ISSN : 1410 - 7104

Implementasi Intrusion Detection System (IDS) Pada Keamanan PC Server Terhadap Serangan Flooding Data

Implementation of Intrusion Detection System (IDS) on PC Server Security Against Data Flood Attack

Achmad Hambali[1] dan Siti Nurmiati[2]
Fakultas Sains dan Teknologi Informasi, Institut Sains dan Teknologi Nasional, Jakarta
[1]Program Studi Teknik Informatika, FSTI – ISTN Jakarta
[2]Program Studi Sistem Informasi, FSTI – ISTN Jakarta
Email : [1]hambaliachmad95@gmail.com,[2]snurmiati@gmail.com

Abstrak --- *Flooding Data adalah jenis serangan Denial of Service (DOS) di mana data flooding menyerang komputer atau server di jaringan lokal atau internet dengan menghabiskan sumber daya yang dimiliki oleh komputer hingga komputer tidak dapat menjalankan fungsinya dengan baik sehingga tidak secara langsung mencegah pengguna lain dari mendapatkan akses ke layanan dari komputer yang diserang. Penelitian ini untuk menganalisis indikasi serangan dan menjaga keamanan sistem dari ancaman banjir data. Untuk itu kita membutuhkan alat deteksi yang dapat mengenali keberadaan serangan flooding data dengan mengetuk paket data dan kemudian membandingkannya dengan aturan basis data IDS (berisi paket serangan tanda tangan). Mesin IDS akan membaca peringatan dari IDS (seperti jenis serangan dan penyadap alamat IP) untuk meminimalkan data serangan flooding terhadap LAN (Local Area Network) dan server. Metode pengujian data serangan banjir dengan menggunakan metode pengujian penetrasi. Tiga sampel uji adalah serangan flooding data terhadap ICMP, UDP dan protokol TCP menggunakan aplikasi Flooding data. Hasil yang diperoleh ketika menguji data serangan flooding di mana sensor sensor deteksi dapat mendeteksi semua serangan dan semua sampel serangan dapat dicegah atau disaring menggunakan sistem keamanan jaringan berbasis firewall.*

Kata kunci: *Intrusion Detection System (IDS), Denial of Service (DOS), Data Banjir, Firewall, (LAN) Local Area Network*

Abstract --- *Flooding Data is a type of Denial of Service (DOS) attack in which flooding data attacks a computer or server in a local network or internet by spending resources owned by the computer until the computer can not perform it's function properly so that does not directly prevent other users from gaining access to services from the computer being attacked. This research is to analyze the indication of attack and maintain the security of a system from the threat of flooding data. For that we need a detection tool that can recognize the existence of data flooding attacks by tapping the data package and then compare it with IDS database rule (contains signature attack packets). IDS engine will read alerts from IDS (such as attack type and IP address intruder) in order to minimize flooding attack data against LAN (Local Area Network) and server. Method of testing flooding attack data by using penetration testing method. The three test samples are data flooding attacks against ICMP, UDP and TCP protocols using data Flooding applications. Results obtained when testing flooding attacks data where detection sensor sensors can detect all attacks and all attack samples can be prevented or filtered using a firewall-based network security system.*

Keywords*: Intrusion Detection System (IDS), Denial of Service (DOS), Flooding Data, Firewall, (LAN) Local Area Network*

1. PENDAHULUAN

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

2. LANDASAN TEORI

2.1 *Flooding Data*

Pengiriman data yang berlebihan baik dari besar paket maupun jumlah paket kedalam suatu jaringan dan umumnya merupakan data yang tidak berguna disebut dengan *Flooding Data*, ada kalanya data

yang berbeda dalam *traffic* merupakan data yang tidak perlu. Data tersebut memang sengaja dikirim oleh seseorang meneruskan jaringan data yang ada. Pengiriman data tersebut dapat mengakibatkan lambatnya jalur *traffic* yang ada dalam jaringan dan juga bisa mengakibatkan kerugian lain yang cukup berarti, misalnya kerusakan *program* karena adanya *intruder* yang masuk kedalam jaringan. *Traffic* data yang ada dalam suatu jaringan akan mengalami turun naik selama pemakaiannya. Pada jam sibuk *traffic* suatu data akan sangat padat, sehingga *traffic* data tersebut akan terganggu. Baik data yang akan dikirim maupun data yang akan datang akan mengalami antrian data yang mengakibatkan keterlambatan dalam pengiriman dan penerimaan data. *Traffic* data yang ada dalam suatu jaringan akan mengalami turun naik selama pemakaiannya. Pada jam sibuk *traffic* suatu data akan sangat padat, sehingga *traffic* data tersebut akan terganggu. Baik data yang akan dikirim maupun data yang akan datang akan mengalami antrian data yang mengakibatkan kelambatan dalam pengiriman dan penerimaan data.

2.1.1 Macam-macam *Flooding Attack*

1. *Ping of Death*

Pengiriman paket *echo request* ICMP kedalam suatu jaringan secara berlebihan. Pengiriman paket ini dapat mengakibatkan sistem *crash*, *hang* atau *reboot.*

2. *Smurf Attack*

Hampir sama dengan *ping of death* tetapi untuk *smurf attack* paket ICMP tidak dikirim secara langsung ke korban, melainkan melalui perantara. Pada awalnya dikirim sebuah paket ICMP *echo request* kesebuah *host* lain, paket ini bertujuan agar host tersebut mengirimkan paket ICMP *ping* secara terus menerus ke korban terakhirnya.

