IMPLEMENTASI SECURITY SYSTEM MENGGUNAKAN SNORT IDPS (INTRUSION DETECTION PREVENTION SYSTEM) DENGAN NOTIFIKASI SMS GATEWAY

Disusun dan Diajukan Untuk Memenuhi Persyaratan Ujian Akhir
Memperoleh Gelar Sarjana Komputer Pada Fakultas Sains Dan Teknologi
Universitas Pembangunan Panca Budi
Medan

SKRIPSI

OLEH :

NAMA	**: DANIEL SAPUTRA**
N.P.M	**: 1514370050**
PROGRAM STUDI	**: SISTEM KOMPUTER**

PROGRAM STUDI SISTEM KOMPUTER
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS PEMBANGUNAN PANCA BUDI
MEDAN
20[illegible]

ABSTRAK

DANIEL SAPUTRA

Implementasi Security System Menggunakan Snort IDPS
(Intrusion Detection Prevention System)
Dengan Notifikasi SMS Gateway
2019

Snort adalah sistem pendeteksi *open source* yang digunakan untuk memonitor lalulintas jaringan. Dalam pengembangannya Snort tidak hanya digunakan sebagai *Intrusion Detection System* (IDS), dengan sedikit pengembangan dan penambahan *netfilter queue* dan Iptables hingga Snort dapat bekerja secara *Prevention*. Keterbatasan seorang administrator dalam memonitor server diluar ruangan server menjadikan SMS Gateway dibutuhkan dalam mendapatkan notifikasi bila adanya *attacker*. Dalam penelitian ini penulis menggunakan *Intrusion Detection Prevention System* (IDPS) yaitu Snort awalnya bekerja sebagai *Intrusion Detection System* (IDS) dikembangkan sehingga dapat melakukan *Prevention* terhadap *attacker*, memonitoring server berbasis web dan menggunakan notifikasi SMS *Gateway*.

Kata Kunci: *Intrusion Detection System (IDS), Prevention, Web Monitoring, SMS Gateway.*

DAFTAR ISI

KATA PENGANTAR

Puji syukur Tuhan yang Maha Esa karena dengan berkat dan kasih anugerah-Nya penulis masih diberikan kesehatan sehingga akhirnya penulis dapat menyelesaikan Skirpsi dengan judul : **"IMPLEMENTASI SECURITY SYSTEM MENGGUNAKAN SNORT IDPS (INTRUSION DETECTION PREVENTION SYSTEM) DENGAN NOTIFIKASI SMS GATEWAY".**

Dalam penyusunan Skirpsi ini penulis menyadari banyak mengalami kesulitan namun berkat bantuan dan dorongan dari berbagai pihak, akhirnya Skirpsi ini dapat juga diselesaikan. Penulis dengan segala kerendahan hati menyampaikan terima kasih kepada:

1. Ayahanda dan Ibunda beserta keluarga yang telah berjasa dalam memberikan dukungan moril dan materil.
2. Bapak H.M. Isa Indrawan, SE, MM, selaku Rektor Universitas Pembangunanan Panca Budi Medan.
3. Rektor I, Bapak Ir. Bhakti Alamsyah, M.T, Ph.D
4. Ibu Sri Shindi Indira, ST., M.Sc, selaku Dekan Fakultas Sains Dan Teknologi Universitas Pembangunan Panca Budi Medan
5. Bapak Dr. Muhammad Iqbal, S.Kom., M.Kom, selaku Ketua Program Studi Sistem Komputer Fakultas Sains Dan Teknologi Universitas Pembangunan Panca Budi Medan.
6. Dosen Pembimbing 1, Bapak Dian Kurnia S.Kom.,M.Kom
7. Dosen Pembimbing 2, Bapak Supiyandi S.Kom.,M.Kom
8. Seluruh Dosen dan Staf Pegawai Fakultas Sains Dan Teknologi yang telah banyak membantu dalam kelancaran seluruh aktivitas perkuliahan.
9. Staf Perpustakaan Universitas Pembanguan Panca Budi yang telah berjasa memberikan pinjaman buku-buku yang ada.
10. Teman-teman yang telah memberikan berbagai saran, inspirasi, dorongan, doa, motivasi dan moril maupan materil yang diperlukan sehingga penulis dapat menyelesaikan Skirpsi ini.

Penulis juga menyadari bahwa penyusunan Skripsi ini belum sempurna baik dalam penulisan maupun isi disebabkan keterbatasan kemampuan penulis. Oleh karena itu, penulis mengharapkan kritik dan saran yang membangun dari pembaca untuk penyempurnaan isi Skirpsi ini.

Medan, Agustus 2019

Penulis,

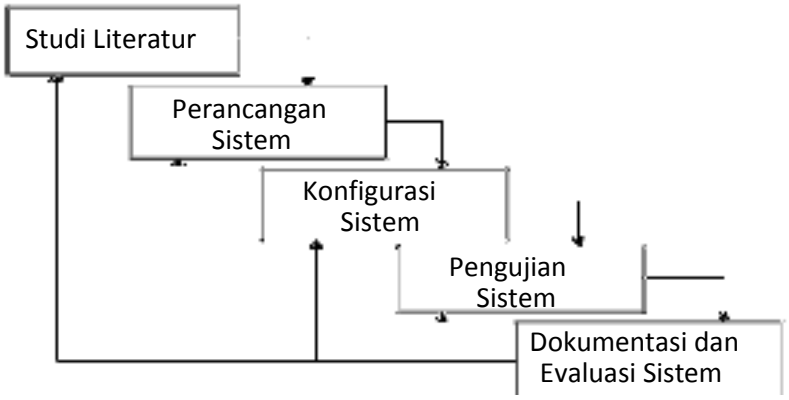
DANIEL SAPUTRA
NPM : 1514370050

BAB III

METODE PENELITIAN

3.1 Tahap Penelitian

Metode yang digunakan dalam membangun sistem ini adalah model *waterfall*. Model ini terdiri dari beberapa tahapan,yaitu: Studi Literatur, perancangan sistem, konfigurasi sistem, penerapan sistem, dokumentasi dan evaluasi sistem. Adapun metode perancangan adalah sebagai berikut :



