UNIVERSITAS INDONESIA

Pengembangan dan Analisa Kinerja *Intrusion Detection Prevention System* (IDPS) pada *Web Server*

TESIS

PANCA HARIWAN
NPM: 0906578106

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK ELEKTRO
DEPOK
JANUARI 2012

UNIVERSITAS INDONESIA

Pengembangan dan Analisa Kinerja *Intrusion Detection Prevention System* (IDPS) pada *Web Server*

TESIS

Diajukan sebagai salah satu syarat untuk memperoleh gelar Magister Teknik

PANCA HARIWAN
NPM: 0906578106

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK ELEKTRO
KEKHUSUSAN TEKNIK JARINGAN INFORMASI DAN MULTIMEDIA
DEPOK
JANUARI 2012

HALAMAN PENYATAAN ORISINALITAS

Tesis ini adalah hasil karya saya sendiri,

dan semua sumber baik yang dikutip maupun dirujuk

telah saya nyatakan dengan benar.

Nama	**: Panca Hariwan**
NPM	**: 0906578106**
Tanda Tangan	**:** 
Tanggal	**: 12 JANUARI 2012**

HALAMAN PENGESAHAN

Tesis ini diajukan oleh

Nama	**: Panca Hariwan**
NPM	**: 0906578106**
Program Studi	**: Teknik Elektro**
Judul Tesis	**:**

Pengembangan dan Analisa Kinerja *Intrusion Detection Prevention System* (IDPS) pada *Web Server*

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

DEWAN PENGUJI

Pembimbing I : Prof. DR.Ing.Ir. Kalamullah Ramli, M.Eng (.....)

Pembimbing II : Muhammad Salman, S.T, MIT (.....)

Penguji : Prof.DR.Ir.Riri Fitri Sari, M.Sc, MM (.....)

Penguji : Yan Maraden, ST, M.Sc (.....)

Ditetapkan di : Depok

Tanggal : 12 Januari 2012

UCAPAN TERIMA KASIH

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya, saya dapat menyelesaikan tesis ini. Penulisan tesis ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Magister 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 tesis ini, sangatlah sulit bagi saya untuk menyelesaikan tesis ini. Oleh karena itu, saya mengucapkan terima kasih kepada:

1. Pembimbing pertama, Bapak Prof.DR.Ing.Ir.Kalamullah Ramli, M.Eng
2. Pembimbing kedua, Bapak. Muhammad Salman, S.T, MIT

Atas bimbingan dan waktu yang disediakan untuk memberi pengarahan, diskusi, dan bimbingan sehingga proses untuk menulis tesis ini dapat terselesaikan. Semoga Allah SWT memberikan selalu memberikan kemudahan untuk keduanya.

Depok, 12 Januari 2012

Penulis,

Panca Hariwan
NPM: 0906578106

HALAMAN PENYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

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

Nama : Panca Hariwan
NPM : 0906578106
Program Studi : Teknik Elektro
Departemen : Teknik Elektro
Fakultas : Teknik
Jenis karya : Tesis

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

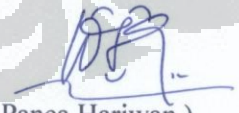
Pengembangan dan Analisa Kinerja *Intrusion Detection Prevention System* (IDPS) pada *Web Server*

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Nonekslusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/format-kan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan tugas akhir 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 : 12 Januari 2012

yang menyatakan



(Panca Hariwan)

ABSTRAK

Nama : Panca Hariwan
Program Studi : Teknik Elektro
Judul : **Pengembangan dan Analisa Kinerja *Intrusion Detection Prevention System* (IDPS) pada *Web Server***
Pembimbing : Prof. DR.Ing.Ir. Kalamullah Ramli, M.Eng
Muhammad Salman, S.T, MIT

Adanya serangan atau akses yang tidak sah dapat mengakibatkan terjadinya kerusakan pada sistem *web server*. Pengamanan *web server* secara umum dilakukan dengan menggunakan *firewall*, namun ternyata hal itu saja belum cukup. *Firewall* secara umum memberlakukan setiap akses secara kaku dengan dua kondisi, yaitu: boleh akses atau tidak. Sehingga sulit untuk mendeteksi apabila serangan itu dilakukan oleh akses yang sah tetapi melampaui kewenangan yang diberikan padanya. Oleh sebab itu *firewall* harus disempurnakan, salah satunya dengan menambahkan perangkat IDPS untuk bekerjasama dengan *firewall* dalam melindungi *web server*. Pada percobaan yang dilakukan memperlihatkan, saat kondisi lalu lintas data *idle*, IDPS mengembalikan nilai prosesor sistem rata-rata sebesar 91,76 % , memori sistem rata-rata sebesar 71,43 %, dan *bandwith* sistem rata-rata sebesar 97,4 %. Pada kondisi lalu lintas data menengah, IDPS mengembalikan nilai prosesor sistem rata-rata sebesar 83 % , memori sistem rata-rata sebesar 89 %, dan *bandwith* sistem rata-rata sebesar 93,1 %. Sedangkan pada kondisi lalu lintas data tinggi, IDPS mengembalikan nilai prosesor sistem rata-rata sebesar 73 % , memori sistem rata-rata sebesar 90 %, dan *bandwith* sistem rata-rata sebesar 87,18 %.

Kata kunci: *intrusion detection system, intrusion prevention system, intrusion detection prevention system, firewall*

ABSTRACT

Name : Panca Hariwan
Study Program: Teknik Elektro
Title : **Development and Performance Analysis of Intrusion Detection Prevention System (IDPS) on a Web Server.**
Lecturer : Prof. DR.Ing.Ir. Kalamullah Ramli, M.Eng
Muhammad Salman, S.T, MIT

Intrusion can damage the web server system. The web server security is usually performed using a firewall, but it is not enough. Firewalls classify data in two conditions, allowing access or not. It is difficult to detect when the legitimate access that goes beyond the authority assign to it. Therefore, the firewall must be refined. We can adding the IDPS to cooperate with firewalls to increase protecting our web server system. Our simulation shows that, when traffic in idle conditions, IDPS return a value of processor system in average of 91.76%, the average of memory system is 71.43%, and the bandwidth system is around 97.4%. In medium traffic conditions, IDPS return a value of processor system in average of 83%, the average of memory system is 89%, and the bandwidth system is around 93.1%. While in high traffic conditions, IDPS return a value of processor system in average of 73%, the average of memory system is 90%, and the bandwidth system is around 87.18%.

keyword: *intrusion detection system, intrusion prevention system, intrusion detection prevention system, firewall*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1
PENDAHULUAN

1.1. Latar Belakang.

Pada era modern seperti saat ini kebutuhan akan informasi menjadi suatu kebutuhan yang tak dapat ditawar-tawar lagi. Setiap pagi setiap hari sejak bangun tidur sampai tidur lagi selalu disuguhkan dengan beragam informasi yang sepertinya tak henti datang, baik yang bernilai positif maupun negatif. Sehingga seakan-akan manusia tidak dapat hidup tanpa berita dan informasi. Setiap orang siapa pun dia pada saat ini mungkin akan merasa sangat tidak nyaman apabila berpergian dan beraktifitas tanpa membawa telephone genggam mereka. Bahkan ada yang rela untuk kembali ke rumah atau menyuruh orang yang ada dirumah untuk mengantarkan telephone genggam mereka yang tertinggal.

Informasi juga merupakan suatu hal yang sangat penting bagi sebuah organisasi. Kemampuan untuk mengakses dan menyediakan informasi secara cepat dan akurat menjadi sangat penting bagi sebuah organisasi, baik yang berupa organisasi komersial (perusahaan), perguruan tinggi, lembaga pemerintahan, maupun individual. Semua hal tersebut dapat dengan mudah dilakukan seiring dengan pesatnya perkembangan teknologi terutama di bidang teknologi komputer dan telekomunikasi. Dahulu komputer belum seperti sekarang, kapasitas dan kemampuannya masih sangat terbatas, baik dari segi perangkat keras maupun perangkat lunak. Evolusi komputer mengantarkannya mulai dari hanya sebagai sebuah alat pengolah data yang bersifat *'stand alone'* kemudian berkembang dengan sangat cepat sehingga terbentuk sebuah komunikasi antar komputer dengan berbagai variasinya, seperti *local area network* (LAN), *metropolitan area network* (MAN), *wide area network* (WAN).

Dengan terhubungnya komputer dalam sebuah jaringan perpindahan data dan informasi dari satu tempat ke tempat yang lain dapat dilakukan dengan mudah dan cepat. Sehingga aliran data dan informasi dalam sebuah jaringan internet dapat berpindah bukan hanya dalam sebuah jaringan yang berada dalam sebuah tempat atau lokal, namun sudah dapat dengan cepat berpindah antar negara dan antar benua dalam waktu yang relatif singkat dan cepat.

Dalam masyarakat modern informasi itu sangat penting diperlukan dalam mengambil sebuah keputusan. Informasi yang tepat dan akurat akan memperkecil kemungkinan kesalahan yang terjadi pada saat sebuah keputusan diambil. Beberapa hal yang menyebabkan sebuah informasi itu amat sangat berharga adalah bagaimana caranya agar informasi yang dibutuhkan dapat diperoleh dengan cepat dan tepat waktu, serta relevan dengan persoalan yang sedang dihadapi. Kesemua hal itu dapat diperoleh dengan alat pengolah data modern yang disebut dengan komputer. Komputer dapat mengolah data dengan cepat, menyimpannya, dan mengirimkannya ke tempat yang diinginkan. Karena kemampuannya tersebut maka manusia modern banyak menggantungkan proses pengolahan data yang diinginkan kepada komputer, termasuk untuk memproses dan menyimpan data dan informasi yang sangat penting.

Karena informasi yang sangat penting tersebut tentunya juga banyak orang yang ingin mengetahui dan menggunakannya, baik mereka itu memang yang berhak menggunakannya maupun pihak-pihak yang dengan sengaja mencari cara untuk dapat mengakses dan menggunakannya untuk dan dengan cara-cara yang tidak bertanggungjawab. Dengan demikian data dan informasi tersebut memang harus dijaga dan dilindungi serta dipastikan hanya dipakai oleh pemakai yang berhak mengakses dan menggunakannya.Selain itu data dan informasi tersebut juga harus dilindungi dan diamankan dari kemungkinan adanya gangguan teknis seperti listrik statis, sumber tegangan yang tidak stabil, pencurian secara fisik, ataupun kemungkinan terjadinya bencana alam, terutama pada data yang diletakan pada jaringan komputer.

Intrusion detection adalah proses pemantauan setiap kejadian yang terjadi pada sebuah sistem komputer atau jaringan komputer dan menganalisisnya untuk memberi tanda atas setiap kemungkinan kejadian yang mungkin terjadi, seperti pelanggaran atau ancaman terhadap kebijakan keamanan sistem komputer, yang ditangani dengan menggunakan kebijakan atau praktek-praktek standar keamanan komputer.

Intrusion prevention adalah proses yang menampilkan *intrusion detection* dan usaha-usaha untuk menghentikan kemungkinan-kemungkinan yang bisa terjadi setelah proses deteksi. Sedangkan penggabungan *intrusion detection* dan

intrusion prevention (IDPS) utamanya fokus pada identifikasi gangguan atau ancaman yang mungkin terjadi, memberikan informasi tentang *logging information* dari gangguan atau ancaman tersebut, usaha-usaha untuk menghentikan gangguan atau ancaman tersebut, dan melaporkan gangguan atau ancaman tersebut kepada pengatur keamanan. Selanjutnya organisasi atau perusahaan juga menggunakan IDPS untuk berbagai keperluan, seperti: mengidentifikasi masalah tentang kebijakan keamanan komputer, pendokumentasian ancaman yang terjadi, dan menghalangi pribadi atau individu dari pelanggaran terhadap kebijakan keamanan jaringan komputer. IDPS menjadi hal yang penting digunakan pada infrastruktur keamanan jaringan komputer pada setiap organisasi/perusahaan.

IDPS secara khusus merekam informasi yang berhubungan dengan objek yang di amati, memberitahukan tentang keamanan dari objek yang di amati kepada administrator, dan menghasilkan sebuah report atau laporan. Banyak IDPS juga dapat merespon sebuah ancaman dengan cara menanggapinya untuk mencegahnya agar tidak mengganggu sistem. Mereka menggunakan beberapa teknik untuk merespon ancaman atau gangguan, diantaranya IDPS mampu menghentikan ancaman terhadap dirinya sendiri, perubahan terhadap keamanan sistem (seperti rekonfigurasi sebuah *firewall*), atau perubahan isi dari ancaman atau gangguan.

Jenis teknologi dari IDPS secara umum dibedakan berdasarkan jenis dari kejadian yang mereka pantau dan cara yang mereka gunakan. Dalam IDPS masalah Keakurasian *signature* sangat ditentukan oleh sensor dan *update* informasi yang ada, dimana sensor membuat *alert*, di suatu kondisi memicu alarm dari sensor (valid atau tidak), jika tidak valid terdeteksi bisa juga sangat memungkinkan sebagai serangan. Tingginya tingkat aktivitas jaringan yang terjadi kadangkala menyebabkan sulitnya untuk mendeteksi adanya sebuah serangan atau bahkan mendeteksi apabila sebuah sistem telah diambil alih.

1.2. Tujuan Penulisan.

Tujuan dari penulisan tesis ini adalah:

a. Membuat aplikasi sistem pemantauan jaringan komputer berbasis *web*.
b. Membuat aplikasi untuk menambah kemampuan dari *intrusion detection system* (IDS) terutama dalam memberikan tanggapan atau tindakan pada saat terjadinya serangan.
c. Meminimalisasi terjadinya *false positive* pada IDS.

1.3. Batasan Masalah.

Pembahasan masalah pada tesis ini akan dibatasi pada:

a. Pembuatan aplikasi sistem pemantauan jaringan komputer berbasis *web*. Menambahkan kemampuan dari *intrusion detection system* (IDS) terutama dalam memberikan tanggapan atau tindakan pada saat terjadinya serangan. Serta meminimalisasi terjadinya *false positive* pada IDS.
b. Perangkat lunak yang digunakan dalam mengimplementasi server, IDS, maupun aplikasi yang akan dibuat semuanya menggunakan perangkat lunak *open source* dan *free software*.
c. Implementasi dan pengujian dari sistem aplikasi yang dibuat diuji dengan membentuk sebuah laboratorium jaringan mini dengan topologi star. Dengan skenario yang sudah ditentukan sebelumnya.

1.4. Sistematika Penulisan.

Pembahasan yang dilakukan pada penelitian ini meliputi empat bab sebagai berikut:

BAB 1 PENDAHULUAN

Bab ini terdiri dari latar belakang masalah, tujuan penulisan, batasan masalah dan sistematika penulisan.

BAB 2 *INTRUSION DETECTION PREVENTION SYSTEM (IDPS)*

Bab ini berisi tinjauan umum tentang IDPS, IDS, dan IPS, Snort, Hypertext Preprocessor (PHP), MySQL, Diagram Aliran Data (DAD), Zenmap, dan Aktifitas *hacking*.

BAB 3 PERANCANGAN SISTEM IDPS.

Bab ini berisi penjelasan tentang jaringan komputer, analisa kebutuhan, perancangan IDPS, IDS dengan snort, IPS, rancangan antarmuka, rancangan basis data.

BAB 4 IMPLEMENTASI DAN PENGUJIAN SISTEM IDPS.

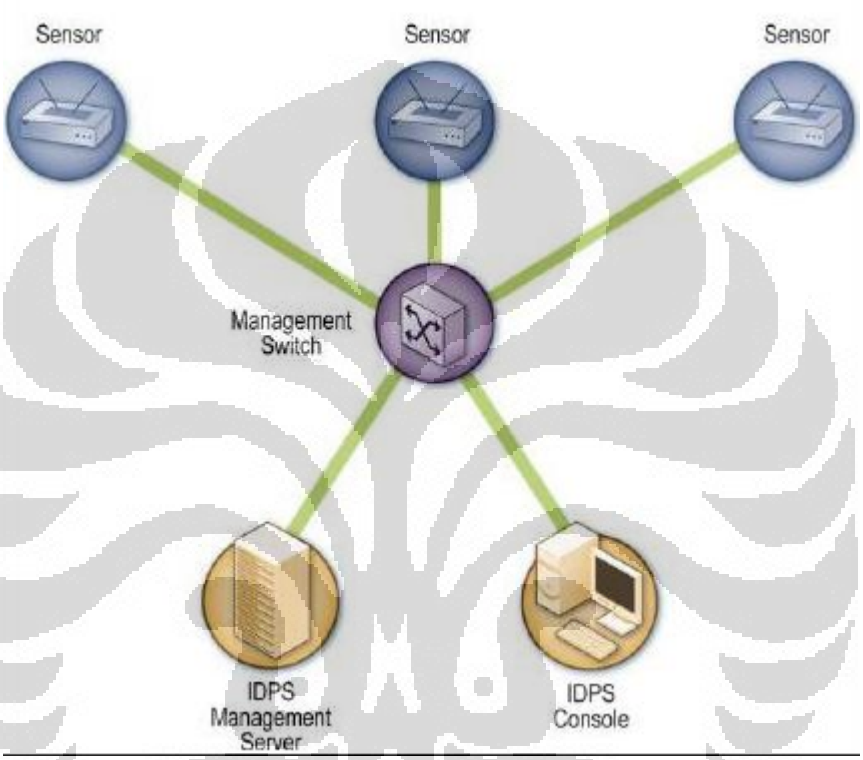
Bab ini berisi penjelasan tentang sistem IDPS, basis data snort, implementasi IPS, perangkat pengujian, web pengujian, skenario pengujian, implementasi pengujian, analisa.

BAB 5 KESIMPULAN.

Bab ini berisi tentang kesimpulan dari apa yang dibahas dalam penulisan tesis ini.

BAB 2
INTRUSION DETECTION PREVENTION SYSTEM (IDPS)

2.1. Intrusion Detection Prevention System (IDPS).



Gambar 2.1 Sistem IDPS

Sebuah *intrusion detection prevention system* (IDPS) memantau lalu lintas jaringan dan menganalisa protokol jaringan tersebut untuk mengidentifikasi aktifitas yang mencurigakan. IDPS secara khusus merekam informasi yang berhubungan dengan objek yang diamati, memberitahukan tentang keamanan dari objek yang diamati kepada *administrator*, dan menghasilkan sebuah *report* atau laporan. Banyak IDPS juga dapat merespon sebuah ancaman dengan cara menanggapinya untuk mencegahnya agar tidak mengganggu sistem. Mereka menggunakan beberapa teknik untuk merespon ancaman atau gangguan, diantaranya IDPS mampu menghentikan ancaman terhadap dirinya sendiri,

perubahan terhadap keamanan sistem (seperti rekonfigurasi sebuah *firewall*), atau perubahan isi dari ancaman/gangguan.

Intrusion detection adalah proses pemantauan setiap kejadian yang terjadi pada sebuah sistem komputer atau jaringan komputer dan menganalisisnya untuk memberi tanda atas setiap kemungkinan kejadian yang mungkin terjadi, seperti pelanggaran atau ancaman terhadap kebijakan keamanan sistem komputer, yang ditangani dengan menggunakan kebijakan atau praktek-praktek standar keamanan komputer.

Intrusion detection and prevention system adalah proses yang menampilkan *intrusion detection* dan usaha-usaha untuk menghentikan kemungkinan-kemungkinan yang bisa terjadi setelah proses deteksi. IDPS utamanya fokus pada identifikasi gangguan atau ancaman yang mungkin terjadi, memberikan informasi tentang *logging information* dari gangguan atau ancaman tersebut, usaha-usaha untuk menghentikan gangguan atau ancaman tersebut, dan melaporkan gangguan atau ancaman tersebut kepada pengatur jaringan. Selanjutnya organisasi atau perusahaan juga menggunakan IDPS untuk berbagai keperluan, seperti mengidentifikasi masalah tentang kebijakan keamanan komputer, pendokumentasian ancaman yang terjadi, dan menghalangi pribadi atau individu dari pelanggaran terhadap kebijakan keamanan jaringan komputer. IDPS menjadi hal yang penting digunakan pada infrastruktur keamanan jaringan komputer pada setiap organisasi atau perusahaan.

Sebuah IDS adalah perangkat lunak yang mengotomatisasi proses *intrusion detection*. Sebuah IPS adalah perangkat lunak yang memiliki semua kemampuan untuk mendeteksi *intrusion* terhadap sistem dan juga melakukan pencegahan terhadap semua ancaman atau gangguan yang mungkin terjadi. Teknologi IDS dan IPS menawarkan banyak kemampuan secara bersama-sama, dan pengatur jaringan biasanya dapat menonaktifkan fasilitas *prevention* pada produk IPS, pada saat alat tersebut difungsikan sebagai sebuah IDS.