3. *SYN Flooding*

SYN Flooding terjadi bila suatu host hanya mengirimkan paket *SYN TCP* saja secara kontinyu tanpa mengirimkan paket ACK sebagai konfirmasinya. Hal ini akan menyebabkan *host* tujuan akan terus menunggu paket tersebut dengan menyimpannya kedalam *backlog*. Meskipun besaran paketnya kecil, tetapi apabila pengiriman *SYN* tersebut terus menerus akan memperbesar *backlog*. Hal yang terjadi apabila *backlog* sudah besar akan mengakibatkan *host* tujuan akan otomatis menolak semua paket *SYN* yang datang, sehingga *host* tersebut tidak bisa dikoneksi oleh *host* yang lain.

4. *UDP Flooding*

Pengiriman data *UDP* secara berlebihan kedalam suatu jaringan, pengiriman *UDP flood* ini akan membentuk suatu jalur hubungan dengan suatu *servis UDP* dari *host* tujuan. *Flood UDP* ini akan mengirimkan karakter yang akan mengetes jaringan korban. Sehingga terjadi aliran data yang tidak perlu dalam jaringan korban tersebut.

2.2 ICMP (*Internet Control Message Protocol*)

ICMP (*Internet Control Message Protocol*) adalah protokol yang bertugas mengirimkan pesan kesalahan dan kondisi lain yang memerlukan perhatian khusus. Pesan atau paket ICMP dikirim jika terjadi masalah pada *layer* IP dan *layer* atasnya (TCP/UDP). Pada konsisi normal, protokol IP berjalan dengan baik. Namun ada beberapa kondisi dimana koneksi IP terganggu, misalnya karena *router crash*, putusnya kabel atau matinya *host* tujuan. Pada saat ini ICMP membantu menstabilkan kondisi jaringan, dengan memberikan pesan tertentu sebagai respons atas kondisi tertentu yang terjadi pada jaringan tersebut.

2.3 TCP (*Transmission Control Protocol*) dan UDP (*User Datagram Protocol*)

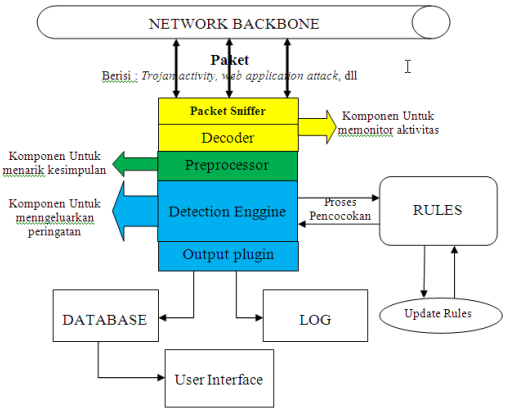
TCP (*Transmission Control Protocol*) adalah protokol yang paling umum digunakan pada dunia internet, karena kelebihan TCP yaitu adanya koreksi kesalahan. Dengan menggunakan protokol TCP, maka proses pengiriman akan terjamin. Hal ini disebabkan adanya bagian untuk sebuah metode yang disebut *flow control*. *Flow control* menentukan kapan data harus dikirim kembali dan kapan menghentikan aliran data paket sebelumnya, sampai data tersebut berhasil ditransfer. Hal ini karena jika paket data berhasil dikirim, dapat terjadi sebuah 'tabrakan'. Ketika ini terjadi, maka klien akan meminta kembali paket dari *server* sampai seluruh paket berhasil ditransfer dan identik dengan aslinya. Sedangkan UDP (*User Datagram Protocol*) adalah protokol umum lainnya yang digunakan pada dunia internet dan merupakan *connectionless*. Hal ini berarti bahwa suatu paket yang dikirim melalui jaringan hingga sampai ke komputer lain tanpa membuat suatu koneksi. UDP tidak pernah digunakan untuk mengirim data penting seperti halaman *web*, informasi *database* dan sebagainya. UDP biasanya digunakan untuk streaming *audio* dan *video*, karena kelebihan UDP yaitu menawarkan kecepatan transfer. UDP dapat lebih cepat daripada TCP karena pada protocol UDP tidak ada bentuk kontrol aliran dan koreksi kesalahan. Artinya UDP tidak mementingkan bagaimana keadaan koneksi, jadi jika terjadi pengiriman data maka tidak dijamin berhasil sampai atau tidaknya data tersebut. Pada UDP juga tidak ada pemecahan data, oleh karena itu tidak dapat melakukan pengiriman data dengan ukuran yang besar.

2.4 *Intrusion Detection System* (IDS)

Intrusion Detection System (IDS) merupakan sebuah metode yang dapat digunakan untuk mendeteksi aktivitas yang mencurigakan dalam sebuah sistem atau jaringan. IDS dapat melakukan inspeksi terhadap lalu lintas *inbound* dan *outbound*

dalam sebuah sistem atau jaringan, melakukan analisis dan mencari bukti dari percobaan intrusi (penyusupan).

Cara kerja IDS yang dilakukan dalam penulisan ini adalah dengan *Signature* (*Misuse*) *Based* IDS yang berbasis pada *signature* akan melakukan pengawasan terhadap paket dalam jaringan dan melakukan perbandingan terhadap paket tersebut dengan basis data *signature* yang dimiliki oleh sistem IDS ini atau atribut yang dimiliki oleh percobaan serangan yang pernah diketahui. Cara ini hampir sama dengan cara kerja aplikasi *antivirus* dalam melakukan deteksi terhadap *malware*. Intinya adalah akan terjadi keterlambatan antara terdeteksinya sebuah serangan di internet dengan *signature* yang digunakan untuk melakukan deteksi yang diimplementasikan didalam basis data IDS yang digunakan.



