PERANCANGAN DAN IMPLEMENTASI NIDS (NETWORK INTRUSION DETECTION SYSTEM) MENGGUNAKAN SNORT DAN BASE PADA FREEBSD 10

[1]Bangun Saputra **[2] Setia Juli Irzal Ismail** **[3] Mochamad Fachru Rizal**

[123]Fakultas Ilmu Terapan – Universitas Telkom
Jl. Telekomunikasi, Dayeuh Kolot Bandung 40257 Indonesia
[1]bgnsptr@gmail.com, [2]jul@tass.telkomuniversity.ac.id, [3]mfrizal@tass.telkomuniversity.ac.id

Abstrak

Teknologi informasi berkembang dengan sangat cepat, khususnya internet. Namun perkembangan teknologi informasi dimanfaatkan oleh pihak-pihak tidak bertanggung jawab untuk mendapatkan informasi-informasi tertentu yang menguntungkan pihak tersebut. Dengan adanya serangan seperti *scanning*, *sniffing* dan DDoS *Attack* menyebabkan penyedia informasi membutuhkan sebuah sistem jaringan yang bisa menangani berbagai jenis serangan tersebut. Solusi yang ditawarkan adalah dengan membangun NIDS (*Network Intrusion Detection System*) menggunakan Snort, IPFilter dan Portsentry. Snort mampu mendeteksi serta melakukan *log event* ketika terjadi serangan DDoS Attack, selain menggunakan snort juga menggunakan portsentry dan IPFilter. Portsentry mampu memblok penyerang yang ingin melakukan *port scanning* terhadap server sedangkan IPFilter digunakan sebagai *firewall* untuk memblok ip *address* yang telah terdeteksi oleh snort sebagai penyerang. Sistem NIDS yang dibangun diintegrasikan dengan web *interface* BASE untuk menampilkan serangan yang telah terdeteksi oleh *rules* snort. Sistem ini dibangun diatas sistem operasi FreeBSD yang dikenal sebagai sistem operasi yang memiliki keamanan yang baik. Sistem ini dapat menangani serangan seperti *port scanning*, dan DDoS *Attack*.

Kata kunci: NIDS, *Sniffing*, *Scanning*, FreeBSD, Snort, *Firewall*

Abstract

Information technology develops very quickly, especially the Internet. But the development of information technology used by hacker for getting certain information. To handle attacks such as scanning, sniffing and DDoS Attack a Network Intrusion Detection System (NIDS) is required. The solution offered is to build NIDS with Snort, Portsentry, and IPFilter. Snort is able to detect and log attacks such as DDoS Attack, Sniffing and Scanning. Portsentry is able to block an attacker who wants to do the scanning of the server. IPFilter used as a firewall to block IP addresses that have been detected by Snort as an attacker. The NIDS system is integrated with the web interface BASE to show all of the attacks that have been detected by the snort rules. The system is built on the FreeBSD operating system, known as the operating system that has a good security. This system can handle attacks such as port scanning and DDoS Attack.

Keywords: NIDS, Sniffing, Scanning, FreeBSD, Snort, Firewall

1. Latar Belakang

Dalam era teknologi informasi saat ini, keamanan informasi sangatlah penting terlebih lagi pada suatu jaringan yang terkoneksi dengan internet. Perkembangan teknologi informasi sayangnya tidak diikuti dengan perkembangan keamanan pada sistem itu sendiri dengan demikian cukup banyak sistem jaringan yang lemah dan harus ditingkatkan keamanannya.

Keamanan suatu jaringan seringkali terganggu dengan adanya ancaman dari dalam ataupun dari luar. Saat ini begitu banyak cara untuk melakukan serangan terhadap suatu sistem jaringan. Cara-cara ini terus berkembang dari zaman dahulu sampai sekarang. Dahulu untuk melakukan suatu serangan membutuhkan pengetahuan dan pemahaman teknis IT yang tinggi, akan tetapi saat ini sangat mudah untuk melakukan serangan bukan hanya orang yang mempunyai keahlian yang tinggi. Metode dan alat-alat yang dipakai semakin banyak dan mudah digunakan, bahkan terhadap sistem keamanan jaringan. Contoh serangan yang sering dilakukan seperti DDoS *Attack*, *Port Scanning*, *Sniffing*, *Cross Site Scripting*, SQL *Injection*, *Malware*, *Phishing*, *Exploit*, dll.