Gambar 3.1 Sistem yang di bangun menggunakan *waterfall*

Dalam menyelesaikan skripsi ini penulis memperoleh data dengan menggunakan beberapa :

1. Studi Literatur

   Dengan pengumpulan data-data berupa teori baik dengan dosen pembimbing maupun dengan orang yang berkompeten dalam kasus ini dan pustaka yang mendukung.

2. Perancangan Sistem

Meliputi beberapa tahap yang terstruktur sebagai berikut :

a. Sistem dirancang menggunakan sistem operasi dan konfigurasi linux, paket pendukung linux dan Snort, menggunakan sistem operasi windows sebagai pengujian.
b. Hasil dan pembahasan dengan cara Implementasi perangkat dan pengujian sistem.

3. Konfigurasi Sistem

   Dalam skripsi ini sistem yang dikonfigurasi yaitu menggunakan linux dan juga Snort sebagai sistem yang dapat mendeteksi adanya serangan dan sebagai pencegahan, untuk web monitoring menggunakan BASE dan Gammu sebagai mesin SMS Gateway.

4. Pengujian Sistem,

   Melakukan pengujian dan penaksiran ulang sistem yang telah melakukan implementasi.

5. Dokumentasi dan Evaluasi System

   Apakah sistem yang telah di miliki mendapatkan kinerja yang baik dan keamanan dengan tingkat yang baik.

3.2 Metode Pengumpulan Data

Untuk mendukung penelitian yang akan dibangun dibutuhkannya metode pengumpulan data. Beberapa teori yang ada pada *website*, jurnal, makalah, dan

penelitian lainnya. Dalam tahap ini juga melakukan analisa sistem yang berjalan dan kebutuhan sistem yang nantinya akan dikonfigurasi.

3.3 Analisis Sistem Yang Sedang Berjalan

Intrusion Detection Prevention System (IDPS) adalah sebuah metode yang dapat mendeteksi aktivitas yang mecurigakan dan mencegah bila adanya serangan yang berbahaya bagi server. Snort adalah salah satu *tools open source* yang di gunakan yang semula snort bekerja sebagai *Intrusion Detection System* (IDS), dengan tambahan paket filtering Iptables dan didalam Snort terdapat modul tambahan DAQ *NetFilter Queue* (NFQ) sebagai *Prevention* bagi Snort. Barnyard2 nantinya berfungsi sebagai output biner Snort yang diproses dan disimpan kedalam database MySQL. Snort menggunakan *Basic Analysis and Security Engine* (BASE) dalam mengelolah data-data kejadian atau *security event* untuk mudah dibaca dan menampilkannya kedalam basis web *interface*. Gammu nantinya berfungsi sebagai SMS *Gatewat* dalam mengirimkan informasi peringatan bila terjadi *attacker* ke *administrator* melalui SMS.

1. Komponen Kerja *Snort Engine Intrusion Detection System* (IDS)
    a. (libpcap) *Library Packet Capture*

       Lipcap bekerja dalam menangkap dan memisahkan paket data yang melalui *Ethernet card* yang selanjutnya akan digunakan Snort.

    b. *Packet Decoder*

       Packet Decoder bekerja dalam mengambil paket dari layer 2 yang dikirim *libpcap*. Dengan memisahkan *Data Link*, *Protocol* IP, paket

TCP dan UDP Snort memiliki informasi protokol yang akan di proses lebih lanjut.

c. *Preprocessor*

Preprocessor adalah komponen yang bekerja dalam menyusun atau mengubah paket data sebelum menuju ke *detection engine* dan beroperasi untuk mencari tahu bila paket data terjadi serangan.

d. *Detection Engine*

Detection Engine adalah bagian penting Snort. Bekerja dengan mendeteksi bila terjadinya kegiatan penyerangan pada paket. *Detection Engine* memproses rule Snort untuk membaca strutur data *internal* yang di cocokkan dengan paket yang ada. Bila paket cocok dengan rule yang ada, tindakan yang di ambil berupa *logging* paket atau *alert*, bila tidak paket akan di biarkan saja.

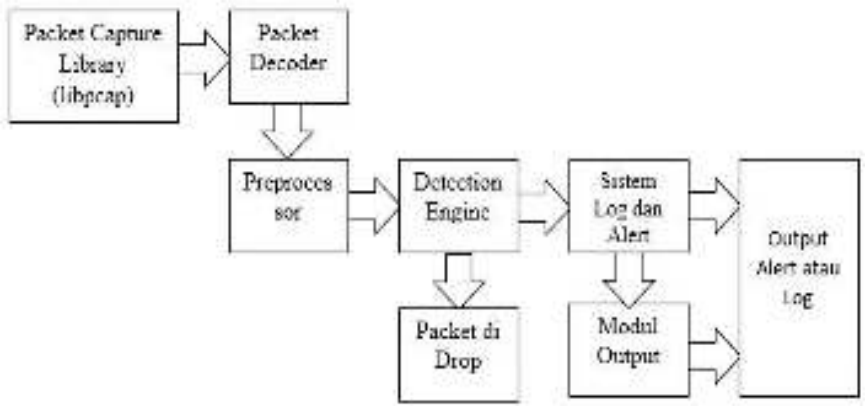
e. Sistem Log dan *Alert*

Telah di dapati oleh *Detection Engine* bila paket cocok dengan rule yang ada, tindakan yang di ambil berupa *logging* paket atau *alert* dan log disimpan pada format teks didalam penyimpanan.

f. Modul Output

Modul Output bekerja bagaimana cara penyimpanan keluaran yang dihasilkan log dan *alert* dari Snort. Modul ini mengatur jenis keluaran yang dihasilkan oleh sistem log dan *alert*.