Gambar 1. Cara Kerja IDS

2.5 *Snort*

Snort merupakan suatu perangkat lunak untuk mendeteksi penyusup dan mampu menganalisa paket yang melintas pada jaringan secara *real time traffic* dan *logging* kedalam *database* serta mampu mendeteksi berbagai serangan yang berasal dari dalam jaringan maupun diluar jaringan. *Snort* adalah sebuah program yang memiliki tiga fungsi atau tiga modus operasi. *Snort* dapat dipakai dalam *packet sniffer mode* sehingga bekerja sebagai *sniffer* sama seperti *Wireshark*. Sama seperti *Wireshark*, *Snort* juga dapat menyimpan setiap *packet* yang di *capture* ke dalam media penyimpan di modus *packet logger mode.* Akan tetapi, berbeda dengan *Wireshark*, *Snort* dapat dipakai sebagai komponen NIDS dengan menjalankannya pada *Network Intrusion Detection System* (NIDS) mode. Pada modus yang terakhir ini, *Snort* akan menganalisa *packet* berdasarkan *rule* yang ada untuk mengenali adanya upaya serangan atau *Intrusion.*

2.6 *Linux Ubuntu Server*

Linux Ubuntu Server adalah sistem operasi turunan dari *Linux Ubuntu* yang di desain khusus dengan *kernel* yang telah dikustomisasi untuk bekerja sebagai sistem operasi *server*. *Kernel Linux Ubuntu Server* di desain khusus untuk bisa bekerja dengan lebih dari satu proses (*multiprocessor*) dengan dukungan NUMA pada 100 Hz *internal timer frequency* dan menggunakan penjadwalan *deadline* I/O. *Linux Ubuntu Server* memiliki lisensi *open source* dan gratis serta merupakan turunan dari *distro linux debian* sehingga memiliki keamanan yang cukup tinggi. *Linux Ubuntu Server* ini mempunyai kebutuhan *minimum* atau *resource* yang harus dipenuhi diantaranya adalah *processor* 300 MHz, *Memory* 64 MB, *Harddisk* 500 MB dan VGA 640×480. Namun untuk meningkatkan kinerja pada komputer *resource* pada komputer harus disediakan lebih tinggi. Berikut kelebihan dan kekurangan *Ubuntu Server*.

2.7 Metode Blokir IP

Untuk metode Blokir IP menggunakan *tools* yang sudah ada pada *Linux Ubuntu Server* yaitu, *IPTABLES*. Pada *tools* ini tersedia fungsi untuk *routing* baik itu *accepting*, *forwarding* dan *blocking*.

3. METODOLOGI PENELITIAN

3.1 Analisa Kebutuhan

3.1.1 Bahan Penelitian

Bahan penelitian diperoleh dari berbagai literatur yang bersumber dari buku, jurnal ilmiah, situs internet dan bacaan lainnya yang berkaitan dengan perancangan yang akan dilakukan.

3.1.2 Alat Penelitian

1. Spesifikasi Perangkat Keras (*Hardware*)

Untuk menjalankan sistem keamanan ini digunakan dua unit komputer, yang satu merupakan komputer *server* dan yang satu lagi adalah komputer *client* dengan spesifikasi minimal sebagai berikut :

- PC *Server* : *AMD E-350* 1,6 GHz, *Harddisk* 320 GB, RAM 4 GB
- PC *Client* : *Intel Atom* 1.7 GHz, *Harddisk* 350 GB, RAM 2 GB

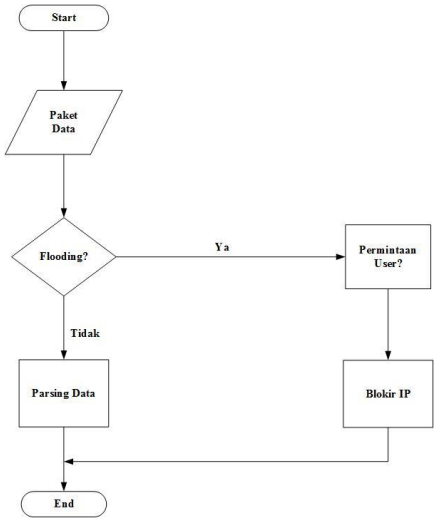
2. Spesifikasi Perangkat Lunak (*Software*)

Adapun untuk spesifikasi perangkat lunak yang digunakan untuk menjalankan sistem ini adalah sebagai berikut :

- PC *Server* : *Ubuntu Server* 16.04, *Snort*, *Barnyard2*, *PulledPork*, ACID, Adodb, Jpgraph, BASE, *MySql*
- PC *Client* : *Windows* 7, *Nmap*, *Command Prompt*

3.2 Desain Sistem Secara Umum

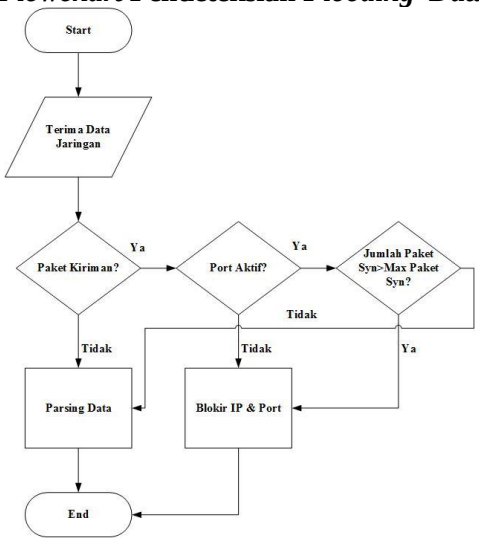
Secara umum sistem yang akan dibangun adalah sebagai berikut :



Gambar 2. *Flowchart* Sistem Secara Umum

Pada gambar 2. Menjelaskan mengenai *Input* dari program adalah data jaringan yang masuk kemudian akan di proses apakah data tersebut melakukan *flooding* atau tidak. Jika data yang datang adalah *flooding* maka alamat IP asal paket dan *port* yang digunakan akan di blok. Jika tidak melakukan *flood* maka paket akan diteruskan.