Oleh karena itu diperlukan solusi untuk menangani serangan yang semakin berkembang. *Intrusion Detection System* (IDS) merupakan solusi untuk menangani serangan-serangan tersebut. Dengan adanya IDS serangan-serangan tersebut dapat dideteksi oleh admin jaringan. IDS berguna untuk mendeteksi adanya serangan dari penyusup. Pada penelitian ini diimplementasikan IDS dengan menggunakan *software* snort dan BASE sebagai web *interface*. Snort merupakan suatu perangkat lunak untuk mendeteksi penyusup dan menganalisa paket-paket yang melewati jaringan secara *real time traffic* dan *logging* ke dalam *database* serta mampu mendeteksi serangan yang berasal dari luar jaringan. Selain mendeteksi adanya penyusup pada penelitian ini juga diimplementasikan pencegahan dari serangan

port scanning dengan menggunakan *portsentry* sehingga seolah-olah server menghilang saat dilakukan *port scanning* dan pencegahan serangan DDoS *Attack* dengan menggunakan IPFilter sebagai *firewall* untuk memblok IP *Address* penyerang yang melakukan DDoS *Attack*.

2. Tinjauan Pustaka

2.1 Intrusion Detection System (IDS)

Intrusion Detection System (IDS) [1] adalah sebuah aplikasi perangkat lunak atau perangkat keras yang dapat mendeteksi aktivitas yang mencurigakan dalam sebuah sistem atau jaringan. IDS dapat melakukan deteksi terhadap lalu lintas *inbound* dan *outbound* dalam sebuah sistem jaringan. Terdapat 2 jenis IDS, yaitu:

1. *Network-based* IDS
   IDS jenis ini melakukan monitoring terhadap semua lalu lintas yang melewati jaringan untuk melakukan analisis terhadap serangan.
2. *Host-based* IDS
   IDS jenis ini melakukan monitoring terhadap suatu *host* untuk melakukan analisis adanya serangan terhadap *host* tersebut.

2.2 Snort

Snort [2] adalah sebuah aplikasi atau *tools* sekuriti yang berfungsi untuk mendeteksi intrusi-intrusi dalam jaringan (penyusupan, penyerangan, dan beragam bentuk ancaman lainnya). Snort memiliki beberapa komponen, yaitu:

1. Libpcap
   Berfungsi memisahkan paket data yang melalui *Ethernet card* untuk selanjutnya digunakan oleh snort.
2. Packet Decoder
   Berfungsi mengambil data pada layer 2 yang dikirim dari paket *capture library*.
3. Preprocessor
   Berfungsi memanipulasi paket sebelum dikirim ke *detection engine*. Manipulasi paket dapat berupa ditandai, dikelompokan atau dihentikan.
4. Detection Engine
   Paket yang datang dari *packet decoder* akan dibandingkan dengan *rule*s yang telah ditetapkan sebelumnya. *Rules* berisi tanda-tanda yang termasuk serangan.
5. 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*.

2.3 Basic Analysis and Security Engine (BASE)

Basic Analysis and Security Engine (BASE) merupakan *tools* yang digunakan untuk menganalisis dan menampilkan serangan-serangan yang telah terdeksi oleh *rules* snort yang berbasis web *interface*.

2.4 FreeBSD

FreeBSD merupakan sistem operasi *open source* yang tangguh dan memiliki tingkat keamanan yang baik. Dari daftar NETCRAFT, FreeBSD tercatat sebagai satu-satunya sistem operasi gratis yang tercantum sebagai web server dengan *uptime* terpanjang. FreeBSD dapat dijalankan di *processor* kompatibel dari keluarga Intel x86, seperti halnya DEC Alpha, *processor* Sun UltraSPARC, IA64 dan AMD64.