Hubungan komponen Snort IDS (*Intrusion Detection System*) diatas dapat digambarkan sebagai berikut:



Gambar 3.2 Komponen Kerja *Snort Engine*

2. Cara Kerja Snort *Intrusion Detection Prevention System* (IDPS)

   Cara kerja dari Snort IDPS yaitu ketika paket masuk ditangkap oleh *libpcap* melewati paket *filtering* Iptables kemudian untuk memaksimalkan paket yang masuk paket tersebut tidak dicocokkan satu per satu dengan rule Iptables tetapi paket yang masuk dapat disaring dulu dengan rule Iptables, kemudian paket yang lolos dialihkan ke NetFilter Queue untuk diolah Snort Engine bila paket terdeteksi adanya suatu serangan maka alert Snort akan di gunakan untuk merekam alamat IP penyerang dan melakukan *Blocking* terhadap alamat IP yang telah terdeteksi.

   Cara kerja Snort IDPS (*Intrusion Detection Prevention System*) dapat digambarkan sebagai berikut:

   a. (libpcap) *Library Packet Capture*

Lipcap bekerja dalam menangkap dan memisahkan paket data yang melalui *Ethernet card* yang selanjutnya akan digunakan Snort.

b. *Filtering* Iptables

Iptables bekerja dalam menyaring paket yang masuk dengan packet tersebut nantinya tidak dicocokkan satu per satu dengan *rules* Iptables akan tetapi langsung diteruskan ke *Netfilter Queue*.

c. *Netfilter Queue*

Bekerja dengan mengolah paket yang masuk untuk diolah oleh Snort nantinya.

d. *Packet Decoder*

Packet Decoder bekerja Dengan memisahkan *Data Link*, *Protocol* IP, paket TCP dan UDP Snort memiliki informasi protokol yang akan di proses lebih lanjut.

e. *Preprocessor*

Preprocessor adalah komponen yang bekerja dalam menyusun atau mengubah paket data sebelum menuju ke *detection engine* dan beroperasi untuk mencari tahu bila paket data terjadi serangan.

f. *Detection Engine*

Detection Engine adalah bagian penting Snort. Bekerja dengan mendeteksi bila terjadinya kegiatan penyerangan pada paket. *Detection Engine* memproses rule Snort untuk membaca strutur data *internal* yang di cocokkan dengan paket yang ada. Bila paket cocok dengan rule yang

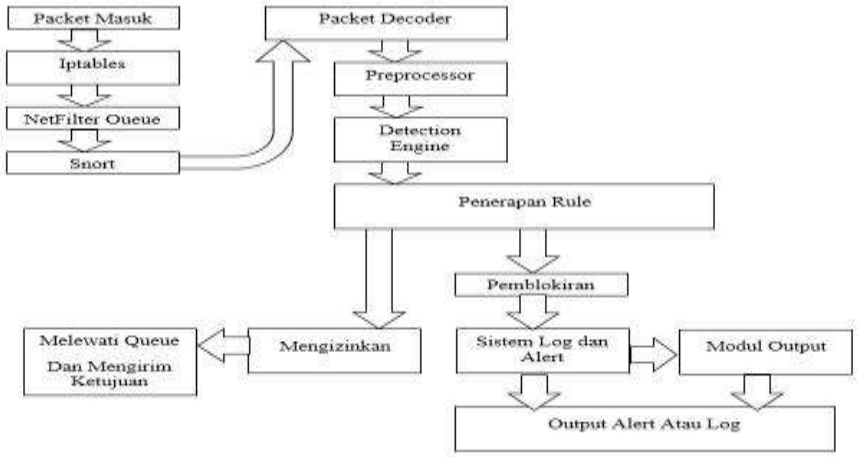
ada, tindakan yang di ambil berupa *logging* paket atau *alert*, bila tidak paket akan di biarkan saja.

g. Penerapan Rules

Telah di dapati oleh *Detection Engine* bila paket cocok dengan rule yang ada, tindakan yang di ambil berupa memisahkan paket apakah paket tersebut berupa ancaman atau tidak, bila itu ancaman *logging* paket atau *alert* dan log disimpan pada format teks didalam penyimpanan dan memblock ancaman tersebut dan bila tidak adanya tanda-tanda dari ancaman logging paket atau alert tidak terpicu berarti akses paket diizinkan.

h. Modul Output

Modul Output bekerja bagaimana cara penyimpanan keluaran yang dihasilkan log dan *alert* dari Snort. Modul ini mengatur jenis keluaran yang dihasilkan oleh sistem log dan *alert*