3.3 *Flowchart* Pendeteksian *Flooding Data*

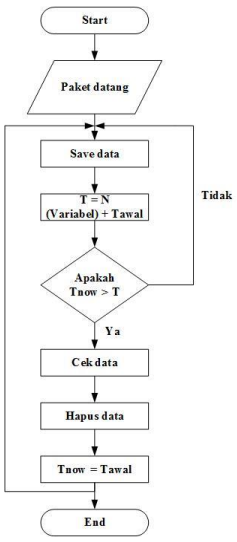


Gambar 3. *Flowchart* Pendeteksian *Flooding Data*

Pada gambar 3 menjelaskan mengenai pertama sistem akan melihat apakah data merupakan data kiriman atau data dari dalam. Jika data merupakan data kiriman, maka dicek *port* yang digunakan tersedia atau tidak. Jika tidak tersedia maka langsung di blok. Jika *port* tersedia, paket tersebut di cek, apakah paket tersebut merupakan paket TCP SYN. Jika paket TCP SYN lebih besar dari TCP SYN maksimum yang di tetapkan, maka akan diblok.

3.4 Desain *Database*

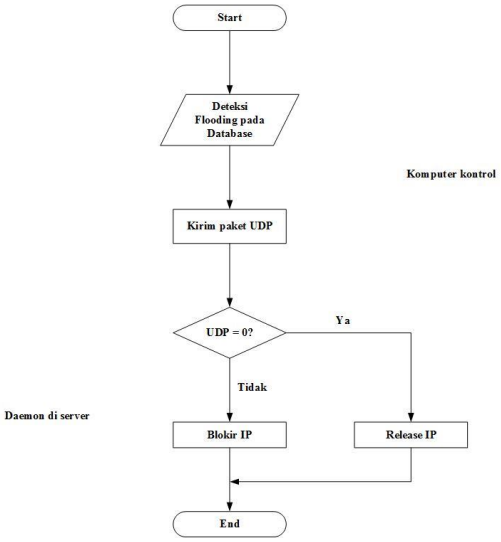
Apabila ada data dari paket masuk akan di ambil kemudian disimpan dalam *database*. Dalam hal ini *database* bersifat sementara dan disimpan hanya di *memory* bukan di *harddisk* karena penyimpanan dalam *harddisk* membutuhkan proses yang cukup lama. Oleh sebab itu data yang sudah tidak diperlukan lagi akan terhapus. Hanya data yang dalam selang waktu yang ditentukan saja yang diolah. Untuk *defaultnya* diambil angka 10. Suatu jaringan komputer dengan *user* 1 atau lebih tentu data yang melewati *router* dalam 10 detik sangatlah banyak, terutama pada jam sibuk, karena pengiriman data dalam jaringan adalah paket yang terpecah, bukan dalam suatu *file* atau f*older*. Sehingga untuk satu koneksi dari *host* ke *host* paket yang tercatat akan banyak. Semakin banyaknya data paket data yang tersimpan maka kebutuhan akan *memory* pun semakin besar. Berikut adalah pengaturan data dari paket dalam *database* :



Gambar 4. *Flowchart Database Umum*

3.5 Desain Pemblokiran IP

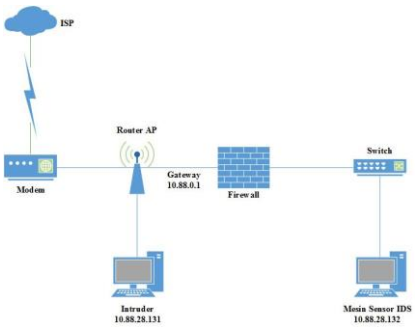
Apabila paket data yang datang terbukti merupakan *flooding*, maka sistem akan mengirim paket UDP ke *server* untuk memberikan perintah blokir kepada IP yang mengirimkan paket data tersebut. *Server* sudah di *install* program *daemon* dan dijalankan, program itu akan menanyakan apakah paket UDP sama dengan nol, jika sama maka data tidak akan diblok dan akan diteruskan ketujuannya, namun jika tidak maka data otomatis akan diblokir.



Gambar 5. Desain Pemblokiran IP

3.6 Perancangan Topologi Jaringan

Design atau perancangan ini mengacu pada konsep dan gambaran perangkat sebenarnya dalam suatu sistem yang digambarkan dalam penulisan ini dengan topologi berikut :



Gambar 6. Topologi Jaringan

Rincian keterangan dari gambar 6 adalah sebagai berikut :

1. Jenis topologi yang diterapkan ialah *Star*.
2. IP *Addres* yang digunakan ialah kelas C.
3. Kabel *straight* untuk menghubungkan *switch* atau *hub* dengan *host*, *router* dengan *hub* atau *switch*.

Dalam penelitian ini kedua jenis kabel tersebut dibutuhkan untuk menghubungkan perangkat jaringan yang digunakan. Berikut penjelasan jenis kabel yang digunakan untuk menghubungkan setiap perangkatnya.