2.5 PortSentry

PortSentry [3] merupakan *tools* yang digunakan untuk merespon berbagai aktivitas *port scanning* yang dilakukan oleh penyerang. Beberapa fitur utama portsentry, yaitu:

1. Berjalan di atas protokol TCP & UDP untuk mendeteksi *port scanning*.
2. Mendeteksi *stealth scan*, seperti SYN/*half-open*, FIN, NULL, X-MAS.
3. Portsentry akan bereaksi secara *real time* dengan cara memblokir IP *address* penyerang. Hal ini dilakukan dengan menggunakan ipchains/ipfwadm dan memasukan ke *file* /usr/local/etc/portsentry.blocked.tcp secara otomatis oleh TCP Wrapper.
4. Portsentry mempunyai mekanisme untuk mengingat *host* mana yang pernah terhubung ke portsentry. Dengan mekanisme tersebut, hanya *host* yang terlalu sering melakukan *port scanning* yang akan di blokir oleh portsentry.
5. Portsentry akan melaporkan semua pelanggaran melalui *history* dan mengindentifikasi nama sistem, waktu serangan *port scannning*, IP *address* penyerang, TCP/UDP *port* tempat serangan dilakukan. *File history* ini dapat dilihat pada *file* /usr/local/etc/portsentry.history.

2.6 IPFilter

IPFilter adalah *firewall* pada FreeBSD yang digunakan untuk melakukan *filter* terhadap lalu lintas paket data.

2.7 Port Scanning

Port scanning [3] merupakan suatu kegiatan atau metode untuk memeriksa *port* apa saja yang aktif pada *host* maupun komputer target. *Port scanning* merupakan gejala awal yang menandakan akan ada suatu serangan atau usaha penyusupan terhadap sistem tersebut.

2.8 Distributed Denial of Service (DDoS) Attack

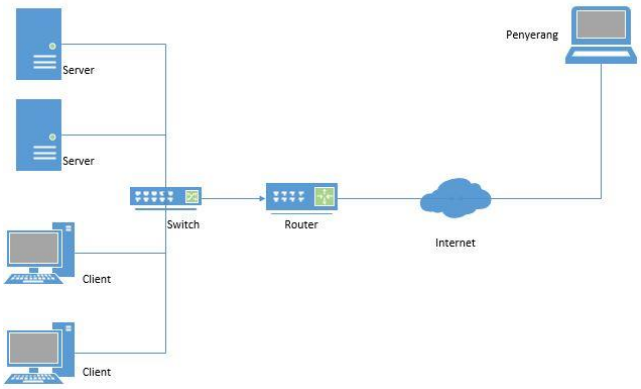
DDoS (*Distributed Denial of Service*) *Attack* [4] merupakan serangan menggunakan banyak komputer untuk menyerang satu titik, komputer-komputer tersebut tersebar tanpa menghiraukan jarak dan waktu dan akan melakukan serangan secara serentak dan bertubi-tubi ke komputer target. Beberapa tipe serangan DDoS *Attack* yang diuji coba dalam penelitian ini, sebagai berikut:

1. SYN Flooding
   Serangan SYN *flooding* dilakukan dengan cara memanfaatkan kelemahan protokol pada saat terjadinya proses *handshake*. Saat dua buah komputer memulai komunikasi, komputer penyerang akan mengirimkan syn, komputer target akan menjawab dengan mengirimkan syn ack kepada komputer penyerang. Seharusnya setelah menerima balasan syn ack dari komputer target, penyerang mengirimkan ack kepada komputer target untuk proses *handshake*. Pada kenyataannya penyerang justru mengirimkan paket syn kepada komputer target yang mengakibatkan komputer target harus terus menjawab permintaan dari penyerang.
2. ICMP Flooding
   Serangan ICMP *flooding* merupakan metode mengirimkan paket data ICMP ke komputer target secara terus menerus.