Terdapat beberapa jenis dari teknologi IDPS, dimana terdapat perbedaan utama berdasarkan ancaman yang dapat dikenali dan metodelogi yang digunakan untuk mengidentifikasi ancaman Selain pemantauan dan analisa kejadian untuk mengidentifikasi aktivitas yang tidak diinginkan, semua jenis teknologi IDPS

biasanya melakukan fungsi-fungsi berikut [7]:

1. Merekam peristiwa yang berkaitan dengan peristiwa yang sedang diamati. Informasi biasanya direkam secara lokal, dan juga dapat dikirimkan ke sistem yang terpisah seperti *logging server* yang terpusat, *security information* and *event management (SIEM) solution*, dan sistem manajemen perusahaan.
2. Memberitahukan administrator keamanan terhadap hal-hal penting dari peristiwa yang diamati. Pemberitahuan ini dikenal sebagai sebuah peringatan, dapat dilakukan dengan beberapa cara, seperti hal-hal sebagai berikut: email, pages, pesan-pesan pada IDPS *user interface*, SNMP, dll. Sebuah jenis pesan pemberitahuan hanya terdiri dari informasi dasar yang berhubungan dengan sebuah peristiwa; *administrator* membutuhkannya untuk mengakses IDPS sebagai informasi tambahan.
3. Menghasilkan laporan. Laporan menyimpulkan pemantauan terhadap peristiwa atau menyediakan detail yang menyangkut peristiwa tertentu.

Beberapa IDPS juga dapat merubah profil keamanannya pada saat sebuah ancaman/gangguan baru terdeteksi. Sebagai contoh, sebuah IDPS mampu mengumpulkan informasi yang lebih detail untuk sebuah sesi tertentu setelah aktifitas illegal terdeteksi. Sebuah IDPS juga dapat merubah setting pada saat peringatan tertentu dipicu atau prioritas tertentu memberi tanda ke sub peringatan setelah sebuah ancaman tertentu terdeteksi.

2.2. Komponen dan Arsitektur IDPS.

Komponen pada IDPS nirkabel adalah: *consoles*, *database servers* (optional), *management servers*, dan sensor [7].

1. *Consoles.* Sebuah *consoles* adalah sebuah program yang menyediakan sebuah antarmuka untuk pengguna dan administrator IDPS. Perangkat lunak *consoles* secara khusus diinstal ke komputer desktop atau komputer laptop. Beberapa *consoles* digunakan hanya untuk mengatur atau administrasi IDPS, seperti mengkonfigurasi sensor atau *agent* dan melakukan *updates* perangkat lunak, sedangkan *consoles* yang lainnya digunakan hanya untuk pemantauan dan analisis. Beberapa *consoles* IDPS juga menyediakan kemampuan untuk administrasi dan pemantauan secara sekaligus.

2. *Database Server.* Sebuah *database server* adalah sebuah gudang untuk rekaman informasi kejadian yang direkam oleh sensor, *agent*, dan atau *management servers*. Banyak IDPS menyediakan fasilitas dukungan untuk *database servers*.
3. *Management Server*. Sebuah *management server* adalah sebuah perangkat terpusat yang menerima informasi dari sensor atau *agent* dan mengaturnya. Beberapa *management server* menampilkan analisis informasi peristiwa atau kejadian yang dihasilkan oleh sensor atau *agent* dan dapat mengenali peristiwa atau kejadian yang tidak dapat dilakukan oleh sensor atau *agent.* Mencocokkan informasi kejadian dari berbagai sensor atau agent, seperti menemukan peristiwa yang dipicu oleh IP *address* yang sama, dikenal sebagai *correlation*. *Management software* tersedia baik dalam bentuk peralatan maupun hanya dalam bentuk produk perangkat lunak saja. Beberapa IDPS skala kecil tidak menggunakan *management server*, tetapi kebanyakan IDPS menggunakannya. Pada implementasi IDPS skala besar, biasanya terdapat beberapa *management server*, dan dalam beberapa kasus ada yang menggunakan *management server* dua tingkat.
4. *Sensor or Agent.* Sensor dan *agent* memantau dan menganalisa aktifitas. Istilah sensor secara khusus digunakan untuk IDPS yang memonitor jaringan, termasuk jaringan dasar, nirkabel, dan menganalisis karakter atau kebiasaan dari jaringan. Sedangkan istilah *agent* khusus digunakan untuk teknologi IDPS *host based*.

2.3. Kapasitas Keamanan pada IDPS.

IDPS nirkabel menyediakan beberapa kemampuan keamanan, yang kita bagi dalam empat kategori, yaitu: *information gathering*, *logging*, *detection*, dan *prevention* [7].

1. Kemampuan Mengumpulkan Informasi.

   Sebagian besar IDPS dapat mengumpulkan informasi dari perangkat nirkabel. Contoh dari kemampuan pengumpulan informasi adalah sebagai berikut:

   a. Mengidentifikasi perangkat LAN. Sebagian sensor IDPS dapat membuat dan merawat sebuah daftar perangkat yang diamati, termasuk router, klien LAN. Daftar tersebut biasanya dibuat berdasarkan MAC *address* dari kartu

jaringan perangkat tersebut, bagian pertama dari setiap MAC *address* menunjukkan identitas vendor dari kartu tersebut. Beberapa sensor juga dapat menggunakan teknik *finger printing* untuk mengamati lalu lintas untuk memverifikasi vendor kartu tersebut, bukan mengandalkan pada informasi MAC. Daftar tersebut juga dapat digunakan sebagai profil untuk menunjukkan perangkat LAN yang baru dan menghapus perangkat yang telah ada.

b. Mengidentifikasi LAN. Sebagian besar sensor IDPS tetap terus mengamati kondisi LAN, mengidentifikasi jaringan tersebut. Administrator dapat kemudian memberi tanda setiap yang masuk menjadi anggota LAN yang sah, LAN tetangga yang tidak berbahaya, atau sebuah perangkat *rogue* WLAN. Informasi ini dapat digunakan untuk mengenali LAN yang baru serta untuk memberikan prioritas respon dan mengenali kejadian.

2. Kemampuan *Logging*.

IDPS secara khusus melakukan data *logging* yang terkait dengan peristiwa atau kejadian yang terdeteksi. Data ini dapat digunakan untuk mencocokkan keabsahan tanda peringatan, untuk menyelidiki insiden yang terjadi, dan untuk menghubungkan peristiwa antara IDPS dan sumber *logging* yang lainnya. Data-data umum yang dicatat dalam log IDPS nirkabel meliputi: *timestamp* (biasanya tanggal dan waktu), peristiwa atau jenis peringatan, peristiwa atau tingkat kerusakan, MAC *address* sumber (dapat digunakan untuk mengidentifikasi vendor), nomor *channel*, ID sensor dari peristiwa yang diamati, tindakan pencegahan yang dilakukan (jika ada).

3. Kemampuan Deteksi.

IDPS dapat mendeteksi serangan, konfigurasi yang kurang baik, dan pelanggaran kebijakan pada level protokol WLAN, terutama protokol komunikasi IEEE 802.11a, b, g, dan i. Beberapa IDPS nirkabel tidak menguji komunikasi pada level yang lebih tinggi, seperti pada alamat IP dan aplikasi yang dimuat. Beberapa produk hanya menampilkan deteksi sederhana dengan teknik *signature based*, sementara yang lainnya menggunakan kombinasi dari teknik deteksi *signature based*, *anomaly based*, dan *stateful protocol analysis*,

organisasi menggunakan produk IDPS nirkabel yang menggunakan berbagai kombinasi dari teknik-teknik tersebut, untuk

mendapatkan akurasi deteksi yang lebih akurat. Pada bagian ini kita coba untuk membahas aspek-aspek dari kemampuan deteksi, yaitu: jenis peristiwa atau kejadian yang dapat terdeteksi, tingkat akurasi deteksi, *tuning* dan *customization*.

a. Jenis peristiwa atau kejadian yang dapat terdeteksi. Jenis dari peristiwa yang pada umumnya dapat terdeteksi oleh sensor IDPS, diantaranya adalah sebagai berikut:
   1. Perangkat LAN yang tidak sah. Melalui kemampuan mengumpulkan informasi sebagian besar sensor IDPS nirkabel dapat mendeteksi adanya *rogue* AP, STA yang tidak sah, dan WLAN yang tidak sah, baik dengan mode infrastruktur maupun mode *ad hoc*.
   2. Perangkat LAN dengan tingkat keamanan yang rendah. Sebagian sensor IDPS dapat mengidentifikasi AP dan STA yang tidak memiliki kontrol keamanan yang tepat. Hal ini termasuk mendeteksi konfigurasi yang tidak benar dan penggunaan protokol LAN yang lemah. Hal ini dilakukan dengan mengidentifikasi penyimpangan dari spesifikasi kebijakan organisasi untuk mengatur beberapa hal seperti: enkripsi, otentifikasi, *data rate*, nama SSID, dan *channel*. Sebagai contoh, sebuah sensor akan mendeteksi bahwa sebuah STA menggunakan WEP dan bukan jenis WPA2 atau IEEE 802.11i. Secara umum jenis dari peristiwa yang seperti ini dapat dideteksi oleh IDPS nirkabel.
   3. Pola penggunaan yang tidak biasa. Beberapa sensor dapat menggunakan metode deteksi *anomaly based* untuk mendeteksi pola penggunaan LAN yang tidak biasa. Sebagai contoh, jika terlalu banyak STA daripada biasanya yang mengakses AP tertentu, atau terdapat lebih banyak jumlah lalu lintas jaringan dari biasanya antara sebuah STA dan AP, salah satu perangkat yang mungkin telah diketahui atau pihak-pihak yang tidak sah mungkin menggunakan LAN. Banyak sensor dapat menunjukkan usaha-usaha yang gagal untuk bergabung ke LAN, seperti peringatan pada beberapa kegagalan usaha dalam periode pendek dari waktu, dimana dapat menunjukkan sebuah

usaha untuk mendapatkan akses yang tidak sah ke LAN. Beberapa sensor juga dapat memberi peringatan jika ada aktifitas LAN yang terdeteksi selama periode *off*.

4. Penggunaan *Scanner* Jaringan. Seperti *scanner* atau pemindai yang digunakan untuk mengidentifikasi LAN yang tidak aman atau yang lemah sistem keamanannya. Sensor IDPS hanya dapat mendeteksi penggunaan aktif *scanner*, *scanner* yang membangkitkan lalu lintas jaringan nirkabel. Sensor tidak dapat mendeteksi penggunaan sensor pasif yang secara sederhana memantau dan menganalisis lalu lintas yang diamati.
5. Kondisi dan Serangan *Denial of Service* (DoS), seperti interferensi pada jaringan. Serangan DoS meliputi serangan secara logika seperti *flooding*, yang melibatkan sejumlah besar pesan ke sebuah AP dengan kecepatan tinggi, dan serangan fisik seperti *jamming*, yang melibatkan emisi energi elektromagnetik pada frekuensi WLAN untuk membuat frekuensi yang tidak dapat digunakan oleh WLAN. Serangan DoS sering dapat dideteksi dengan metode analisis *stateful protocol* dan deteksi *anomaly*, yang dapat ditentukan jika aktifitas yang diamati konsisten secara terus menerus menampakkan aktifitas yang tidak diharapkan. Banyak serangan DoS terdeteksi dengan menghitung lama periode dari waktu dan sinyal pada saat nilai ambang batas telah terlampaui. Contohnya, sejumlah besar dari kejadian yang mengakibatkan berhentinya sebuah sesi dari jaringan nirkabel dapat diindikasikan sebagai serangan DoS.
6. Penyamaran dan *man in the middle attack*. Beberapa sensor IDPS dapat mendeteksi ketika sebuah perangkat mencoba untuk memalsukan identitas dari perangkat lain. Hal ini dapat dilakukan dengan mengidentifikasikan perbedaan karakteristik aktifitas, seperti nilai-nilai tertentu dari *frame*.

Kebanyakan sensor IDPS nirkabel dapat mengenali lokasi fisik dari ancaman yang terdeteksi dengan menggunakan *triangulation*, memperkirakan perkiraan jarak ancaman dari beberapa sensor oleh kekuatan sinyal ancaman yang diterima oleh setiap sensor, kemudian menghitung lokasi fisik dimana ancaman itu akan diperkirakan jaraknya dari setiap sensor. Cara ini membuat sebuah organisasi dapat mengirimkan petugas keamanan ke alamat lokasi untuk mengatasi ancaman

tersebut. Produk IDPS nirkabel juga dapat digunakan untuk mengetahui apakah ancaman itu berasal dari dalam atau luar gedung, atau jika ancaman berasal dari umum atau daerah aman. Informasi ini sangat membantu tidak hanya dalam menemukan atau menghentikan ancaman, namun juga dalam menentukan prioritas dari respon yang harus dilakukan terhadap ancaman. Sensor IDPS nirkabel dapat diatur berdasarkan prioritas dari peringatan berdasarkan lokasi dari setiap ancaman. *Handheld* sensor IDPS juga dapat digunakan untuk menentukan lokasi ancaman, pada sebagian *fixed* sensor tidak menawarkan kemampuan *triangulation* atau jika ancaman tersebut bergerak.

b. Akurasi dari Pendeteksian. Dibandingkan dengan bentuk lain dari IDPS. IDPS nirkabel secara umum lebih akurat, hal ini sebagian besar diakibatkan terbatasnya ruang lingkup, yaitu hanya menganalisis *protocol* jaringan nirkabel. Kesalahan positif seringkali disebabkan oleh adanya metode deteksi *anomaly based*, terutama jika nilai ambang tidak diperbaharui dengan tepat. Demikian pula banyak peringatan terjadi berdasarkan aktifitas yang tidak berbahaya, seperti WLAN yang dimiliki organisasi lain yang masuk dalam jangkauan WLAN organisasi kita, peringatan ini bukan kesalahan positif yang sebenarnya karena mereka dideteksi secara akurat sebagai WLAN yang tidak dikenal dalam fasilitas organisasi.
c. *Tuning* dan *Customization*. Teknologi IDPS nirkabel biasanya membutuhkan beberapa pengaturan dan kustomisasi untuk meningkatkan akurasi dari kemampuan deteksinya. Terutama spesifikasi dari WLAN, AP, dan STA yang sah, dan memasukan karakteristik dari kebijakan kedalam perangkat lunak IDPS nirkabel.
d. Kemampuan Pencegahan. Sensor IDPS menawarkan dua jenis kemampuan pencegahan terhadap intrusi:
    1. Jaringan Nirkabel. Beberapa sensor dapat menghentikan hubungan antara sebuah *rogue devices* atau STA yang tidak dikonfigurasi dengan benar dan sebuah AP yang sah atau antara sebuah STA yang sah dan sebuah *rogue devices* atau AP yang tidak dikonfigurasi dengan benar melalui udara. Hal ini biasanya dilakukan dengan mengirimkan pesan ke *endpoint*,

memberitahu mereka untuk menghentikan sesi yang sedang dilakukan. Sensor kemudian menolak permintaan hubungan baru yang akan dilakukan.

2. Jaringan Kabel. Beberapa sensor dapat menginstruksikan sebuah switch pada jaringan kabel untuk menghentikan aktifitas jaringan yang melibatkan STA tertentu atau AP berdasarkan alamat MAC perangkat atau port switch. Contohnya, jika sebuah STA mengirimkan serangan ke server pada jaringan kabel, sebuah sensor akan memerintahkan switch untuk memblok semua aktifitas ke dan dari STA. Teknik ini hanya efektif untuk memblok STA tidak sah atau komunikasi AP jaringan kabel. Hal ini tidak akaa menghentikan sebuah STA atau AP dari melanjutkan untuk melakukan aktifitas ancaman atau gangguan pada *protocol* nirkabel.

Kebanyakan sensor IDPS memungkinkan administrator untuk menspesifikasi kemampuan pencegahan untuk setiap jenis peringatan atau ancaman. Termasuk mengaktifkan dan menonaktifkan sistem pencegahan, serta menspesifikasikan jenis kemampuan pencegahan yang akan digunakan. Beberapa sensor IDPS memiliki fasilitas belajar atau simulasi yang menekankan berbagai aksi pencegahan, serta memberikan peringatan atau tanda pada saat sebuah aksi pencegahan dilakukan. Hal ini memungkinkan administrator untuk memantau dan memastikan sistem pencegahan terkonfigurasi dengan baik sebelum aktifitas pencegahan dilakukan, sehingga dapat mengurangi resiko pada saat melakukan aktifitas pencegahan diperlukan.

Hal yang perlu diperhatikan adalah akibat dari aksi pencegahan pada sensor pemantauan. Misalnya, jika sebuah sensor mengirimkan sinyal untuk menghentikan hubungan, dia tidak mampu untuk melakukan *channel scanning* untuk memantau saluran komunikasi yang lain sampai dia selesai melakukan aksi pencegahan. Untuk mengatasi hal ini, Beberapa sensor memiliki dua radio, satu untuk pemantauan dan deteksi, dan yang lainnya untuk melakukan aksi pencegahan. Pada saat memilih sensor, organisasi harus memperhatikan aksi pencegahan seperti apa yang diperlukan dan bagaimana kemampuan deteksi dari sensor akan mempengaruhi aktifitas pencegahan.

2.4.Snort

Snort adalah *network intrusion detection system* (NIDS) yang bekerja dengan menggunakan *signature detection*, berfungsi juga sebagai *sniffer* dan *packet logger*. Snort pertama kali di buat dan dikembangkan oleh Marti Roesh, lalu menjadi sebuah opensource project. Versi komersial dari snort dibuat oleh Sourcefire (www.sourcefire.com). Snort memiliki karakteristik, sebagai berikut [12]:

a. Berukuran kecil – Source code dan rules untuk rilis 2.1.1 hanya 2256k.

b. Portable untuk banyak OS – Telah diporting ke Linux, Windows, OSX, Solaris, BSD,dll.

c. Cepat – Snort mampu mendeteksi serangan pada network 100Mbps.

d. Mudah dikonfigurasi – Snort sangat mudah dikonfigurasi sesuai dengan kebutuhan network kita. Bahkan kita juga dapat membuat rule sendiri untuk mendeteksi adanya serangan baru.

e. Free – Kita tidak perlu membayar sepeser pun untuk menggunakan snort. Snort bersifat open source dan menggunakan lisensi *general public licence* (GPL).

1. Komponen Snort.

Snort merupakan *packet sniffing* yang sangat ringan. *Snifing interface* yang digunakan berbasis libpcap (pada Unix tersedia dengan tcpdump, www.tcpdump.org). Pembuat snort sangat fokus pada *engine* yang digunakan untuk mendeteksi serangan dan memanfaatkan *tools* tcpdump untuk mengambil paket *network*. Salah satu keunggulan snort adalah bahwa snort memiliki *plugin* sistem yang sangat fleksibel untuk dimodifikasi.

Snort memiliki beberapa komponen yang tiap komponennya mempunyai tugas masing-masing. Pada saat ada paket *network* yang melewati Ethernet di tempat snort dipasang, maka ada beberapa hal yang dilalui [12]:

a. *Packet capture library* (libpcap).

Packet capture library – akan memisahkan paket data yang melalui *ethernet card* untuk selanjutnya digunakan oleh snort.

b. *Packet decoder*.

Packet decoder – mengambil data di layer 2 yang dikirim dari packet capture library (proses 1). Pertama ia akan memisahkan Data *link* (seperti ethernet,

TokenRing, 802.11) kemudian protokol IP, dan selanjutnya paket TCP dan UDP. Setelah pemisahan data selesai, snort telah mempunyai informasi protokol yang dapat diproses lebih lanjut.

c. *Preprocessor*.

Selanjutnya dilakukan analisis (*preprocessor*) atau manipulasi terhadap paket sebelum dikirim ke *detection engine*. Manipulasi paket dapat berupa ditandai, dikelompokan atau malah dihentikan.

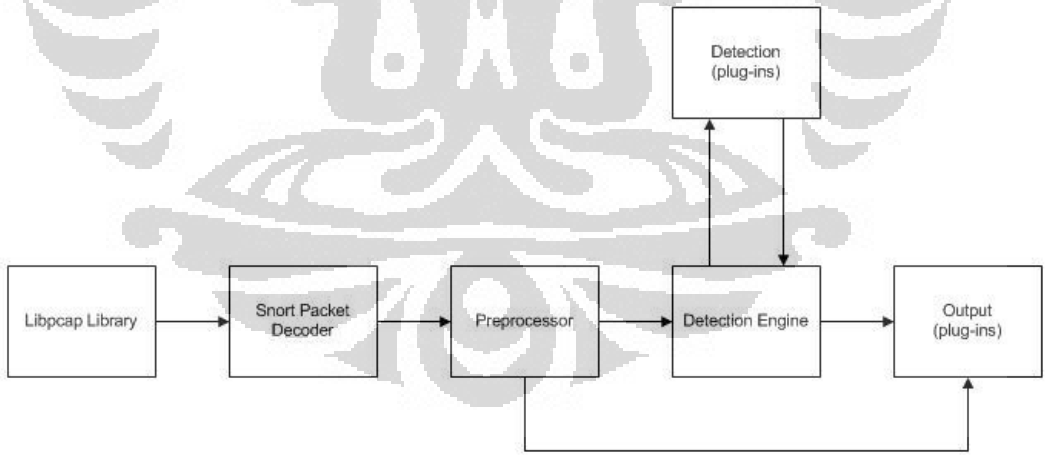
d. *Detection Engine*.

Inilah jantung dari snort. Paket yang datang dari *packet decoder* akan di uji dan dibandingkan dengan *rule* yang telah ditetapkan sebelumnya. *Rule* berisi tanda-tanda (*signature*) yang termasuk serangan.

e. *Output*.