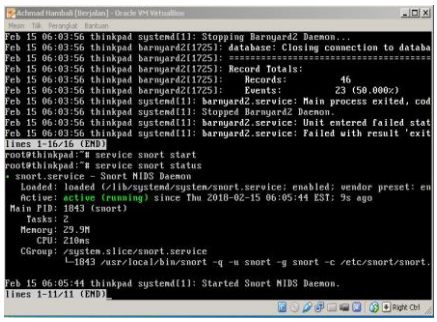
1. Jenis kabel yang digunakan untuk menghubungkan *router* dan komputer IDS adalah tipe *straight*.
2. Dari komputer IDS ke switch menggunakan kabel *straight*.
3. Jenis kabel yang digunakan untuk menyambungkan antara switch dengan *client* adalah tipe *straight*.

Tipe koneksi yang digunakan dari komputer intruder menuju *router* adalah melalui media transmisi *wireless*.

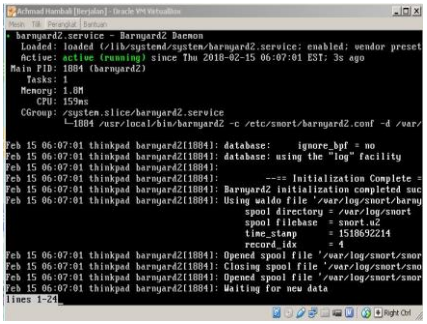
4. HASIL DAN PEMBAHASAN

4.1 Pengujian *Snort*

Pengujian *snort* pada mesin sensor IDS dilakukan dengan menggunakan *rules* yang sudah ada. Dimana pengujian tersebut dengan melakukan attack atau serangan ke mesin sensor IDS. Sebelum melakukan pengujian aktifkan dahulu *snort* dan *barnyard2* dengan menulis *command* atau perintah *service snort start* dan *service barnyard2 start*. Lalu setelah itu lihatlah status dari *snort* apakah sudah aktif atau belum dengan menulis *command* atau perintah *service snort status* dan *service barnyard2 status.* Jika sudah aktif atau berjalan maka akan terlihat seperti gambar 7 dan 8.



Gambar 7. *Snort* Sudah Aktif Atau Berjalan



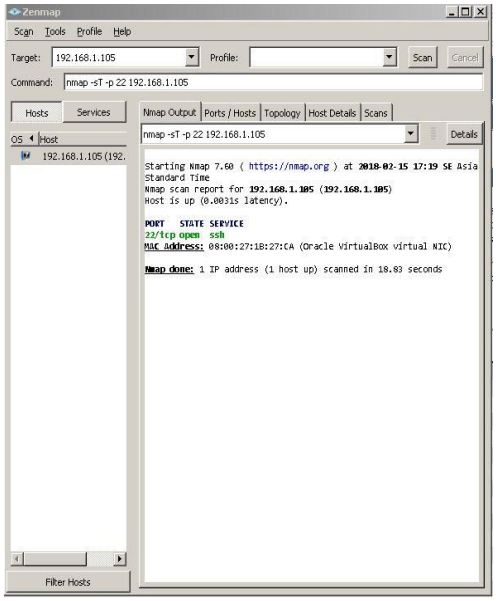
Gambar 8. *Barnyard2* Sudah Aktif Atau Berjalan

4.1.1 Pengujian 1 : *Ping Attack* (ICMP *Traffic*)

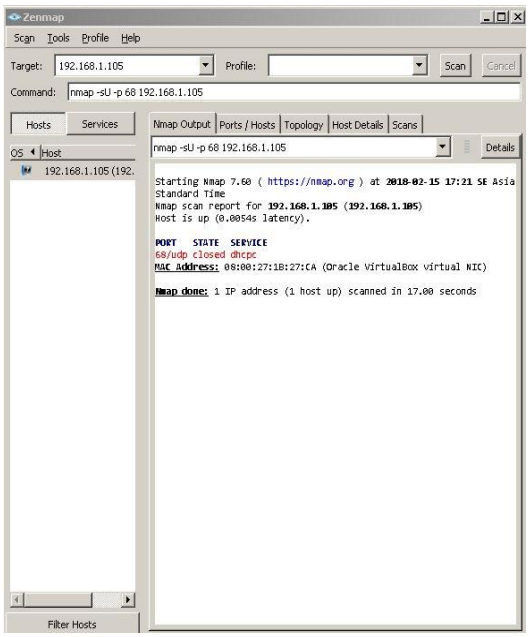
Gambar 9. *Ping Attack*

Pada gambar 9 menjelaskan mengenai Dalam kasus ini, jenis serangan dan pengujian disimulasikan dengan serangan berprotokol ICMP. Pada mesin sensor IDS, dimana *client* mencoba *ping attack* ke alamat IP mesin sensor IDS yaitu 192.168.1.105 dengan menggunakan cmd atau *command prompt*.

