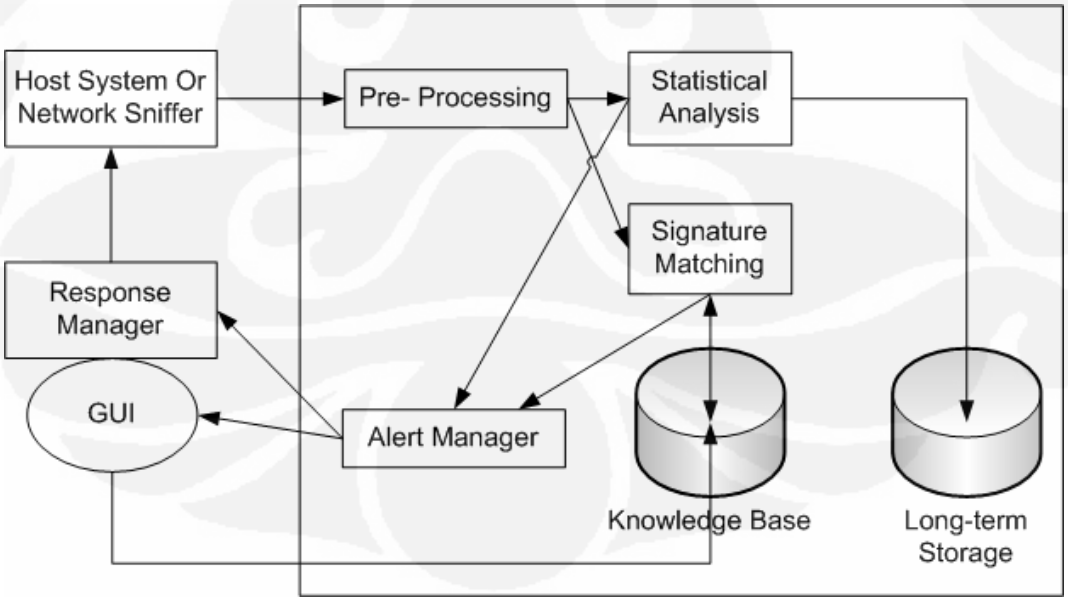


Gambar 2.3 Pola Pengenalan Serangan [2]

Beberapa fitur yang diharapkan pada generasi lanjutan IDS antara lain: [1]

1. Integrasi antara *network-based IDS* dan *client-based IDS*.
2. Manajemen konsol yang generik untuk semua produk.
3. Integrasi basisdata event.
4. Integrasi sistem report.
5. Kemampuan menghungkan event dengan serangan.
6. Integrasi dengan *on-line help* untuk respons insiden.
7. Prosedur instalasi yang ringakas dan terintegrasi dengan seluruh produk yang ada.

Proses dasar dari IDS, baik pada NIDS atau HIDS adalah mengumpulkan data, melakukan pre-process, dan mengklasifikasikan data tersebut. Dengan analisis statistik suatu aktifitas yang tidak normal akan bisa dilihat, sehingga IDS bisa mencocokkan dengan data dan pola yang sudah ada. Jika pola yang ada cocok dengan keadaan yang tidak normal maka akan dikirim respons tentang aktifitas tersebut. Hal ini dapat diilustrasikan pada Gambar 2.4.



Gambar 2.4 Sistem IDS Standart [1]

2.7.2 Intrusion-Prevention System (IPS)

Teknologi *Intrusion Detection System* (IDS) diperkirakan akan tertinggal dan akan digantikan dengan *Intrusion Prevention System* (IPS) yang memiliki kemampuan lebih lengkap. IDS hanya mampu mendeteksi adanya penyusupan dalam jaringan, lalu mengaktifkan peringatan kepada pengguna untuk segera mengambil langkah-langkah mitigasi sedangkan IPS dapat langsung mengatasi penyusupan tersebut.

Awalanya IDS adalah pengembangan dari firewall, yakni sistem yang memisahkan antara jaringan internal dan eksternal. Firewall dinilai tidak cukup karena hanya sebagai pemisah saja dan tidak memeriksa paket-paket data yang berbahaya sehingga bisa lolos. “Pada perkembanganya kemudian IDS tidak berguna karena pada saat alarm berbunyi, jaringan sudah terinfeksi dan pengguna tidak bisa berbuat banyak”. Kata Ken Low, praktisi keamanan berkualifikasi Certified Information Sistem Keamanan Professional (CISSP). [1]

IDS berkembang karena pada awal kelahiranya, pasar saat itu memandang skeptis terhadap keberhasilan teknologi IPS yang menggunakan filter dalam menangkal serangan dan penyusupan. Pada 2002, lembaga riset Gartner merilis laporan yang isinya mengingatkan para pengguna untuk menunda investasi IDS yang dinilai gagal. IDS bahkan dinilai sebagai investasi mahal namun tidak efektif untuk meningkatkan keamanan jaringan.

Saat itu aktif IDS, yang merupakan teknologi asal mula dari IPS yang mampu secara aktif mendeteksi serangan dan mengubah aturan firewall dan router untuk mengantisipasi serang, dinilai mengganggu, serhingga tidak diterima oleh administrator jaringan. Pada perkembangannya frekuensi serangan terhadap jaringan meninggkat sementara rentang waktu antara ditemukannya celah keamanan dan tersedia patch untuk menutup celah itu semakin sempit. Akibatnya para administrator jaringan tidak memiliki cukup waktu untuk mengantisipasi serangan dengan memasang patch. Kondisi dimana serangan muncul sebelum tersedia patch desebut juga *zero day attack* semakin dekat. Kemudian perkembangan teknologi memungkinkan IPS bekerja lebih cepat dalam mengatisipasi pola-pola serangan.

Salah satu peranti IPS bahkan memiliki kecepatan maksimal hingga 5 Gigabit per detik (Gbps). Menurut Wikipedia, IPS pertama kali diperkenalkan One Secure yang kemudian dibeli NetScreen Technologies sebelum akhirnya diakusisi Juniper Network pada 2004. Salah satu produsen *IPS- Tip-ping Point* Juga dibeli penyedia peranti jaringan 3Com Corp. IPS memutuskan memberikan akses kepada jaringan berdasarkan isi paket data, bukan berdasarkan alamat IP (*Internet Protocol*) atau port seperti *firewall*.[1]

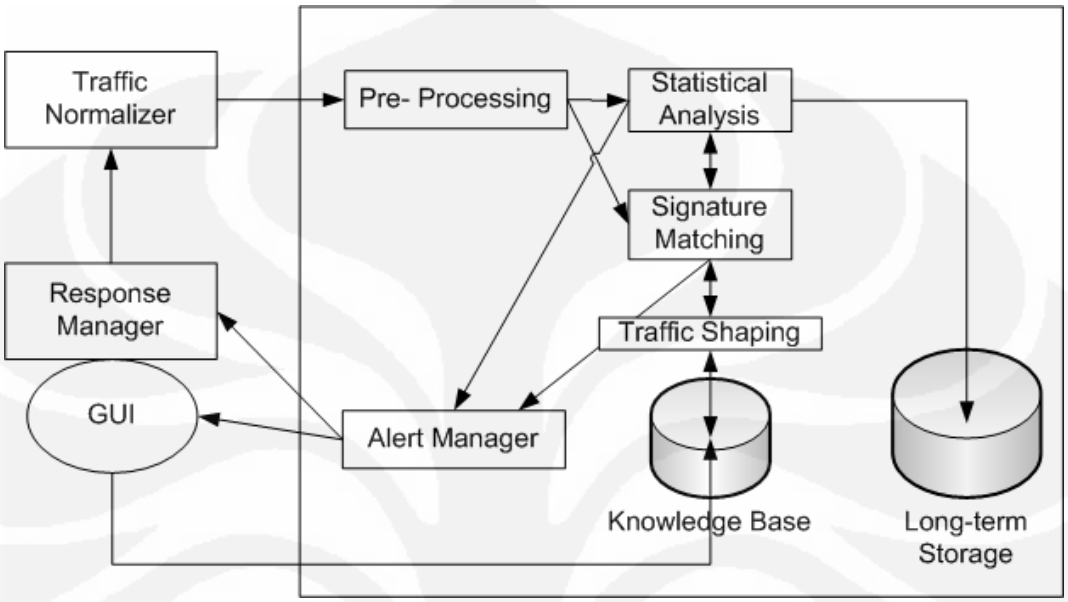
Sistem setup pada IPS sama dengan sistem setup pada IDS. IPS bias sebagai *client*-based IPS (HIPS) yang bekerja untuk melindungi aplikasi dan juga sebagai network based IPS (NIPS). Mengapa IPS lebih unggul dari IDS? Karena IPS mampu mencegah serangan yang datang dengan bantuan administrator secara minimal atau bahkan tidak sama sekali. Tidak seperti IDS, secara logic IPS akan menghalangi suatu serangan sebelum terjadi eksekusi pada memori, metode lain dari IPS membandingkan *file checksum* yang tidak semestinya *dengan file chechksum* yang semestinya mendapatkan izin untuk dieksekusi.

Secara khusus IPS memiliki empat komponen utama:

- Normalisasi traffic
- Services scanner
- Detection engine
- Traffic shaper

Normalisasi *traffic* akan menginterpretasikan lalu-lintas jaringan dan melakukan analisis terhadap paket yang disusun kembali, seperti halnya fungsi blok sederhana. Lalu-lintas paket bias dideteksi dengan *detection engine* dan *service scanner*. *Service scanner* membangun suatu tabel acuan untuk mengelompokkan informasi dan membantu pembentukan lalu-lintas serta mengatur lalu-lintas informasi. *Detection engine* melakukan *pattern matching* terhadap tabel acuan dan *respons* yang sesuai. Gambar 2.4 mengilustrasikan proses tersebut secara garis besarnya.

Teknologi IDS dan IPS masing-masing mempunyai kemampuan dalam melindungi suatu sistem. Teknologi IPS merupakan teknologi yang diperbarui dari IDS. Perbedaan dari kedua program itu adalah seperti pada Tabel 2.5



Gambar 2.5 Sistem IPS Stadart [1]

Tabel 2.4 Perbandingan IDS dan IPS

IDS	IPS
Install pada segmen jaringan (NIDS) dan pada *client* (HIDS)	Install pada segmen jaringan (NIDS) dan pada *client* (HIDS)
Berada pada jaringan sebagai sistem yang pasif	Berada pada jaringan sebagai sistem yang aktif
Tidak bias menguraikan lalu-lintas enkripsi	Lebih baik untuk melindungi aplikasi
Managemen control terpusat	Managemen control terpusat
Baik untuk mendeteksi serangan	Ideal untuk memblokir serangan
Alerting (reaktif)	Blocking (proaktif)

2.8 SNORT

Snort IDS merupakan IDS open source yang secara *de facto* menjadi standar IDS di industri. Snort dapat didownload di situs www.snort.org. Snort dapat diimplementasikan dalam jaringan yang multiplatform, salah satu kelebihannya adalah mampu mengirimkan alert dari mesin Unix atupun Linux ke platform Microsoft Windows dengan melalui SMB. Snort dapat berkerja dalam 3 mode:

- *Sniffer mode (penyadap)*: untuk melihat paket yang lewat di jaringan.
- *Packet logger*: untuk mencatat semua paket yang lewat di jaringan untuk dianalisi di kemudian hari
- *Network Intrusion Detection* (NIDS) mode: pada mode ini snort akan berfungsi untuk mendeteksi serangan yang dilakukan melalui jaringan komputer. Untuk menggunakan mode IDS ini diperlukan setup dari berbagai rules atau aturan yang akan membedakan sebuah paket normal dengan paket yang membawa serangan.

Pada dasarnya cara kerja snort hampir sama dengan alarm, yaitu memberi tahu apabila terjadi penyusup yang akan masuk ke jaringan.

2.8.1 Komponen – Komponen Snort

Snort mempunyai lima komponen dasar yang bekerja saling berhubungan satu dengan yang lain seperti berikut:[1]

1. **Decoder**: sesuai dengan paket yang di-capture dalam bentuk struktur data dan melakukan identifikasi protol, decode IP dan kemudian TCP atau UDP tergantung informasi yang dibutuhkan, seperti port number, IP address. Snort akan memberikan alert jika menemukan paket yang cacat.
2. **Preprocessors**: Merupakan suatu saringan yang menidentifikasi berbagai hal yang harus diperiksa seperti Detection Engine. Pada dasarnya preprocessors berfungsi mengambil paket yang mempunyai potensi berbahaya yang kemudian dikirim ke detection engine untuk dikenali polanya.

   Example: HTTPInspect

   HTTPInspect menggantikan http_decode sebagai preprocessor yang bertanggung jawab untuk mendecodekan lalu-lintas http dan mendeteksi lapisan aplikasi serangan eksploit http design atau implementasi. Hal ini akan terlihat dalam buffer data paket yang berusaha mencari celah-celah di dalam lalu lintas http dan berusaha melakukan normalisasi data.
3. **Rules Files:** Merupakan suatu file teks yang berisi daftar aturan sintaks-nya sudah diketahui. Sintaks ini meliputi protokol, address, output plug-ins dan hal-hal yang berhubungan dengan berbagai hal. Rules file akan selalu diperbaharui setiap ada kejadian di dunia maya. Rule snort lebih dari 100 ribu

tipe. Setiap hari bisa diupdate melalui situs resmi snort ww.snort.org atau dari forum yang disediakan oleh komunitas snort.