3. Analisis dan Perancangan

3.1 Gambaran Sistem Saat Ini (atau Produk)

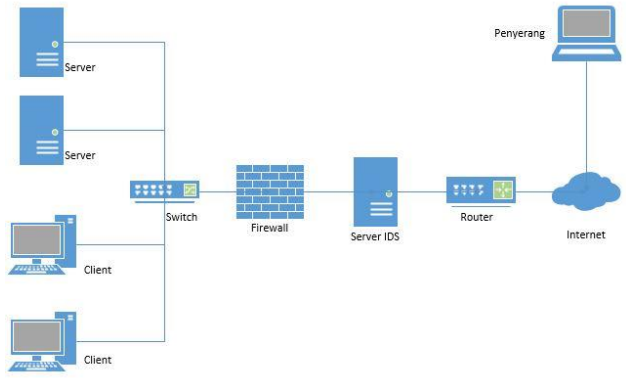
Perkembangan jaringan komputer saat ini sudah sangat maju tetapi masih banyak yang mengabaikan sisi keamanan dari jaringan komputer itu sendiri. Topologi jaringan tanpa adanya fitur keamanan seperti IDS sangatlah rentan terhadap serangan yang berasal dari luar selain itu admin jaringan juga sangat sulit ketika ingin mendeteksi adanya serangan tersebut.



Gambar 1 Topologi jaringan tidak menggunakan IDS

3.2 Sistem Usulan

Sistem usulan pada penelitian ini adalah dengan menerapkan fitur keamanan IDS dengan menggunakan snort dan BASE sebagai web *interface*, sehingga admin jaringan lebih mudah dalam mendeteksi adanya serangan. Untuk pencegahannya dengan menggunakan IPFilter sebagai *firewall* dan *portsentry*.



Gambar 2 Topologi jaringan dengan menggunakan IDS

Dengan banyaknya serangan DDoS *Attack* yang berasal dari luar maka admin jaringan perlu untuk memblok protokol icmp dan tidak memblok protokol icmp untuk beberapa IP *Address* tertentu. Terdapat beberapa alasan mengapa protokol icmp diblok dan tidak diblok.

Alasan protokol icmp diblok, yaitu:

1. Mengurangi beban server agar kinerja server lebih ringan karena tidak dapat dilakukan ping dari jaringan luar.
2. Meminimalisir resiko serangan DDoS *Attack*.

Alasan protokol icmp tidak diblok, yaitu:

1. Protokol icmp seperti ping tidak diblok untuk beberapa IP *address* tertentu yang digunakan oleh admin jaringan,

hal ini untuk memudahkan admin dalam mengawasi kinerja jaringan.

3.3 Perangkat keras yang digunakan

1. Spesifikasi perangkat keras untuk server
   Processor: Intel (R) Core i3(R) CPU 2.40GHz
   Memori: 4 GB DDR3
   Drive: 500 GB *Harddisk*, 1 DVD-RW
   Perangkat tambahan: 1 LAN *card*
2. Spesifikasi perangkat keras untuk penyerang
   Processor: Intel Celeron 887 Dual-core 1.50 GHz
   Memori: 4 GB DDR3
   Drive: 320 GB *Harddisk*
   Perangkat tambahan: 1 LAN *card*
3. Switch
   1 Switch TP-LINK TL-SF1008D 8 *port*.
4. Kabel UTP
   2 kabel UTP tipe *straight*.

3.4 Perangkat lunak yang digunakan

Tabel 1 Tabel perangkat lunak yang dibutuhkan

No	Nama	Versi	Deskripsi
1.	FreeBSD	10	Sistem operasi yang digunakan untuk server IDS.
2.	Kali Linux	1.1.0	Sistem operasi yang digunakan sebagai penyerang.
3.	Snort	2.9.7	Perangkat lunak *open source* yang digunakan untuk mendeteksi dan mencegah serangan.
4.	IPFilter	1.0	Perangkat lunak yang digunakan sebagai *firewall*.
5.	Portsentry	1.2-12	Perangkat lunak yang digunakan untuk pencegahan *port scanning*.
6.	Nmap	6.40	Perangkat lunak yang digunakan untuk *port scanning* pada jaringan target.
7.	Hping3	1.0	Perangkat lunak yang digunakan untuk melakukan penyerangan DDoS *Attack*.

3.5 Langkah Pengerjaan

1. Konfigurasi IP *Address* server IDS.
2. Instalasi dan konfigurasi web server apache.
3. Konfigurasi switch yang akan menjadi penghubung antara server IDS dan penyerang.
4. Konfigurasi Snort, BASE, IPFilter, dan Portsentry dengan menggunakan sistem operasi FreeBSD 10.
5. Pengujian.