4.1.2 Pengujian 2 : *Nmap Port Scanning*



Gambar 10. Pengujian *Nmap* ke Mesin Sensor IDS Dengan Protokol TCP

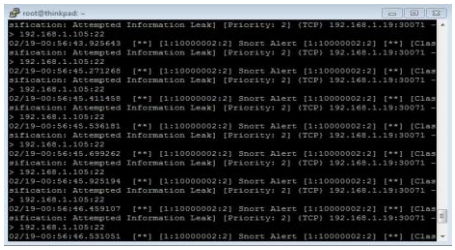


Gambar 11. Pengujian *Nmap* ke Mesin Sensor IDS Dengan Protokol UDP

Pada gambar 11 menjelaskan mengenai Pada kasus yang kedua, jenis serangan dan pengujian, disimulasikan dengan metode *port scanning* menggunakan aplikasi *Nmap* yang dilakukan dari *client* atau mesin penyerang menggunakan protokol TCP seperti yang terlihat pada gambar dan UDP yang terlihat pada gambar 10 dan 11. Pengujian kali ini, *client* mencoba *port scanning* dengan menggunakan *Nmap* ke mesin sensor IDS dengan IP *addres* 192.168.1.105. Aplikasi *Nmap* yang sudah ter*install* pada mesin *client* berfungsi untuk menguji coba mesin sensor IDS dengan target IP *addres*s 192.168.1.105. Pada gambar 10 dan 11 saat *scanning* dilakukan oleh *client* ke mesin sensor IDS bahwa mesin sensor IDS pada *port scanning* yang dilakukan oleh *client* dengan menggunakan aplikasi *Nmap* ke mesin sensor bahwa ada *port* yang terbuka pada mesin sensor IDS yaitu *port* 22 dan 68.

4.1.3 Hasil Pengujian *Snort*

Sistem keamanan yang berupa *Intrusion Detection System* (IDS) ini merupakan sistem yang didalamnya memiliki sistem keamanan dan pendeteksi serangan yang cukup baik. Sistem ini dirancang agar bila *server* diserang oleh *client* atau dari perangkat komputer lain dapat segera langsung mendeteksi serangan tersebut dengan baik. Karena sistem ini tidak memiliki *Graphic User Interface* (GUI), maka dalam penelitian ini penulis menggunakan *tools* tambahan untuk dapat menampilkan hasil dari deteksi dan pencegahan yang dilakukan oleh sistem tersebut, yaitu dengan menggunakan BASE (*Basic Analysis and Security Engine*) yang dapat menampilkan informasi dalam tampilan *web browser*. Laporan ini dibuat untuk dapat mengetahui siapa saja yang melakukan penyerangan terhadap *server*. Diwah ini adalah bentuk tampilan peringatan adanya serangan yang dilakukan oleh *intruder* dalam bentuk CUI (*Command User Interface*).

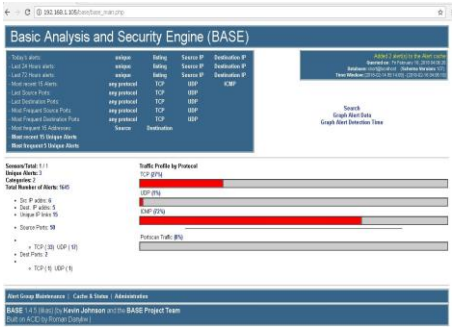


Gambar 12. Tampilan Serangan Dalam Bentuk CUI (*Command User Interface*)

Pada gambar 12 merupakan tampilan dari alert serangan dalam bentuk CUI (*Command User Interface*). Bisa dilihat bahwa serangan berasal dari alamat IP 192.168.1.19 dengan protokol TCP dengan target yang beralamat IP 192.168.1.105 yaitu mesin sensor *snort*. Namun, dalam penelitian ini menggunakan aplikasi tambahan yang berbasis *web*, yaitu BASE (*Basic Analysis and Security Engine*) untuk menampilkan GUI (*Graphic User Interface*) agar lebih menunjang dalam proses analisis data kejadian dan lebih efektif dalam memonitoring serangan yang masuk. BASE Akan dijelaskan pada sub bab 4.1.4.

4.1.4 Analisis Data BASE (*Basic Analysis and Security Engine*)

Sub bab ini menjelaskan proses analisis data kejadian melalui fugsionalitas BASE. Di bawah ini adalah analisis data BASE.



Gambar 13. Total Jumlah *Alert* Serangan

Pada gambar 13 adalah halaman utama BASE yang berisi total jumlah *alert* serangan, lalu disisi kiri atas terdapat *link* yang menjelaskan informasi mengenai *alert* atau pemberitahuan tentang apa yang terjadi hari ini, 24 jam terakhir dan 72 jam terakhir yang dapat ditampilkan berdasarkan perimeter *unique*, *listing*, IP *address* berdasarkan sumber dan tujuan.

Disisi kanan atas gambar terdapat informasi search dimana pencarian berdasarkan waktu, jam dan bulan. Pada *graph alert* data menjelaskan informasi berdasarkan grafik pemberitahuan dan pada *graph alert detection time* berdasarkan waktu yang sudah dikonfigurasikan.

Pada sisi kanan bawah menginformasikan *traffic profile by protocol* atau trafik berdasarkan protokol dan pada sisi kiri bawah menginformasikan jumlah sensor, *unique alert*, jumlah *alert* dan IP *address* berdasarkan sumber tujuan dan *unique alert*. Pada *traffic profile by protocol* dilihat dari protokol TCP dan *port scanning traffic* terjadi peningkatan sinyal berwarna merah yang artinya pada protokol TCP terjadi presentase alert sebesar 27% dan pada protokol ICMP presentase alert sebesar 72%.