Sebagai contoh rule pada Snort sebagai berikut:

```
alert tcp $EXTERNAL NET
alert tcp $EXTERNAL NET any -> $HTTP SERVERS $HTTP PORTS
(msg:"WEB-IIS unicode directory traversal attempt";
flow:to server, established; content:¨/..%c0%af../";
nocase; classtype:web-application-attack; reference:cve,
CVE-2000-0884; sid:981; rev:6;)
```

Rule di atas terdiri dari 2 bagian: header dan option. Bagian ”alert tcp $EXTERNAL NET any -> $HTTP SERVERS $HTTP PORTS” adalah header dan selebihnya merupakan option. Dari *rule* seperti di ataslah IDS Snort menentukan apakah sebuah paket data dianggap sebagai penyusupan atau serangan atau bukan, paket data dibandingkan dengan *rule IDS*, jika terdapat dalam *rule*, maka paket data tersebut dianggap sebagai penyusupan atau serangan dan demikian juga sebaliknya jika tidak ada dalam rule maka dianggap bukan penyusupan atau serangan.

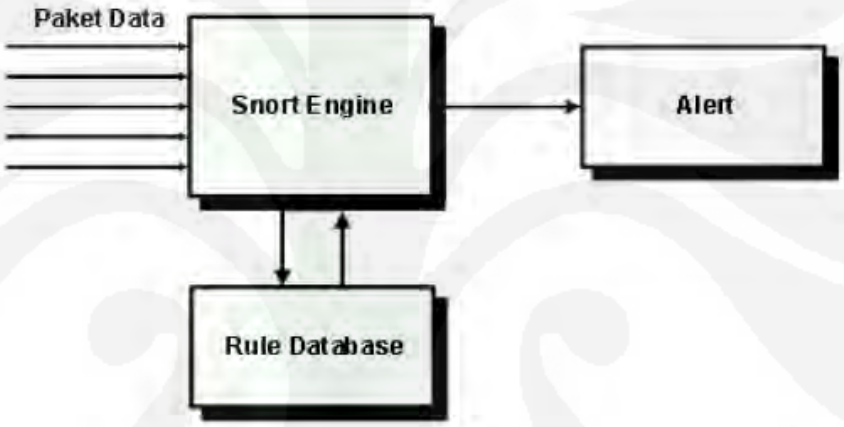
4. **Detection Engine:** Menggunakan *detection plug-ins*, jika ditemukan paket yang cocok maka snort akan menginisialisasi paket tersebut sebagai suatu serangan.
5. **Output Plug-ins**: Merupakan suatu modul yang mengatur format dari keluaran untuk alert dan file logs yang biasa diakses dengan berbagai cara serperti console, extern file, database dan sebagainya.
6. **Alert:** merupakan catatan serangan pada deteksi penyusupan. Jika snort engine menghukumi paket data yang lewat sebagai serangan, maka snort engine akan mengirimkan alert berupa log file. Untuk kebutuhan analisa, alert dapat disimpan di dalam database, sebagai contoh ACID (Analysis Console for Intrusion Databases) sebagai modul tambahan pada Snort.

   Contoh alert sebagai berikut:

```
[**] [1:499:3] ICMP Large ICMP Packet [**] [Classification:
Potentially Bad Traffic] [Priority: 2] 05/09-20:15:14.
895348 10.1.4.113 -> 10.1.3.126 ICMP TTL:128 TOS:0x0
ID:6316 IpLen:20 DgmLen:65528 Type:8 Code:0 ID:512
Seq:3072 ECHO [Xref => http://www.whitehats.com/info/IDS246]
```

Contoh alert di atas merupakan alert ketika terdapat paket data dalam ukuran besar dari IP Address 10.1.4.113 ke 10.1.3.126 yang dianggap sebagai serangan oleh Snort karena pola paket data tersebut terdapat dalam rule Snort.

Hubungan komponen Snort dijelaskan dalam Gambar 2.5



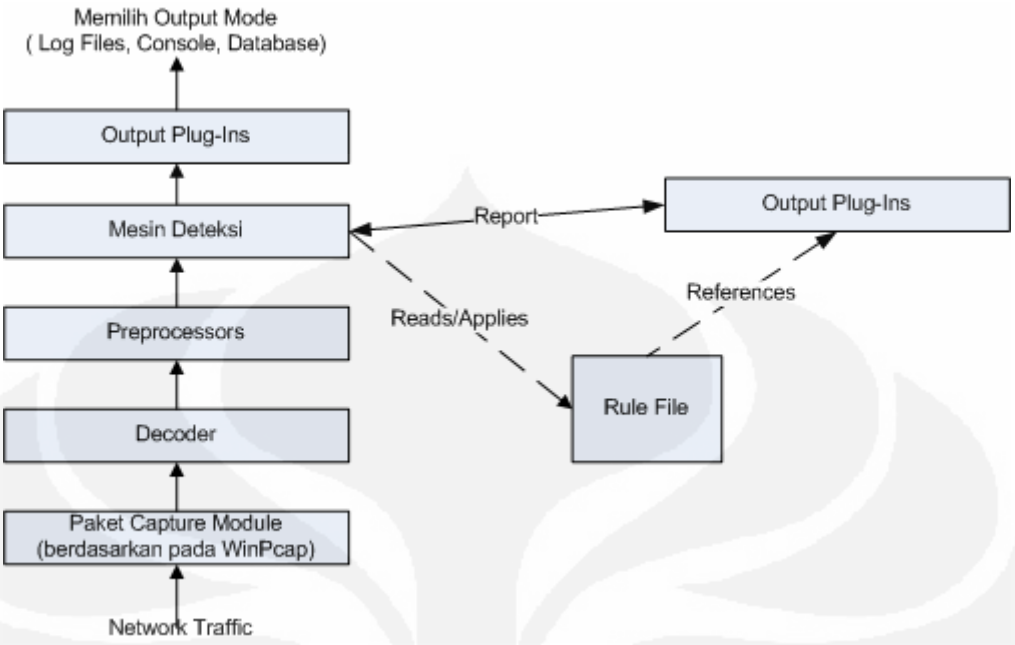
Gambar 2.6 Komponen – komponen Snort

2.8.2 Snort.conf file

File snort.conf merupakan suatu otak dari snort itu sendiri. Dari sini kita bisa melakukan konfigurasi apa yang diingikan. File snort.conf meliputi network dan *configuration variables*, *snort decoder* dan *detection engine configuration, preprocessor configuration, output configuration* dan *file inclusion.*

2.8.3 Proses Deteksi pada Snort

Proses deteksi snort sebagai intrusion detection sistem secara menyeluruh digambarkan sebagai berikut:[1]



Gambar 2.7 Proses Deteksi Snort [1]

Pada snort sebagai *intrusion detection system*, *rule* merupakan suatu hal yang sangat penting. Dengan adanya suatu r*ule* maka IDS bisa berfungsi untuk mendeteksi suatu kejadian. Sifat dari rule snort adalah seperti berikut:

- Dynamic
- Activation
- Alert
- Pass
- Log

2.8.4 Klasifikasi Serangan

Berikut adalah tabel klasifikasi serangan yang nantinya menjadi priority di dalam snort. Priority 1 = high, priority 2 = medium dan priority 3 = low. Klasifikasi serangan dapat dirubah sesuai keinginan administrator. Untuk merubah klasifikasi serangan dengan merubah isi pada *file classification.conf* yang berada pada */etc/snort/classification.conf*

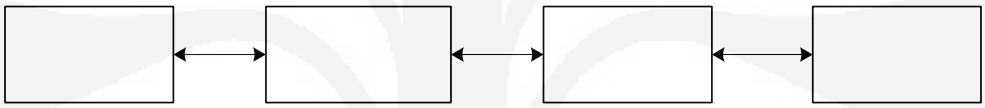
Tabel 2.5 Klasifikasi Serangan Berdasarkan Tingkat Prioritas

Classtype	Description	Priority
attempted-admin	Attempted Administrator Privilege Gain	high
attempted-user	Attempted User Privilege Gain	high
kickass-porn	SCORE! Get the lotion!	high
policy-violation	Potential Corporate Privacy Violation	high
shellcode-detect	Executable code was detected	high
successful-admin	Successful Administrator Privilege Gain	high
successful-user	Successful User Privilege Gain	high
trojan-activity	A Network Trojan was detected	high
unsuccessful-user	Unsuccessful User Privilege Gain	high
web-application-attack	Web Application Attack	high
attempted-dos	Attempted Denial of Service	medium
attempted-recon	Attempted Information Leak	medium
bad-unknown	Potentially Bad Traffic	medium
default-login-attempt	Attempt to login by a default username and password	medium
denial-of-service	Detection of a Denial of Service Attack	medium
misc-attack	Misc Attack	medium
non-standard-protocol	Detection of a non-standard protocol or event	medium
rpc-portmap-decode	Decode of an RPC Query	medium
successful-dos	Denial of Service	medium
successful-recon-largescale	Large Scale Information Leak	medium
successful-recon-limited	Information Leak	medium
suspicious-filename-detect	A suspicious filename was detected	medium
suspicious-login	An attempted login using a suspicious username was detected	medium
system-call-detect	A system call was detected	medium
unusual-client-port-connection	A client was using an unusual port	medium
web-application-activity	Access to a potentially vulnerable web application	medium
icmp-event	Generic ICMP event	low
misc-activity	Misc activity	low
network-scan	Detection of a Network Scan	low
not-suspicious	Not Suspicious Traffic	low
protocol-command-decode	Generic Protocol Command Decode	low
string-detect	A suspicious string was detected	low
unknown	Unknown Traffic	low
tcp-connection	A TCP connection was detected	very low

BAB 3
PERANCANGAN IDS *SERVER*

3.1 PERANCANGAN SISTEM

Pada skripsi ini, konsep jaringan NAC dibagi menjadi 2 bagian yaitu *policy server* dan IDS *server*. IDS *server* pada NAC berfungsi sebagai pendeteksi adanya serangan terhadap *server*, serangan bisa berupa *denial of service (DOS), ping of death (POD), port scanning* dan lain-lain. IDS *server* akan menginformasikan kepada *policy server* untuk melakukan tindakan untuk mengamankan *server* dan memberikan laporan kepada administrator melalui sms.



Gambar 3.1 Disain Jaringan NAC

Pada jaringan ini network device yang digunakan adalah switch CISCO 2950. *Server* jaringan menggunakan linux *server*, media penyimpanan data pada *server* menggunakan database MySQL dan *SMS gateway* menggunakan modem itegno 3000. Prinsip kerja jaringan diatas adalah sebagai berikut:

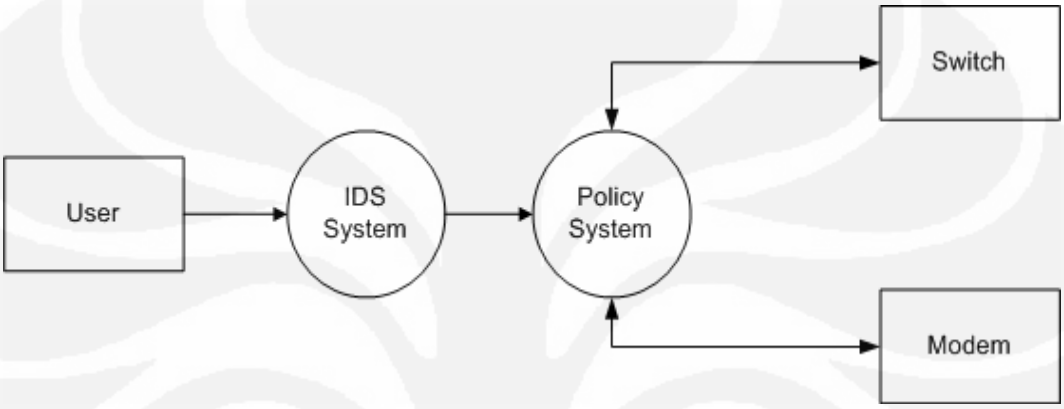
a) User baru melakukan akses ke jaringan, secara default user mendapatkan IP secara automatis.
b) NAC *server* mengirimkan autentifikasi menggunakan web browser.
c) User mengirimkan username dan password.
d) NAC *Server* melakukan cek ke database dan mengirimkan respon ke end device
e) NAC *Server* melakukan setting VLAN pada switch sesuai dengan username.
f) User dapat masuk ke jaringan dan memperoleh previledge sesuai dengan username yang digunakan.
g) Bila terdeteksi suatu serangan terhadap *server*, NAC akan menutup port VLAN yang terdeteksi melakukan serangan dan mengirimkan pesan kepada administrator jaringan menggunakan sms.

SMS Gateway

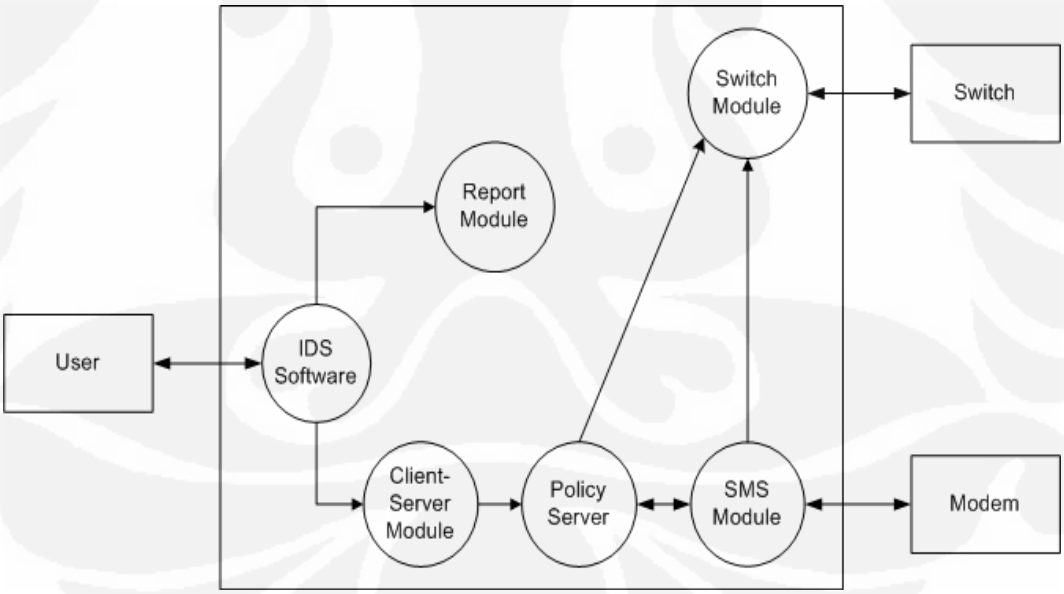
NAC Server

h) Administrator jaringan dapat mengubah setting switch dengan cara mengirimkan perintah tertentu melalui SMS ke nomor yang digunakan oleh *sms gateway*.