Output yang dihasilkan berupa *report* dan *alert*. Ada banyak variasi *output* yang dihasilkan snort, seperti teks (ASCII), XML, syslog, tcpdump, binary format, atau Database (MySQL, MsSQL, PostgreSQL, dsb).

Diagram komponen snort seperti ditunjukan oleh Gambar 2.2 berikut ini.



Gambar 2.2 Komponen Snort

2. Mengoperasikan Snort.

Secara umum snort dapat dioperasikan dalam tiga macam mode, yaitu [12]:

a. *Sniffer mode*, untuk melihat paket yang lewat di jaringan.
b. *Packet logger mode*, untuk mencatat semua paket yang lewat di jaringan untuk dianalisa di kemudian hari.
c. *Intrusion Detection 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.

Sniffer Mode, untuk menjalankan snort pada mode ini tidaklah sukar, beberapa contoh perintah pada mode ini adalah sebagai berikut.

```
./snort –v
./snort –vd
./snort –vde
./snort –v –d –e
```

Dengan menambahkan beberapa switch –v, -d, -e akan menghasilkan beberapa keluaran yang berbeda yaitu.

-v, untuk melihat header TCP/IP paket yang lewat.
-d, untuk melihat isi paket.
-e, untuk melihat *header link layer* paket seperti *ethernet header*.

Packet Logger Mode, cukup membingungkan untuk melihat paket lewat sedemikian cepat di layar terutama jika kita menggunakan ethernet berkecepatan 100Mbps, layar monitor kita akan terlihat *scrolling* dengan cepat sekali susah untuk melihat paket yang diinginkan. Cara paling sederhana untuk mengatasi hal ini adalah menyimpan dulu semua paket yang lewat ke sebuah file untuk kemudian dilihat untuk dianalisa.

Beberapa perintah yang mungkin dapat digunakan untuk mencatat paket yang lewat, yaitu.

```
./snort –dev –l ./log
./snort –dev –l ./log –h 192.168.0.0/24
./snort –dev –l ./log –b
```

Perintah yang paling penting untuk menangkap log paket yang lewat adalah

```
-l ./log
```

Perintah ini yang menentukan bahwa paket yang lewat akan di log atau di catat ke file ./log. Beberapa perintah tambahan dapat digunakan seperti –h 192.168.0.0/24 yang menunjukan bahwa yang dicatat hanya paket dari host mana saja, dan –b yang memberitahukan agar file yang di log dalam format binary, bukan ASCII.

Untuk membaca file log dapat dilakukan dengan menjalankan snort dengan ditambahkan perintah –r nama file log-nya, seperti.

```
./snort –dv –r packet.log
./snort –dvr packet.log icmp
```

Intrusion Detection mode, mode ini merupakan mode operasi snort yang paling rumit, yaitu sebagai pendeteksi penyusup di jaringan yang kita gunakan. Ciri khas mode operasi untuk pendeteksi penyusup adalah dengan menambahkan perintah ke snort untuk membaca file konfigurasi –c nama-file-konfigurasi.conf. Isi file konfigurasi ini lumayan banyak, tapi sebagian besar telah di set secara baik dalam file snort.conf yang dibawa oleh source snort.

Beberapa contoh perintah untuk mengaktifkan snort untuk melakukan pendeteksian penyusup, seperti.

```
./snort –dev –l ./log –h 192.168.0.0/24 –c snort.conf
./snort –d –h 192.168.0.0/24 –l ./log –c snort.conf
```

Untuk melakukan deteksi penyusup secara prinsip snort harus melakukan logging paket yang lewat dapat menggunakan perintah –l nama-file-logging, atau membiarkan snort menggunakan *default* file logging-nya di direktori

/var/log/snort. Kemudian menganalisa catatan atau logging paket yang ada sesuai dengan isi perintah snort.conf.

Ada beberapa tambahan perintah yang akan membuat proses deteksi menjadi lebih efisien, mekanisme pemberitahuan alert di Linux dapat di set dengan perintah –A sebagai berikut.

-A fast, mode alert yang cepat berisi waktu, berita, IP dan port tujuan.
-A full, mode alert dengan informasi lengkap.
-A unsock, mode alert ke unix socket
-A none, mematikan mode alert.

Agar snort beroperasi secara langsung setiap kali *worksation* atau *server* di boot, kita dapat menambahkan ke file /etc/rc.d/rc.local perintah sebagai berikut.

/usr/local/bin/snort –d –h 192.168.0.0/24 –c /root/snort/snort.conf –A full –s -D

Atau

/usr/local/bin/snort –d –c /root/snort/snort.conf –A full –s –D

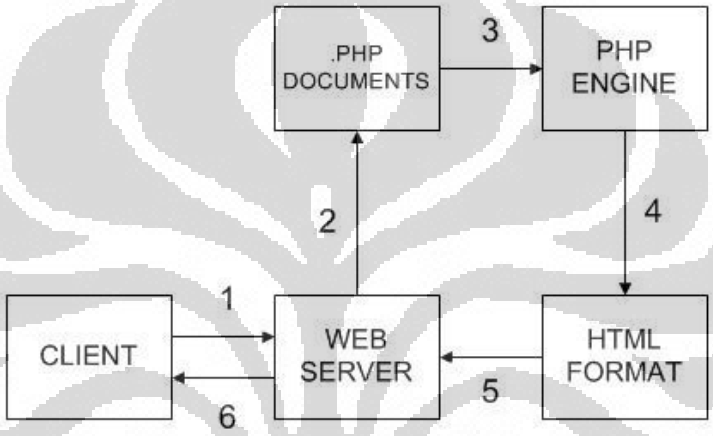
Pada perintah di atas –D adalah switch yang menset agar snort bekerja sebagai Daemon (bekerja dibelakang layar).

Bagi pengguna yang hendak menggunakan snort untuk melindungi jaringan komputer mereka, akan lebih nudah jika kita juga menggunakan *Analysis Console for Intrusion Databases* (ACID). ACID menggunakan PHPlot, sebuah *library* untuk membuat grafik yang baik di PHP, dan ADODB, sebuah *library* abstraksi untuk menggabungkan PHP ke berbagai database seperti MySQL dan PostgreSQL.

2.5 Hypertext Preprocessor (PHP).

PHP adalah bahasa *server-side scripting* yang menyatu dengan *Hypertext Markup Language* (HTML) untuk membuat halaman *web* yang dinamis. Maksud dari *server-side scripting* adalah sintaks dan perintah-perintah yang diberikan akan sepenuhnya dijalankan di server tetapi disertakan pada dokumen HTML.

Server-side scripting, adalah sebuah bahasa pemrograman yang digunakan untuk mengatur alur kerja dari sebuah program atau aplikasi lain. Dalam hal ini skrip yang dibuat dengan menggunakan PHP akan digunakan untuk mengatur bagaimana *web server* memproses masukan dan memberikan hasil yang diinginkan oleh pengguna *web*. Prosesnya seperti ditunjukan pada Gambar 2.3 berikut ini.



Gambar 2.3 Struktur pembacaan *Web Server*

Dalam penggunaannya, PHP akan menyisipkan instruksi-instruksi khusus di dalam sebuah halaman *web* untuk mengatur alur prosesnya. Apabila halaman *web* tersebut diminta oleh sebuah *web browser*, *web server* akan terlebih dahulu menjalankan instruksi-instruksi tersebut sebelum memberikan halaman tersebut kepada *web browser*.

Ketika seorang pengguna Internet akan membuka suatu situs yang menggunakan fasilitas *server-side scripting* PHP, maka terlebih dahulu server yang bersangkutan akan memproses semua perintah PHP di server lalu mengirimkan hasilnya dalam format HTML ke *web* browser pengguna Internet. Dengan demikian seorang pengguna Internet tidak dapat melihat kode program yang ditulis dalam PHP sehingga keamanan dari halaman *web* menjadi lebih terjamin.

Instruksi-instruksi pemrosesan yang dapat dilakukan oleh PHP sangatlah beragam, seperti memproses masukan data dari formulir, menghubungkan dan mengambil data dari basis data, melakukan manipulasi terhadap data yang terdapat di dalam basis data, menentukan apakah seorang pengguna memiliki hak untuk mengakses sebuah halaman.

Kelebihan yang lain PHP merupakan perangkat lunak *free software* dan *open source* dan mampu bekerja lintas *platform*, yaitu dapat digunakan dengan sistem operasi dan *web server* apapun. PHP mampu bekerja pada sistem operasi Windows dan Linux. PHP juga dapat dibangun sebagai modul pada *web server* Apache dan sebagai *binary* yang dapat berjalan sebagai *Common Gateway Interface* (CGI).

PHP dapat mengirim HTTP *header*, dapat mengeset *cookies*, mengatur *authentication* dan *redirect users*. PHP juga menawarkan koneksitas yang baik dengan beberapa basis data, antara lain Oracle, Sybase, MySQL, PostgreSQL, dan semua basis data yang menggunakan *interface* ODBC.

PHP juga dapat berintegrasi dengan beberapa *library* eksternal yang membuat mudah untuk melakukan segalanya mulai dari membuat dokumen PDF hingga mem-*parse* XML.

PHP juga mendukung komunikasi dengan layanan lain melalui protokol IMAP, SNMP, NNTP, POP3 atau bahkan HTTP. Bila PHP berada dalam halaman *web*, maka tidak lagi dibutuhkan pengembangan lingkungan khusus atau direktori khusus. Hampir seluruh aplikasi berbasis *web* dapat dibuat dengan PHP. Namun kekuatan utama adalah konektifitas basis data dengan *web*. Dengan kemampuan ini kita akan mempunyai suatu sistem basis data yang dapat diakses dari *web*.

2.6. MySQL.

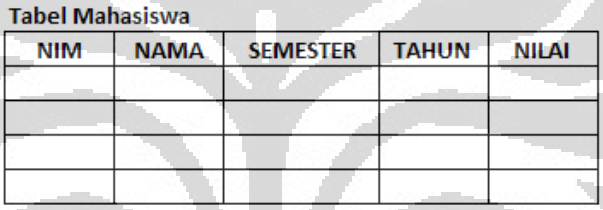
MySQL adalah basis data *multiuser* yang menggunakan bahasa *Structured Query Language* (SQL). MySQL dalam operasi *client-server* melibatkan *server daemon* MySQL di sisi *server* dan berbagai macam program serta *library* yang berjalan di sisi *client*.

SQL adalah bahasa standar yang digunakan untuk mengakses *server* basis data. Bahasa ini pada awalnya dikembangkan oleh IBM, namun telah diadopsi dan

digunakan sebagai standar industri. Dengan menggunakan SQL, proses akses basis data menjadi lebih mudah.

Secara umum basis data (*database*) merupakan kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya.

Dalam konteks bahasa SQL, pada umumnya informasi tersimpan dalam tabel-tabel yang secara logika merupakan struktur dua dimensi yang terdiri atas baris-baris data yang berada dalam satu atau lebih kolom. Baris pada tabel sering disebut sebagai *instance* atau *record* dari data, sedangkan kolom sering disebut sebagai *attribut* atau *field*. Bentuk tabel seperti yang digambarkan seperti contoh pada Gambar 2.4 berikut ini.

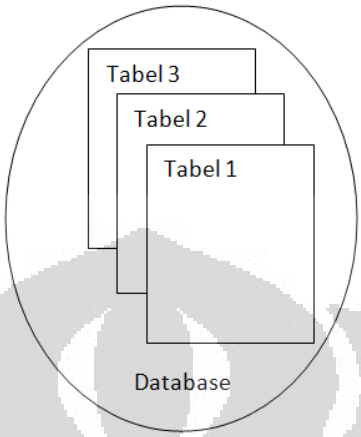


Gambar 2.4 Contoh tabel dalam Basis Data

Pada Gambar 2.4 di atas diperlihatkan sebuah tabel mahasiswa yang memiliki lima buah kolom atau yang biasa dalam istilah basis data disebut dengan lima buah *field*, yaitu: NIM, NAMA, SEMESTER, TAHUN, NILAI. Sedangkan data-data yang dimasukan dalam setiap baris pada tabel mahasiswa tersebut disebut dengan istilah *record*.

Keseluruhan tabel itu dihimpun dalam satu kesatuan yang sering kita sebut dengan istilah basis data (*database*).

Hubungan antara tabel-tabel yang menyusun sebuah basis data dapat diperlihatkan seperti pada Gambar 2.5 berikut ini.



Gambar 2.5 Hubungan antara Tabel dan Basis Data

2.7 Diagram Arus Data (DAD).

Pada tahap analisa, digunakan notasi Diagram Arus Data (DAD). Penggunaan notasi ini sangat membantu sekali di dalam komunikasi dengan pemakai sistem untuk memahami sistem secara logika.

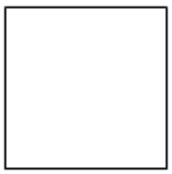
DAD sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertimbangkan lingkungan fisik dimana data tersebut mengalir (misalnya lewat telephone, surat, dan sebagainya) atau lingkungan fisik dimana data tersebut akan disimpan (misalnya *hard disk*, DVD room, dan lain sebagainya).

Beberapa simbol digunakan untuk melakukan analisa dengan metode DAD, yaitu [1]:

a. *External entity* (kesatuan luar) atau *boundary* (batas sistem).

Setiap sistem pasti mempunyai batas sistem (*boundary*) yang memisahkan suatu sistem dengan lingkungan luarnya. Sistem akan menerima input dan menghasilkan output kepada lingkungan luarnya. Kesatuan luar (*external entity*) merupakan kesatuan (*entity*) di lingkungan luar sistem yang dapat berupa orang, organisasi, atau sistem lainnya yang berada di lingkungan luarnya yang akan memberikan input atau menerima output dari sistem.

Simbol dari kesatuan luar seperti ditunjukan pada Gambar 2.6 berikut ini.



Gambar 2.6 Simbol kesatuan luar

b. *Data flow* (arus data).

Arus data di DAD diberi simbol suatu panah. Arus data ini mengalir diantara proses, simpanan data, dan kesatuan luar. Arus data ini menunjukan arus dari data yang dapat berupa masukan untuk sistem atau hasil dari proses sistem.

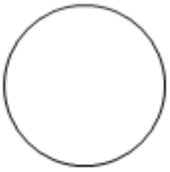
Simbol dari arus data seperti ditunjukan pada Gambar 2.7 berikut ini.

Gambar 2.7 Simbol arus data

c. Process (proses).

Suatu proses adalah kegiatan atau kerja yang dilakukan oleh orang, mesin, atau komputer dari hasil suatu arus data yang masuk ke dalam proses untuk dihasilkan arus data yang akan keluar dari proses.

Simbol dari proses seperti ditunjukan pada Gambar 2.8 berikut ini.



Gambar 2.8 Simbol proses

d. Data store (simpanan data).

Simpanan data merupakan simpanan dari data, yang dapat berupa:

1. Suatu file atau database di sistem komputer.
2. Suatu arsip atau catatan manual.
3. Suatu tabel acuan manual.

Simbol dari simpanan data seperti ditunjukan pada Gambar 2.9 berikut ini.

Gambar 2.9 Simbol simpanan data

2.8 Zenmap.

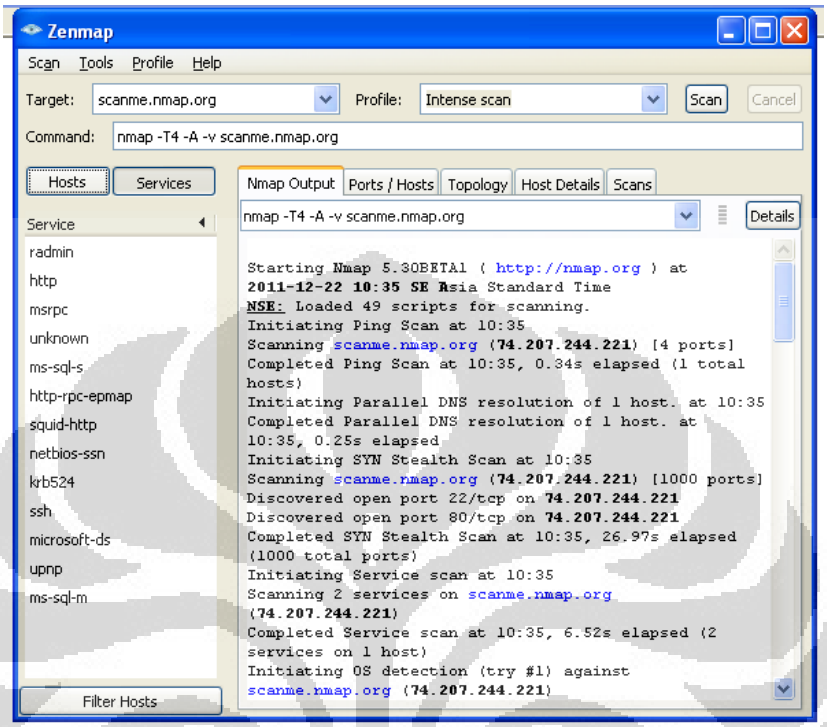
Para administrator jaringan tentu sudah tidak asing lagi dengan perangkat lunak Nmap, adalah singkatan dari "*network mappers*" yang berarti pemeta jaringan. fungsi utama dari Nmap adalah scanning terhadap suatu jaringan, mengidentifikasi service atau aplikasi apa saja yang berjalan di komputer korban, lalu mencetaknya.

Kini Nmap sudah hadir dalam format GUI yang bernama Zenmap, Zenmap mempunyai semua kemampuan Nmap. kelebihan Zenmap adalah menampilkan report dalam bentuk gambar yang tentunya lebih mudah untuk dimengerti.

Sebuah program pemindai sebenarnya adalah memindai port TCP/IP, yaitu sebuah program yang menyerang port TCP/IP dan servis-servisnya (telnet, ftp, http, dan lain-lain) dan mencatat respon dari komputer target. Dengan cara ini, pengguna program pemindai dapat memperoleh informasi yang berharga dari host yang menjadi target.

Teknik-teknik yang dapat digunakan untuk memindai port dari host yang dituju ada berbagai macam cara, yaitu: TCP *connect() scanning* (-sT). Adalah teknik yang paling dasar dalam TCP scanning, dengan menggunakan teknik ini, scanning dilakukan setelah 3-*way handshaking* terjadi, 3-*way handshaking* terjadi

setelah melewati proses-proses seperti berikut: Paket SYN dikirimkan dari terminal yang ingin melakukan scanning ke terminal tujuan.



Gambar 2.10 Tampilan Program Zenmap

Paket SYN|ACK dikirimkan oleh terminal tujuan sebagai balasan dari paket SYN yang diterimanya. Paket ACK dikirimkan oleh terminal yang ingin melakukan scanning sebagai balasan dari paket SYN|ACK yang diterimanya.

Program pemindai seperti Nmap atau Zenmap memiliki beberapa kemampuan memindai yang berbeda-beda, seperti [13]:

a. *Intense Scan*

Perintah = nmap -T4 -A -v

Melakukan pemindaian secara intensif dan menyeluruh. Meliputi pendeteksian sistem operasi, versi dari sistem operasi, pemindaian skrip, dan traceroute. Tanpa akses ke root tujuan, hanya untuk melakukan pemindaian, aktifitas ini termasuk dalam pemindaian yang bersifat mengganggu.

b. *Intense scan plus UDP*

Perintah = nmap -sS -sU -T4 -A -v

Melakukan pendeteksian sistem operasi (-O), pendeteksian versi (-sV), pemindaian skrip (-sC), dan traceroute (--traceroute) ditambah dengan pemindaian terhadap port TCP dan UDP.

c. *Intense scan, all TCP ports*

Perintah = nmap -p 1-65535 -T4 -A -v

Pemindaian terhadap seluruh port TCP, kemudian melakukan pendeteksian sistem operasi (-O), pendeteksian versi (-sV), pemindaian skrip (_sC), dan traceroute (--traceroute).

d. *Intense scan, no ping*

Perintah = nmap -T4 -A -v -Pn

Melakukan pemindaian secara intensif tanpa melakukan pemeriksaan apakah target aktif atau tidak. Pemindaian ini dapat berguna pada saat sebuah target tampaknya mengabaikan probe pencarian host biasa.

e. *Ping scan*

Perintah = nmap -sn

Pemindaian jenis ini hanya mencari target yang aktif dan tidak melakukan pemindaian terhadap port.

f. *Quick scan*

Perintah = nmap -T4 -F

Pemindaian jenis ini lebih cepat daripada pemindaian normal karena hanya dilakukan dalam waktu yang cepat dan pemindaian terhadap beberapa port saja.

g. *Quick scan plus*

Perintah = nmap -sV -T4 -O -F --version-light

Melakukan pemindaian hanya terhadap sistem operasi dan versinya.

h. *Quick traceroute*

Perintah = nmap -sn --traceroute

Melakukan penelusuran terhadap path target tanpa melakukan pemindaian terhadap port secara lengkap.

i. *Regular scan*

Perintah = nmap

Melakukan pemindaian dasar tanpa ada perintah tambahan.

j. *Slow comprehensive scan*

Perintah = nmap -sS -sU -T4 -A -v -PE -PS80,443 -PA3389 -PP -PU40125 -PY --source-port 53 --script all

Melakukan pemindaian secara menyeluruh, pemindaian secara lambat. Melakukan pemindaian terhadap setiap port TCP dan UDP. kemudian melakukan pendeteksian sistem operasi (-O), pendeteksian versi (-sV), pemindaian skrip (_sC), dan traceroute (--traceroute). Melakukan probing terhadap banyak host. Termasuk dalam pemindaian yang sangat mengganggu.