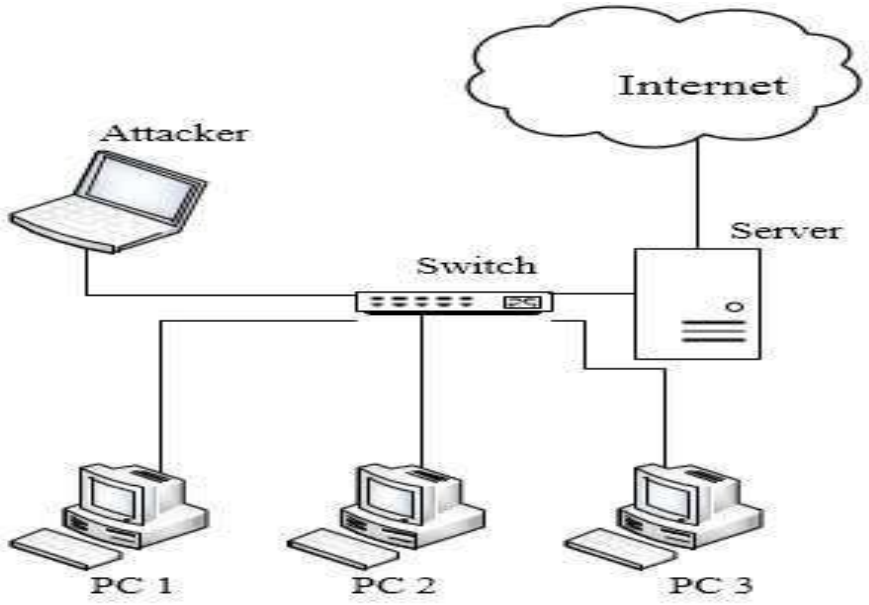


Gambar 3.3 Komponen Kerja Snort *IDPS*

3.4 Rancangan Penelitian

Dalam tugas akhir ini akan mengkonfigurasi sebuah *security system* yang memiliki kemampuan dalam *memonitoring* jaringan, mendeteksi (*detection*) dan mencegah (*prevention*) aktifitas mencurigakan didalam jaringan, Barnyard2 nantinya berfungsi sebagai output biner Snort yang diproses dan disimpan kedalam database MySQL, BASE (*Basic Analysis and Security Engine*) berfungsi dalam mengelolah data-data kejadian atau *security event* untuk mudah dibaca dan menampilkannya kedalam basis web *interface*, serta menggunakan SMS *Gateway* dalam memberikan notifikasi SMS ke *administrator*.

Sistem yang akan di bangun dapat digambarkan dengan topologi berikut:



Gambar 3.4 Topologi Sistem *IDPS* yang akan di bangun

Dalam gambar 6. Diatas dapat dijelaskan dengan pengalamatan IP pada tabel berikut ini:

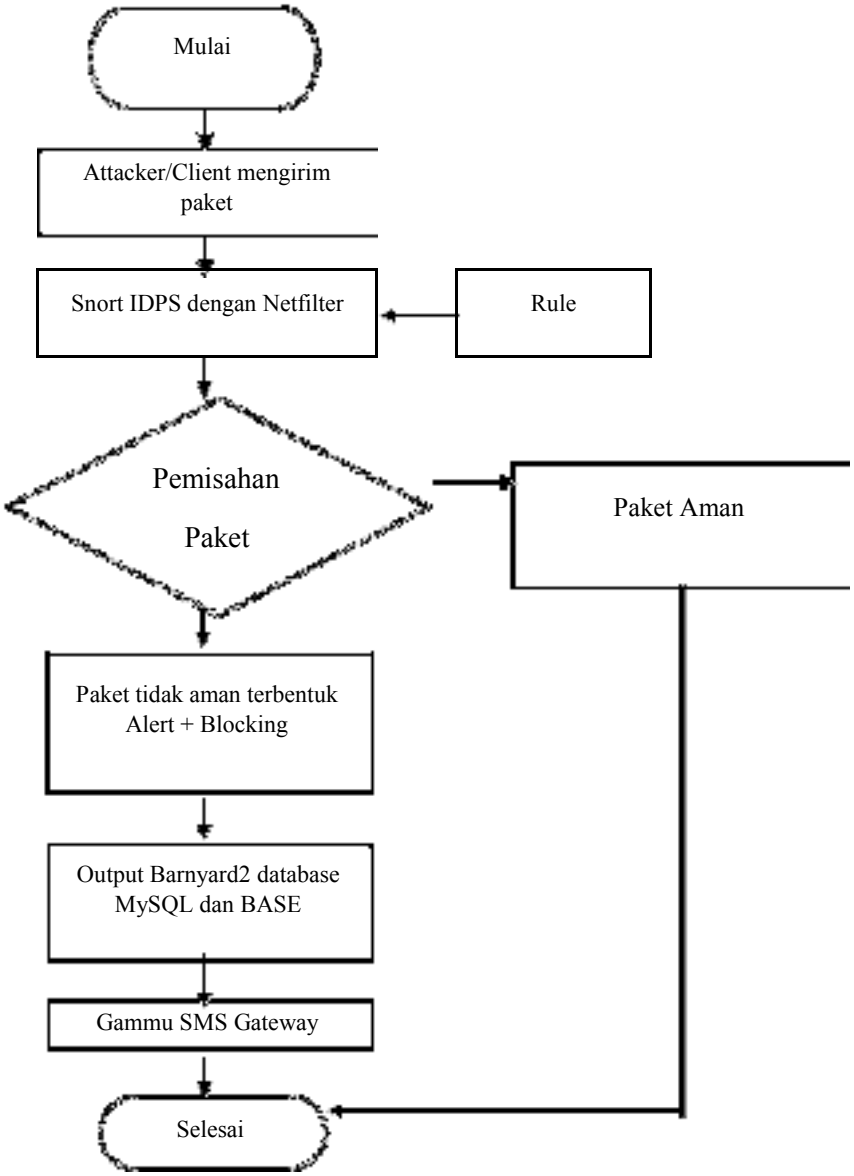
Tabel 3.1 Pengalamatan Ip Address

NO	Hardware/Software Network	Port Ethernet	Alamat IP / IP Address
1	Internet melalui Hotspot	-	Address 192.168.43.1
2	Server	Wlan0	*Dynamic Host Configuration Protocol* (DHCP)
		Ether 0	Address 192.168.1.5
3	Attacker	Ether 0	*Dynamic Host Configuration Protocol* (DHCP)
		Ether 1	-
4	Client Terhubung Jaringan Lokal	Port switch	Address 192.168.1.25-192.168.1.30 Gateway 192.168.1.5