Disain sistem dari IDS *server* yang digunakan pada skripsi ini adalah sebagai berikut:



Gambar 3.2 Data Flow Diagram Level 0



Gambar 3.3 Data Flow Diagram IDS *Server* Level 1

3.2 Instalasi dan Konfigurasi IDS Software

Snort IDS merupakan IDS *open source* yang secara *de facto* menjadi standar IDS di industri. Snort dapat didownload di situs www.snort.org. Snort dapat di implementasikan dalam jaringan yang *multiplatform,* salah satu kelebihannya adalah mampu mengirimkan alert dari mesin UNIX atupun Linux. Berikut adalah cara instalasi SNORT: [8]

- Download SNORT
  http://dl.snort.org/snort-current/snort-2.8.1.tar.gz
  http://dl.snort.org/snort-current/snort-2.8.1-1.FC7.i386.rpm
  http://dl.snort.org/snort-current/snort-mysql-2.8.1-1.FC7.i386.rpm

- Download ADODB
  http://sourceforge.net/project/showfiles.php?group_id=42718

- Download BASE
  http://sourceforge.net/project/showfiles.php?group_id=103348

Salah satu metode yang dapat digunakan untuk melakukan instalasi *software* dapat menggunakan YUM yaitu suatu *tools software management* yang dimiliki fedora. Untuk menggunakan YUM pastikan terlebih dahulu kita menjadi *super user*. Kemudian jalankan perintah berikut:

```
Root> yum install snort snort-mysql php php-mysql mysql-server
msyql pear phpmyadmin
```

- Menjalankan snort

```
Root> /usr/sbin/snort –c /etc/snort/snort.conf
……
          --== Initialization Complete ==--

      ,,_     -*> Snort! <*-
     o"  )~   Version 2.8.1 (Build 28)
      ''''            By   Martin   Roesch   &   The   Snort   Team:
http://www.snort.org/team.html
              (C) Copyright 1998-2008 Sourcefire Inc., et al.
              Using PCRE version: 7.8 2008-09-05

              Rules  Engine:  SF_SNORT_DETECTION_ENGINE    Version  1.8
<Build 13>
              Preprocessor Object: SF_FTPTELNET  Version 1.1  <Build 10>
              Preprocessor Object: SF_SSLPP  Version 1.0  <Build 1>
              Preprocessor Object: SF_DCERPC  Version 1.1  <Build 4>
              Preprocessor Object: SF_SMTP  Version 1.1  <Build 7>
              Preprocessor Object: SF_DNS  Version 1.1  <Build 2>
              Preprocessor Object: SF_SSH  Version 1.1  <Build 1>
Not Using PCAP_FRAMES
```

Apabila telah muncul pesan di atas maka proses inisialisasi snort telah berjalan dengan baik. Dari perintah di atas snort berjalan dalam mode *intrusion detection system* sehingga apabila terjadi serangan, snort akan memberikan peringatan atau *alerting*. File alert disimpan di dalam folder /var/log/snort/alert. Berikut adalah salah satu contoh alerting:

```
05/19-22:41:34.719623    [**]  [1:480:6]  ICMP  PING  speedera  [**]
[Classification:  Misc  activity]  [Priority:  3]  {ICMP}  192.168.0.2  ->
192.168.0.1
  [**] [1:480:6] ICMP PING speedera [**]
  [Classification: Misc activity] [Priority: 3]
  05/19-22:41:34.719623 192.168.0.2 -> 192.168.0.1
  ICMP TTL:64 TOS:0x0 ID:40838 IpLen:20 DgmLen:64028
  Type:8  Code:0  ID:19214   Seq:13  ECHO
```

3.3 Instalasi Modul Report

Modul report yang banyak digunakan dan telah terintegrasi dengan baik dengan snort adalah *Basic Analysis Security Engine (BASE)* digunakan untuk mengelola data-data *security event,* keuntungan menggunakan BASE diantaranya:

- Log-log yang tadinya susah dibaca menjadi mudah di baca.
- Data-data dapat dicari dan difilter sesuai dengan kriteria tertentu.
- Managing Large Alert Databases (Deleting and Archiving).
- Untuk kasus-kasus tertentu dapat merujuk alert pada situs database security seperti *Securityfocus, CVE, arachNIDS*.

Untuk installasi BASE diperlukan beberapa *software* pendukung yaitu MySQL sehinga terlebih dahulu dilakukan konfigurasi MySQL.

3.3.1 Konfigurasi MySQL

- Start MySql

```
Root> /etc/init.d/mysql start
```

- Set root password for MySql

```
Root> /usr/bin/mysqladmin -u root password 'passwordhere'
```

- Login to MySql

```
Root> mysql -u root –p
```

- Membuat database, user dan security.

```
Mysql>create database snort; grant INSERT,SELECT on root.* to
snort@localhost;          SET          PASSWORD          FOR
snort@localhost=PASSWORD('passwordhere');                grant
CREATE,INSERT,SELECT,DELETE,UPDATE      on      snort.*      to
snort@localhost;  grant  CREATE,INSERT,SELECT,DELETE,UPDATE  on
snort.* to snort; exit
```

- Membuat struktur database snort menggunakan perintah berikut:

```
Root>  zcat  /usr/share/doc/snort-2.4.3/schemas/create_mysql.gz
| mysql -p snort
```

- Memeriksa struktur database snort:

```
Root>mysql -u root -p snort
Mysql> use snort;

Mysql> show tables;
+------------------+
| Tables_in_snort |
+------------------+
| data             |
| detail           |
| encoding         |
| event            |
| icmphdr          |
| iphdr            |
| opt              |
| reference        |
| reference_system |
| schema           |
| sensor           |
| sig_class        |
| sig_reference    |
| signature        |
| tcphdr           |
| udphdr           |
+------------------+
16 rows in set (0.00 sec)
```

3.3.2 Konfigurasi snort.conf

Agar snort dapat menyimpan report ke MySql maka perlu merubah konfigurasi pada snort.conf yaitu pada baris 382 rubah menjadi uncomment dengan menghilangkan tanda # pada baris tersebut. Kemudian sesuaikan untuk username, password, database dan *client* yang digunakan pada MySql.

```
output database: log, mysql, user=snort password=password
dbname=snort client=localclient
```

Jalankan snort agar menyimpan log dan alert kedalam database MySql dengan perintah:[3]

```
Root>/usr/sbin/snort – c /etc/snort/snort.conf

……
Initializing Network Interface eth0
OpenPcap() device eth0 network lookup:
        eth0: no IPv4 address assigned
Decoding Ethernet on interface eth0
database: compiled support for ( mysql )
database: configured to use mysql
database:          user = snort
database: password is set
database: database name = snort
database:          host = localhost
database:   sensor name = localhost.localdomain:eth0
database:     sensor id = 6
database: schema version = 107
database: using the "alert" facility

[ Port Based Pattern Matching Memory ]
+-[AC-BNFA Search Info Summary]------------------------------
| Instances        : 251
| Patterns         : 45145
| Pattern Chars    : 402529
| Num States       : 108025
| Num Match States : 14954
| Memory           :   3.54Mbytes
|   Patterns       :   1.24M
|   Match Lists    :   0.82M
|   Transitions    :   1.45M
+-------------------------------------------------

        --== Initialization Complete ==--
```

Apabila telah muncul pesan di atas maka snort telah dapat menyimpan report di dalam database mysql dengan baik.

3.3.3 Installasi dan Konfigurasi BASE

- Install ADODB

```
Root>cp adodb465.tgz /var/www/html
Root>cd /var/www/html
Root>tar –xvzf adodb465.tgz
Root>mv adodb465 adodb
```

- Install pear:

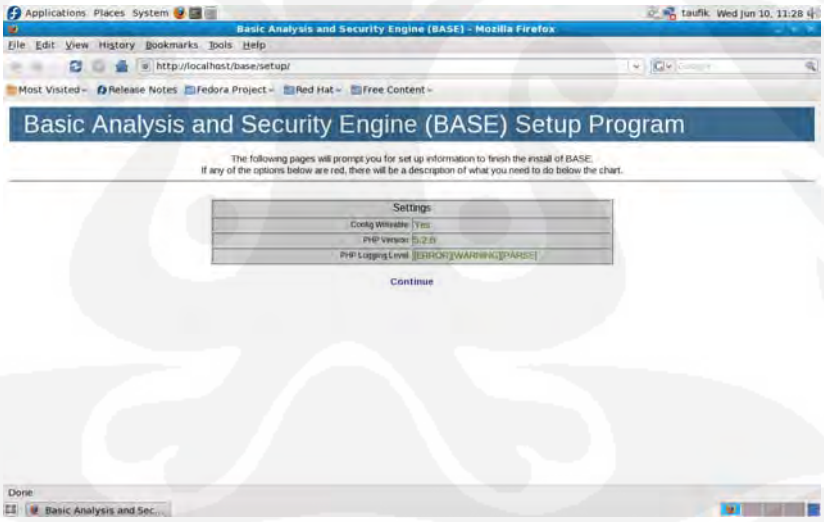
```
Root>pear install Image_Color
Root>pear install Log
Root>pear install Numbers_Roman
Root>pear install http://pear.php.net/get/Numbers_Words-0.13.1.tgz
```

- Install BASE:

```
Root> cp base-1.2.1.tgz /var/www/html/
Root> cd /var/www/html
Root>tar –xvzf base-1.2.1
Root>mv base-1.2.1 base
```

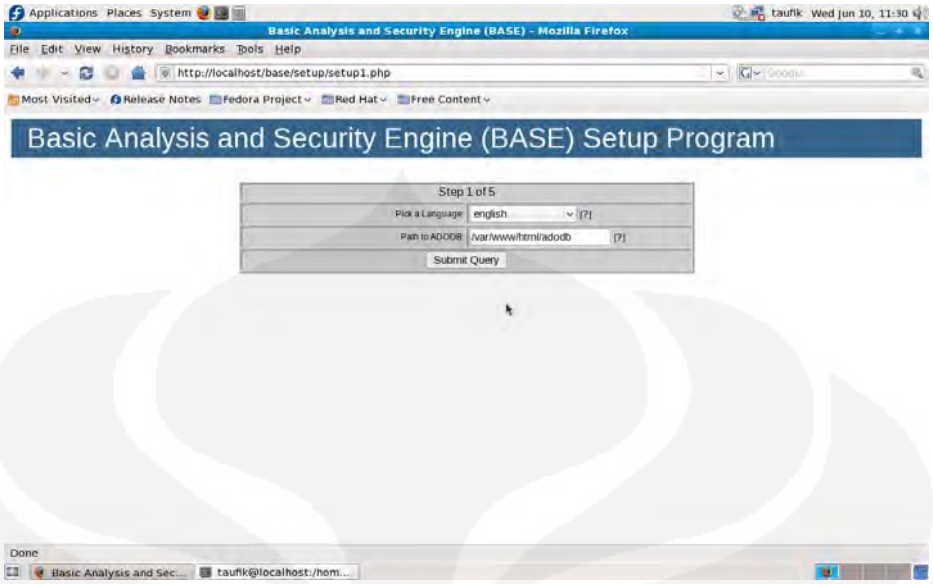
- Konfigurasi BASE:

  Buka browser dan ketik alamat http://local*client*/base/setup/ maka akan keluar tampilan sebagai berikut:



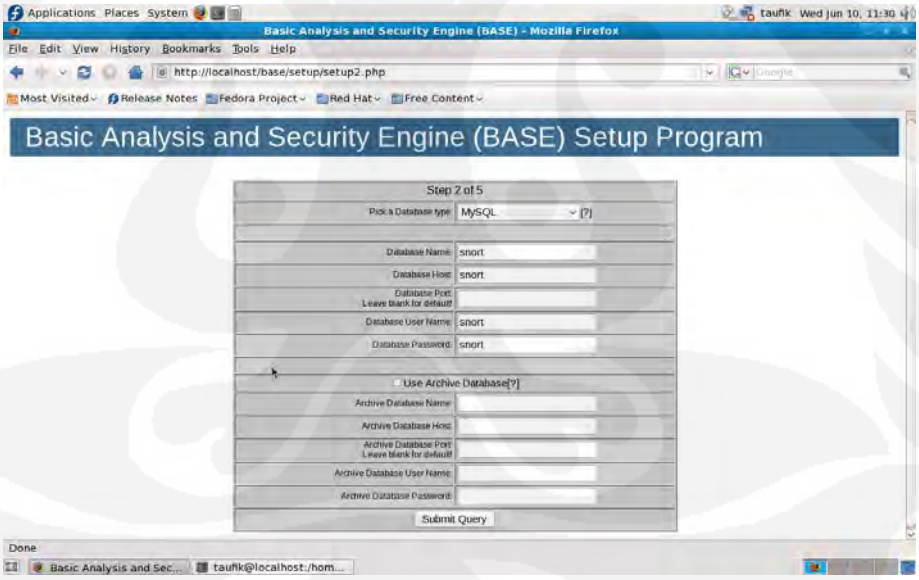
Gambar 3.4 Setup BASE

Pastikan tidak ada error pada settingan di atas. Kemudian klik “continue” untuk mengisi path letak dari adodb seperti pada Gambar 3.5.