2.9 Aktifitas "*Hacking*".

Hacking secara umum merupakan kegiatan yang dilakukan seseorang mencakup bidang komputer dan teknologi dimana ia mencoba dan berusaha masuk ke sebuah sistem komputer orang lain, dengan demikian ia dapat hak akses untuk mengendalikannya.

Berikut ini adalah langkah-langkah yang biasa dilakukan dalam proses *hacking* [2]:

a. *Footprinting*. Mencari rincian informasi terhadap sistem-sistem untuk dijadikan sasaran, mencakup pencarian informasi dengan *search engine*, *whois,* dan *DNS zone transfer*.
b. *Scanning.* Terhadap sasaran tertentu dicari pintu masuk yang paling mungkin. Digunakan *ping sweep* dan *port scan*.
c. *Enumeration*. Meneliti sasaran secara intensif ,yang mencari *user account* absah, *network resource and share* dan aplikasi untuk mendapatkan mana yang proteksinya lemah.
d. *Gaining Access*. Mendapatkan Akses Mendapatkan data lebih banyak lagi untuk mulai mencoba mengakses sasaran. Meliputi mengintip dan merampas *password*, menebak *password*, serta melakukan *buffer overflow*.
e. *Escalating Privilege*. Menemukan Akses Khusus (Root) Bila baru mendapatkan *user password* di tahap sebelumnya, di tahap ini diusahakan

mendapat *priviledge* admin jaringan dengan *password cracking* atau *exploit* sejenis getadmin, sechole, atau lc_messages.

f. *Pilfering*. Proses pengumpulan informasi dimulai lagi untuk identifikasi mekanisme untuk mendapatkan akses ke *trusted system*. Mencakup evaluasi *trust* dan pencarian *cleartext password* di *registry*, *config file*, dan *user* data.

g. Menutupi Jejak (*Covering Tracks*). Begitu kontrol penuh terhadap sistem diperoleh, maka menutup jejak menjadi prioritas. Meliputi membersihkan *network log* dan penggunaan *hide tool* seperti macam-macam *rootkit* dan *file streaming.*

h. Membuat Pintu Belakang (*Creating Backdoors*). Pintu belakang diciptakan pada berbagai bagian dari sistem untuk memudahkan masuk kembali ke sistem ini dengan cara membentuk *user account* palsu,menjadwalkan *batch job*, mengubah *startup file*, menanamkan service pengendali jarak jauh serta *monitoring tool*, dan menggantikan aplikasi dengan *trojan.*

i. *Denial of Service* (DOS), Bila semua usaha di atas gagal, penyerang dapat melumpuhkan sasaran sebagai usaha terakhir Meliputi SYN flood, teknik-teknik ICMP, Supernuke,land/latierra, teardrop, bonk, newtear, trincoo, trini00, dan lain-lain.

j. *Phising*. Sebuah tindakan memperoleh informasi pribadi seperti *User ID*, *password*, PIN, nomor rekening bank, nomor kartu kredit Anda secara ilegal.

k. *Keylogger*. Sebuah mesin atau software yang dipasang atau diinstal di komputer agar mencatat semua aktivitas yang terjadi pada keyboard (bekerja diam diam alias tidak terketahui oleh kita secara kasat mata)

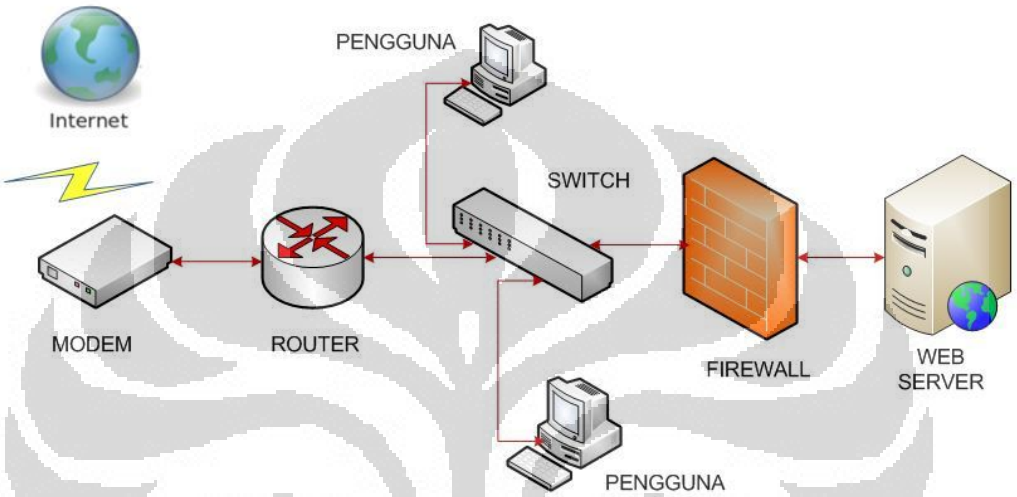
l. *Fake Login*. Halaman tiruan/palsu yang dibuat untuk mengelabui user, bertujuan untuk mencuri informasi penting dari user (eg. username, password, email). Seperti pada kasus pencurian email n password Friendster, Facebook, dll.

m. *Fake Process*. Proses tiruan yang dibuat untuk menyembunyikan nama proses asli, bertujuan untuk mengelabui admin sistem. Seperti mem-fake "./backdoor" menjadi "usr/sbin/httpd", sehingga ketika di "ps -ax", proses "./backdoor" berubah menjadi "usr/sbin/httpd".

n. *Malicious Code*. Kode yg dibuat untuk tujuan jahat atau biasa disebut kode jahat.
o. Virus, Kode jahat yg sistim kerjanya seperti virus pada manusia, menggandakan diri dan seperti parasit menopang pada file yg diinfeksinya. File yg terinfeksi menjadi rusak atau ukurannya bertambah. Sekarang kode jenis ini akan sangat mudah terdeteksi pada aplikasi yg memeriksa crc32 dari dirinya.
p. *Worm* (Cacing), Kode jahat yg sistim kerjanya seperti cacing, menggandakan diri dan menyebar, tidak menopang pada file. Kebanyakan di Indonesia adalah kode jenis ini.
q. *Trojan* (*Horse*), Kode jahat yg sistim kerjanya seperti kuda trojan pada zaman kerajaan Romawi, masuk ke dalam sistem untuk mengintip dan mencuri informasi penting yg ada didalamnya kemudian mengirimnya kepada pemilik trojan.

BAB 3
PERANCANGAN SISTEM IDPS

3.1. Jaringan Komputer.



Gambar 3.1 Topologi Jaringan Komputer

Pada Gambar 3.1 diatas diperlihatkan sebuah jaringan komputer dari sebuah perusahaan atau kantor yang memiliki fasilitas server yang mereka miliki, yaitu: *web server*. *Web server* digunakan oleh perusahan atau kantor sebagai media untuk publikasi dan berinteraksi dengan masyarakat yang membutuhkan layanan informasi yang dimiliki oleh perusahaan atau kantor tersebut. Mengingat pentingnya data dan informasi yang terdapat dalam jaringan kantor tersebut, terutama pada *web server*, maka harus diupayakan agar jaringan komputer tersebut dapat aman terutama dari niat buruk pihak-pihak yang tidak diinginkan.

Berkenaan dengan mengamankan *web server* tersebut, maka semua potensi yang mungkin dapat digunakan oleh pihak-pihak yang tidak bertanggung jawab harus diantisipasi sejak awal.

Dalam bab 3 ini kita akan melakukan perancangan terhadap sistem keamanan yang dapat diterapkan pada *web server* tersebut.

Merujuk pada Gambar 3.1 di atas terlihat bahwa *web server* yang ada diletakan tepat di belakang sebuah *firewall*. *Firewall* adalah benteng pertahanan yang melindungi *web server* dari serangan luar.

Firewall bisa didefinisikan sebagai komponen yang membatasi akses antara komputer atau jaringan dengan Internet. *Firewall* bekerja dengan cara memfilter lalu lintas data yang melewati jaringan. *Firewall* dapat berupa *hardware, software* atau kombinasi keduanya.

Pada saat ini keamanan komputer tidak cukup hanya dilakukan oleh *firewall* semata. Biasanya untuk mengamankan jaringan komputer pengatur jaringan akan mengkombinasikan *firewall* dengan pendeteksi penyusup yang sering dikenal dengan istilah *Intrusion Detection System* (IDS).

Jika dianalogikan *firewall* lebih kepada benteng penjaga, maka IDS adalah petugas pengawas dari benteng tersebut. IDS akan memantau sistem komputer kita setiap saat dan kita dapat mendapatkan laporan lengkap setiap saat.

IDS bertugas melakukan identifikasi akses oleh siapa saja yang menggunakan sistem komputer tanpa hak. Termasuk didalamnya adalah percobaan untuk masuk secara paksa (*hacking*).

3.2 Analisa Kebutuhan.

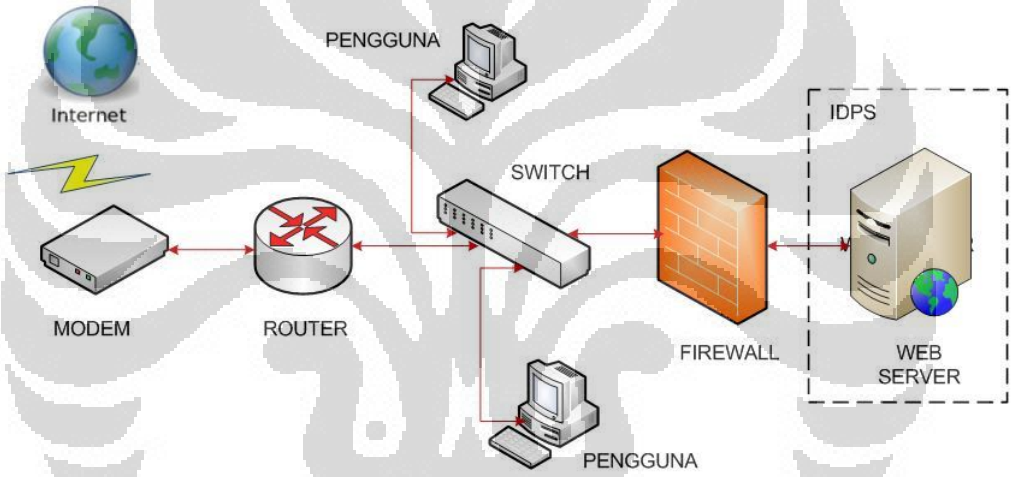
Pada penjelasan diatas telah dijabarkan secara singkat tentang fungsi-fungsi *firewall* dan IDS. Bila kita perhatikan dengan baik, maka kita dapat melihat bahwa baik *firewall* dan IDS hanya dapat melakukan tindakan *blocking* dan pemantauan dari sebuah jaringan ataupun komputer. Tanpa melibatkan didalamnya tindakan *preventif* yang dapat mereka lakukan sebelum menyadari bahwa sistem mereka telah dimasuki oleh pihak-pihak yang tidak berwenang.

Dalam penulisan tesis ini penulis merancang sebuah sistem yang dapat menambah kemampuan dari IDS terutama dalam hal memberikan tanggapan atau tindakan pada saat terjadinya serangan serta menambahkan fasilitas pemantauan berbasis web. Sehingga fungsi IDS menjadi lebih maksimal dan dapat meningkatkan fungsinya sehingga menjadi sebuah *Intrusion Detection Prevention System* (IDPS).

3.3 Perancangan IDPS.

Perancangan IDPS yang akan dilakukan adalah merancang sebuah aplikasi sederhana untuk diimplementasikan pada *web server* yang akan menyempurnakan fungsi IDS terutama dalam hal memberikan tanggapan atau tindakan pada saat terjadinya serangan serta menambahkan fasilitas pemantauan berbasis web.

Dengan fasilitas pemantauan berbasis *web* diharapkan dapat dengan mudah aktifitas pemantauan dilakukan karena data-data tentang akses legal dan akses ilegal dapat ditampilkan dalam bentuk tabel, sehingga diharapkan dapat dengan mudah dilakukan analisa terhadap jaringan atau *web server* tersebut.



Gambar 3.2 Topologi Jaringan Komputer dengan IDPS

Pada Gambar 3.2 diatas memperlihatkan sebuah jaringan komputer lokal dengan topologi star yang terdiri atas beberapa komponen yang melengkapinya, diantaranya adalah: modem dan sebuah router yang menghubungkan jaringan lokal atau jaringan internal dengan jaringan eksternal atau Internet, kemudian jaringan tersebut juga dilengkapi dengan Switch yang mengatur lalu lintas data internal dengan IP *address* kelas C dengan *subnetting* 192.168.1.0/24, terdapat pula sebuah *firewall* dan IDPS yang nanti akan diimplementasikan dengan meningkatkan kemampuan sebuah software IDS yaitu Snort yang dijalankan pada sebuah komputer yang berplatform Linux Ubuntu dengan alamat IP 192.168.1.1,

lalu terdapat pula sebuah komputer user dan sebuah komputer yang nantinya akan difungsikan untuk menyerang atau menguji sistem yang masing-masing komputer tersebut berplatform Windows XP dengan alamat IP 192.168.1.2 dan 192.168.1.3.

3.4. IDS dengan Snort.

Pada perancangan sistem ini digunakan perangkat lunak *free software* untuk mendukung aplikasi yang nanti akan kita jalankan. Sehingga dengan demikian kita bisa lebih menghemat biaya dan menghindarkan kita dari aktifitas pembajakan perangkat lunak yang marak akhir-akhir ini.

```
[**] [122:1:0] (portscan) TCP Portscan [**]
[Priority: 3]
06/09-15:22:50.322046 192.168.1.3 -> 192.168.1.1
PROTO:255 TTL:0 TOS:0x0 ID:0 IpLen:20 DgmLen:159 DF

[**] [1:100000160:2] COMMUNITY SIP TCP/IP message flooding directed to
SIP proxy [**]
[Classification: Attempted Denial of Service] [Priority: 2]
06/09-15:22:50.580544 192.168.1.3:54579 -> 192.168.1.1:1081
TCP TTL:48 TOS:0x0 ID:42625 IpLen:20 DgmLen:44
******S* Seq: 0x8BFE691D  Ack: 0x0  Win: 0x400  TcpLen: 24
TCP Options (1) => MSS: 1460
```

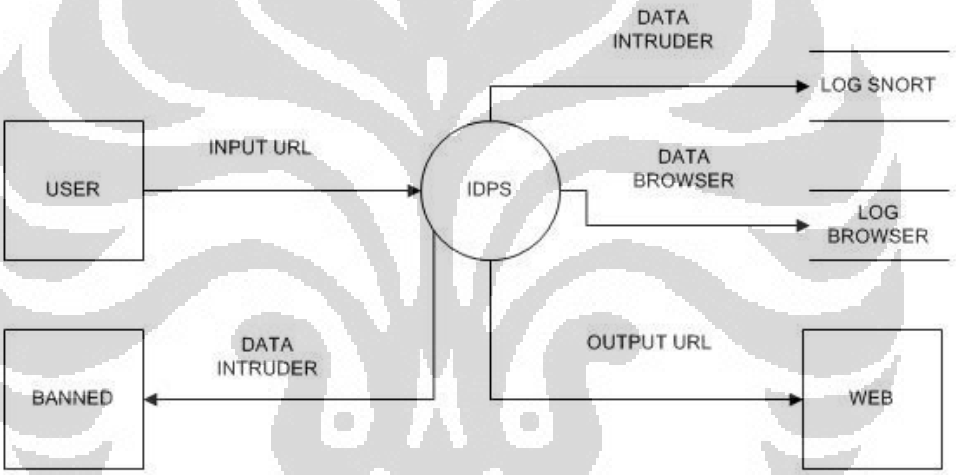
Gambar 3.3 *Alert* dari *Signature* pada IDS Snort

Pada Gambar 3.3 diatas diperlihatkan contoh dari sebuah *alert* dari snort. Snort adalah IDS yang bekerja dengan menggunakan *signature detection* dan berfungsi sebagai *sniffer* dan paket *logger*. Sehingga di dalam sebuah jaringan komputer snort dapat kita manfaatkan untuk mendeteksi adanya serangan pada sistem kita berdasarkan *signature* yang dimilikinya dan fasilitas atau kemampuannya dalam paket *logger* dapat kita manfaatkan untuk menyimpan informasi penyerang, sehingga identitas penyerang dapat terekam dengan baik. Dengan harapan apabila penyerang tersebut berusaha masuk ke dalam sistem kita akan langsung dikenali dan tidak akan diberikan akses masuk ke dalam *web server*.

3.5 *Intrusion Prevention System* (IPS).

Untuk menambahkan kemampuan dalam memberikan tanggapan dan tindakan terhadap pihak-pihak yang tidak berhak, maka akan dirancang sebuah program mini *Intrusion Prevention System* (IPS). Program mini IPS ini nanti akan diiimplementasi di satu tempat bersama dengan *web server*, dengan harapan akan mampu merespon tindakan dari pihak-pihak yang tidak berhak berdasarkan *log file* yang dimiliki oleh IDS snort.

Algoritma dari mini IPS yang akan kita implementasi seperti ditunjukan Diagram Konteks Diagram Alir Data (DAD) pada Gambar 3.4 dibawah ini.



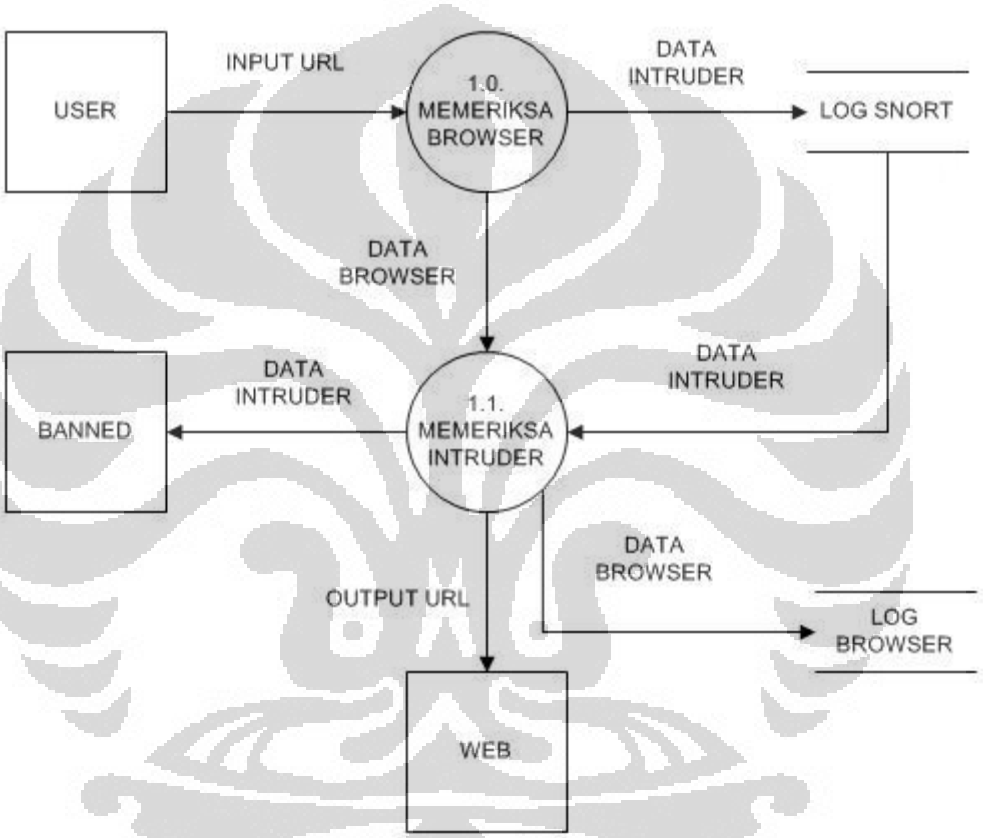
Gambar 3.4 Diagram Konteks Sistem IDPS

Dari Gambar 3.4 diatas dapat kita lihat sebuah sistem IDPS yang terhubung dengan tiga buah kesatuan luar, yaitu: *user, banned, web*. Serta dua buah simpanan data, yaitu: *log snort* dan *log browser*. Dilengakapi pula dengan 4 buah arus data, yaitu: *input url*, data *intruder*, data *browser*, dan *output url*.

User akan memberikan input berupa alamat *web* yang dituju kepada sistem IDPS. Sistem IDPS akan memeriksa apakah akses yang dilakukan oleh *user* dilakukan dengan menggunakan *browser* atau menggunakan program untuk memindai *web server*. Apabila sistem IDPS membaca akses dari *browser* maka akan diidentifikasi sebagai akses yang legal, tercatat di *log browser* dan berhak

mengakses *web server*. Sedangkan bila sistem IDPS memindai bahwa *input user* bukan berasal dari *browser* atau dari perangkat atau dari program pemindai, maka sistem IDPS akan mengenalinya sebagai akses ilegal, otomatis tercatat pada *log intruder* dan tidak berhak mengakses *web server* (*banned*).