Dalam tabel 1. Pengalamatan alamat Ip dapat dejalaskan bahwa sumber internet berasal dari hotspot atau menggunakan wifi yang terhubung dengan Server. Kemudian didalam Server terinstall sebuah operasi sistem linux ubuntu dengan

konfigurasi Snort IDPS, untuk pengalamatan Ip pada Server dapat melakukan penyetingan jaringannya dengan menggunakan beberapa perintah.

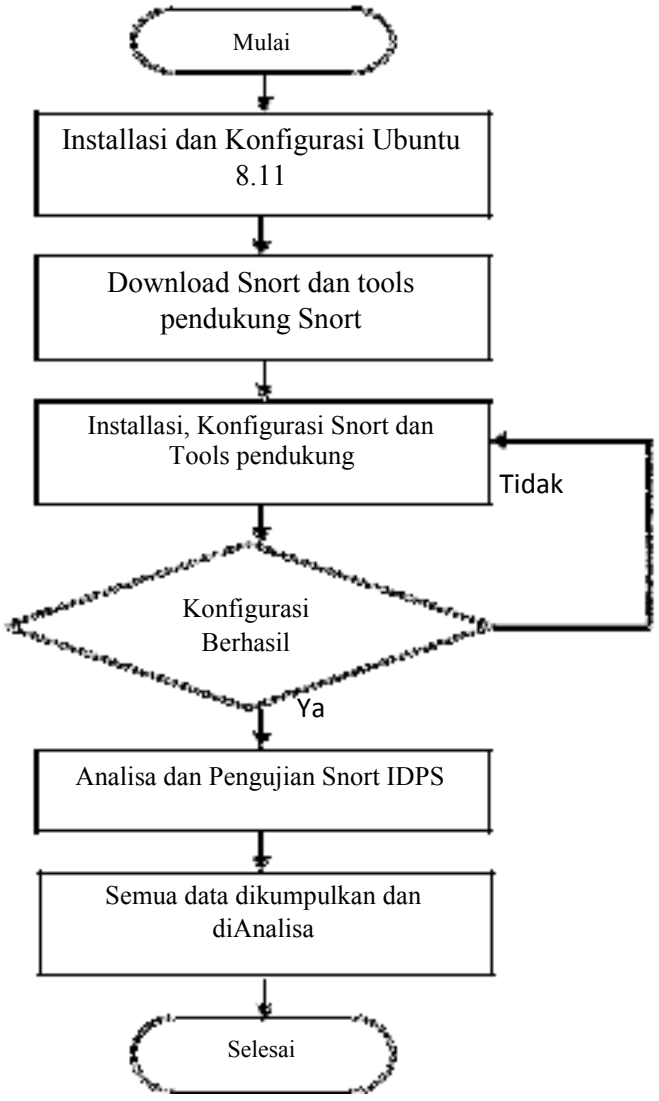
Flowchart Sistem IDPS dapat dilihat pada gamabar berikut:



Gambar 3.5 Flowchart Sistem *IDPS* yang akan dibangun

3.4.1 Perancangan Konfigurasi IDPS

Dalam membangun IDPS (Intrusion Detection Prevention System) agar berjalan sesuai dengan apa yang di inginkan dengan baik, dibutuhkannya proses yang akan dibuat dalam bentuk diagram alir berikut :



Gambar 3.6 Flowchart Perancangan Konfigurasi

IDPS Untuk penjelasan pada gambar diatas sebagai berikut :

1. Diawali dengan melakukan installasi Linux Ubuntu 8.11 kemudian mengikuti alur installasi hingga selesai penginstallan. Saat telah selesai penginstallan lakukan penyesuaian IP *Address* dan konfigurasi pengroutingan.
2. Setelah selasai dalam penyetingan IP *Address* kemudian penginstallan paket-paket yang dibutuhkan dalam mendukung kinerja Snort agar penginstallan Snort nanti tidak terjadi kesalahan dengan menginstall paket-paket yang di butuhkan berupa build-essential, libpcap-dev libpcre3-dev libdumbnet-dev, bison, flex, zlibb1g-dev, liblzma-dev, openssl, libssl-dev, autoconf, libtool. Pkg-config, mysql-server, libmysqlclient-dev, mysql-client, libcrypt-ssleay-perl, liblwp-useragent-determined-perl, apache2, libnetfilter-queue-dev, php5, dan tools php5 lainnya.
3. Setelah semua paket di install kemudian tahap download dan install DAQ, SNORT, Barnyard2, Nghttp2, Adodb, Base, Iptables dan mengkonfigurasinya.
4. Bila semua tahap telah berhasil, lakukan tahap akhir yaitu pengujian sistem yang telah dibangun dan pengumpulan data dan menganalisa.