Gambar 3.5 Letak Path ADODB

Kemudian masukkan konfigurasi MySQL. Seperti nama *database, username, password*, dan *client*

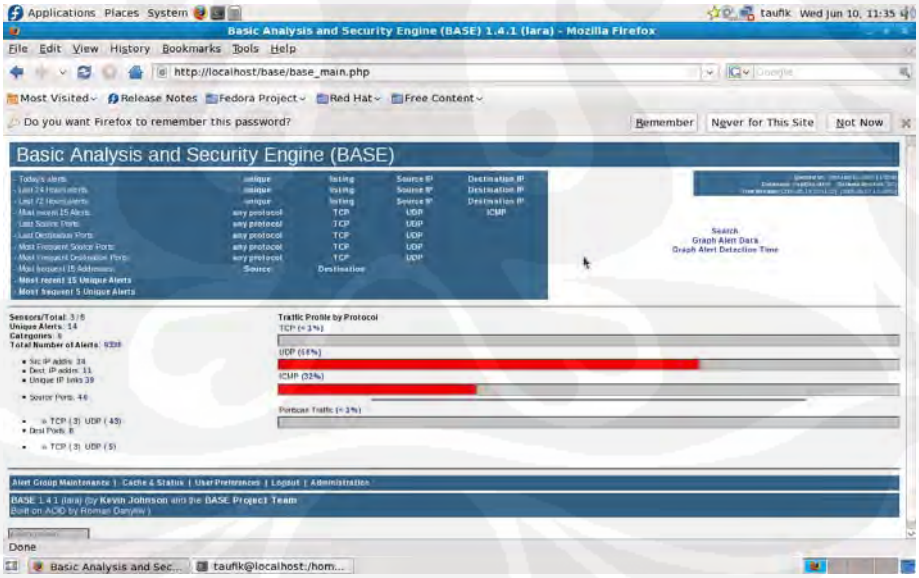


Gambar 3.6 Konfigurasi MySQL

Klik “Create BASE AG”

Gambar 3.7 Penambahan Tabel

Dari proses tersebut BASE menambahkan tabel acid_ag, acid_ag_alert, acid_ip_cache, acid_event, base_roles, base_user ke dalam database snort di MySql. Konfigurasi BASE selesai kemudian klik “Main page” untuk masuk ke menu utama. Pada Gambar 3.8 dapat ditunjukkan tampilan awal dari BASE.



Gambar 3.8 Tampilan Utama BASE

3.4 *Client – Server Module*

Modul ini bertujuan untuk menghubungkan antara IDS *server* dengan *policy server*. Modul dibangun menggunakan bahasa pemrograman perl dengan

menggunakan prisip *client – server* dimana yang bertindak sebagai *client* adalah IDS *server* dan yang bertindak sebagai *server* adalah *policy server*. Program *client* yang terletak di IDS *server* berfungsi untuk membaca logfile berupa *alerting* yang dihasilkan oleh snort. Data hasil pembacaan kemudian dikirimkan ke *policy server* melalui jaringan yang menggunakan prinsip *socket programming*. Pada sisi *policy server* akan mambaca kiriman data dari IDS *server* kemudian melakukan klasifikasi *alerting* berdasarkan priority sebuah serangan. Dari klasifikasi tersebut *policy server* akan melakukan perubahan konfigurasi pada switch sesuai dengan prioritasnya hal ini dapat dilihat pada tabel dibawah ini. *Policy server* juga akan meneruskan pesan yang dikirimkan IDS *server* ke admin melalui sms.

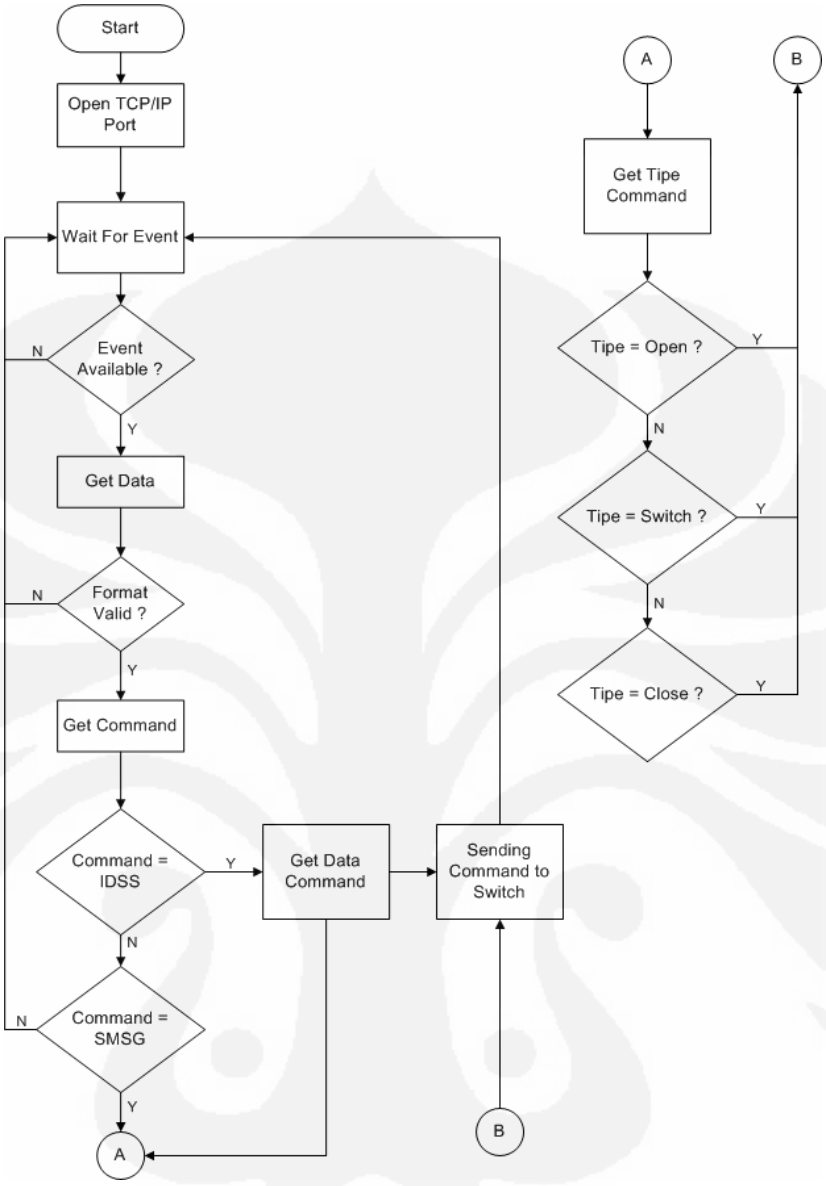
Table 3.1 Prioritas Serangan dan Action

Prioritas Serangan	Action
1	Shutdown Port
2	Change VLAN-ID
3	No Action

3.5 SMS Module

Modul ini bertujuan untuk melakukan reporting terhadap administrator jaringan bila terjadi gangguan pada jaringan. Selain melakukan *reporting*, modul ini juga akan melakukan eksekusi command untuk melakukan sesuai dengan command yang diberikan oleh admin. Salah satu contoh ini sms yang dikirimkan untuk pemberitahuan telah terjadi serangan kepada *server* adalah sebagai berikut. Pada pesan sms di bawah menunjukkan telah terjadi serangan terhadap IDS *server* dengan priotritas 1 dengan IP dari sumber serangan 192.168.20.7

```
IDSS,Priority 1|source 192.168.20.7
```



Gambar 3.9 Flowchart Program SMS Modul

Pada proses *"Sending Message"* prosedur yang digunakan adalah sebagai berikut:

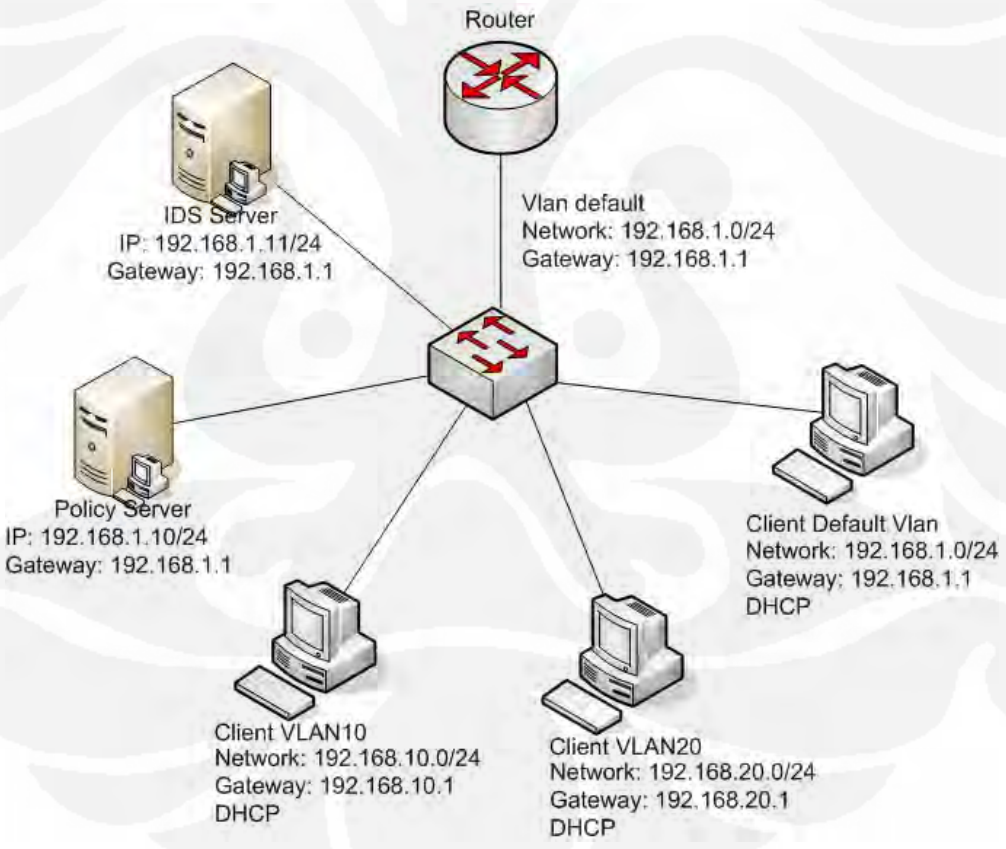
1. `Get Data`
2. `Create PDU from Data`
3. `Sending AT+Command to Modem`
4. `Sending PDU Data`
5. `Wait Response from Modem`

Sementara pada prosedur *"execute command"*, algoritma yang digunakan adalah sebagai berikut:

1. `Get Data From Command`
2. `Cek Tipe Command`
3. `If command Valid then`
   `Sending data to switch module`

3.6 DISAIN JARINGAN

Pada skripsi ini, jaringan yang akan digunakan adalah jaringan sederhana yang terhubung menggunakan sebuah switch CISCO 2950-24. Untuk mempermudah pengelompokan user jaringan, user jaringan dikelompokan dalam sebuah vlan sesuai dengan tipe dari *user*. Ada 3 macam vlan yang digunakan yaitu vlan default untuk user yang baru, vlan10 yang digunakan untuk user dengan tipe karyawan dan vlan20 untuk user dengan tipe staff. *Policy server* dan IDS *server* terletak pada *default vlan*. Router disini bertugas untuk memfilter packet akan menuju ke jaringan vlan menggunakan *access list*. Selain melakukan pemfilteran paket, router juga digunakan sebagai DHCP *server* jaringan vlan. Disain jaringan yang digunakan dapat dilihat pada Gambar 3.10 Perangkat jaringan yang digunakan pada skripsi ini adalah switch CISCO 2950-24 dan router.