Sistem IDPS secara lebih detail dapat dilihat pada diagram detail yang ditunjukan pada Gambar 3.5 berikut ini.

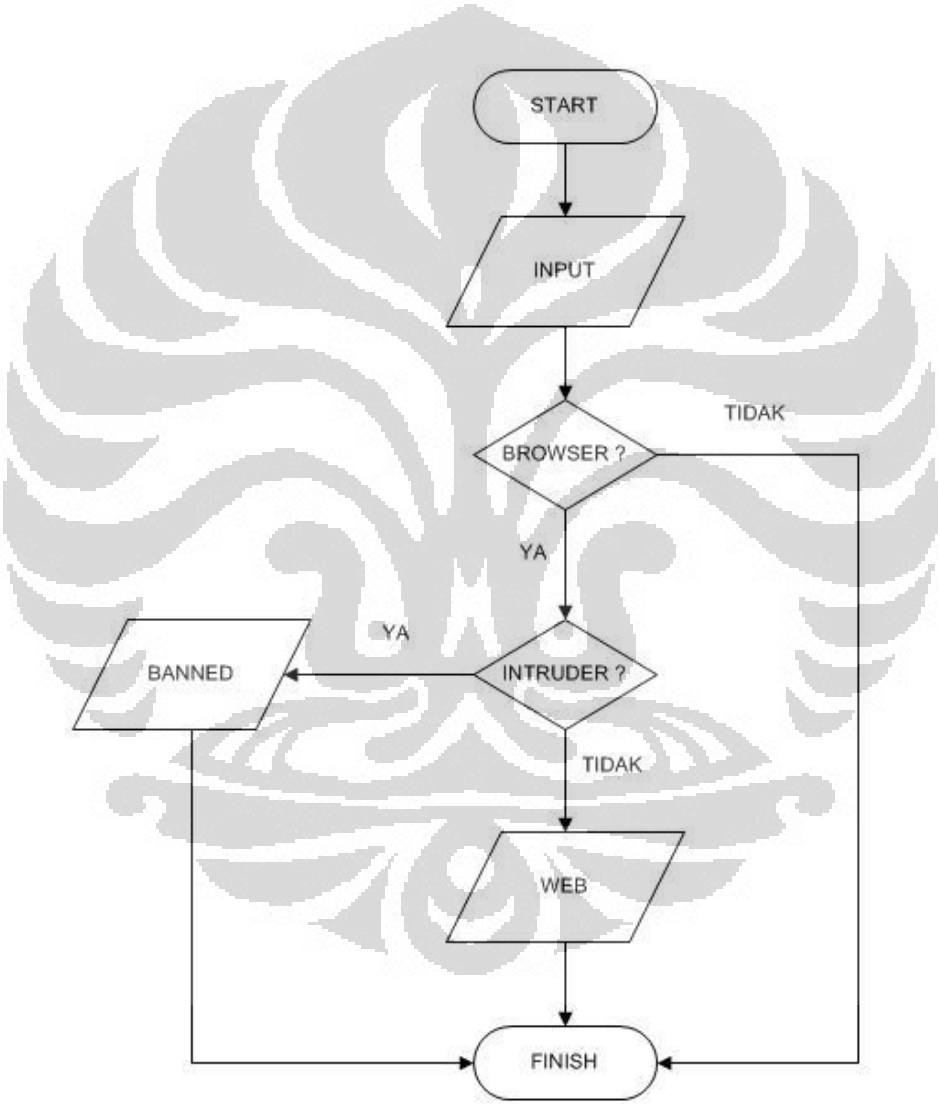


Gambar 3.5 Diagram Detail Sistem IDPS

Pada Gambar 3.5 diatas diperlihatkan adanya detail dari sistem IDPS yang terdiri atas 2 proses, yaitu: proses 1.0 memeriksa browser dan proses 1.1 memeriksa intruder. Untuk proses 1.0 memeriksa browser, proses ini dilakukan oleh sebuah program sederhana yang diletakan pada halaman depan *website,* yang berfungsi untuk melakukan pengecekan apakah akses yang dilakukan oleh *user,* berasal dari *browser* atau bukan. Apabila akses berasal dari sebuah *browser* program akan meneruskan ke proses 1.1. Pada proses 1.1 alamat IP yang

digunakan oleh *browser* tersebut akan dicocokan dengan basis data yang ada pada *log snort*, apabila cocok maka permintaan browser tersebut akan dianggap sebagai akses ilegal (banned). Demikian pula sebaliknya apabila tidak cocok maka secara otomatis akan dianggap sebagai akses legal, program akan mencatatnya pada *log browser* dan boleh mengakses *web server*.

Bagan alir dari proses program pendeteksi browser ini dapat dilihat pada Gambar 3.6 dibawah ini.



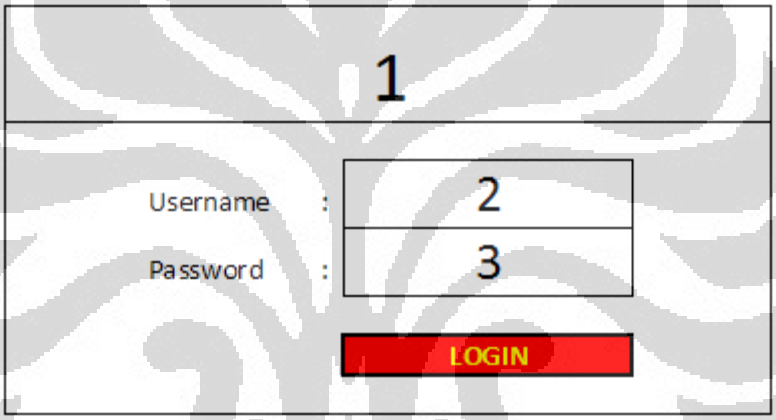
Gambar 3.6 Bagan Alir pendeteksi *browser* atau *intruder*

3.6 Rancangan Antarmuka.

Beberapa antarmuka telah dirancang sebagai sarana komunikasi antara pengatur jaringan dengan sistem IDPS yang nantinya akan digunakan. Beberapa antarmuka tersebut mempunyai fungsi masing-masing yang secara keseluruhan sistem menjadi satu kesatuan yang saling terkait.

Untuk masuk kedalam sistem IDPS ini kita akan dihadapkan dengan sebuah halaman *login*, yang memiliki *username* dan *password* tertentu agar tidak dengan mudah dapat diakses oleh pihak-pihak yang tidak berhak.

Rancangan halaman *login* sistem IDPS seperti ditunjukan pada Gambar 3.7 dibawah ini.



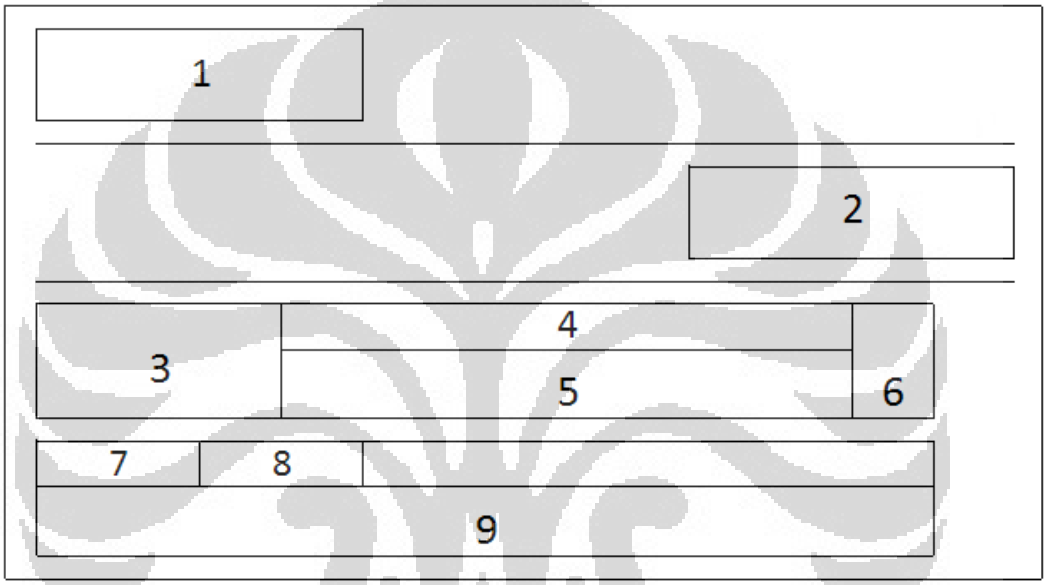
Gambar 3.7 Rancangan antarmuka halaman login

Pada Gambar 3.7 diatas ada beberapa bagian yang diberikan nomor untuk mempermudah dalam menjelaskan tentang halaman *login* tersebut. Bagian dengan nomor 1 adalah tempat untuk meletakan judul, kemudian bagian dengan nomor 2 dan 3, masing-masing adalah tempat untuk memasukan *username* dan *password*. *Command button LOGIN* berfungsi mengeksekusi masukan dari *username* dan *password*.

Kemudian rancangan antarmuka yang berikutnya adalah untuk halaman utama dari sistem IDPS. Halaman ini akan memberikan informasi tentang: data sensor, jumlah *user* yang *online*, data *browser*, data *intruder*. Data sensor menunjukan identitas dari sensor yang berupa data *ethernet card* beserta alamat IP

yang digunakan oleh *web server* sebagai sensor. Jumlah *user online* menunjukan jumlah *user* legal yang sedang mengakses web server. Data browser berisi identitas dari user legal yang sedang mengakses web server, seperti: alamat IP, hari, tanggal, waktu, dan identitas dari *browser*. Sedangkan data *intruder* berisikan informasi akses ilegal yang mencoba menyerang *web server* kita.

Rancangan tampilan antarmuka halaman utama dari sistem IDPS kita seperti ditunjukan pada Gambar 3.8 berikut ini.



Gambar 3.8 Halaman utama sistem IDPS

Pada Gambar 3.8 diatas diperlihatkan bagian gambar dengan nomor 1 adalah tempat untuk meletakan judul dan identitas dari aplikasi halaman utama. Bagian nomor 2 memperlihatkan informasi tentang basis data snort. Bagian nomor 3 memperlihatkan tentang data sensor yang memperlihatkan identitas dari *ethernet card* yang digunakan sebagai sensor. Bagian nomor 4 akan menghitung jumlah dari pengguna yang sah. Bagian nomor 5 adalah *log file* untuk pengguna yang sah. Bagian nomor 6 adalah halaman untuk memanggil ulang halaman utama. Bagian nomor 7 untuk menghitung jumlah terjadinya akses yang tidak sah atau ilegal. Bagian nomor 8 digunakan untuk memanggil detail dari *log file*

pengguna yang tidak sah. Bagian nomor 9 adalah tampilan dari *log file* pengguna yang tidak sah yang terbaru.

3.7 Rancangan Basis Data.

Sistem IDPS yang kita akan implementasi akan menggunakan beberapa basis data untuk keperluan penyimpanan data yang akan digunakan pada berbagai keperluan pemeriksaan.

Untuk keperluan halaman *login* sistem IDPS, kita memerlukan basis data phplogin dengan tabel *users* yang terdiri dari 3 field,yaitu *id*, *username* dan *password*, seperti ditunjukan pada Gambar 3.9 berikut ini.

tabel: users

id	username	password

Gambar 3.9 basis data halaman *login*

Dari Gambar 3.9 diatas diperlihatkan *field id* yang merupakan kolom yang nantinya diisi secara otomatis oleh basis data MySQL, yang berfungsi untuk memberikan identitas maupun penomoran untuk memudahkan kegiatan pencarian data maupun perhitungan. Kolom *username* dan password akan berisi identitas orang-orang yang nantinya akan diberi hak untuk masuk ke dalam sistem IDPS.

Untuk menyimpan data *browser* dari *user* kita membutuhkan tabel *useronline*, yang memiliki tujuh buah field, yaitu: *timestamp*, ip, harian, tanggal, waktu, *browser*, *file*. Tabel *useronline* tersebut diperlihatkan seperti pada Gambar 3.10 berikut ini.

tabel: useronline

timestamp	ip	harian	tanggal	waktu	browser	file

Gambar 3.10 basis data *useronline*

Pada Gambar 3.10 diperlihatkan kolom *timestamp* yang berfungsi untuk memberikan waktu bagi pengguna untuk dapat mengakses *web server*. Kolom *ip* digunakan untuk menyimpan alamat IP dari pengunjung. Kolom harian digunakan untuk menyimpan data hari. Kolom tanggal digunakan untuk menyimpan data tanggal. Kolom waktu digunakan untuk menyimpan data jam dari pengunjung. Kolom *browser* digunakan untuk menyimpan data tentang *browser* yang digunakan oleh pengunjung. Kolom file digunakan untuk menyimpan data tentang *file* yang diakses oleh *browser* pengunjung.

Untuk menyimpan data sensor dari sistem IDPS kita harus membuat tabel sensor dari basis data snort. Dengan susunan *field* pada tabel sensor seperti ditunjukan pada Gambar 3.11 sebagai berikut.

tabel: sensor

id	hostname	interface

Gambar 3.11 basis data sensor

Untuk menyimpan data akses ilegal dari sistem IDPS kita harus membuat tabel serangandari basis data snort. Dengan susunan *field* pada tabel serangan seperti ditunjukan pada Gambar 3.12 sebagai berikut.

tabel: serangan

waktu	jenis serangan	ip sumber	port sumber	ip tujuan	port tujuan

Gambar 3.12 basis data serangan

BAB 4
IMPLEMENTASI DAN PENGUJIAN SISTEM IDPS

4.1 Sistem IDPS.

Sistem IDPS yang akan kita implementasi pada bagian ini akan kita lakukan dengan menggunakan *open source* dan *free software*. Dalam implementasi sistem ini kita akan menggunakan sistem operasi Linux Ubuntu 10.10, *web server* apache, bahasa pemrograman PHP, dan basis data menggunakan MySQL, serta Snort.

Kebutuhan akan perangkat lunak yang digunakan dapat kita lihat pada Tabel 4.1 berikut ini.

Tabel 4.1 Perangkat Lunak Implementasi Sistem IDPS

Sistem Operasi	Linux Ubuntu
Web server	Apache
Bahasa pemrograman	PHP 5.0
Basis data	MySQL
IDS	Snort

4.2 Basis Data Snort.

Kita akan menggunakan *file log* snort untuk mendeteksi adanya *intruder*, maka kita harus membuat agar *file log* snort tersebut dapat dibaca oleh aplikasi yang akan kita buat. Sehingga kita perlu membuat agar file log snort tersebut yang berbentuk teks dapat ditampilkan dalam basis data MySQL.

Untuk itu perlu kita membuat sebuah basis data dinamis pada basis data MySQL dengan nama snort, yang akan menyimpan semua perubahan yang terjadi pada *file log* snort ke dalam basis data MySQL snort tersebut.

Kita akan membuat dulu basis data snort dalam MySQL dengan masuk ke MySQL terlebih dahulu, dengan perintah seperti yang ditunjukan pada Gambar 4.1 berikut ini.

```
mysql -u root -p
```

Gambar 4.1 Masuk ke Lingkungan MySQL

Setelah itu kita buat database dengan nama snort, diikuti dengan perintah untuk membuat tabel-tabel dinamis dalam MySQL seperti ditunjukan pada Gambar 4.2 berikut ini.

```
mysql>create database snort;
Query OK, 1 row affected (0.01 sec)

mysql>GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP,INDEX,ALTER,CREATE
    TEMPORARY TABLES, LOCK TABLES ON snort.*'your_directory'@'host'
    IDENTIFIED BY 'your_password';

mysql>flush privileges;
```

Gambar 4.2 Membuat Tabel Dinamis Basis Data Snort.

Untuk basis data yang lain seperti untuk tabel halaman *login* dan *log browser* kita buat juga dengan perintah-perintah MySQL. Seperti untuk tabel *users,* kita buat dengan perintah seperti ditunjukan pada Gambar 4.3 berikut ini.

```
CREATE TABLE users (
        id       int      (11),
        username varchar  (25),
        password varchar  (25),
        PRIMARY KEY (id),
);
```

Gambar 4.3 Membuat Tabel *Users*

Untuk basis data *useronline* kita buat juga dengan perintah MySQL seperti ditunjukan pada Gambar 4.4 berikut ini.

```
CREATE TABLE useronline (
       timestamp int     (15), AUTO INCREMENT,
       ip        varchar (40),
       harian    varchar (10),
       tanggal   varchar (10),
       waktu     varchar (10),
       browser   varchar (10),
       file      varchar (100),
       PRIMARY KEY (timestamp),
);
```

Gambar 4.4 Membuat Tabel *Useronline*

4.3 Implementasi IPS.

Untuk menambahkan kemampuan dalam memberikan tanggapan dan tindakan terhadap pihak-pihak yang tidak berhak, maka akan dirancang sebuah program mini *Intrusion Prevention System* (IPS). Program mini IPS ini diiimplementasi di satu tempat bersama dengan *web server*, dengan harapan akan mampu merespon tindakan dari pihak-pihak yang tidak berhak berdasarkan *log file* yang dimiliki oleh IDS snort.

Program sederhana ini diimplementasi dengan menggunakan bahasa pemrograman PHP dengan menambahkan programnya pada web server. Program sederhana tersebut memiliki pseudocode seperti diperlihatkan pada Gambar 4.5 berikut ini.

```
1: INPUT IP ADDRESS
2: CONNECT TO DATABASE
3: OPEN TO SNORT DATABASE
4: QUERY TO ACID_EVENT TABLE WHERE IP_SRC=”$IP1”
5: COUNTING THE ROWS
6: IF ROWS > 0 {
   MESSAGE.PHP } ELSE {
   OPEN.PHP }
```

Gambar 4.5 Pseudocode Program IPS Sederhana

4.4 Perangkat Pengujian.

Pada tahap pengujian ini kita akan menguji sistem IDPS yang telah kita implementasi, apakah bekerja dengan baik atau tidak. Dengan menggunakan laboratorium mini yang kita buat untuk melakukan pengujian terhadap sistem IDPS yang telah selesai kita buat.

Dalam pengujian ini akan diterapkan beberapa skenario baik yang mewakili akses legal maupun yang mewakili oleh adanya kegiatan pemindaian (akses ilegal). Akses legal akan diwakili oleh adanya akses yang menggunakan *browser*. Sedangkan akses ilegal akan diwakili dengan menggunakan program pemindai yang banyak digunakan oleh para pengatur jaringan untuk menguji jaringan mereka yaitu *network mapper* (Nmap) yang pada pengujian kali ini kita gunakan versi GUI, yaitu Zenmap.

Topologi yang akan kita gunakan seperti ditunjukan oleh Gambar 4.6 berikut ini.

Gambar 4.6 Topologi Pengujian Sistem IDPS

Pada topologi yang terlihat pada Gambar 4.6 diatas memperlihatkan sebuah laboratorium mini sebagai sarana untuk melakukan pengujian terhadap jaringan dan sistem IDPS yang telah kita buat. Diperlihatkan adanya komputer yang berperan sebagai *web server*, IDPS, *firewall*, komputer pemindai, komputer pengguna, dan *traffic generator*.

Dalam pengujian ini spesifikasi perangkat keras yang digunakan untuk *web server* tempat sistem IDPS diletakan seperti yang ditunjukan pada Tabel 4.2 berikut.

Tabel 4.2 Spesifikasi Perangkat Keras *Web Server*

No.	Item	Spesifikasi
1	Processor	Intel core 2 Duo 2,1 GHz
2	Memori	DDR2 2 GB
3	Harddisk	160 GB
4	Ethernet/LAN Card	D-LINK

Sedangkan Perangkat Lunak yang digunakan, kita gunakan perangkat lunak yang berbasis *free software* seperti pada Tabel 4.3 berikut ini:

Tabel 4.3 Spesifikasi Perangkat Lunak *Web Server*

No.	Item	Spesifikasi
1	Sistem Operasi	Linux Ubuntu
2.	Modul Pendeteksi	*Snort Open Source*
3.	Modul Pencegah	Sistem IPS

Sebuah komputer pengguna dalam jaringan ini adalah klien yang akan mengakses fasilitas yang terdapat pada komputer server yang dalam hal ini diperankan oleh komputer dengan sistem operasi Windows XP.

Komputer pengguna akan berperan sebagai klien sah dari server yang nantinya akan kita lihat reaksinya pada saat komputer pengguna ini mengakses komputer server pada fasilitas IDPS yang terdapat pada komputer server. Secara otomatis apabila komputer pengguna ini terbaca sebagai pengguna yang sah atau legal maka akan diberikan akses oleh *web server* dan akan tercatat pada basis data *useronline*.

Spesifikasi komputer pengguna ditunjukan seperti Tabel 4.4 berikut ini.

Tabel 4.4 Spesifikasi Komputer Pengguna

No.	Item	Spesifikasi
1	Processor	Intel atom 1,66 GHz
2	Memori	DDR2 1 GB
3	Harddisk	120 GB
4	Ethernet/LAN Card	On Board
5	Sistem Operasi	Windows XP

Komputer penyerang dirancang berplatform windows XP, karena memang sistem operasi yang paling banyak digunakan oleh sebagian besar pengguna komputer. Apabila kita tambahkan beberapa perangkat lunak pengujian keamanan, maka akan bermanfaat sekali untuk menguji sistem keamanan dari jaringan komputer yang kita miliki baik keamanan server maupun user yang terdapat dalam jaringan tersebut.