3.4.2 Installasi Operasi Sistem dan Paket-Paket Tambahan

Pada penelitian ini jenis Operasi Sistem yang digunakan pada Server IDPS menggunakan Linux Ubuntu 8.11 dan untuk komputer client dan laptop penyerang menggunakan windows 7. Melakukan penginstallan Operasi Sistem dilakukan dengan

menggunakan CD/DVD atau Flashdisk, dalam tahap penginstallan selalu di temui pemilihan Bahasa, nama perangkat, berapa ruang partisi dan membagi ukuran untuk partisi Primer dan Sekunder.

3.4.3 Installasi Paket IDPS

Untuk berjalannya penginstallan IDPS menggunkan Snort dibutuhkan paket pendukung agar penginstallan IDPS tidak mendapatkan kesalahan nantinya. dalam melakukan penginsatallan memerlukan perintah dalam linux yaitu :

```
# apt-get install
```

Kemudian tambahkan paket-paket yang di butuhkan dalam membangung Snort IDPS agar Snort berjalan dengan baik dan penginstallan Snort tidak terkendala oleh error nantinya berupa seperti berikut :

```
# apt-get install -y build-essential libpcap-dev libpcre3-dev libdumbnet-dev bison flex zlibb1g-dev liblzma-dev openssl libssl-dev autoconf libtool Pkg-config, mysql-server libmysqlclient-dev mysql-client libcrypt-ssleay-perl liblwp-useragent-determined-perl apache2 php5 apache2 libapache2-mod-php5 php5-mysql php5-common php5-gd php5-cli php-pear libnetfilter-queue-dev gammu –alldeps Image_Graph autoconf libtool pkg-config
```

3.4.4 Installasi DAQ, Snort, Barnyard2, BASE dan SMS Gateway

Dengan diinstalnya paket-paket yang mendukung dalam berjalannya pembangunan IDPS kemudian menginstall tools tambahan untuk Snort agar Snort dapat bekerja dengan maksimal dan handal dengan cara beriukut :

1. Penginstallan DAQ

   DAQ berfungsi sebagai pengaktif semua fitur inline yang ada pada Snort dan juga dengan semula Snort hanya mendeteksi akan tetapi dengan DAQ ini Snort dapat mencegah penyerangan dengan cara menjalankan Snort pada mode inline menggunakan DAQ.

   Sebelum melakukan installasi pastikan dahulu mengunduh DAQ dengan menggunakan wget seperti berikut :

```
# wget https://Snort.org/downloads/Snort/daq-2.0.6.tar.gz
# tar -xvzf daq-2.0.6.tar.gz ==> mengekstrak file yang telah di unduh #
cd daq-2.0.6 ==> masuk kedalam folder yang di ekstrak
# ./configure && make && make install  ==> melakukan pre-installasi
```

2. Installasi Snort

   Setelah DAQ di install kemudian lakukan hal yang sama pada installasi Snort seperti berikut :

```
wget https://Snort.org/downloads/Snort/Snort-2.9.9.0.tar.gz
tar -xvzf Snort-2.9.9.0.tar.gz
cd Snort-2.9.9.0
./configure --enable-sourcefire
make
```

```
# make install
```

Lakukan perintah berikut ini untuk memperbarui libraries pada Snort jika tidak akan terjadi kesalahan ketika menjalankan Snort.

```
ldconfig
ln -s /usr/local/bin/Snort /usr/sbin/Snort     ==> menempatkan symlink
Snort -V                                       ==> menguji Snort
```

Kemudian konfigurasi agar Snort berjalan pada mode IDPS dengan cara berikut ini :

```
=> Membuat penyimpanan untuk Snort
# mkdir /etc/Snort
# mkdir /etc/Snort/rules
# mkdir /etc/Snort/rules/iplists
# mkdir /etc/Snort/preproc_rules
# mkdir /usr/local/lib/Snort_dynamicrules
# mkdir /etc/Snort/so_rules

=> Membuat beberapa file
# touch /etc/Snort/rules/iplists/black_list.rules
# touch /etc/Snort/rules/iplists/white_list.rules
# touch /etc/Snort/rules/local.rules
# touch /etc/Snort/sid-msg.map

=> Membuat ruang penyimpanan log
# mkdir /var/log/Snort
# mkdir /var/log/Snort/archived_logs
```

```
=> Memberikan hak akses
# chmod -R 5775 /etc/Snort
# chmod -R 5775 /var/log/Snort
# chmod -R 5775 /var/log/Snort/archived_logs
# chmod -R 5775 /etc/Snort/so_rules
# chmod -R 5775 /usr/local/lib/Snort_dynamicrules
```

Copy file konfigurasi dan dynamic preprocessor dengan perintah :

```
cd /Snort-2.9.9.0/etc/
# cp *.conf* /etc/Snort
# cp *.map /etc/Snort
# cp *.dtd /etc/Snort
# cd /Snort-2.9.9.0/src/dynamic-preprocessors/build/usr/local/
lib/Snort_dynamicpreprocessor/
# cp * /usr/local/lib/Snort_dynamicpreprocessor/
```

Modifikasi file “Snort.conf” dengan melakukan perintah #nano /etc/Snort/Snort.conf dan modifikasi isi seperti berikut

```
var RULE_PATH /etc/Snort/rules
var SO_RULE_PATH /etc/Snort/so_rules
var PREPROC_RULE_PATH /etc/Snort/preproc_rules
var WHITE_LIST_PATH /etc/Snort/rules/iplists var
BLACK_LIST_PATH /etc/Snort/rules/iplists

=> Hilangkan tanda pagar # pada perintah
berikut include $RULE_PATH/local.rules
include $RULE_PATH/Snort.rules
```

```
=> Edit pada baris 168 seperti berikut :
Config daq: nfq
Config daq_mode: inline
Config daq_var: queue=4

=> Tambahkan perintah berikut di baris 521
output unified2: filename Snort.u2, limit 128

=> Setelah konfigurasi di rasa benar, uji Snort dengan
perintah # Snort -T -i eth0 -c /etc/Snort/Snort.conf
```