Gambar 3.10 Disain Jaringan NAC *Server*

Konfigurasi switch pada jaringan diatas adalah sebagai berikut:

```
hostname Switch
enable password cisco
vlan 10
  name karyawan
vlan 20
  name staff
interface FastEthernet0/1
  switchport mode trunk
  no ip address
interface Vlan1
  ip address 192.168.1.2 255.255.255.0
  no ip route-cache
line con 0
line vty 0 4
  password cisco
  login
line vty 5
  password cisco
  login
line vty 6 15
  login
```

Konfigurasi router yang digunakan adalah sebagai berikut:

```
interface FastEthernet0/0
 no ip address
 duplex auto
 speed auto
interface FastEthernet0/0.1
 encapsulation dot1Q 1 native
 ip address 192.168.1.1 255.255.255.0
 ip access-group 101 in
interface FastEthernet0/0.10
 encapsulation dot1Q 10
 ip address 192.168.10.1 255.255.255.0
 ip access-group 110 in
interface FastEthernet0/0.20
 encapsulation dot1Q 20
 ip address 192.168.20.1 255.255.255.0
 ip access-group 120 in
ip dhcp pool vlan10
   network 192.168.10.0 255.255.255.0
   default-router 192.168.10.1
ip dhcp pool vlan20
   network 192.168.20.0 255.255.255.0
   default-router 192.168.20.1
ip dhcp pool vlan1
   network 192.168.1.0 255.255.255.0
   default-router 192.168.1.1
ip dhcp excluded-address 192.168.1.10
ip dhcp excluded-address 192.168.10.1
ip dhcp excluded-address 192.168.20.1
access-list 101 permit ip 192.168.1.10 0.0.0.1 192.168.10.0
0.0.0.255
access-list 101 permit ip 192.168.1.10 0.0.0.1 192.168.20.0
0.0.0.255
```

```
access-list 101 deny   ip 192.168.1.0 0.0.0.254 192.168.10.0
0.0.0.255
access-list 101 deny   ip 192.168.1.0 0.0.0.254 192.168.20.0
0.0.0.255
access-list 101 permit ip any any
access-list 110 permit ip 192.168.10.0 0.0.0.255 192.168.1.10
0.0.0.1
access-list 110 deny   ip 192.168.10.0 0.0.0.255 192.168.1.0
0.0.0.254
access-list 110 deny   ip 192.168.10.0 0.0.0.255 192.168.20.0
0.0.0.255
access-list 110 permit ip any any
access-list 120 permit ip 192.168.20.0 0.0.0.255 192.168.1.10
0.0.0.1
access-list 120 deny   ip 192.168.20.0 0.0.0.255 192.168.1.0
0.0.0.254
access-list 120 deny   ip 192.168.20.0 0.0.0.255 192.168.10.0
0.0.0.255
access-list 120 permit ip any any
line vty 0 5
 password cisco
 login
```

BAB 4
PENGUJIAN DAN ANALISA

4.1 UMUM

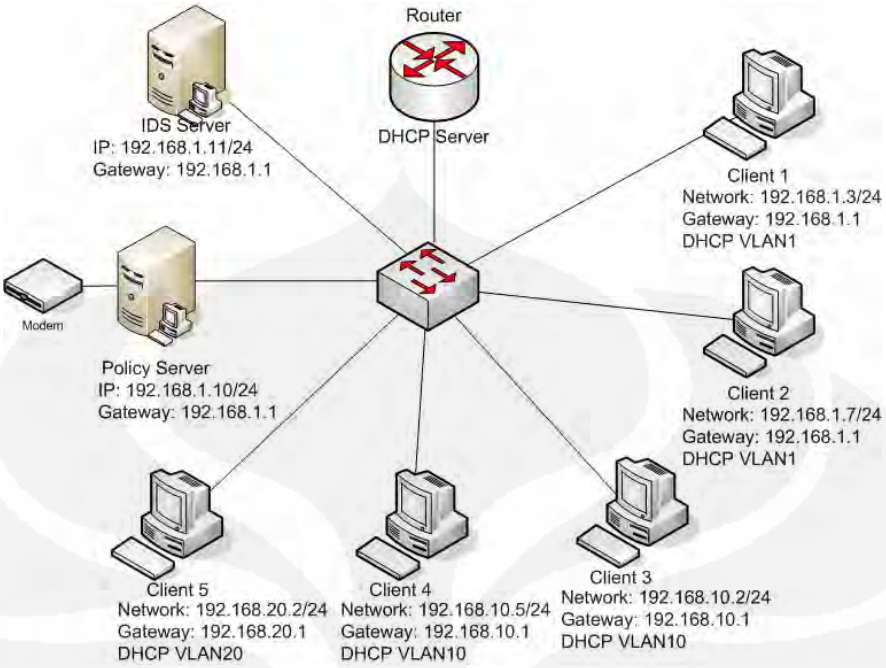
Pada bagian ini akan dilakukan pengujian sistem yang sudah dibuat berdasarkan perancangan pada bab sebelumnya. Pengujian sistem dilakukan dengan melakukan beberapa variasi serangan. Untuk mengetahui apakah IDS *server* dapat berfungsi dengan baik.

4.2 METODE PENGUJIAN

Pada skripsi untuk menguji IDS *server* apakah dia berfungsi dengan baik dan memiliki tingkat *reliability* atau kehandalan maka akan dilakukan dua metode pengujian yaitu:

1. *Functionality Test*
2. *Response time* dan *Action time*

Pada pengujian ini akan diguanakan 1 *client* dan 5 *client.* Untuk menghitung *response time* dan *action time* akan digunakan sotware Wireshark yang diletakkan di IDS *Server* yang fungsinya melakukan *sniffing packet* yang masuk ke dalam IDS *server*. Untuk *response time* akan dihitung mulai terjadi serangan sampai IDS memberikan respon dengan mengirimkan *alerting* ke *policy server*. Sedangkan untuk *action time* dihitung dari IDS *server* mengirimkan *alerting* sampai dengan *policy server* menghentikan serangan. Disain jaringan yang digunakan di dalam pengujian dapat dilihat di dalam gambar 4.1.



Gambar 4.1 Desain Jaringan untuk Pengujian IDS *Server*

Metode pengujian dilakukan dengan menggunakan 1 *client* dan 5 *client*. Setiap pengujian berupa serangan ke IDS *server* juga di monitoring oleh *software* Wireshark, yaitu *software packet sniffer* yang nantinya berfungsi untuk menganalisa *response time* dan *action time* dari IDS *server*.

4.2.1 Functionality Test

Functionality test bertujuan untuk menguji *server* apakah dapat berfungsi dengan baik sesuai dengan skenario yang diinginkan. Adapun skenario itu adalah ketika IDS *server* mendapat serangan dari jaringan, maka *server* akan memberikan *alerting* kemudian dikirim ke *client-server module* yang nantinya dikirim ke *policy server* untuk dilakukan klasifikasi berdasarkan prioritas serangan. Untuk tindakan yang akan dilakukan oleh *policy server* berdasarkan prioritas serangan dapat dilihat pada Tabel 4.1

Table 4.1 Prioritas Serangan dan Action

Prioritas Serangan	**Action**
1	Shutdown Port
2	Change VLAN-ID
3	No Action

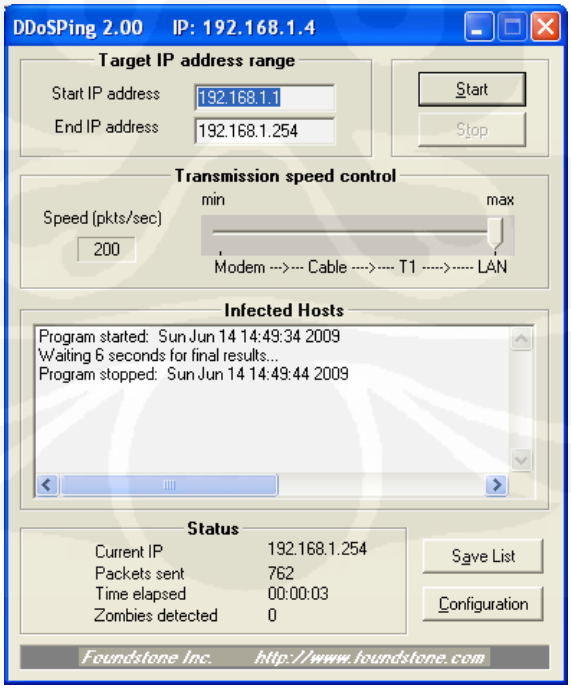
Pada *functionality test* akan dilakukan beberapa pengujian menggunakan tipe serangan yang berbeda. Tipe serangan yang digunakan dalam pengujian ini adalah:

1. DOS Attack
2. *Port scan*ning
3. ICMP *Flood*

4.2.1.1 DOS Attack

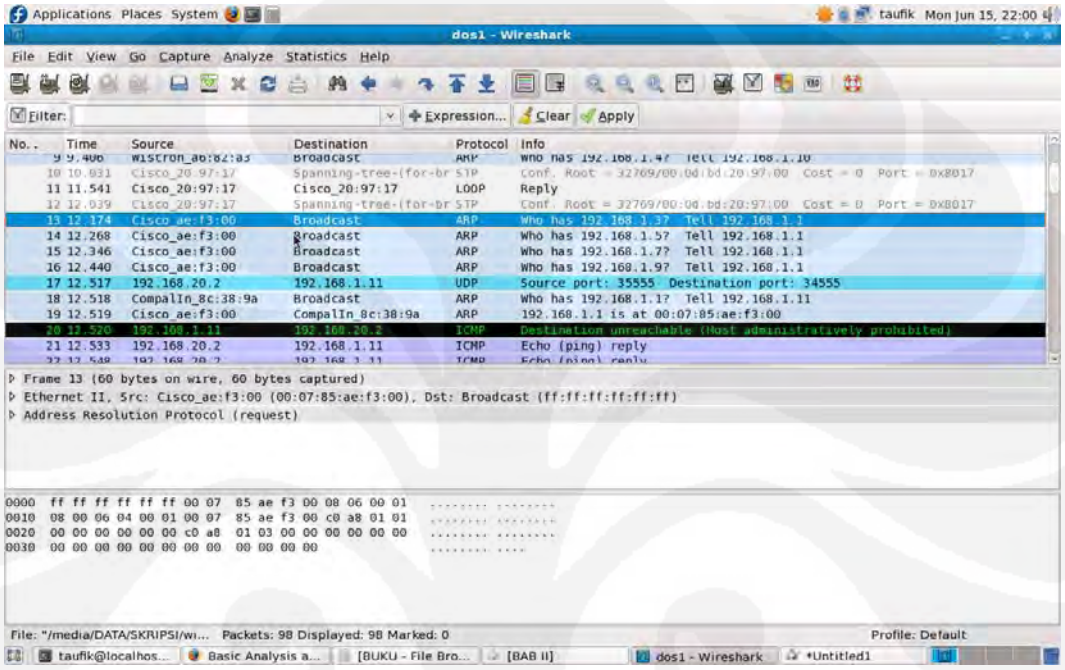
Denial of Service (DOS) adalah jenis serangan terhadap sebuah komputer atau *server* di dalam jaringan internet dengan cara menghabiskan sumber *(resource)* yang dimiliki oleh komputer tersebut sampai komputer tersebut tidak dapat menjalankan fungsinya dengan benar sehingga secara tidak langsung mencegah pengguna lain untuk memperoleh akses layanan dari komputer yang diserang tersebut.

Untuk melakukan DOS attack akan digunakan tool bernama dDoSping. Program ini akan mengenerate paket ping broadcast terhadap range ip yang telah ditentukan di dalam program. Berikut adalah tampilan dDoSping.exe:



Gambar 4.2 Tampilan DDoSPing

Apabila program dijalankan maka program akan mingirimkan paket *broadcast* keseluruh alamat IP yang berada di dalam range 192.168.1.1 – 192.168.1.254 sehingga apabila dilakukan monitoring terhadap jaringan dengan menggunakan wireshark akan ditunjukkan pada Gambar 4.3.



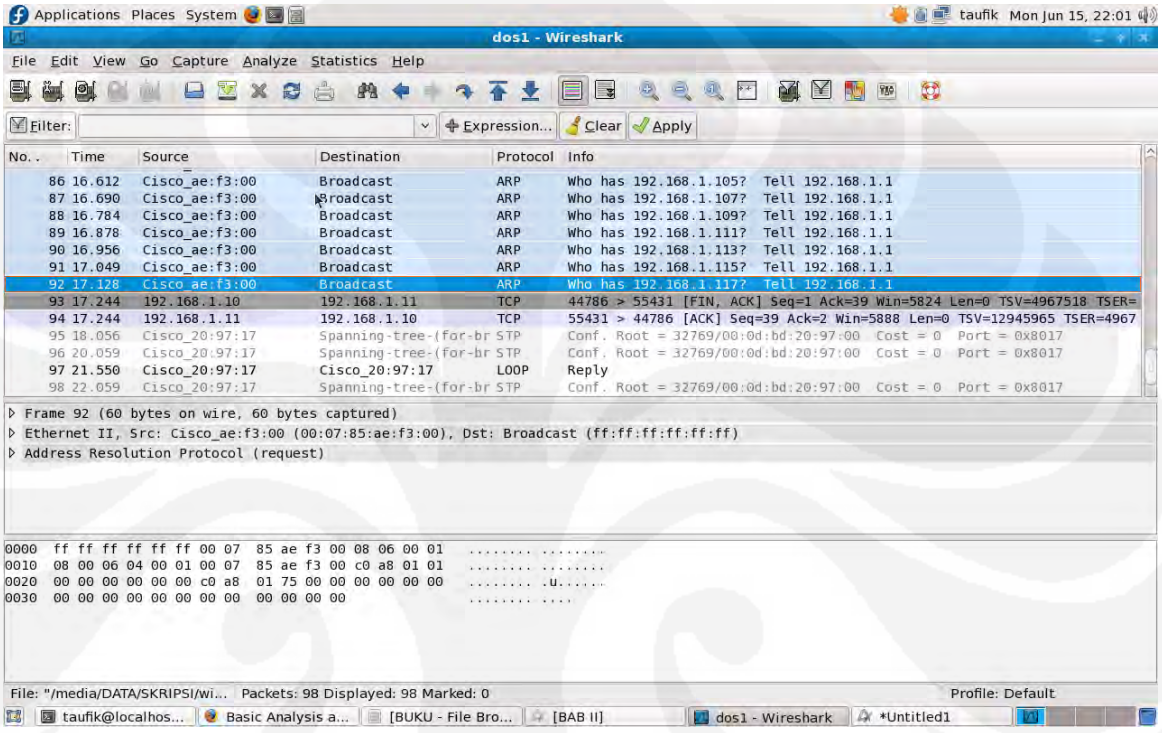
Gambar 4.3 Hasil *Capture* Paket DoS

Dari Gambar 4.3 ditunjukkan serangan DOS mulai berjalan. Paket *broadcast* dikirim ke setiap IP dari range 192.168.1.1 sampai 192.168.1.254. IDS *server* akan mendeteksi adanya serangan yang kemudian akan memberikan reaksi berupa *alerting*. Bentuk *alerting* dari serangan DOS adalah sebagai berikut:

```
06/14-05:25:30.358308 [**] DDOS Stacheldraht client check gag [**]
[Cassification: Attempted Denial of Service] [Priority: 1] {ICMP}
192.168.1.5 -> 192.168.1.11
```