Spesifikasi komputer penyerang ditunjukan pada Tabel 4.5 berikut ini.

Tabel 4.5 Spesifikasi Komputer Penyerang

No.	Item	Spesifikasi
1	Processor	Intel atom 1,66 GHz
2	Memori	DDR2 1 GB
3	Harddisk	120 GB
4	Ethernet/LAN Card	On Board
5	Sistem Operasi	Windows XP
6	Perangkat Lunak Pengujian Keamanan	Zenmap

Sebuah komputer lagi kita siapkan untuk memerankan fungsi sebagai pembangkit jaringan dan pengukur *bandwidth*. Komputer ini sangat penting

peranannya dalam pengujian, sebagai pembangkit lalu lintas jaringan. Lalu lintas jaringan yang akan kita bangkitkan dengan komputer ini terdiri atas tiga kondisi, yaitu, *idle,* menengah, dan tinggi. Spesifikasi dari komputer yang digunakan sebagai komputer pembangkit jaringan, seperti ditunjukan pada Gambar 4.6 berikut ini.

Tabel 4.6 Spesifikasi Komputer Pembangkit Jaringan

No.	Item	Spesifikasi
1	Processor	Intel core 2 Duo, 2.1 GHz
2	Memori	DDR2 1 GB
3	Harddisk	120 GB
4	Ethernet/LAN Card	On Board
5	Sistem Operasi	Windows XP
6	Perangkat Lunak	Iperf

Semua komputer yang digunakan dalam pengujian tersebut, dihubungkan dengan sebuah switch D-Link delapan port. Hubungan antar komputer tersebut melalui transmisi kabel UTP dan menggunakan konektor standar RJ-45.

4.5 Web Pengujian.

Untuk melakukan pengujian terhadap akses yang legal dan tidak, dalam pengujian ini juga telah disiapkan dan dibuat sebuah web sederhana yang nantinya akan dijadikan target untuk melakukan pengujian ini.

Halaman *web* ini nantinya hanya akan dapat diakses oleh pengguna yang menggunakan browser dan alamat IP-nya belum tercatat sebagai *intruder* pada *log file* snort. Apabila pengguna yang mengakses menggunakan browser dan alamat IP-nya terekam pada *log file* snort sebagai *intruder*, maka secara otomatis permintaan akses ke halaman web akan tidak diberikan (*banned)*.

Tampilan halaman *web* yang akan dijadikan target pengujian, terlihat seperti ditunjukan pada Gambar 4.7 berikut ini.

Gambar 4.7 Halaman *Website* Target

4.6 Skenario Pengujian.

Pada skenario pengujian akan disimulasikan bentuk-bentuk kegiatan atau aktifitas yang secara umum terdiri dari dua hal, yaitu:

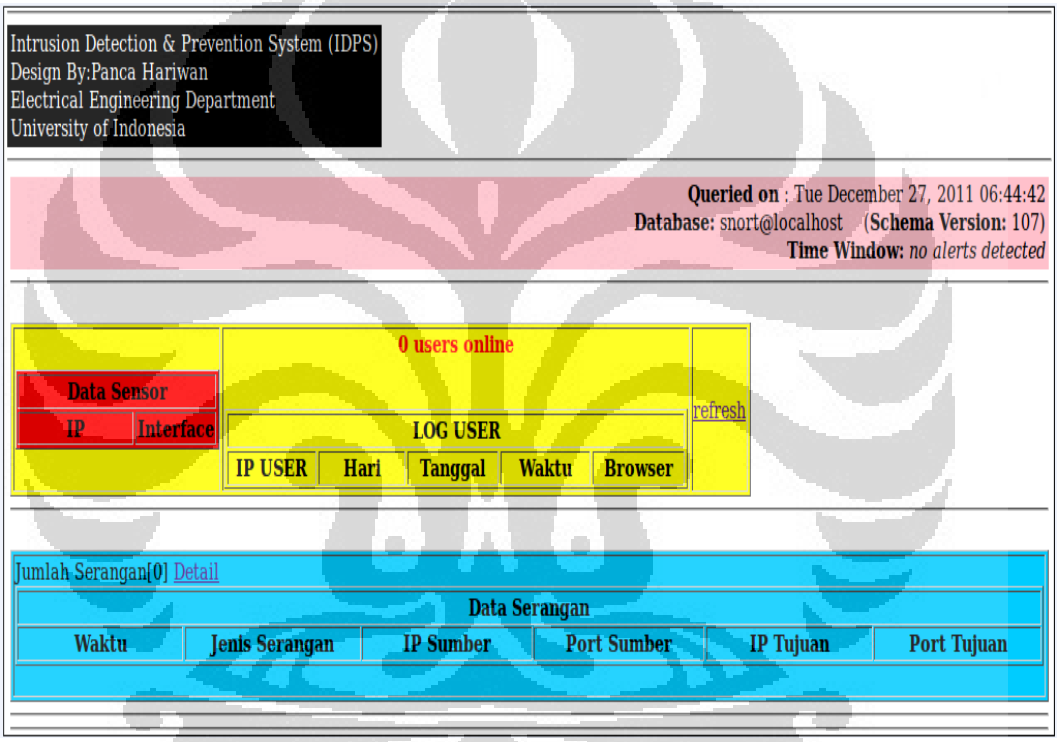
a. Akses pengguna yang sah atau legal. Pengguna sah ini dapat diidentifikasi dengan dua hal, yaitu: akses masuk ke *web server* dengan menggunakan *browser*, dan yang kedua alamat IP-nya belum pernah tercatat sebagai *intruder* pada *log file* snort.
b. Akses pengguna yang tidak sah atau ilegal. Pengguna tidak sah ini dapat diidentifikasi dengan dua hal, yaitu: akses masuk ke *web server* dengan menggunakan *browser* atau *scanning tools*, dan yang kedua alamat IP-nya sudah pernah tercatat sebagai *intruder* pada *log file* snort.

Akses pengguna yang tidak sah ini nantinya akan diwakili oleh komputer penyerang, yang sudah dilengkapi dengan program Zenmap. Program Zenmap nantinya akan dicoba untuk melakukan pemindaian yang dimiliki oleh program tersebut, dan kemudian kita akan lihat tanggapan dari sistem IDPS dalam menangani atau menghadapi kegiatan pemindaian tersebut.

Dalam pengujian ini kita akan mencoba untuk membangkitkan tiga jenis lalu lintas data yang akan kita umpankan kepada *web server* yang mewakili kondisi *idle,* menengah, dan tinggi.

4.7 Pengujian Fungsional.

Sebagai tahap awal implementasi pengujian kita akan memperlihatkan terlebih dahulu tampilan dari aplikasi mini IDPS yang di buat. Aplikasi mini IDPS memiliki tampilan seperti yang ditunjukan pada Gambar 4.8 berikut ini.

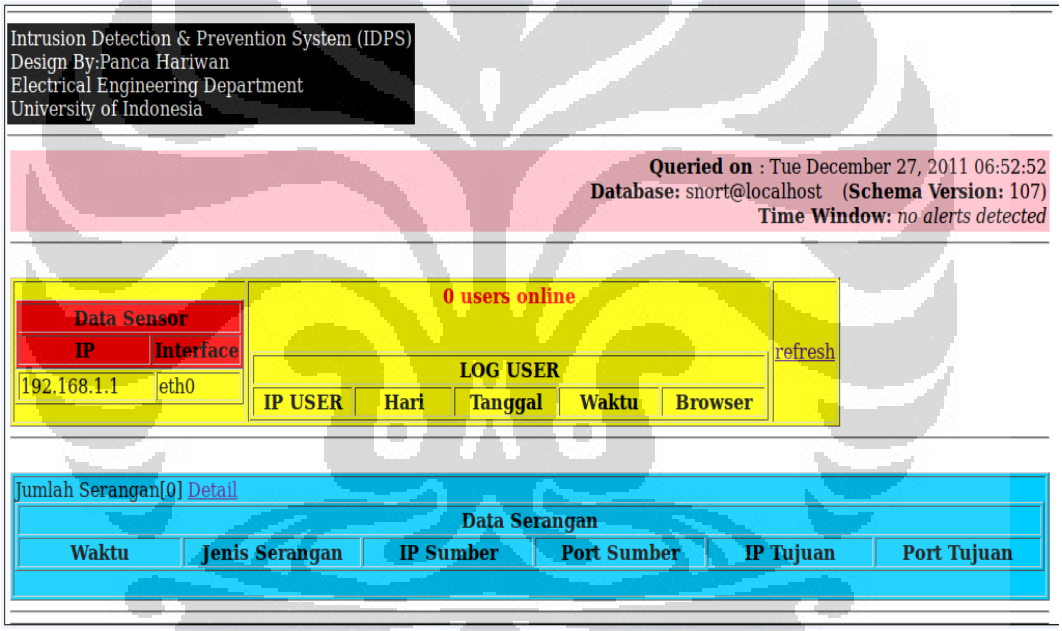


Gambar 4.8 Tampilan Awal Mini IDPS

Dari tampilan mini IDPS seperti yang diperlihatkan pada Gambar 4.8 diatas, memperlihatkan beberapa bagian seperti yang telah dirancang pada tahap perancangan pada bab III. Bagian kotak yang berwarna hitam adalah bagian judul. Bagian kotak yang berwarna merah muda adalah bagian yang menunjukan informasi tentang basis data snort dan informasi dari *alert* snort. Pada Kotak yang berwarna kuning terdiri atas dua bagian. Bagian yang berwarna merah menunjukan informasi dari sensor yang digunakan.

Bagian berikutnya menunjukan *log file* dari pengguna yang sah atau legal. Bagian yang berwarna biru menginformasikan tentang *log file* dari pengguna yang tidak sah.

Sekarang kita akan mencoba untuk mengakses *web server* dengan menggunakan browser yang ada pada komputer penguji. *Web* yang akan dijadikan target pengujian adalah *web* yang sudah kita buat sebelumnya. Sebelum aplikasi mini IDPS ini digunakan kita harus memastikan bahwa data sensor pada tabel sensor berisi tentang informasi *ethernet card* dari *web server* kita. Tampilan data dari sensor mini IDPS seperti diperlihatkan pada Gambar 4.9 berikut ini.



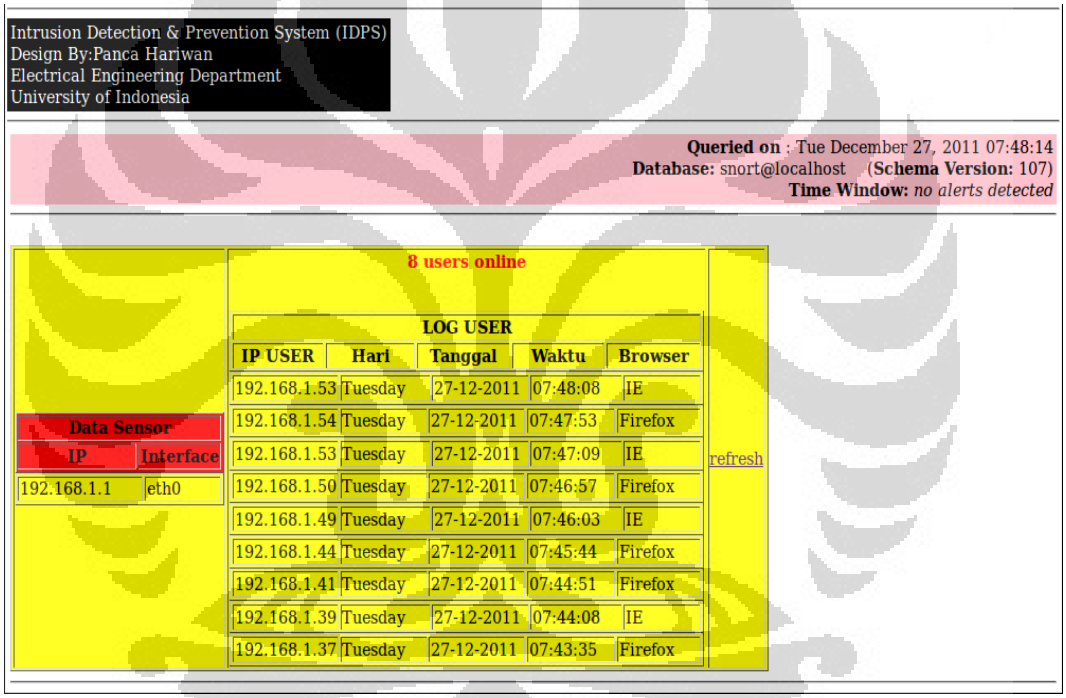
Gambar 4.9 Data Tabel Sensor

Pada Gambar 4.9 diatas terlihat bahwa tabel data sensor telah terisi sebuah sensor dengan alamat IP 192.168.1.1 dengan identitas *interface* eth0. Jika tabel sensor ini telah terisi dengan data sensor yang akan kita gunakan, maka aplikasi mini IDPS ini sudah dapat kita gunakan.

Tahap selanjutnya kita akan mencoba untuk melakukan pengujian untuk mengakses *web server*, dengan menggunakan komputer penguji. Komputer penguji akan melakukan pengujian untuk mengakses halaman *web* yang telah

dibuat sebelumnya sebagai target untuk pengujian. Pada tahap pengujian ini akan digunakan dua macam *browser* yang paling sering digunakan oleh para pengguna komputer yang sering berselancar di dunia maya, yaitu: *Internet Explorer* dan *Mozilla Firefox*. Untuk pengguna yang mengakses menggunakan selain dua *browser* tersebut, aplikasi mini IDPS akan membacanya sebagai *unknown browser*, namun tetap dapat digunakan untuk mengakses halaman web yang digunakan sebagai target.

Hasil yang ditangkap oleh mini IDPS sebagai akibat dari diaksesnya halaman *web* oleh *browser* pengguna diperlihatkan pada Gambar 4.10 berikut ini.



Gambar 4.10 Data Akses Halaman Web

Pada Gambar 4.10 diatas diperlihatkan bahwa sensor dengan alamat IP 192.168.1.1 menangkap aktifitas akses halaman web dari delapan alamat IP pengguna. Diperlihatkan juga pada tabel *user online* tersebut data tentang hari, tanggal, waktu akses, serta *browser* yang digunakan.

Pengguna yang tercantum dalam tabel tersebut adalah pengguna yang dikategorikan sebagai pengguna yang sah. Mereka hanya mengakses halaman web

dengan menggunakan *browser* yang mereka miliki, seperti data yang disajikan pada Gambar 4.10 tersebut.

Bagian berikutnya *web server* akan diskenario untuk diserang dengan menggunakan program eksploit Zenmap yang merupakan bentuk GUI dari program Nmap (*network mapper*). Pada pengujian ini Zenmap akan mencoba untuk melakukan pemindaian terhadap port *web server* (port 80), dan akan kita lihat bagaimana kemampuan dari sistem mini IDPS dalam mendeteksi serangan tersebut. Hasil dari deteksi oleh sistem mini IDPS seperti ditunjukan pada Gambar 4.11 berikut ini.

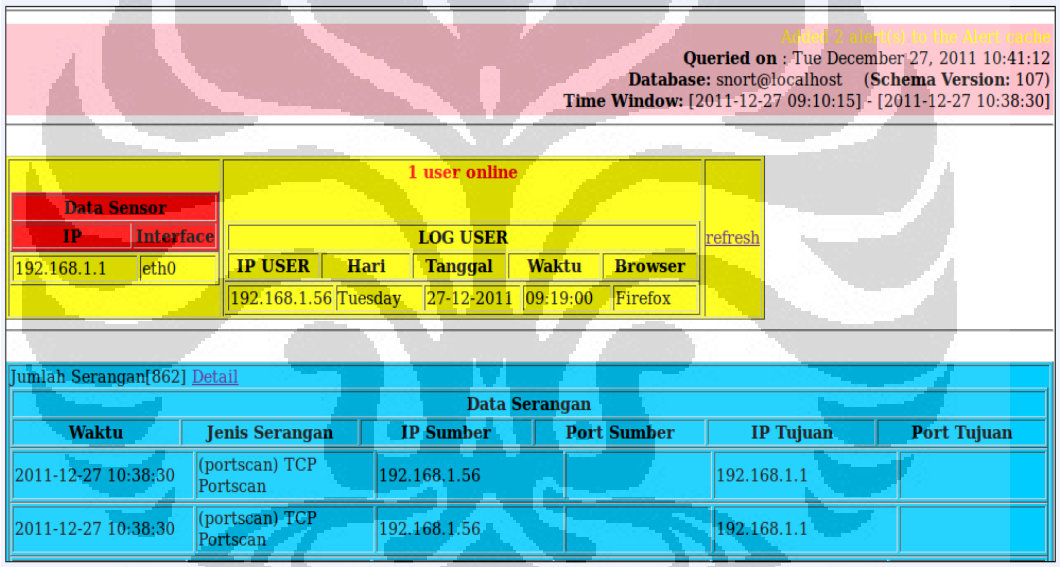
Gambar 4.11 Tampilan Deteksi IDPS terhadap Eksploit Zenmap

Pada Gambar 4.11 diatas pada tabel yang berwarna biru memperlihatkan bahwa telah terjadi 282 kali serangan pemindaian port. Diantaranya adalah serangan dilakukan pada pukul 09:12:04 berasal dari alamat IP 192.168.1.53, port sumber adalah port 53, alamat IP yang diserang atau yang dijadikan target adalah 192.168.1.1 dan port target adalah port 554.

Agar pada saat terjadinya atau terdeteksinya suatu serangan dapat diambil suatu tindakan tertentu maka ditambahkan ke dalam sistem mini IDPS kita sebuah mekanisme program yang dapat melakukan tanggapan atau tindakan pada saat terjadinya serangan.

Program ini akan membaca alamat IP dari penyerang yang akan diambil dari basis data snort dan kemudian pada saat penyerang dengan alamat IP tersebut akan mengakses *web server* akan dilakukan tindakan pencegahan dengan tidak diperbolehkan masuk untuk mengakses halaman web tersebut.

Apabila kita melakukan penyerangan pada *web server* hasilnya akan tampak seperti diperlihatkan pada Gambar 4.12 berikut ini.



Gambar 4.12 Tampilan Hasil Serangan

Terlihat pada Gambar 4.12 diatas pada bagian tabel yang berwarna merah muda terdapat tambahan dua buah peringatan (*alert*) dari snort. Hal ini menandakan telah terjadi aktifitas ilegal atau tidak sah yang mencoba untuk melakukan pemindaian terhadap *web server*. Jika hal ini dibiarkan dan tidak diamankan maka akan mengganggu aktifitas dari *web server*.

Dan pada *browser* pengguna apabila mereka akan berusaha mengakses *web server* akan muncul pesan seperti ditunjukan pada Gambar 4.13 berikut ini.

Gambar 4.13 Pesan Tidak dapat Akses

Pesan seperti ditunjukan pada Gambar 4.13 diatas akan selalu muncul pada saat pengguna mengakses *web server*, apabila pengatur jaringan belum melepas akses alamat IP dari basis data snort.

Untuk menghindari terulangnya kembali penyerang dapat mengakses *web server* maka dilakukan pemutusan dengan menjalankan fungsi *firewall* terhadap akses penyerang. Dengan melakukan pemutusan terhadap akses ke *web server* dengan menggunakan identitas dari alamat MAC dari pengguna tersebut. Hal ini dilakukan dengan melakukan perintah seperti pada Gambar 4.14 dibawah ini.

```
Iptables –A INPUT –i eth0 –p tcp –destination-port90 –m mac –mac-source-bc:ae:c5:87:0f:c5 –j DROP
```

Gambar 4.14 Perintah Aktifkan Firewall dengan Iptables

4.8. Pengujian Jaringan.

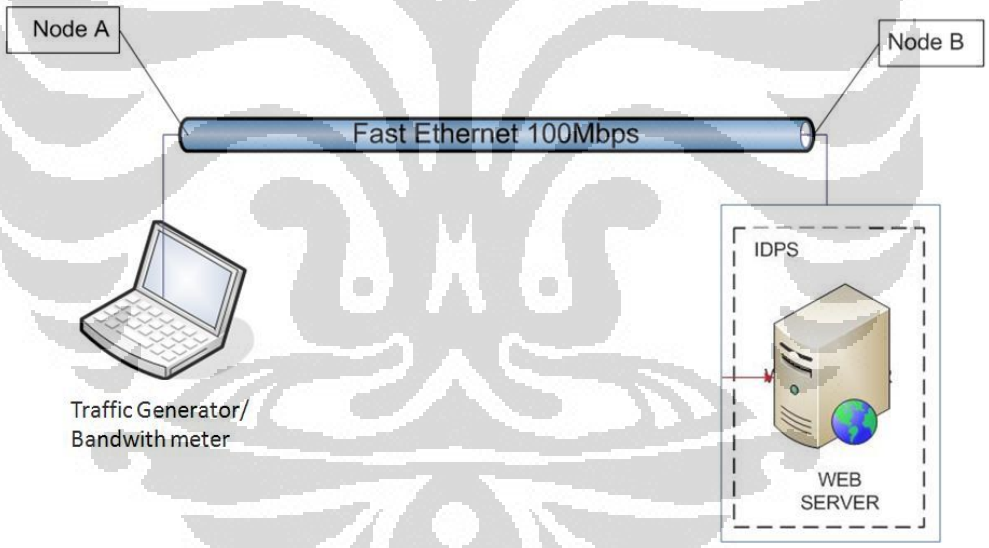
Setelah sebelumnya kita melakukan pengujian terhadap fungsi-fungsi dari bagian-bagian yang ada pada sistem mini IDPS yang telah dibuat, maka selanjutnya kita akan melakukan pengujian dengan mengimplementasikannya dalam suatu jaringan komputer.