Kemudian menambahkan beberapa rule dengan pertama kali mengedit file local.rules dan tambahkan contoh rules berikut :

```
drop icmp any any -> $HOME_NET any (msg:"ICMP detected"; GID:1;
sid:10000001; rev:001; classtype:icmp-event;)
```

3. Installasi Barnyard2

Download terlebih dahulu dan Install Barnyard2 seperti berikut :

```
# wget https://github.com/firnsy/barnyard2/archive/master.tar.gz -
O barnyard2-Master.tar.gz
# tar zxvf barnyard2-Master.tar.gz
# cd barnyard2-master
# autoreconf -fvi -I ./m4
=> Memberikan Barnyard akses dnet.h dan menghubungkan ke database
# ln -s /usr/include/dumbnet.h /usr/include/dnet.h
# ldconfig
# ./configure --with-mysql --with-mysql-libraries=/usr/lib/i386-linux-gnu
```

Setelah tahap installasi kemudian konfigurasi barnyard berikut :

```
# cp /barnyard2-master/etc/barnyard2.conf /etc/Snort/
# mkdir /var/log/barnyard2
# chown Snort.Snort /var/log/barnyard2
# touch /var/log/Snort/barnyard2.waldo
# chown Snort.Snort /var/log/Snort/barnyard2.waldo
```

Membuat database untuk Barnyard2

```
# mysql -u root -p
mysql> create database Snort;
mysql> use Snort;
mysql> source ~/Snort_src/barnyard2-master/schemas/create_mysql
mysql> CREATE USER 'Snort'@'localhost' IDENTIFIED BY
'MySqlSNORTpassword';
mysql> grant create, insert, select, delete, update on Snort.* to
'Snort'@'localhost';
mysql> exit
```

4. Installasi BASE

Download dan kemudian ekstrak base lalu pindahkan folder hasil ekstrak tersebut seperti berikut :

```
# wget http://sourceforge.net/projects/secureideas/files/BASE/
base-1.4.5/base-1.4.5.tar.gz
# tar xzvf base-1.4.5.tar.gz
# mv base-1.4.5 /var/www/html/base/
```

Kemudian konfigurasi beberapa baris yang ada di dalam file base_conf.php dengan cara nano /var/www/html/base/base_conf.php berikut :

```
Pada bari 50 : $BASE_urlpath = '/base'; #
Pada bari 80 : $DBlib_path = '/var/adodb/';
```

```
Pada bari 102 : $drop_dbname = 'Snort';
Pada bari 103 : $drop_host = 'localhost';
Pada bari 104 : $drop_port = '';
Pada bari 105 : $drop_user = 'Snort';
Pada bari 106 : $drop_password = 'password';
```

5. Menambahkan Snort Rules

   Snort membutuhkan Rules untuk membuat aturan atau notifikasi jika ada serangan. Untuk menambah atau mengedit lakukan perintah nano /etc/Snort/rules/local.rules kemudian tambahkan beberapa perintah seperti berikut :

```
alert udp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS (msg:"DOS LOIC UDP)"; threshold: type threshold,track by_src, count 100 , seconds 5;classtype:misc-activity; sid:1234590; rev:1;)

alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS (msg:"DOS LOIC TCP)"; threshold: type threshold,track by_src, count 100 , seconds 5;classtype:misc-activity; sid:1234590; rev:1;)
```

   Dari rules yang telah dibuat secara manual seperti printah diatas, tentu rules Snort juga dapat ditambahkan dengan lebih banyak lagi rules, dengan memanfaatkan Pulledpork dari *website* snort yang telah disediakan oleh pengembang snort itu sendiri dan ekstrak file tersebut sehingga isi dari file tersebut nantinya adanya bagian file rules itu sendiri. Installasi Pulledpork seperti berikut:

```
Wget https://github.com/shirkdog/pulledpork/archive/master.tar.gz -O
pulledpork-master.tar.gz
tar xzvf pulledpork-master.tar.gz
cd pulledpork-master/
sudo cp pulledpork.pl /usr/local/bin
sudo chmod +x /usr/local/bin/pulledpork.pl
sudo cp etc/*.conf /etc/snort
```

6. Menambahkan Iptables

```
Iptables -t nat -A PREROUTING -j NFQUEUE –queue-num=2

Iptables –t nat –A POSTROUTING –j MASQUERADE
```

7. Konfigurasi Gammu

```
=> konfigurasi Gammu agar dapat mendeteksi Modem
#nano /etc/gammu.config
Port = /dev/ttyUSB0
Connection =at
Logfile = /var/log/smsdlog

=> konfigurasi gammu-smdrc menghubungkan gammu dengan
database Service =sql
Driver = native_mysql
User = Snort
Password = 1234
Logfile=/var/log/smsdlog
Pc = localhost
Database = Snort
Pin = 1234
Checksecurity = 0

=> menjalankan Gammu
# gammu –identify
```

```
# gammu-smsd
```