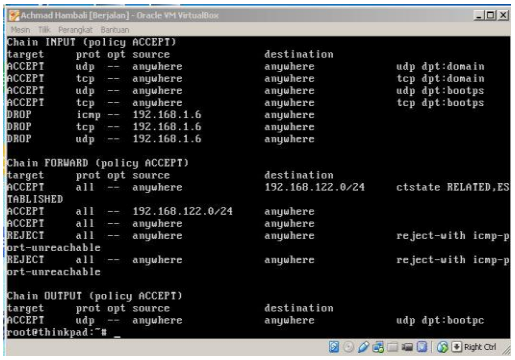
4.2 Solusi mengatasi *Ping Attack* dan *Port Scanning*

Untuk mengatasi *ping attack* dan *Port Scanning* dari intruder, dalam penulisan ini menggunakan sebuah *rule IPTABLES* untuk memblokir berdasarkan IP *Addres*.

```
root@thinkpad:~# sudo iptables -A INPUT -s 192.168.1.6 -p icmp -j DROP
root@thinkpad:~# sudo iptables -A INPUT -s 192.168.1.6 -p tcp -j DROP
root@thinkpad:~# sudo iptables -A INPUT -s 192.168.1.6 -p udp -j DROP
root@thinkpad:~# _
```

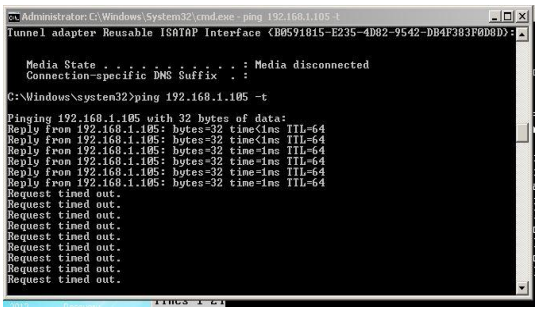
Gambar 14. *Command* Untuk Blokir IP Pada *IPTABLES*

Pada gambar 14 adalah *command* atau perintah untuk memblokir IP *addres* 192.168.1.6 yang menyerang mesin sensor dengan protokol ICMP, TCP dan UDP. Jika sudah, coba tulis *command* atau perintah iptables -L untuk melihat konfigurasi *IPTABLES* yang sudah dibuat, seperti gambar 15.



Gambar 15. IP *Address* yang Sudah Diblokir Pada *IPTABLES*

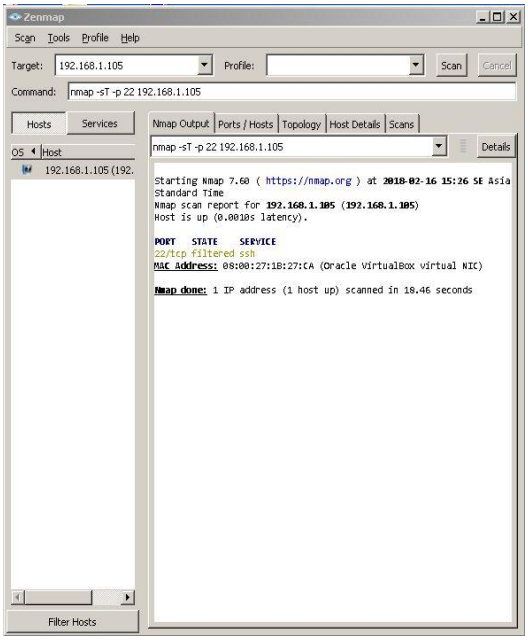
Pada gambar 15 Jika pemblokiran IP *address* berhasil, akan terlihat jika alamat IP 192.168.1.6 dengan protokol ICMP, TCP dan UDP telah berhasil di *DROP* seperti yang terlihat pada sisi kiri atas gambar. Setelah itu, akan terlihat pada mesin *client* atau penyerang tulisan *request time out*, seperti gambar 16.



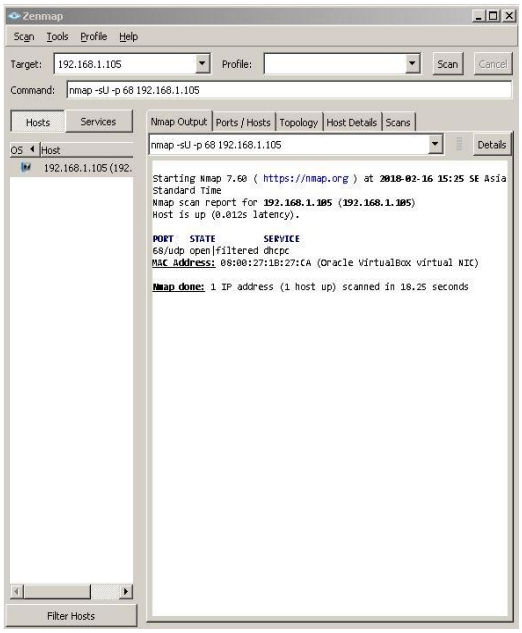
Gambar 16. Memblok *Client* yang Melakukan *Ping Attack*

Pada Gambar 4.10 menandakan bahwa blokir IP dengan menggunakan *IPTABLES* telah berhasil dilakukan. Terlihat pada saat *ping* ke alamat IP 192.168.1.105 yang merupakan alamat IP mesin sensor sedang berjalan, secara mendadak keluar tulisan RT0 atau *request time out* pada mesin *client* atau penyerang yang artinya *client* tidak dapat melakukan *ping* ke alamat IP yang dituju.

Pada mesin *client* atau penyerang saat melakukan *port scanning* dengan menggunakan *tools Nmap*, *port* ter*filtered* atau tidak terbuka, seperti gambar 17 dan 18.