Jaringan komputer ini akan diimplementasi dalam sebuah laboratorium mini yang kita bentuk dari beberapa komputer yang terhubung satu sama lain dan

menjalankan beberapa fungsi-fungsi tertentu, seperti yang ditunjukan pada Gambar 4.6 diatas.

Pada pengujian jaringan ini kita akan implementasikan jaringan dalam tiga kondisi, yaitu: *idle*, menengah, dan tinggi. Untuk menentukan klasifikasi kondisi *idle*, menengah, dan tinggi itu maka akan dilakukan serangkaian percobaan untuk menentukan *bandwidth* yang dapat mewakili ketiga kondisi tersebut. Bila telah kita dapat kondisi tersebut maka nanti nilai dari *bandwidth* itu akan kita jadikan ukuran untuk mengimplementasi apakah lalu lintas jaringan berada dalam kondisi idle, menengah, atau tinggi.

Untuk menentukan kondisi tersebut kita akan mengimplementasikannya dengan menggunakan bentuk topologi jaringan seperti ditunjukan pada Gambar 4.15 berikut ini.



Gambar 4.15 Topologi Penentuan Lalu Lintas Jaringan

Diperlihatkan pada Gambar 4.15.diatas adanya sebuah komputer yang berperan sebagai *traffic generator* sekaligus berfungsi sebagai pengukur *bandwith* terhubung dengan *web server* melalui sebuah kabel atau saluran 100Mbps. Dalam hal ini implementasi saluran kita gunakan kabel UTP kategori 5 dengan konektor RJ 45 yang memang memiliki standar kecepatan 100Mbps.

Komputer *traffic generator* akan memberikan sinyal masukan kepada *web server* kondisi ini dianggap belum ada satupun pengguna yang mengakses *web server*, kita namakan kondisi ini dengan kondisi *idle*. Kemudian setelah itu bandwidth yang terjadi selama hubungan tersebut kita ukur, hal ini dilakukan secara berulang sebanyak sepuluh kali. Hasil pengukurannya dapat dilihat pada Tabel 4.7 berikut ini.

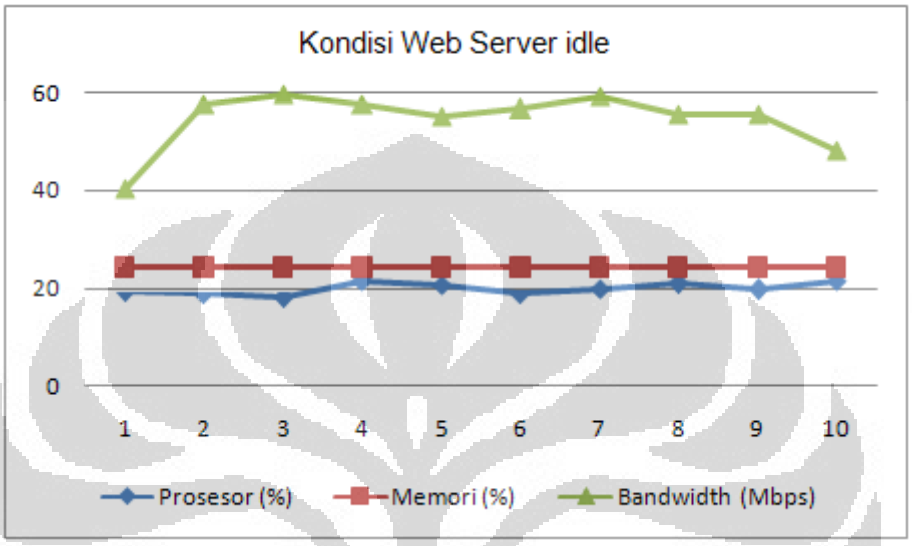
Tabel 4.7 Pengukuran *Bandwidth* Kondisi *Idle*

iterasi ke-	Prosesor (%)	Memori (%)	Bandwidth (Mbps)
1	19.2	24.2	40.4
2	18.8	24.2	57.7
3	18	24.2	59.8
4	21.4	24.2	57.7
5	20.6	24.2	55.1
6	18.8	24.2	56.6
7	19.8	24.2	59.2
8	20.8	24.2	55.6
9	19.6	24.2	55.6
10	21.6	24.2	48.2
Rata-rata	**19.86**	**24.2**	**54.59**

Dari Tabel 4.7 diatas kita lihat dilakukan pengujian berulang-ulang, dari iterasi pertama sampai sepuluh. Ada beberapa parameter yang terukur, yaitu: aktifitas prosesor, aktifitas memori, dan *bandwith*. Aktifitas prosesor dan memori yang tercatat pada Tabel 4.7 tersebut adalah aktifitas dari prosesor dan memori yang dimiliki oleh *web server*. Parameter ini penting untuk mengetahui kondisi kinerja dari *web server* terutama yang terkait dengan kedua komponen tersebut diatas. Karena apabila terjadi gangguan pada kedua komponen tersebut akan mengakibatkan sistem *web server* menjadi terganggu dan tidak dapat bekerja secara maksimal.

Pada kondisi *idle* Tabel 4.7 menunjukan bahwa rata-rata kinerja prosesor adalah 19,86 % , kinerja memori adalah 24,2 %, dan *bandwidth* menunjukan angka 54,59 Mbps. Nilai-nilai inilah yang nantinya akan kita gunakan untuk mewakili kondisi *idle* dalam implementasi jaringan.

Hasil dari Tabel 4.7 diatas dapat kita tunjukan dalam bentuk grafik untuk memperlihatkan fluktuasi nilai yang terjadi pada komponen-komponen tersebut baik: prosesor, memori, maupun *bandwidth*. Gambar 4.16 memperlihatkan fluktuasi prosesor, memori, dan *bandwith* dalam kondisi *idle*.



Gambar 4.16 Grafik Lalu Lintas Jaringan dalam Kondisi *Idle*

Selanjutnya berikutnya kita akan menggunakan juga topologi yang sama seperti ditunjukan pada Gambar 4.15 untuk menentukan lalu lintas jaringan dalam kondisi menengah dan tinggi.

Pada percobaan untuk menentukan bandwidth pada lalu lintas jaringan menengah ditentukan dengan mentargetkan tercapainya *bandwidth* sebesar 70% dari besarnya *bandwidth* dalam kondisi *idle*. Dalam kondisi ini diharapkan dapat mewakili kondisi *bandwith* baik lalu lintas jaringan dalam rendah maupun menengah. Hasil ini nantinya yang kita kita ujikan pada saat terjadi pengujian pada kasus yang membutuhkan jaringan dengan lalu lintas data menengah.

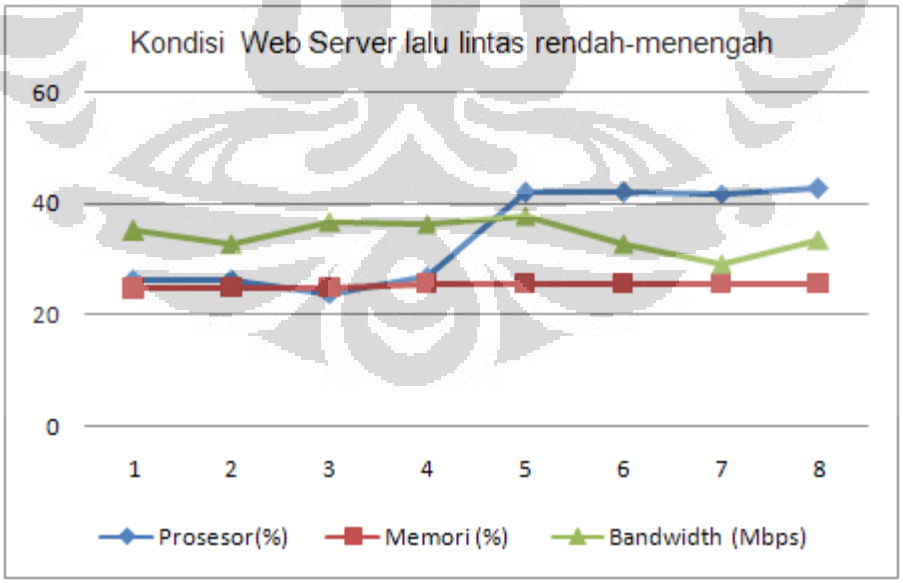
Parameter yang dihasilkan juga sama dengan pengujian *idle*, yaitu: kinerja prosesor, memori, serta bandwidth. Hal ini penting untuk tetap menjaga agar kinerja *web server* dapat tetap terjaga dengan baik.

Hasil dari pengujian untuk kondisi menengah ini seperti diperlihatkan pada Tabel 4.8 berikut ini.

Tabel 4.8 Pengukuran *Bandwidth* Kondisi Menengah

bytes	Prosesor(%)	Memori (%)	Bandwidth (Mbps)
50000	26.17	24.7	35.18
45000	26.13	24.9	32.76
40000	23.91	24.9	36.86
35000	27.11	25.5	36.48
20000	41.93	25.6	37.73
15000	41.95	25.5	32.78
10000	41.71	25.6	29.32
5000	42.63	25.5	33.54
Rata-rata	**33.94**	**25.28**	**34.33**

Dari Tabel 4.8 diatas diperlihatkan rata-rata nilai untuk kondisi menengah adalah, kinerja prosesor sebesar 33,94 %, kinerja memori 25,28 %, dan *bandwidth* sebesar 34,33 Mbps. Jika data-data tersebut ditunjukan dalam bentuk grafik maka akan didapatkan grafik seperti pada Gambar 4.17 berikut ini.



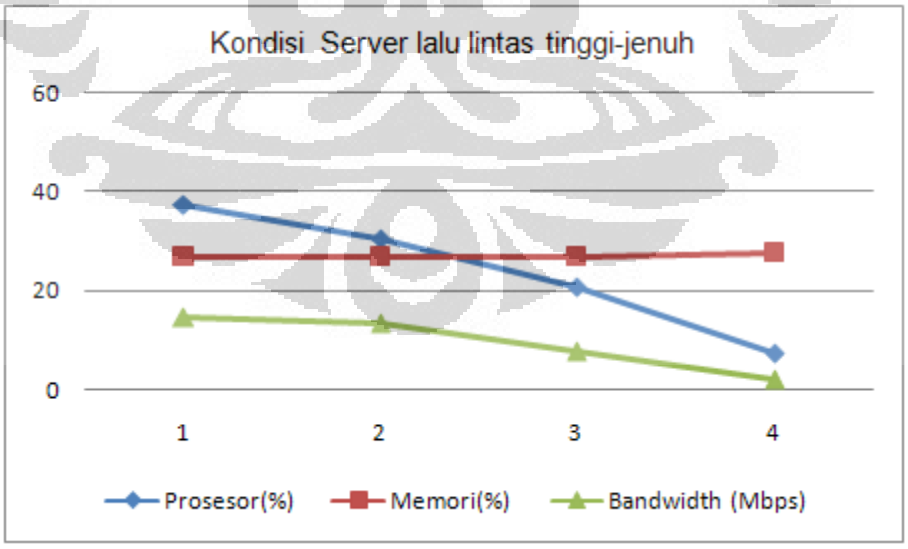
Gambar 4.17 Grafik Lalu Lintas Jaringan dalam Kondisi Menengah

Selanjutnya kita akan menentukan kondisi lalu lintas jaringan dalam kondisi padat. Kita ambil nilai sekitar sepuluh sampai lima belas persen dari kondisi *idle*. Dari percobaan kita dapat data-data seperti yang ditunjukan oleh Tabel 4.9 berikut ini.

Tabel 4.9 Pengukuran Bandwidth Kondisi Tinggi

bytes	Prosesor(%)	Memori(%)	Bandwidth (Mbps)
500	37.44	26.9	14.8
400	30.52	26.8	13.53
300	20.68	26.9	7.76
200	7.47	27.8	2
Rata-rata	**24.03**	**27.1**	**9.52**

Dari hasil pengukuran tersebut didapatkan bahwa pada kondisi yang mewakili lalu lintas padat didapatkan nilai rata-rata kinerja prosesor 24,03 %, kinerja memori 27,1 %, dan bandwidth 9,52 Mbps. Apabila data-data dari Tabel 4.9 tersebut ditunjukan dalam bentuk grafik, maka akan didapatkan grafik seperti yang ditunjukan dalam Gambar 4.18 berikut ini.



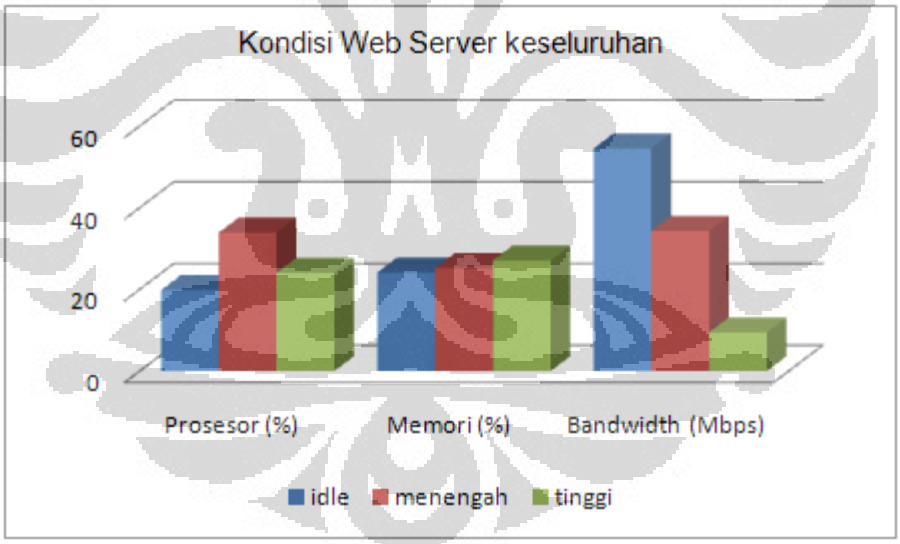
Gambar 4.18 Grafik Lalu Lintas Jaringan dalam Kondisi Tinggi

Dari hasil pengujian terhadap ketiga kondisi, baik: *idle*, menengah, maupun tinggi, dapat kita simpulkan data-data tersebut dalam sebuah tabel seperti yang ditunjukan pada Tabel 4.10 berikut ini.

Tabel 4.10 Parameter Lalu Lintas Jaringan Keseluruhan

lalu lintas	Prosesor (%)	Memori (%)	Bandwidth (Mbps)
idle	19.86	24.2	54.59
menengah	33.94	25.28	34.33
tinggi	24.03	27.1	9.52
Rata-rata	**25.94**	**25.52**	**32.81**

Nilai-nilai pada Tabel 4.10 diatas nanti akan kita gunakan sebagai parameter pada saat akan dilakukan pengujian terhadap jaringan dengan kondisi *idle*, menengah, dan tinggi. Data-data tersebut apabila ditunjukan dalam bentuk grafik akan seperti ditunjukan pada Gambar 4.19 berikut ini.



Gambar 4.19 Grafik Lalu Lintas Jaringan dalam Berbagai Kondisi

Dari Gambar 4.19 dapat dilihat bahwa kinerja prosesor terutama pada kondisi menengah maupun tinggi lebih tinggi nilainya daripada pada kondisi *idle*. Hal ini menunjukan adanya peningkatan pelayanan yang harus diberikan oleh prosesor pada kondisi menengah dan tinggi jika dibandingkan pada kondisi *idle*.

Demikian pula halnya untuk kinerja memori yang mengalami peningkatan pada kondisi menengah dan tinggi apabila dibandingkan dengan kondisi *idle*. Namun tidak demikian halnya dengan kondisi *bandwidth*, kondisi pada saat menengah dan tinggi memiliki nilai yang lebih rendah dibandingkan dengan kondisi *idle*. Hal ini menunjukan tingkat kepadatan yang dimiliki oleh kondisi menengah dan tinggi lebih besar dibandingkan dengan kondisi *idle*.

Selanjutnya dicoba untuk melakukan pemindaian terhadap *web server* dengan mengimplementasi tiga kondisi tadi, yaitu: *idle*, menengah, dan tinggi. Dari percobaan tersebut didapatkan data seperti ditunjukan pada Tabel 4.11 berikut ini.

Tabel 4.11 Kondisi Web Server Setelah Pemindaian

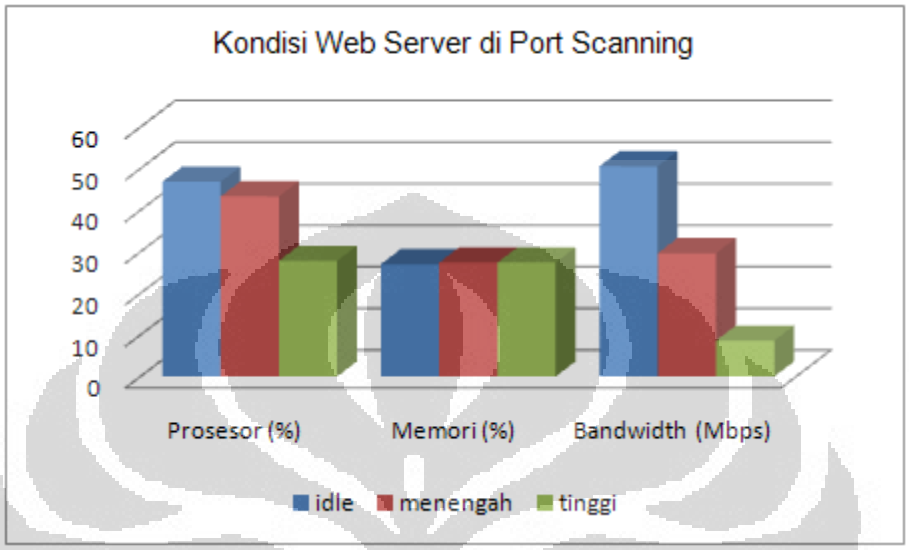
lalu lintas	Prosesor (%)	Memori (%)	Bandwidth (Mbps)
idle	46.81	27	50.54
menengah	43.3	27.4	29.64
tinggi	27.81	28.2	8.62
Rata-rata	**39.31**	**27.53**	**29.6**

Dari data-data yang dihasilkan pada berbagai kondisi lalu lintas jaringan, baik: *idle*, menengah dan tinggi, didapatkan nilai rata-rata untuk kinerja prosesor 39,31 %, kinerja memori 27,53 %, dan *bandwidth* bernilai 29,6 Mbps. Nilai-nilai ini nanti akan kita analisa dengan cara membandingkan dengan nilai-nilai yang didapatkan pada kondisi seperti ditunjukan pada Tabel 4.10 diatas.

Bila nanti sudah kita bandingkan dengan nilai pada Tabel 4.10 diatas, dapatlah kiranya kita menyimpulkan apakah aktifitas pemindaian ini merupakan aktifitas yang berbahaya dan dapat mengganggu aktifitas dari *web server* maupun pada lalu lintas jaringan yaitu terjadinya perubahan nilai pada *bandwidth*.

Dari data pada Tabel 4.11 dapat dilihat terjadinya penurunan nilai dari kinerja prosesor baik pada kondisi *idle*, menengah, maupun tinggi. Bila hal ini tidak segera diantisipasi oleh pengatur jaringan maka akan mengakibatkan *web server* akan lumpuh dan akan berhenti total sehingga segala bentuk layanan yang diberikan oleh *web server* tersebut akan terputus.

Data-data pada Tabel 4.11 diatas dapat kita tampilkan dalam bentuk grafik seperti ditunjukan pada Gambar 4.20 berikut ini.



Gambar 4.20 Grafik Web Server Setelah Pemindaian

Pada grafik di Gambar 4.20 terlihat bahwa terjadinya pemindaian selain menurunkan kinerja prosesor juga menyebabkan terjadinya penurunan pada *bandwidth*. Hal ini tidak boleh dibiarkan begitu saja oleh pengatur jaringan. Karena akan menyebabkan kondisi dari *web server* akan kehilangan fungsi secara cepat atau lambat, dan akan menyebabkan kerusakan yang lebih besar dan mungkin sistem akan berhenti secara total.

Akibatnya adalah layanan-layanan yang biasanya tersedia pada *web server* tersebut tidak dapat diakses oleh para pengguna yang membutuhkan layanan *web server* tersebut. Untuk mengantisipasi hal ini maka dalam sistem jaringan perlu ditambahkan satu sistem yang mampu mendeteksi dan melakukan tindakan pada saat terjadinya aktifitas ilegal dan yang mengganggu pada web server.