4. Pengujian dan Analisis

Metode pengujian yang dilakukan adalah *blackbox testing*, yaitu menguji fungsionalitas implementasi sistem yang sudah dibangun. Tujuannya adalah untuk memastikan semua fungsionalitas pada sistem yang sudah dibangun dapat berjalan dengan baik. [5]

1. Pengujian Distributed Denial of Service (DDoS) Attack

Pengujian serangan DDoS *Attack* adalah mekanisme penyerangan dengan mengirimkan paket data secara terus menerus ke server. Pada penelitian ini sistem diuji dengan menggunakan serangan DDoS *Attack* seperti syn *flooding* dan icmp *flooding*. Berikut merupakan langkah-langkah penyerangan DDoS *Attack* dengan menggunakan metode syn *flooding* dan icmp *flooding*:

1. Pengujian serangan DDoS *Attack* menggunakan hping3 dengan menggunakan sistem operasi kali linux. Berikut merupakan perintah untuk DDoS *Attack* dengan metode syn *flooding* yang menyerang *port* 80.

```
root@kali:~# hping3 -i u100 -S -p 80
192.168.1.1
```

2. Berikut merupakan perintah DDoS *Attack* dengan metode penyerangan icmp *flooding*.

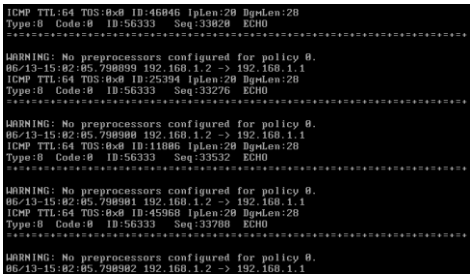
```
root@kali:~# hping3 --icmp -d 10000 -i
u100 192.168.1.1
```

2. Pengujian Snort

1. Setelah pada pengujian sebelumnya melakukan penyerangan DDoS *Attack* maka pada pengujian snort dapat dilihat bahwa penyerang melakukan DDoS *Attack*. Pada gambar 3 dapat

dilihat hasil dari deteksi serangan menggunakan snort.

```
root@bangunsaputra:~ # snort -v
```

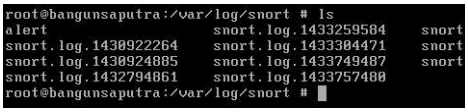


Gambar 3 Hasil deteksi serangan menggunakan snort

2. Snort dapat mencatat paket-paket yang lewat dan menyimpannya ke dalam /var/log/snort.

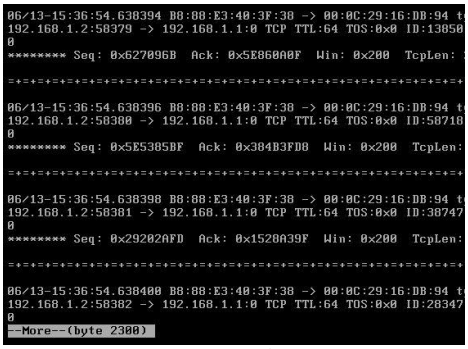
```
root@bangunsaputra:~ # snort –dev –l /var/log/snort
```

3. Pada gambar 4 dapat dilihat hasil *log* pada snort.



Gambar 4 *File log* snort

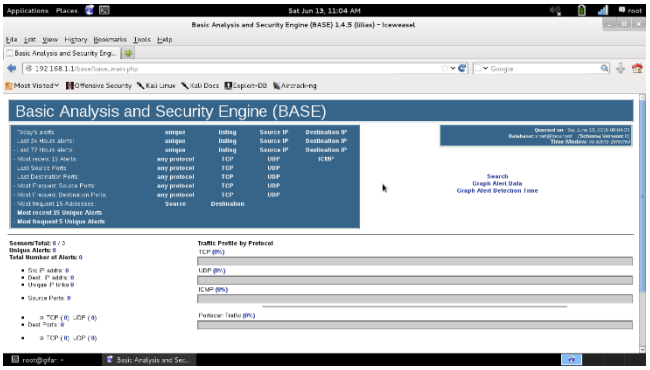
4. Pada gambar 5 dapat dilihat isi dari *file log* pada snort.