Gambar 17. Tampilan *Nmap* Pada Mesin *Client* Dengan Protokol TCP Yang Telah Diblokir



Gambar 18. Tampilan *Nmap* Pada Mesin *Client* Dengan Protokol UDP Yang Telah Diblokir

Pada gambar 17 menjelaskan bahwa, *client* atau penyerang tidak dapat melakukan *port scanning* ke alamat IP 192.168.1.105 dilihat dari tabel keterangan bahwa *port* 22 dengan protokol TCP telah ter*filtered*, begitu juga dengan gambar 18 yang menjelaskan bahwa, *client* atau penyerang tidak dapat melakukan *port scanning* ke alamat IP yang sama dilihat dari tabel keterangan bahwa *port* 68 dengan protokol UDP telah ter*filtered*.

5. PENUTUP

5.1 Simpulan

Simpulan dari keseluruhan proses penelitian yang telah dilakukan dari pembahasan yang sudah diuraikan adalah berikut :

1. Sistem IDS (*Intrusion Detection System*) yang diimplementasikan telah berhasil dibangun dan dikembangkan dengan baik. Keseluruhan sistem mesin sensor IDS dapat bekerja dengan efektif sebagai sistem keamanan jaringan komputer yang berbasis open source dalam mendeteksi sebuah serangan pada mesin sensor IDS, dimana dalam pendeteksian serangan dianalisis oleh aplikasi BASE. Sistem ini bekerja diluar *server* itu sendiri, karena jika ada didalam *server* kemungkinan besar ikut menjadi target serangan.
2. Sistem IDS dalam mendeteksi serangan yang terjadi adalah dengan melakukan *scanning* terhadap sejumlah *source* dan *traffic* jaringan yang terjadi didalam jaringan, sehingga seluruh kejadian yang dianggap sah atau tidak sah dilihat melalui kegiatan *monitoring* dengan menggunakan aplikasi yang digunakan untuk melakukan *monitoring* jaringan.
3. Mekanisme kerja dari *snort* dan ACID yang telah diimplementasikan dengan menggunakan *ping attack* dan *port scanning* (*Nmap*).
4. Pencegahan yang dapat dilakukan terhadap serangan adalah dengan menggunakan *IPTABLES*. Untuk mengatasi serangan dari *intruder*, yaitu dengan menulis sebuah *rule* pada *IPTABLES* dimana rule dimasukkan kedalam *rule IPTABLES*, maka akan terlihat pada mesin penyerang atau *client*, yaitu *request time out*.
5. Kelebihan dalam menggunakan IDS ini adalah suatu jaringan komputer dapat dipantau atau hanya dengan sebuah mesin atau komputer yang bertindak sebagai sensor didalam jaringan dan terhubung kedalam sebuah jaringan itu dan dapat melihat semua kejadian yang terjadi didalamnya. Selain keuntungan yang didapat dalam implementasi IDS ini, penulis juga mendapatkan hasil dari sistem IDS dalam mengamankan jaringan, yaitu jika terdapat sebuah masalah pada jaringan, maka dapat diketahui secara langsung oleh IDS ini yang menggunakan aplikasi *snort*, dari mana serangan itu datang, melalui *port* berapa dan protokol apa yang digunakan.

5.2 Saran

Saran yang dapat diberikan pada penelitian ini adalah *Intrusion Detection System* (IDS) yang diimplementasikan sudah baik, namun itu hanya dari segi pendeteksian atau monitoring jaringan saja sehingga untuk pencegahannya membutuhkan *tools* tambahan. Alangkah baiknya jika *Intrusion Detection System* (IDS) yang diimplementasikan

dikembangkan lagi dari segi pencegahannya, agar serangan dari *flooding data* yang masuk dapat langsung dicegah atau diblokir secara otomatis tanpa *tools* tambahan.

DAFTAR PUSTAKA

1. https://thesolidsnake.wordpress.com/2011/06/05/pengenalan-snort/ (Diakses pada tanggal 25 November 2017, pukul 13.51 WIB)
2. http://www.laptopsiipat.com/2017/12/macam-macam-jenis-jaringan-berdasarkan-geografis.html (Diakses pada tanggal 26 November 2017, pukul 14.00 WIB)
3. https://www.nesabamedia.com/topologi-jaringan-komputer/ (Diakses pada tanggal 26 November 2017, pukul 14.30 WIB)
4. http://www.transiskom.com/2011/03/pengertian-icmp-internet-control.html (Diakses pada tanggal 26 November 2017, pukul 14.45 WIB)
5. https://lizahotmauli.wordpress.com/2013/01/12/pengertian-serta-perbedaan-tcp-dan-udp/ (Diakses pada tanggal 26 November 2017, pukul 15.00 WIB)
6. Dony Ariyus, *Intrusion Detection System*, C.V. Andi Offset, Yogyakarta 2007
7. https://thesolidsnake.wordpress.com/2011/06/05/pengenalan-snort/ (Diakses pada tanggal 28 November 2017, pukul 16.15 WIB)
8. https://belajarkomputersekarang.wordpress.com/2011/03/15/snort/ (Diakses pada tanggal 28 November 2017, pukul 17.30 WIB)
9. http://bazkomblogs.blogspot.co.id/2017/09/kelebihan-dan-kekurangan-linux-ubuntu.html (Diakses pada tanggal 28 November 2017, pukul 19.45 WIB)
10. Budi Triandi. Skripsi : "Sistem Kemanan Jaringan Dalam Mencegah *Flooding Data* Dengan Metode *Bloking* IP dan *Port*" Universitas Potensi Utama 2015
11. https://arkatkj.wordpress.com/2014/11/13/pengertian-contoh-dan-simbol-simbol-flowchart/ (Diakses pada tanggal 3 Desember 2017, pukul 10.30 WIB)