Alerting menunjukkan adanya percobaan *Denial of Service* dengan prioritas serangan 1 dengan alamat sumber 192.168.1.5 dengan tujuan 192.168.1.11. Dari *alerting* ini akan dikirimkan ke *policy server* melalui *client – server module*. *Policy server* akan mengklasifikasi serangan berdasarkan tingakat prioritas. Tingkat prioritas dari serangan ini adalah 1 yaitu masuk kategori tingkat ancaman

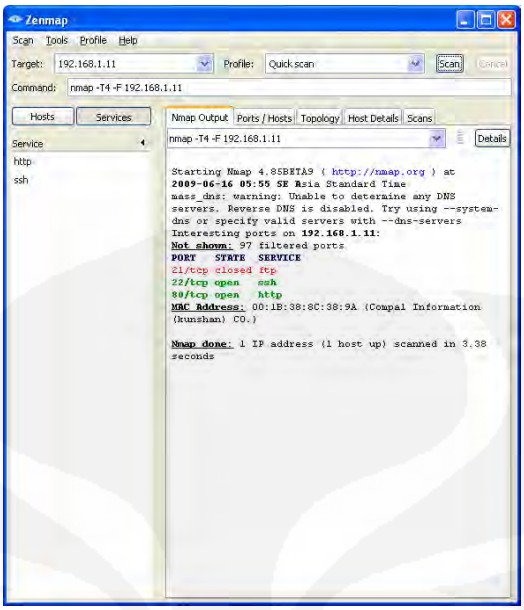
yang tinggi terhadap jaringan. Sehingga *policy server* akan mematikan port dimana ip 192.168.1.5 berada. Dari wireshark dapat dilihat serangan DOS dapat dihentikan. Hal ini ditunjukkan pada gambar 4.4 paket broadcast berhenti pada ip 192.168.1.117



Gambar 4.4 Penghentian DOS attack

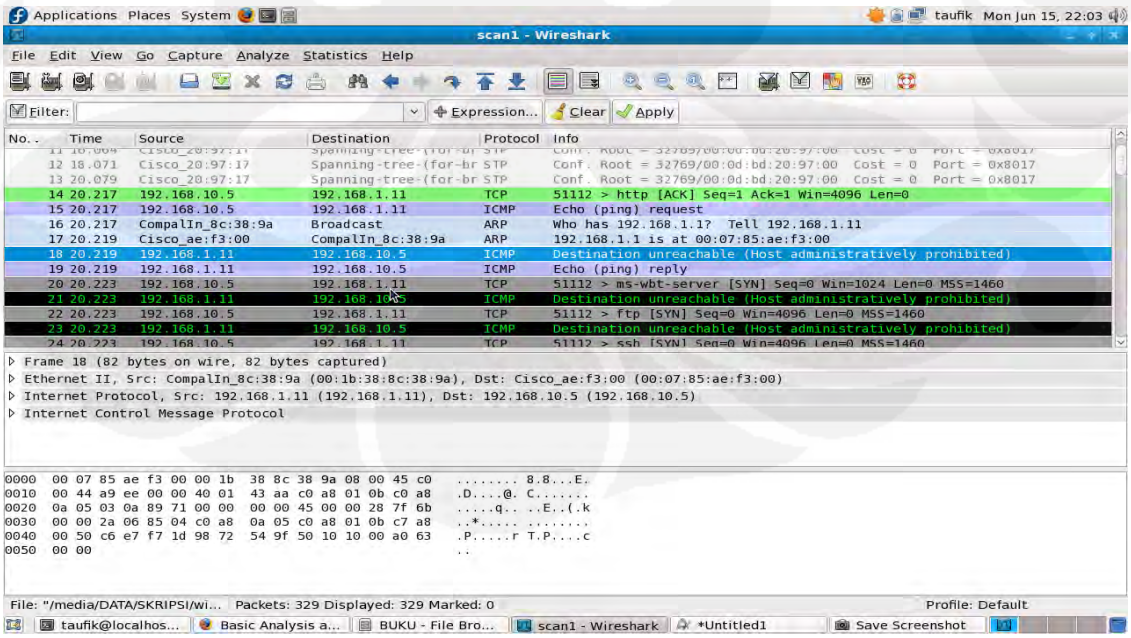
4.2.1.2 *Port scan*ning

Port scanning merupakan suatu proses untuk mencari dan membuka port pada suatu jaringan computer. Dari hasil scanning akan didapat letak kelemahan sistem tersebut. Pada dasarnya sistem *port scan*ning mudah untuk dideteksi, tetapi penyerang akan menggunakan berbagai metode untuk menyembunyikan serangan. Salah satu program scan yang terkenal adalah nmap. Berikut adalah tampilan *software* nmap:



Gambar 4.5 Tampilan nmap

Pada saat program dijalankan proses *scanning port* dimulai. Nmap akan mencari port yang aktif pada IP target. Hasil dari proses *scanning* di atas menunjukkan port 22 SSH dan 80 HTTP sedang terbuka. Hasil proses *scanning* ini sering digunakan penyerang untuk melakukan serangan disalah satu port tersebut. Untuk melihat cara kerja scanner port ini dapat dilihat dari hasil monitoring wireshark pada jaringan ketika nmap bekerja.

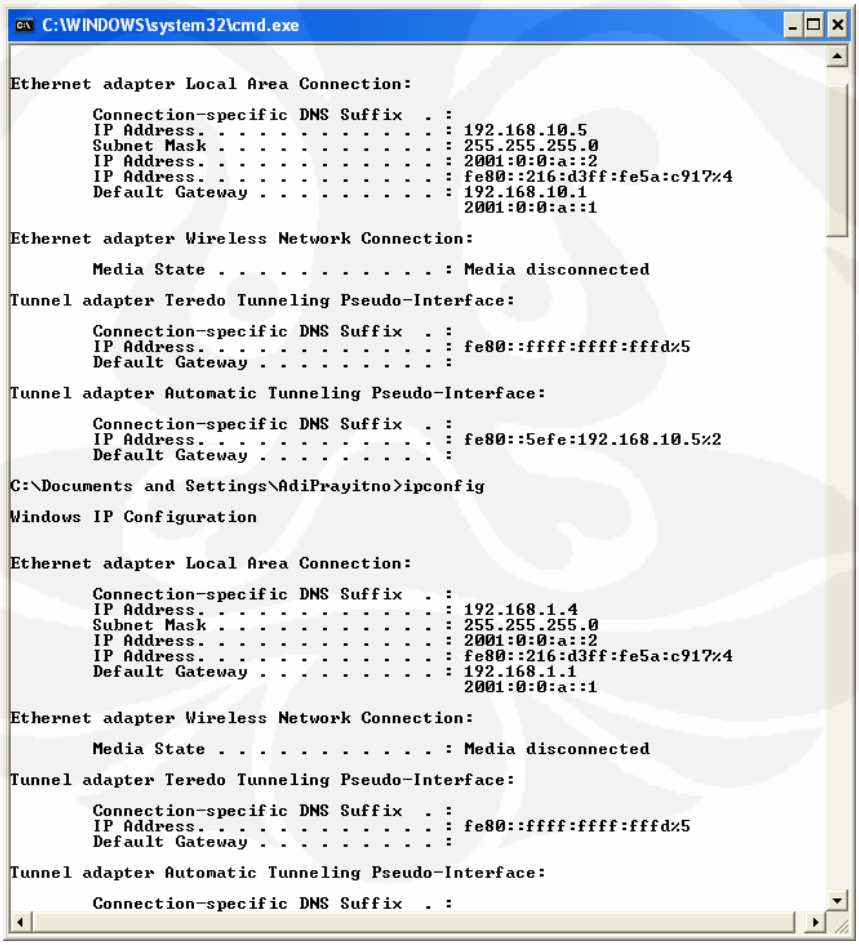


Gambar 4.6 Capture Paket *Port Scanning*

Apabila IDS *server* mendeteksi adanya *port scan*ning, IDS *server* akan mengeluarkan *alerting* berbentuk:

```
06/13-20:38:56.897016     [**] [1:469:4] ICMP PING NMAP [**]
[Classification: Attempted Information Leak] [Priority: 2] {ICMP}
192.168.10.5 -> 192.168.1.11
```

Dari *alerting* yang dikeluarkan oleh IDS *server*, menunjukkan adanya kebocoran informasi dari *server*. Sehingga *alerting* ini masuk kedalam priority 2 atau medium. Kemudian *alerting* dikirim ke *policy server*, sesuai dengan kebijakan *policy server* untuk priority 2 port akan dirubah ke vlan 1 atau *default* vlan. Pada sekenario ini pada saat terjadi *port scan*ning *client* berada pada vlan 10. Sehingga *client* akan dirubah ke vlan 1, hal ini dapat ditunjukkan pada Gambar 4.7 adanya perubahan ip *client* menjadi 192.168.1.4



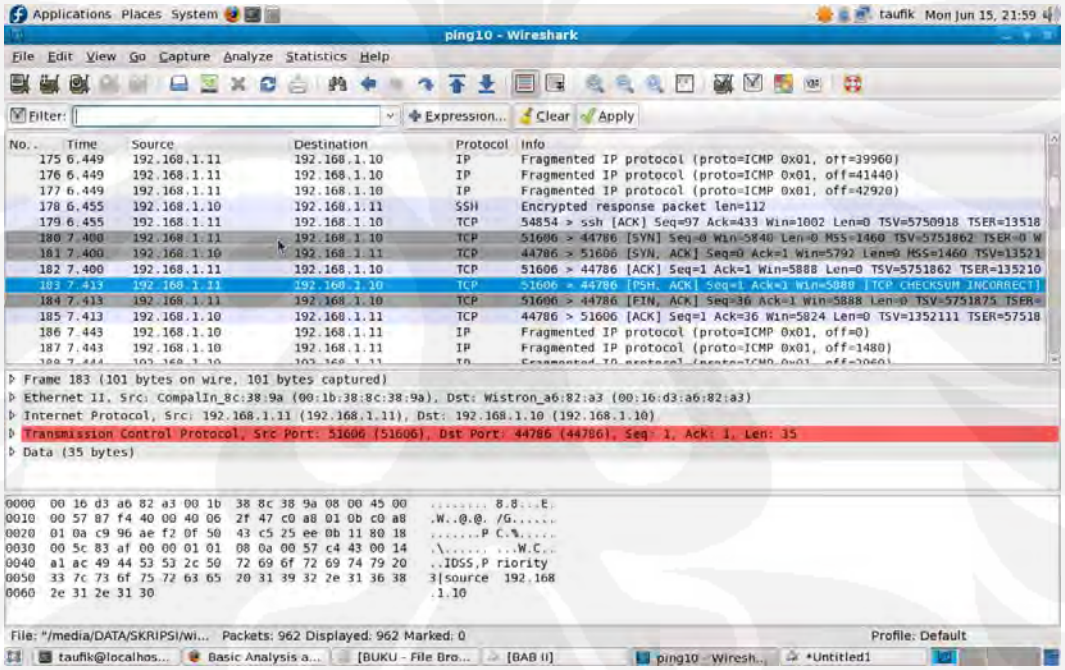
Gambar 4.7 Perubahan IP pada *Client*

4.2.1.3 ICMP *flood*

Penyerang melakukan eksploitasi sistem dengan tujuan untuk membuat suatu target *client* menjadi *crash*, yang disebabkan oleh pengiriman sejumlah paket yang besar ke arah target *client.* Bahkan hal ini dapat mengakibatkan *denial of service*. Pada skenario ini penyerang akan mengirimkan ping dengan ukuran 64000byte ke komputer target.

```
Root@192.168.1.10> ping 192.168.1.11 –s 64000
```

Dapat dilihat hasil monitoring jaringan pada komputer target dengan menggunakan wireshark sebagai berikut:



Gambar 4.8 Capture Paket ICMP *Flood*

Pada gambar di atas dapat dilihat proses penyerangan dengan menggunakan metode ICMP *flood*. Dari penyerangan di atas IDS *server* akan melakukan respon penyerangan dengan mengirimkan *alerting* berupa:

```
06/14-21:34:46.661446  [**] [1:480:6] ICMP PING speedera [**]
[Classification: Misc activity] [Priority: 3] {ICMP} 192.168.1.10 -
> 192.168.1.11
```

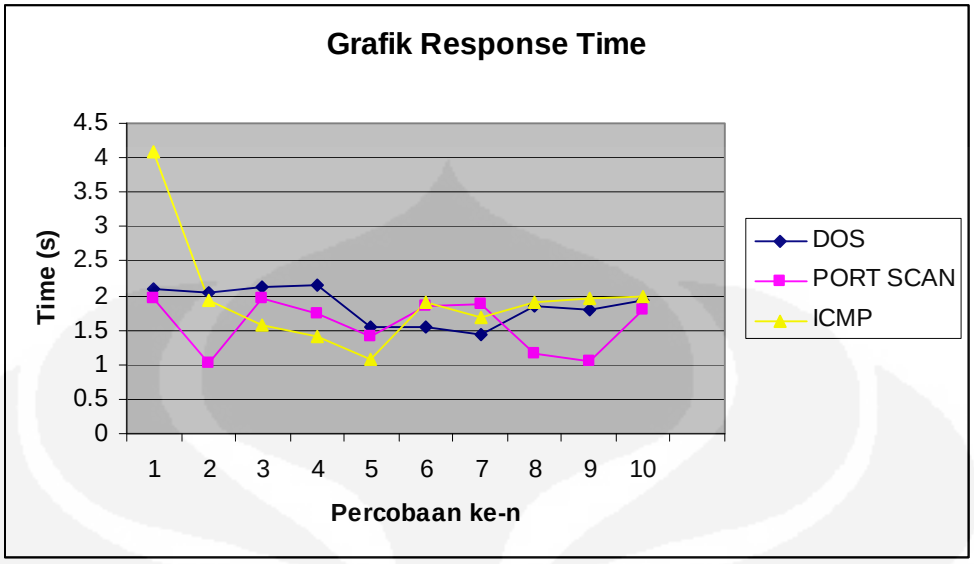
Dari *alerting* tersebut dikirimkan ke *policy server* melalui *client – server module*. Dari *policy server* didapatkan kebijakan bahwa priorty 3 tidak melakukan tindakan apapun. Sehingga serangan masih terus berjalan. Hal ini disebabkan karena IDS menganggap serangan tersebut masuk ke dalam prioritas serangan yang rendah.