Gambar 5 Isi dari *file log* snort

3. Pengujian Basic Analysis and Security Engine (BASE)

Setelah BASE dikonfigurasi dan diintegrasikan dengan snort maka BASE dapat diakses melalui alamat 192.168.1.1/base. Pada gambar 6 dapat dilihat tampilan awal dari BASE.



Gambar 6 Tampilan awal BASE

4. Pengujian Firewall IPFilter

Setelah IP *Address* yang melakukan serangan DDoS *Attack* terdeteksi oleh snort selanjutnya admin melakukan blok terhadap IP *Address* tersebut karena serangan DDoS *Attack* dapat mengganggu kinerja dari server. Pada penelitian ini *firewall* IPFilter diimplementasikan untuk memblok IP *address* yang melakukan serangan DDoS *Attack* dengan metode syn *flooding*, icmp *flooding* dan serangan *port scanning*. Berikut merupakan langkah-langkah membuat *rules* IPFilter untuk memblok serangan DDoS *Attack* dan *port scanning*:

1. Buat *file* konfigurasi *rules* IPFilter.

```
root@bangunsaputra:~ # edit /etc/ipf.rules
```

2. Buat *rules* pada *file* /etc/ipf.rules untuk memblok IP *Address* yang melakukan DDoS *Attack* SYN *Flooding* yang menyerang *port* 80.

```
block in quick on em0 proto tcp from 192.168.1.2 to 192.168.1.1 port 80
```

3. Buat *rules* pada *file* /etc/ipf.rules untuk memblok IP *Address* yang melakukan DDoS *Attack* ICMP *Flooding*.

```
block in quick on em0 proto icmp from 192.168.1.2 to 192.168.1.1
```

4. Buat *rules* pada *file* /etc/ipf.rules untuk memblok *port scanning* yang melakukan *port scanning* ke server.

```
block in quick on em0 proto tcp from 192.168.1.0/24
```

5. Tambahkan ipfilter_enable=”YES” pada *file* /etc/rc.conf untuk mengaktifkan *service* IPFilter pada saat server diaktifkan.

```
root@bangunsaputra:~ # echo 'ipfilter_enable="YES"' >> /etc/rc.conf
```

6. *Restart service* IPFilter.

```
root@bangunsaputra:~ # service ipfilter restart
```

7. Setelah *firewall* IPFilter diaktifkan maka penyerang sudah tidak bisa melakukan DDoS *Attack* syn *flooding* ke server. Pada gambar 7 dapat dilihat hasil DDoS Attack syn *flooding* setelah diblok.

Gambar 7 Tes DDoS *Attack* syn *flooding* setelah *firewall* diaktifkan

8. Setelah *firewall* IPFilter diaktifkan maka penyerang sudah tidak bisa melakukan DDoS *Attack* icmp *flooding* ke server. Pada gambar 8 dapat dilihat hasil DDoS Attack icmp *flooding* setelah diblok.

Gambar 8 Tes DDoS *Attack* icmp *flooding* setelah *firewall* diaktifkan

9. Setelah *firewall* IPFilter diaktifkan maka penyerang sudah tidak bisa melakukan *port scanning* ke server. Pada gambar 9 dapat dilihat hasil *port scanning* setelah diblok oleh IPFilter.

Gambar 9 Hasil port scanning setelah diblok oleh firewall IPFilter

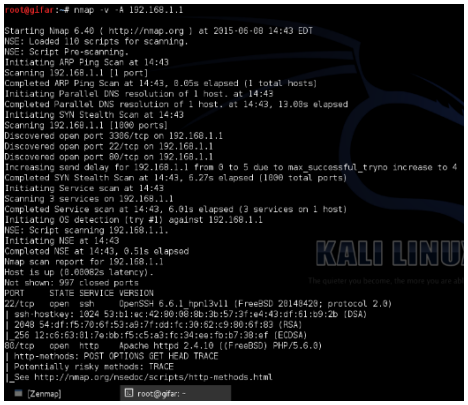
5. Pengujian PortSentry

Pengujian portsentry dilakukan dengan melakukan *port scanning* ke server. Pengujian *port scanning* terdiri dari 2 bagian yaitu *port scanning* dengan menggunakan portsentry pada server dan tidak menggunakan portsentry pada server. Berikut merupakan langkah-langkah pengujian portsentry:

1. Pengujian *port scanning* tanpa menggunakan portsentry.

```
root@bangunsaputra:~ # killall portsentry
```

2. Pengujian *port scanning* dilakukan dengan menggunakan aplikasi nmap. Tanpa adanya portsentry penyerang dapat dengan mudah mengetahui port yang terbuka pada server. Pada gambar 10 dan gambar 11 dapat dilihat hasil *port scanning* yang tidak menggunakan portsentry.



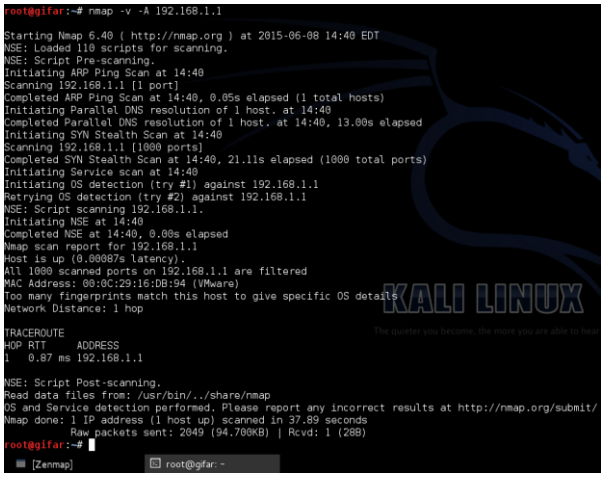
Gambar 10 Hasil *port scanning* tanpa menggunakan portsentry 1

Gambar 11 Hasil *port scanning* tanpa menggunakan portsentry 2

3. Pengujian *port scanning* dengan menggunakan portsentry.

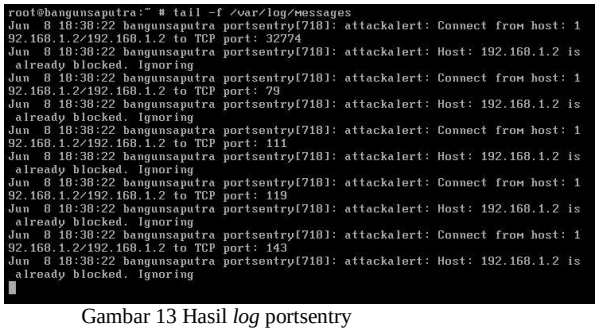
```
root@bangunsaputra:~ # portsentry –tcp && portsentry -udp
```

4. Pengujian *port scanning* dengan menggunakan portsentry. Pengujian *port scanning* dilakukan dengan menggunakan aplikasi nmap. Dengan menggunakan portsentry disisi server, maka penyerang tidak dapat melihat *port* yang terbuka pada server. IP *Address* penyerang otomatis diblok oleh portsentry apabila melakukan *port scanning*. Pada gambar 12 terlihat hasil *port scanning* menggunakan portsentry.



Gambar 12 Hasil *port scanning* dengan menggunakan portsentry

5. Portsentry secara otomatis menyimpan ke dalam *log* apabila ada penyerang yang melakukan *port scanning*. Pada gambar 13 dapat dilihat bahwa IP *Address* penyerang yaitu 192.168.1.2 sudah diblok oleh portsentry.



Gambar 13 Hasil *log* portsentry

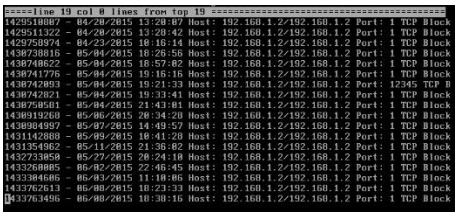
6. *File* portsentry dapat dilihat di direktori /usr/local/etc. Pada gambar 14 dapat dilihat *file* portsentry pada direktori /usr/local/etc.