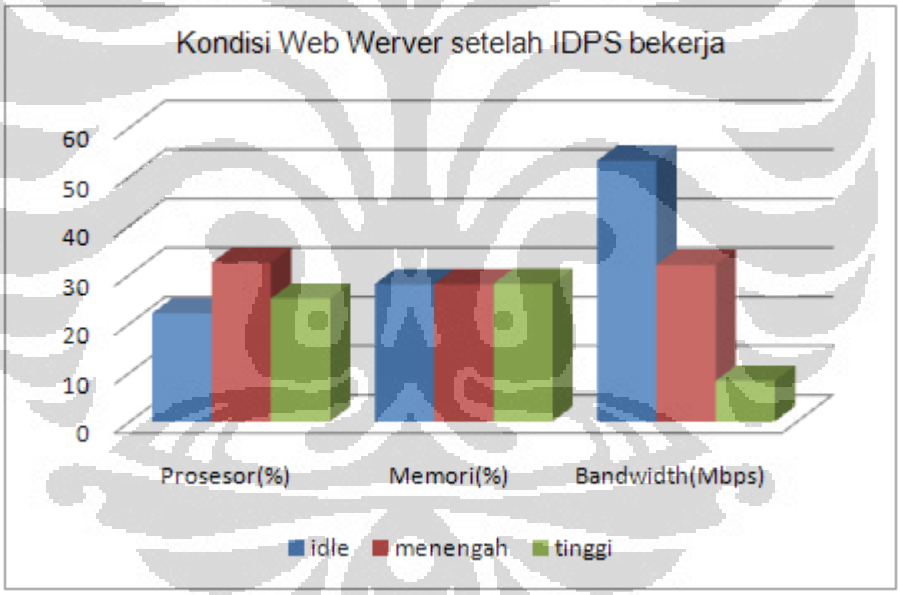
Untuk mengantisipasi adanya gangguan tersebut maka kita perlu mengaktifkan sistem *firewall* dan *intrusion detection prevention system* (IDPS) pada *web server* agar masalah-masalah gangguan tersebut dapat diantisipasi sejak awal.

Berikut ini diperlihatkan Tabel 4.12 yang menunjukan data-data setelah IDPS bekerja untuk mengantisipasi terjadinya pemindaian terhadap *web server*.

Tabel 4.12 Kondisi Web Server Setelah IDPS Bekerja

lalu lintas	Prosesor(%)	Memori(%)	Bandwidth(Mbps)
idle	22.08	25	53.17
menengah	32.33	25.6	31.96
tinggi	25.05	27.8	8.3
Rata-rata	**26.49**	**26.13**	**31.14**

Demikian pula ditampilkan grafik dari tabel 4.12 diatas, seperti ditunjukan pada Gambar 4.21 berikut ini.



Gambar 4.21 Grafik Web Server Setelah IDPS Bekerja

4.9 Hasil Analisa.

Dari pengujian fungsional yang dilakukan terhadap sistem IDPS, dapat dikatakan secara umum sistem IDPS berfungsi dengan baik, dalam pengertian bahwa sistem dapat menjalankan semua layanan yang diujikan dengan baik, seperti:

a. Untuk memulai fungsi mendeteksi adanya akses yang sah maupun tidak sah, sensor dalam hal ini *ethernet card* (eth0) berfungsi dengan baik.
b. Sistem dapat mengenali akses yang sah dan tidak sah serta mencatatnya dalam *log file*. Pemisahan antara *log file* pengguna yang sah dan tidak sah berjalan cukup efektif, terutama dalam mengurangi adanya *false positive*.
c. Sistem dapat mengambil tindakan pengamanan berupa tidak dapat diaksesnya *web server* oleh komputer yang melakukan pemindaian dan diputusnya koneksi melalui aktifitas *firewall*.

Sedangkan untuk menganalisa dan mengambil kesimpulan dari pengujian yang dilakukan pada jaringan *web server*, perlu kiranya kita rangkum beberapa data dari tabel-tabel sebagai berikut.

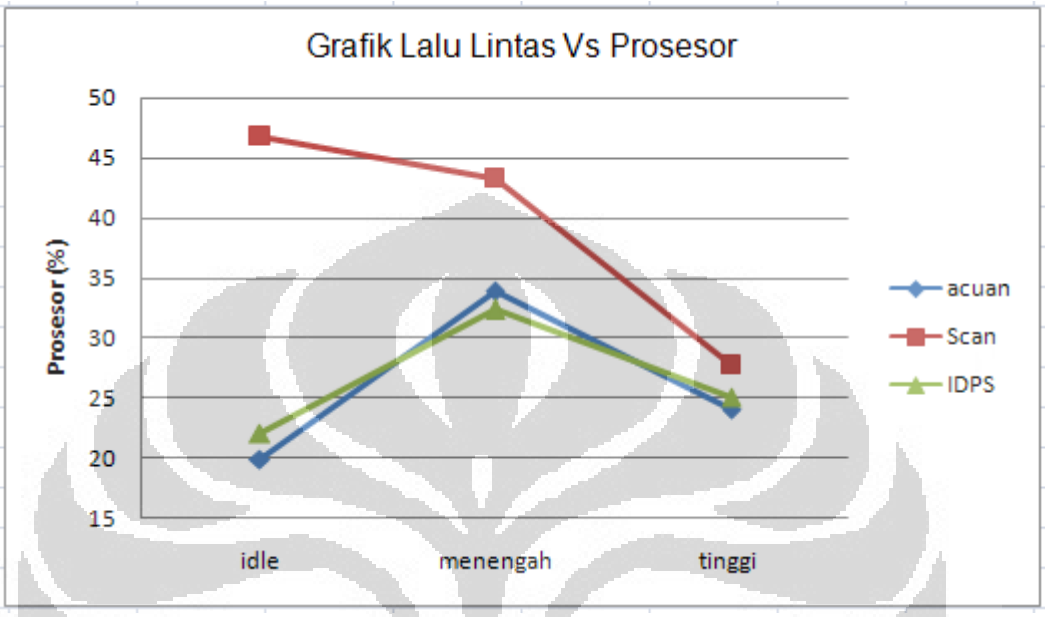
Tabel 4.13 Aktifitas Prosesor Dalam Berbagai Kondisi

Aktifitas	idle	menengah	tinggi
acuan	19.86	33.94	24.03
Scan	46.81	43.3	27.81
IDPS	22.08	32.33	25.05

Pada Tabel 4.13 diatas kita disajikan data tentang aktifitas prosesor pada *web server* dalam berbagai aktifitas. Akan kita lihat perbedaan antara kinerja prosesor pada nilai acuan dibandingkan dengan kondisi pada saat terjadi pemindaian dan pada saat setelah IDPS bekerja. Agar lebih mudah melihat perbedaan tersebut ada baiknya kita implementasikan data-data dalam Tabel 4.13 tersebut ke dalam bentuk grafik.

Grafik tersebut akan memperlihatkan kecenderungan ataupun karakteristik dari setiap parameter yang kita jadikan acuan, terhadap kondisi pada saat terjadinya pemindaian maupun pada saat setelah IDPS bekerja.

Kinerja prosesor pada Tabel 4.13 diatas dapat kita perlihatkan dalam bentuk grafik, seperti ditunjukan oleh Grafik 4.22 berikut ini.



Grafik 4.22 Lalu Lintas Vs Prosesor

Berikut ini juga disajikan Tabel 4.14 yang menunjukan tentang aktifitas memori dalam berbagai kondisi.

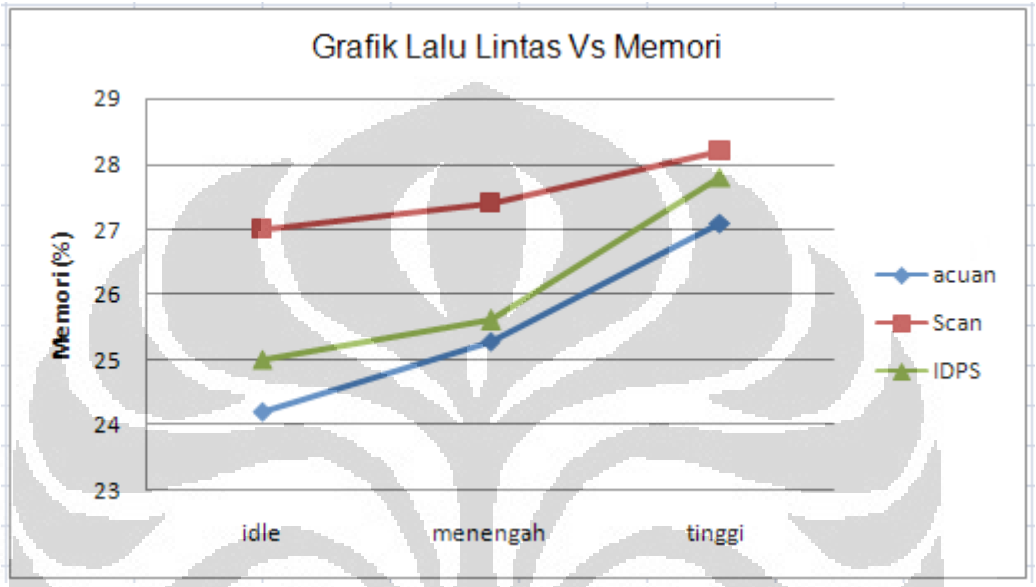
Tabel 4.14 Aktifitas Memori Dalam Berbagai Kondisi

Aktifitas	idle	menengah	tinggi
acuan	24.2	25.28	27.1
Scan	27	27.4	28.2
IDPS	25	25.6	27.8

Pada Tabel 4.14 diatas kita disajikan data tentang aktifitas memori pada *web server* dalam berbagai aktifitas. Akan kita lihat perbedaan antara kinerja memori pada nilai acuan dibandingkan dengan kondisi pada saat terjadi pemindaian dan pada saat setelah IDPS bekerja. Agar lebih mudah melihat

perbedaan tersebut ada baiknya kita implementasikan data-data dalam Tabel 4.14 tersebut ke dalam bentuk grafik.

Kinerja memori pada Tabel 4.14 diatas dapat kita perlihatkan dalam bentuk grafik, seperti ditunjukan oleh Grafik 4.23 berikut ini.



Grafik 4.23 Lalu Lintas Vs Memori

Berikutnya juga disajikan Tabel 4.15 yang menunjukan tentang aktifitas *bandwidth* dalam berbagai kondisi.

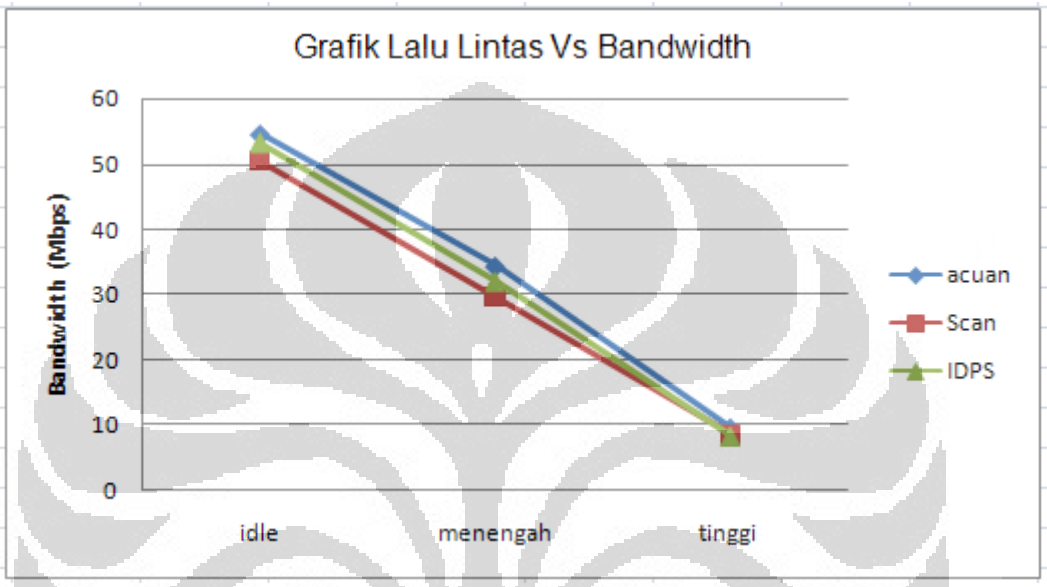
Tabel 4.15 Aktifitas *Bandwidth* Dalam Berbagai Kondisi

Aktifitas	idle	menengah	tinggi
acuan	54.59	34.33	9.52
Scan	50.54	29.64	8.62
IDPS	53.17	31.96	8.3

Pada Tabel 4.15 diatas kita disajikan data tentang aktifitas *bandwidth* pada *web server* dalam berbagai aktifitas. Akan kita lihat perbedaan antara kinerja *bandwidth* pada nilai acuan dibandingkan dengan kondisi pada saat terjadi pemindaian dan pada saat setelah IDPS bekerja. Agar lebih mudah melihat

perbedaan tersebut ada baiknya kita implementasikan data-data dalam Tabel 4.15 tersebut ke dalam bentuk grafik.

Kinerja *bandwidth* pada Tabel 4.15 diatas dapat kita perlihatkan dalam bentuk grafik, seperti ditunjukan oleh Grafik 4.24 berikut ini.



Grafik 4.24 Lalu Lintas Vs *Bandwidth*

Dari beberapa tabel dan grafik yang didapatkan dari percobaan pada jaringan *web server*, dapat kita ambil beberapa analisa sebagai berikut.

1. Pada ketiga kondisi lalu lintas yang diujikan baik pada saat kondisi *idle*, menengah, maupun tinggi terdapat kecenderungan yang sama pada saat dilakukan pemindaian dan setelah IDPS bekerja. Pada saat dilakukan pemindaian dalam semua kondisi lalu lintas yang ada parameter prosesor dan memori kinerjanya meningkat. Kecuali bandwith yang mengalami penurunan. Sehingga dari data-data tersebut kita bisa menyimpulkan bahwa dalam percobaan kita ini ternyata aktifitas pemindaian dapat meningkatkan kinerja prosesor dan memori pada *web server* serta menurunkan *bandwith* pada jaringan. Apabila pengatur jaringan tidak mengantisipasi hal ini maka dikhawatirkan web server akan lumpuh dan jaringan tidak akan berfungsi dengan baik. Oleh sebab itu disarankan agar

jaringan diatur untuk tetap berada pada bandwith menengah. Agar sumber daya yang ada dapat berfungsi maksimal dan baik.

2. Pada kondisi setelah IDPS bekerja terjadi penurunan nilai yang cukup berarti pada berbagai kondisi lalu lintas, yaitu: *idle*, menengah, dan tinggi untuk semua parameter, baik prosesor, memori, maupun *bandwidth.* Untuk prosesor dalam kondisi *idle*, nilai acuan 19.86 % setelah pemindaian naik menjadi 46.81 %, dan setelah IDPS bekerja turun menjadi 22.08 %. Prosesor dalam kondisi menengah, nilai acuan 33.94 % setelah pemindaian naik menjadi 43.3 %, dan setelah IDPS bekerja turun menjadi 32.33 %. Prosesor dalam kondisi Tinggi, nilai acuan 24.03 % setelah pemindaian naik menjadi 27.81 %, dan setelah IDPS bekerja turun menjadi 25.05 %. Sedangkan memori dalam kondisi *idle*, nilai acuan 24.2 % setelah pemindaian naik menjadi 27 %, dan setelah IDPS bekerja turun menjadi 25 %. Memori dalam kondisi menengah, nilai acuan 25.28 % setelah pemindaian naik menjadi 27.4 %, dan setelah IDPS bekerja turun menjadi 25.6 %. Memori dalam kondisi tinggi, nilai acuan 27.1 % setelah pemindaian naik menjadi 28.2 %, dan setelah IDPS bekerja turun menjadi 27.8 %. Pada *bandwidth* dalam kondisi *idle*, nilai acuan 54.59 % setelah pemindaian naik menjadi 50.54 %, dan setelah IDPS bekerja turun menjadi 53.17 %. Kondisi *bandwidth* dalam kondisi menengah, nilai acuan 34.33 % setelah pemindaian naik menjadi 29.64 %, dan setelah IDPS bekerja turun menjadi 31.96 %. Kondisi *bandwidth* dalam kondisi tinggi, nilai acuan 9.52 % setelah pemindaian naik menjadi 8.62 %, dan setelah IDPS bekerja turun menjadi 8.3 %. Dari data-data tersebut diatas dapat kita lihat bahwa pada saat setelah terjadinya pemindaian dan kemudian sistem IDPS bekerja dapat dilihat terjadi penurunan nilai dalam berbagai kondisi baik *idle*, menengah, maupun tinggi. Sehingga dapat diambil kesimpulan bahwa sistem IDPS kita selain dapat mendeteksi adanya aktifitas pemindaian sekaligus mengambil tindakan terhadap aktifitas pemindaian tersebut, serta mengembalikan kinerja *web server* dan jaringan rata-rata mendekati nilai 90%.

BAB 5
KESIMPULAN

Beberapa kesimpulan dapat diambil dari tulisan ini:

1. Untuk menunjang aktifitas yang sangat beragam bantuan akan jaringan komputer khususnya *web server* memang sangat diandalkan. Sebagai tempat mencari informasi dan pertukaran data untuk lebih mengefektifkan dan mengefisienkan dalam menyelesaikan berbagai macam pekerjaan. Oleh karena itu adanya gangguan atau ancaman yang bisa menyebabkan terjadinya kerusakan pada sistem *web server* akan menyebabkan kerugian yang sangat besar.
2. Pengamanan *web server* biasanya dilakukan dengan cara menggunakan *firewall*, namun ternyata hal itu saja belum cukup. *Firewall* memberlakukan setiap data secara umum dengan dua kondisi boleh akses atau tidak. Sehingga sulit untuk mendeteksi apabila serangan itu dilakukan oleh akses yang sah tetapi melampaui kewenangan yang diberikan padanya. Oleh sebab itu *firewall* harus disempurnakan, salah satunya dengan menambahkan perangkat IDPS untuk bekerjasama dengan *firewall* dalam melindungi jaringan komputer.
3. Dibuatnya sebuah antarmuka pemantauan yang berbasis grafik dapat memudahkan pengatur jaringan untuk memantau kondisi jaringan komputer serta menganalisis setiap kemungkinan gangguan atau ancaman yang mungkin terjadi.
4. Dari data-data yang didapatkan dari hasil pengujian IDPS didapatkan fakta bahwa setelah terjadinya aktifitas pemindaian, IDPS dapat mengembalikan sistem mendekati nilai normalnya dalam setiap kondisi. Saat kondisi lalu lintas data *idle*, IDPS mengembalikan nilai prosesor sistem rata-rata sebesar 91,76 % , memori sistem rata-rata 71,43 %, dan *bandwith* sistem rata-rata sebesar 97,4 %. Pada kondisi lalu lintas data menengah, IDPS mengembalikan nilai prosesor sistem rata-rata sebesar 83 % , memori sistem rata-rata 89 %, dan *bandwith* sistem rata-rata sebesar 93,1 %. Sedangkan pada kondisi lalu lintas data tinggi, IDPS mengembalikan nilai prosesor sistem rata-rata sebesar 73 % , memori sistem rata-rata 90 %, dan *bandwith* sistem rata-rata sebesar 87,18 %.

DAFTAR REFERENSI

[1] Braude, Eric J. 2001. Software Engineering an Object Oriented Perspective. New York : John Wiley & Sons, Inc.

[2] CEH Exam Objectives. 2007. Introduction to Ethical Hacking, Ethics, and Legality.

[2] Cullingford. 2009. Correlation of Network Behavior in Intrusion Detection Systems. Trusted Computer Solutions.

[3] Datasheet. 2010. Intrusion Prevention System (IPS). Enterasys Secure Network.

[4] Jacob V, Sreenivasa, Venkaiah. Agustus 2010. Intrusion Detection Systems Analysis and Containment of False Positive Alerts. International Journal of Computer Applications (0975 – 8887) Volume 5– No.8

[5] Jain, Raj. 2007. Intrusion Detection Systems. Saint Louis: Washington University.

[6] Karen Scarfone, and Peter Mell. Februari 2007. Guide to Intrusion Detection and Prevention System. Gaithersburg: National Institute of Standard and Technology.

[7] Khanchi, Sara, and Fazlollah Adibnia. 2009. False Alert Reduction on Network-based Intrusion Detection Systems by Means of Feature Frequencies. International Conference on Advances in Computing, Control, and Telecommunication Technologies.

[8] Nadeem Khawaja. November 2007. Intrusion Detection and Prevention System. University College University of Denver.

[9] Omar .A, Homam .E, Ahmed .M, Sureswaran .R. 2009. False Positif Reduction in Intrusion Detection System: A Survey. Universiti Sains Malaysia, National Advanced IPv6 Centre (NAV6). Proceeding of IC-BNMT.

[10] Rehman, Ur Rafeeq. 2003. Intrusion Detection Systems with Snort. New Jersey: Prentice Hall.

[11] Sourcefire, Inc. 2011. The Snort Project. Snort Users Manual 2.9.1

[12] "Zenmap GUI Users Guide". 2007. Types of Scanning. Nmap.org. < http://nmap.org/book/zenmap-scanning.html>