4.2.2 *Response time* dan *Action time*

Tingkat kehandalan dari IDS *server* dapat dilihat dari beberapa parameter. Salah satu parameter yang penting adalah *response time* dan *action time*. *Response time* adalah waktu yang dibutuhkan untuk *server* merespon sebuah serangan. Pada percobaan ini pengukuran *response time* dilakukan pada saat serangan dimulai sampai pada saat *server* pertama kali memberikan respon. Sedangkan *action time* adalah waktu yang dibutuhkan sebuah *server* untuk bereaksi terhadap serangan tersebut. Pada percobaan *action time* diukur mulai pada saat *server* bereaksi pertama kali sampai serangan berhenti. Di bawah ini adalah respone time dan *action time* yang dihasilkan dari tiga metode penyerangan terhadap *server*. Setiap metode dilakukan sepuluh kali percobaan sehingga didapatkan data seperti pada Tabel 4.2.

Tabel 4.2 *Response time* dengan 1 *client*

	Response time (s)		
Percobaan Ke-n	**DOS**	***PORT SCAN***	**ICMP**
1	2.094	1.955	4.078
2	2.052	1.017	1.94
3	2.138	1.956	1.575
4	2.142	1.737	1.403
5	1.548	1.418	1.082
6	1.547	1.84	1.898
7	1.431	1.886	1.672
8	1.844	1.147	1.907
9	1.808	1.042	1.957
10	1.938	1.799	1.974
Average	**1.8542**	**1.5797**	**1.9486**

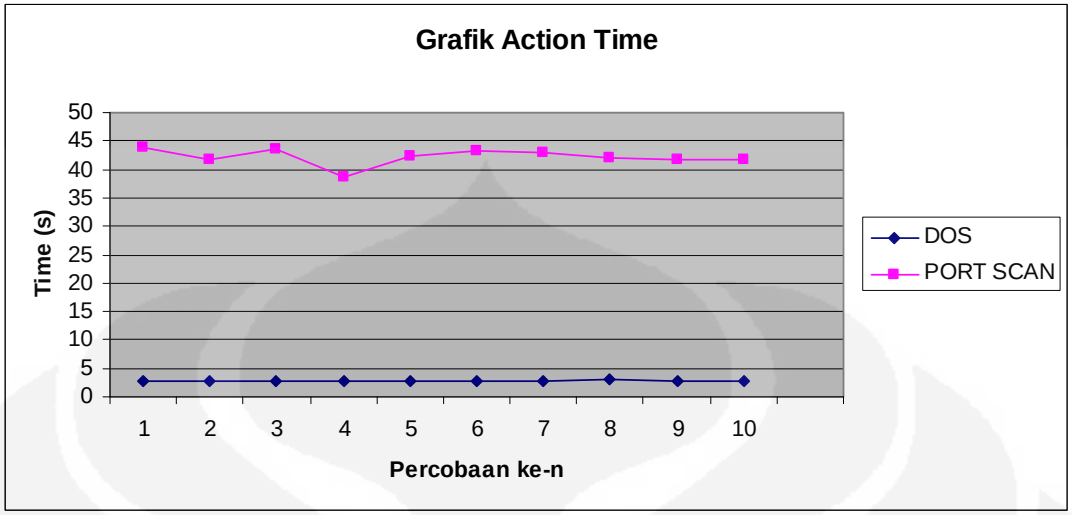


Gambar 4.9 Grafik *Response time* dengan 1 *Client*

Dari data *response time* ditunjukkan untuk jenis serangan denial of service (DOS) *server* memiliki rata – rata *response time* 1.85 detik. Untuk *port scan* memiliki rata – rata *response time* sebesar 1.57 detik dan ICMP *flood* dengan rata – rata *response time* 1.9486 detik. Hal ini menunjukkan IDS *server* dapat merespon setiap serangan kurang dari 2 detik. Untuk data *action time* dapat dilihat pada Tabel 4.3.

Table 4.3 *Action time* dengan 1 *Client*

	***Action time* (s)**	
Iterasi	**DOS**	***PORT SCAN***
1	2.86	43.904
2	2.902	41.793
3	2.91	43.685
4	2.687	38.53
5	2.89	42.322
6	2.891	43.19
7	2.834	43.086
8	3.203	41.953
9	2.895	41.861
10	2.843	41.652
Average	**2.8915**	**42.1976**



Gambar 4.10 Grafik *Action time* dengan 1 *Client*

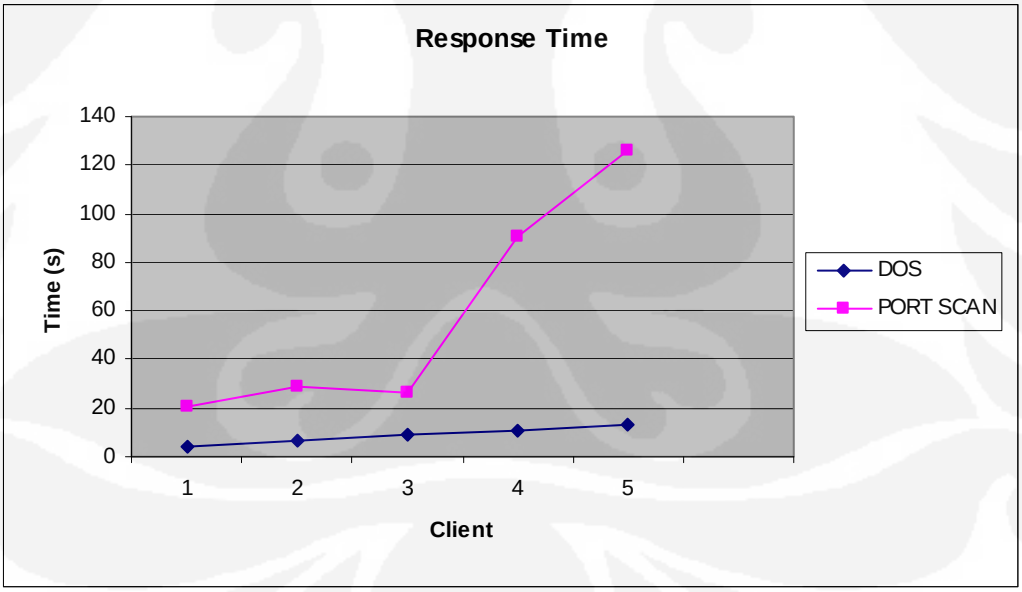
Dari data di atas ditunjukkan pada saat terjadi serangan denial of service (DOS) menunjukkan rata – rata *action time* sebesar 2.89 detik. Hal ini disebabkan karena pada saat terjadi serangan DOS. Jaringan penuh oleh paket *broadcast* yang dikirim oleh penyerang sehingga menghambat pengiriman *alerting* dari IDS *server* ke *policy server*. Yang menyebabkan *policy server* menjadi terhambat dalam menghentikan serangan DOS dengan cara mematikan port pada switch dimana komputer penyerang berada. Sedangkan pada saat serangan berupa *port scan* rata – rata *action time* sebesar 42.19 detik. Hal ini disebabkan *port scan* dikenali IDS *server* dengan prioritas 2, sesuai dengan kebijakan maka apabila ada serangan berupa *port scan server* akan merubah VLAN yang berada pada *client* ke default VLAN yaitu VLAN 1. Pada pengujian di atas sebelum melakukan *port scan client* berada pada vlan 10 dengan IP 192.168.10.2. Setelah dilakukan penyerangan menggunakan *port scan* vlan akan dirubah ke vlan 1 yaitu *default vlan* sehingga *server* memberikan IP DHCP 192.168.1.4. Pada prioritas 2 perhitungan *action time* dilakukan mulai dari *server* merespon sampai dengan *client* dirubah ke vlan 1 dan mendapatkan IP DHCP. Sehingga *action time* lebih lama dari prioritas 1.

Untuk ICMP *flood* tidak memiliki *action time* karena serangan ini berada pada prioritas 3. Apabila serangan berada pada prioritas 3 maka dianggap tingkat bahaya yang rendah bagi *server* sehingga *policy server* tidak melakukan tindakan apapun.

Salah satu parameter kehandalan dari suatu *server* juga harus dapat melayani beberapa *client* secara sekaligus. Maka untuk menguji IDS *server* juga dilakukan dengan menggunakan beberapa user. Pada pengujian ini akan digunakan lima *client* yang akan melakukan serangan secara berasamaan sehingga dapat dilihat *response time* dari *server*. Berikut adalah hasil pengujian dengan menggunakan lima *client*.

Table 4.4 *Response time* dengan 5 *client*

	***Response time* (s)**	
Client	**DOS**	***PORT SCAN***
1	3.97	20.2
2	6.327	28.675
3	9.243	26.236
4	10.679	90.568
5	13.198	126.36
Average	**8.6834**	**41.41975**



Gambar 4.11 Grafik *Response time* dengan 5 *Client*

Dari hasil pengujian dapat dilihat apabila jumlah *client* bertambah akan sangat berpengaruh terhadap performa *server*. Hal ini dapat ditunjukkan dengan semakin lamanya *response time* yang dihasilkan oleh IDS *server*. Ini disebabkan

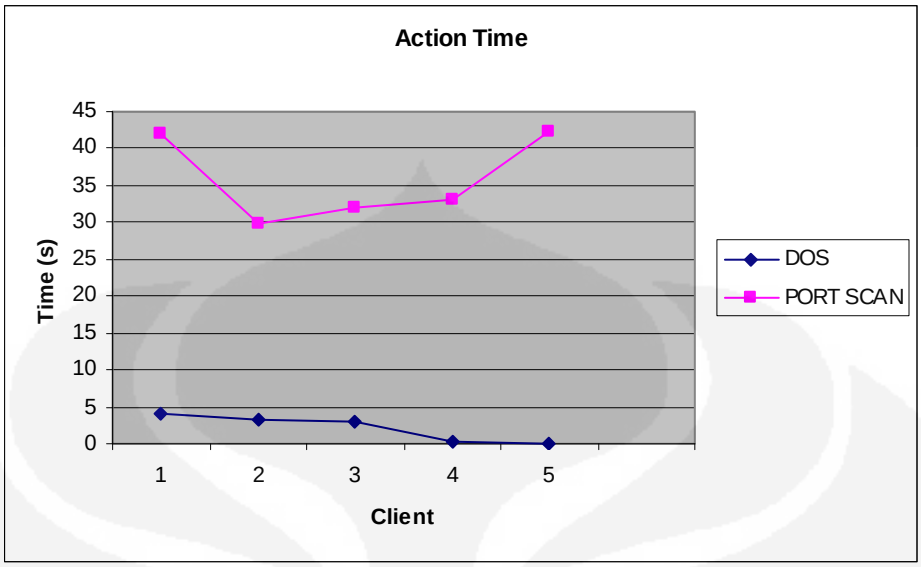
karena apabila serangan berupa *Denial of Service (DoS)* paket yang dikirimkan adalah paket *broadcast*. Apabila lima *client* mengirimkan paket *broadcast* secara bersamaan maka jaringan akan penuh sehingga menyebabkan kemampuan IDS *server* dalam merespon serangan akan terganggu. Begitu pula dengan *port scan* apabila lima *client* melakukan *scanning* secara bersamaan maka *server* menerima paket yang ditunjukkan ke *server* dalam jumlah besar, yang dapat menyebabkan *response time* semakin lambat.

Semakin lambatnya *response time* selain disebabkan karena proses pengiriman paket dalam jumlah besar yang berasal dari *client* juga dapat disebabkan karena pada modul *client – server* ditambahkan fungsi `sleep (2)`. Hal ini berfungsi untuk menghentikan pengiriman alerting selama 2 detik dari IDS *server* ke *policy server*, ini dilakukan agar memberikan jeda kepada switch untuk melakukan perubahan konfigurasi yang disebabkan karena penerimaan alerting.

Dengan bertambahnya *client* yang melakukan serangan secara bersamaan dapat menyebabkan *response time* yang dimiliki IDS *server* semakin lambat. Semakin lambatnya *response time* berpengaruh juga terhadap *action time* hal ini dapat dilihat di dalam tabel di bawah ini.