Gambar 14 *File* portsentry

7. Pada *file* portsentry.blocked.tcp terlihat IP *Address* penyerang 192.168.1.2 telah diblok oleh portsentry. Pada gambar 15 dapat dilihat isi dari *file* portsentry.blocked.tcp.

Gambar 15 Tampilan IP *Address* yang sudah diblok pada *file* portsentry.blocked.tcp

8. Pada *file* portsentry.history terdapat IP *Address* yang sudah melakukan penyerangan terhadap server. Pada gambar 16 dapat dilihat isi dari *file* portsentry.history.



Gambar.16 Tampilan *history* penyerangan pada *file* portsentry.history

6. Analisa hasil pengujian

Berdasarkan hasil pengujian maka dapat ditarik kesimpulan sebagai berikut:

1. Pengujian koneksi antara server dan penyerang berhasil dilakukan begitu juga sebaliknya.
2. Penyerangan DDoS *Attack* dengan metode syn *flooding* dan icmp *flooding* berhasil dilakukan dan snort dapat mendeteksi adanya serangan tersebut.
3. Penyerangan *port scanning* dengan menggunakan aplikasi nmap berhasil dilakukan.
4. Penyerangan DDoS *Attack* dengan metode syn *flooding* dan icmp *flooding* berhasil diblok dengan bantuan *firewall* IPFilter.
5. Penyerangan *port scanning* dapat diblok dengan *firewall* IPFilter dan portsentry.
6. Perbedaan pencegahan serangan *port scanning* dengan menggunakan *firewall* IPFilter dan portsentry adalah apabila menggunakan *firewall* IPFilter pendeteksian serangan *port scanning* tidak dapat mencatat *log* serangan tersebut, sedangkan dengan menggunakan portsentry pendeteksian *port scanning* dapat mencatat *log* serangan tersebut.

5. Kesimpulan dan Saran

5.1 Kesimpulan

Berdasarkan hasil pengujian dapat ditarik kesimpulan sebagai berikut:

1. Snort dan BASE berhasil diimplementasikan pada sistem operasi FreeBSD.
2. Snort mampu mendeteksi adanya serangan dan mencatat serangan kedalam *log* snort tetapi tidak bisa mengkategorikan serangan tersebut.
3. *Firewall* IPFilter mampu memblok IP *Address* yang melakukan penyerangan DDoS *Attack* dan IPFilter mampu memblok penyerang yang ingin melakukan penyerangan *port scanning* ke server tetapi tidak dapat mencatatnya ke dalam *log*.
4. Portsentry mampu memblok IP *Address* yang melakukan penyerangan *port scanning* dan mencatatnya ke dalam *log*.

5.2 Saran

1. Untuk penelitian selanjutnya disarankan menggunakan *firewall* pfsense, karena *firewall* pfsense memiliki banyak fitur dan memiliki tampilan web *interface* yang *user friendly*.
2. Admin harus selalu melakukan *update* dan *maintenance* secara berkala terhadap sistem yang telah dibuat.
3. Perlu dikembangkan sistem *alert* atau notifikasi agar admin tidak harus secara rutin mengecek *log*.

Daftar Pustaka

[1] D. Setiawan, Sistem Keamanan Komputer, Jakarta: Gramedia, 2012.

[2] R. Rafiudin, Mengganyang hacker dengan snort, Yogyakarta: Andi, 2012.

[3] I. Surabaya, Modul Praktikum Keamanan Jaringan, Surabaya: ITS Surabaya, 2013.

[4] EC-Council, Denial of Service, Albuquerque: EC-Council, 2007.

[5] B. Saputra, Perancangan dan Implementasi NIDS (Network Intrusion Detection System) Menggunakan Snort dan Base pada FreeBSD 10, Bandung: Telkom University, 2015.

[6] S. Architecture, "Intrusion Detection Systems Learning with Snort," Security Architecture, 15 January 2015. [Online]. Available: http://www.securityarchitecture.com/learning/intrusion-detection-systems-learning-with-snort/installing-base/. [Diakses 7 April 2015].

[7] FreeBSD, "IPFilter," FreeBSD, 7 August 2014. [Online]. Available: https://www.freebsd.org/doc/handbook/firewalls-ipf.html. [Diakses 3 Mei 2015].