Table 4.5 *Action time* dengan 5 *client*

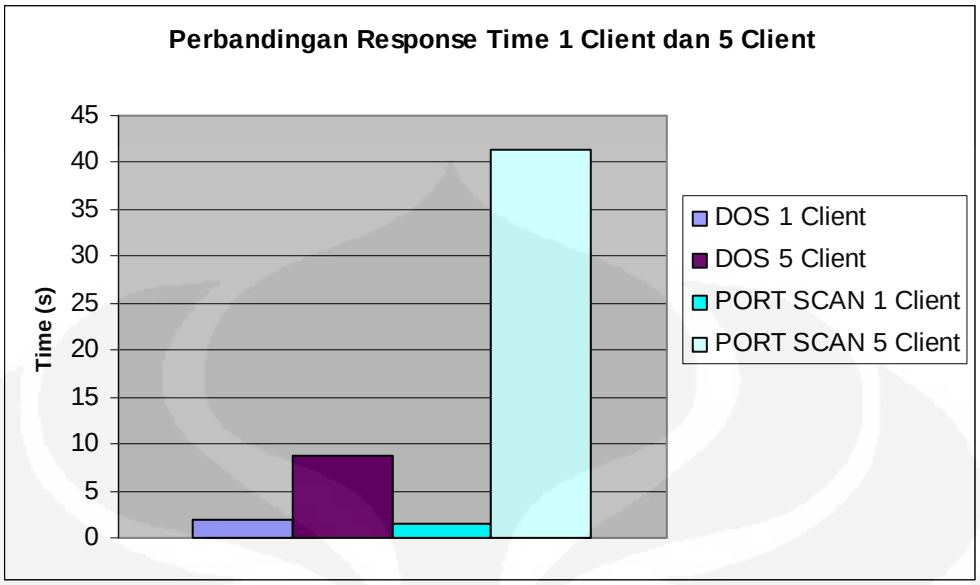
	***Action time* (s)**	
User	**DOS**	***PORT SCAN***
1	3.97	42.147
2	3.141	29.942
3	3.067	32.112
4	0.388	33.079
5	0.039	42.203
Average	**2.121**	**34.32**



Gambar 4.12 Grafik *Action time* dengan 5 *Client*

Dari data pengujian didapatkan data *action time* untuk serangan *denial of service (DoS)* dengan rata – rata 2.121 detik. *Action time* dihitung pada saat IDS *server* mengirimkan *alerting* ke *policy server* sampai dengan *policy server* menghentikan serangan ini dengan mematikan port pada switch dimana *client* berada. Pada *client* 4 dan 5 nilai *action time* menunjukkan nilai 0, hal ini disebabkan karena program dDoSping.exe pada sisi *client* hanya mengirimkan paket *broadcast* selama kurang lebih 10 detik. Sehingga pada saat *response time* yang dimiliki *client* 4 dan 5 berada di atas 10 detik serangan DoS sudah berhenti. Sehingga nilai *action time* bernilai 0, karena tidak terlihat IP terkhir yang mendapat kiriman paket *brodcast* dari *client*. Tetapi karena *alerting* telah dikirim ke *policy server* maka switch tetap mematikan port pada *client* 4 dan 5. Sedangkan untuk pengujian menggunakan *port scan* data rata – rata *action time* sebesar 34.32 detik.

Untuk mengetahui perbandingan nilai rata – rata *response time* antara pengujian menggunakan 1 *client* dan 5 *client* dapat dilihat pada Gambar 4.13.

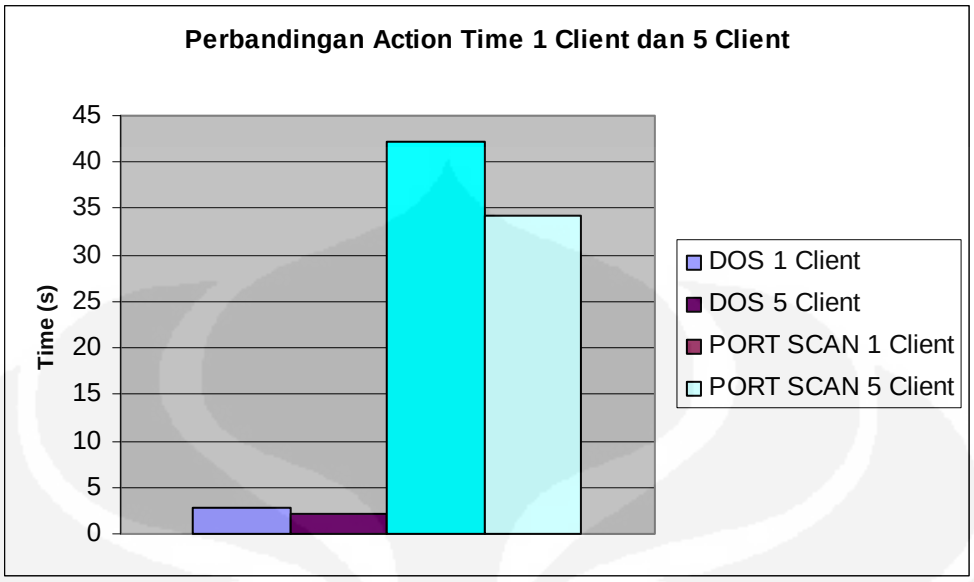


Gambar 4.13 Perbandingan Response Time 1 *Client* dan 5 *Client*

Pada gambar grafik di atas ditunjukkan perbandingan antara *response time* IDS *server* terhadap serangan yang berasal dari 1 *client* dan 5 *client*. Untuk searangan *denial of service (DoS) response time response time* yang berasal dari 1 *client* menunjukkan nilai rata – rata *response time* sebesar 1.8542 detik. Sedangkan yang berasal dari 5 *client* menujukkan nilai rata – rata *response time* sebesar 8.6834 detik. Hal ini menunjukkan IDS *server* mengalami penurunan performa dalam merespon sebuah serangan berupada DoS sebesar 64.81%.

Untuk pengujian menggunakan *port scan* nilai rata – rata *response time* apabila menggunakan 1 *client* sebesar 1.5797 detik, sedangkan apabila menggunakan 5 *client* sebesar 41.41975 detik. . Hal ini menunjukkan IDS *server* mengalami penurunan performa dalam merespon sebuah serangan berupada DoS sebesar 92.65%. Penurunan performa dari IDS *server* disebabkan karena pada saat terjadi serangan IDS *server* dibanjiri oleh paket *broadcast* atau paket berupa *scan port* yang berasal dari *client*. Sehingga ketika IDS *server* akan mengirimkan respon berupa pengiriman *alerting* ke *policy server* menjadi terhambat dengan adanya paket yang diterima dalam jumlah besar oleh IDS *server*.

Penurunan performa pada IDS *server* yang disebabkan bertambahnya jumlah *client* yang melakukan penyerangan terhadap IDS *server* tidak berpengaruh pada *action time*. Hal ini dapat dilihat pada Gambar 4.14.



Gambar 4.14 Perbandingan *Action Time* 1 *Client* dan 5 *Client*

Pada Gambar 4.14 ditunjukkan adanya perbandingan *action time* dimana rata – rata *action time* apabila terjadi serangan berupa *denial of service (DoS)* yang dihasilkan oleh 1 *client* sebesar 2.8915 detik sedangkan apabila menggunakan 5 *client* sebesar 2.121 detik. Sedangkan serangan berupa *port scan* nilai rata – rata *action time* apabila menggunakan 1 *client* sebesar 42.1976 detik, apabila menggunakan 5 *client* nilai rata – rata *action time* sebesar 34.32 detik.

Hal ini menunjukkan adanya nilai *action time* semakin cepat apabila menggunakan 5 *client* sebesar 15.37% apabila terjadi serangan berupa DoS dan sebesar 10.30% apabila terjadi serangan berupa *port scanning*. yang disebabkan oleh *action time* tidak terpengaruh secara langsung dengan *response time* karena *action time* baru dihitung setelah *response time* selesai. Nilai *action time* lebih dipengaruhi oleh kondisi dari *policy server,* apabila *policy server* dalam kondisi *idle policy server* dapat melakukan klasifikasi *alerting* dan kemudian melakukan tindakan untuk menghentikan serangan secara cepat.

BAB 5
KESIMPULAN

1. Pada *functionality test* IDS *server* dapat merespon adanya serangan berupa denial of service (DoS), *port scan* dan ICMP *flood*.
2. Bertambahnya jumlah *client* yang melakukan penyerangan terhadap IDS *server* akan menyebabkan penurunan performa IDS *server* dalam merespon sebuah serangan. Hal ini dapat ditunjukkan nilai *response time* semakin meningkat sebesar 64.81% apabila terjadi serangan berupa *denial of service (DoS)* dan 92.65% apabila terjadi serangan berupa *port scanning.*
3. Terjadinya penurunan performa IDS *server* dalam merespon serangan disebabkan banyaknya paket yang dikirimkan masing – masing *client* ke IDS *server* sehingga IDS server terhambat dalam pengiriman *alerting* ke *policy server*.
4. Untuk *action time* dengan bertambahnya *client* nilai *action time* semakin cepat. Hal ini dapat ditunjukkan nilai *action time* semakin menurun dengan bertambanya client. Untuk serangan berupa DoS nilai *action time* menurun sebesar 15.37% dan untuk serangan berupa *port scanning* nilai *action time* menurun sebesar 10.30%.
5. Nilai *action time* lebih dipengaruhi oleh kondisi dari *policy server*. Apabila *policy server* dalam kondisi *idle* maka *policy server* dapat melakukan klasifikasi serangan dan penghentian serangan secara lebih cepat.

DAFTAR REFERENSI

[1]. Ariyus, Dony. " Intrusion Detection System", 2007

[2]. Laing, Brian. "How To Guide-Implemening a Network Based Intrusion Detection System", 2000

[3]. Roesch, Martin. Green, Chris. Sourcefire,Inc, " Snort User Manual 2.8.3". September 15, 2008

[4]. Hartono, Puji. " Sistem Pencegahah Penyusupan pada Jaringan berbasis Snort IDS dan IPTables Firewall", June 5, 2006

[5]. Suparsa, I Ketut. " *Client* – Based Intrusion Detection Systems (HBIDS), 2004

[6]. www.interop.com/archive/pdfs/CISCONAC.pdf, "What is CISCO NAC". Diakses 21 Januari 2009

[7]. http://www.cisco.com/en/US/netsol/ns466/networking_solutions_package.html, " Network Admission Control". Diakses 30 Januari 2009

[8]. Harper, Patrick. "Snort Install Manual on Fedora Core 3", November 2, 2005

LAMPIRAN

Lampiran 1 : *Client-Server* Module

```
#!/usr/bin/perl -w
#
use strict;
use warnings;


  use File::Tail;
  use IO::Socket;

  my($name) = "/var/log/snort/logfile/alert";
  my($file) =File::Tail->new(name=>$name, maxinterval=>1);
  my($buffer)=0;
  my($buffer2)=0;
  my($local)="192.168.1.11";
  while (defined(my $_=$file->read))
{
        #print "$line";
      no warnings 'uninitialized';
      chomp($_);

      #extract sid and description format: [1:882:4] description
[**]
      #$_ =~ /\[(\d+):(\d+):(\d)\]/;
      #my($sid) = $2;
      #my($desc) = $4;

      $_ =~ /\[Priority: (\d+)\]/;
      my($prio) = $1;

      $_=~ /\[(\d+):(\d+):(\d)\]\s(.*) \[\*\*\]\s(.*) \[Priority:
(\w+)]/;
      my($event) = $5;

      # extract date and time format: 04/02-18:56:25.123340
#     $_ =~ /((\d+)\/(\d+))-((\d\d):(\d\d):(\d\d))/;
#     my($date) = $1;
#     my($tm) = $4;

      #extract protocol format: {TCP}
      #$_ =~/{(\w+)}/;
      #my($proto) = $1;

      #extract src and tgt addresses format: 65.29.19.186:40668 ->
205.174.16.50:80
      $_ =~ /(\d+).(\d+).(\d+).(\d+):(\d+) ->
(\d+).(\d+).(\d+).(\d+):(\d+)/;
#     $_ =~ /(\d+).(\d+).(\d+).(\d+) -> (\d+).(\d+).(\d+).(\d+)/;

      my($src) = "$1.$2.$3.$4";
      my($dst) = "$6.$7.$8.$9";
```

```
      #extract sid and description format: [1:882:4] description
[**]
        $_ =~ /\[(\d+):(\d+):(\d)\]/;
        my($sid) = $2;
        my($desc) = $4;

      #print out the formatted values
      if ($buffer != $sid && $src ne $local && $sid != 486)
      {
            if ( $sid == 5 || $sid == 469 || $sid == 236 || $sid
== 1 || $sid == 485 || $sid == 480)
            {
                  $_ =~ /(\d+).(\d+).(\d+).(\d+) ->
(\d+).(\d+).(\d+).(\d+)/;
                    my($src) = "$1.$2.$3.$4";
                  my($dst) = "$5.$6.$7.$8";
                  my($mydata) = "IDSS,Priority $prio|source $src";
                  system ("./client 192.168.1.10 44786
'$mydata'");
                  print ("Sid = $sid Priority = $prio source IP=
$src \n");
            }
            else
            {
                  my($mydata) = "IDSS,Priority $prio|source $src";
                        system ("./client 192.168.1.10 44786
'$mydata'");
                  print ("Sid = $sid Priority = $prio source IP=
$src \n");
            }
      }
      $buffer = $sid;
      $buffer2 = $src;
      sleep (2);
}
```

Lampiran 2: Listing Program *client*.c

```
#include <stdio.h>
#include <sys/socket.h>
#include <string.h>
#include <arpa/inet.h>
#include <stdlib.h>

#define SERV_PORT 44786

void err(char* str)
{
fprintf(stdout,str);
fflush(stdout);
exit(EXIT_FAILURE);
}


void sendString(char* str)
{
fprintf(stdout,str);
fflush(stdout);
}

int main(int argc,char** argv)
{
int sockfd;
struct sockaddr_in servaddr;
char msg[1024];
int nread;

if(argc<4){
      printf("%s ip port pesan\n", argv[0]);
      exit(0);
}


if((sockfd=socket(AF_INET,SOCK_STREAM,0))<0)
err("Socket Failed");

memset(&servaddr,0,sizeof(servaddr));

servaddr.sin_family=AF_INET;
servaddr.sin_port=htons(atoi(argv[2]));
inet_pton(AF_INET,argv[1],&servaddr.sin_addr);

connect(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));

write(sockfd,argv[3],strlen(argv[3]));


}
```