8. Perintah dalam menjalankan Snort

```
=> Perintah menjalankan Snort mode console
/usr/local/bin/Snort -A console -q -u Snort -g Snort -c
/etc/Snort/Snort.conf –Q
=> perintah menjalankan Snort dengan barnyard2 /usr/local/bin/Snort -q
-u Snort -g Snort -c /etc/Snort/Snort.conf -Q => perintah menjalan
barnyard2 untuk membaca alert atau log file barnyard2 -c
/etc/Snort/barnyard2.conf -d /var/log/Snort -f Snort.u2 -w
/var/log/Snort/barnyard2.waldo
```

9. Script PHP SMS Gateway

Untuk menjalakan Service Gammu agar dapat mengirimkan SMS kepada administrator dibutuhkan sedikit Script PHP yaitu:

```
=> Membuat Script Index.php
        <?php
                class sms {
                        public function sendsms($data)
                        {
                                $DestinationNumber =
        $data['DestinationNumber'];
                                $TextDecoded = $data['TextDecoded'];
                                $timestamp = $data['timestamp'];
                                $query = mysql_query("insert into outbox
        (DestinationNumber,TextDecoded)
                                values ('$DestinationNumber','$TextDecoded')");
                                if($query)
                                {
                                $this->CIDEvent($timestamp);
                                }
                        }
                        public function CIDEvent($timestamp)
                        {
                                $query = "UPDATE acid_event set
        status='processed' where
                                timestamp='$timestamp'";
                                $result = mysql_query($query);
                        }
```

```
                }
        ?>
        <?php ?>
                <div id="wrap">
                <legend style="text-align:center;">ALERT</legend>
                <table class="table table-striped table-hover table-bordered">
                        <tr>
                                <th>ip_src</th>
                                <th>ip_dst</th>
                                <th>sig_name</th>
                                <th>timestamp</th>
                        </tr>

<?php

                                $query ="SELECT  timestamp,  inet_ntoa(ip_src)
        as ip_src,inet_ntoa(ip_dst)
                                as ip_dst,sig_name,timestamp
                                from acid_event order by timestamp";
                                $result = mysql_query($query);
                                while($row = mysql_fetch_array($result)):

?>

                        <tr>
                                <td><?php echo $row['ip_src'];?></td>
                                <td><?php echo $row['ip_dst'];?></td>
                                <td><?php echo $row['sig_name'];?></td>
                                <td><?php echo $row['timestamp'];?></td>

<?php

        $sms = new sms();
        $TextDecoded = $row['ip_src'].' melakukan : '.$row['sig_name'].' kepada
        :'.$row['ip_dst'].' pada jam '.$row['timestamp'];
                        $data = array (


                'DestinationNumber'=>'+6282215565330',
                                        'TextDecoded'=>$TextDecoded,
                                        'timestamp'=>$timestamp,
                                        );
                                $sms-> sendsms($data);
                                endwhile;
                        ?>
                </table>
```

```
<?php

$query ="SELECT timestamp, inet_ntoa(ip_src) as ip_src,inet_ntoa(ip_dst)
as ip_dst, sig_name,timestamp from acid_event order by timestamp";

                if(mysql_num_rows($query)==null)
                {

?>

                        <div class="alert alert-block">
                                <button type="button" class="close" data-
        dissimiss="alert">&times;</button>
                                <h4>Ops..!</h4>
                                nggak ada alert yang harus dikirim ke sms...
                        </div>
        <?php
                {
                        $row = mysql_fetch_array($result);
                        $timestamp = $row['timestamp'];
                        $ip_src = $row['ip_src'];
                        $ip_dst = $row['ip_dst'];
                        $sig_name = $row['sig_name'];
                }
        ?>
        </div>
        <?php include("PEAR.php");}?>

=> Membuat Script Koneksi.php

<?php

        class koneksi {
                public function __construct()
                {
                        $host = "localhost";
                        $username = "Snort";
                        $password = "123";
                        $database = "Snort";
                                $connect =
        mysql_connect($host,$username,$password);
                                if($connect)
```

```
                                {
                                $select_db = mysql_select_db($database);
                                        if($select_db)
                                {
                                        //echo "bisa konek";
                                }
                                else
                                {
                                        echo "cannot select database";
                                }
                        }
                        else
                        {
                                echo "cannot connect server";
                        }

                        $query = "SELECT inet_ntoa(ip_src) as aipi from
        acid_event order by timestamp";
                        $result = mysql_query($query);
                        while ($row = mysql_fetch_array($result)){
                                echo $row ['aipi'];
                        }
                }
        }
        $koneksi = new koneksi();
?>
```

BAB II

LANDASAN TEORI

2.1 Pengertian Security System

Security system (keamanan sistem) adalah menjamin dan menjaga sumber daya yang ada dari ancaman *attacker*. Dalam keamanan *system* yang perlu dilakukan adalah memberikan batasan akses orang lain yang dapat menggagnggu sistem, baik itu menggunakan komputer yang bersifat *stand alone*, jaringan intranet maupun jaringan global (Hardi, 2017),. Dengan mencapai tujuan sebuah sistem dapat beroperasi dengan baik tanpa ada kendala.

2.2 Pengertian Client Server

Client Server merupakan model konektivitas jaringan yang berfungsi menjadikan komputer menjadi client dan server. Pada konektivitas ini sebuah komputer ditempatkan sebagai server dan sebuah komputer lainnya menjadi client dimana server berfungsi untuk mengelola data dan melayani jalur atau terminal yang terhubung pada sebuah sistem jaringan yang dapat dikatakan sebagai client. Dan sebaliknya sebuah client tidak dapat menjadi server, akan tetapi server dapat menjadi client yang dimana prinsip kerja dari server akan menunggu permintaan client, memproses dan membagikan hasilnya kepada client sedangkan client mengirimkan permintaan kepada server untuk diproses dan mendapatkan hasil prosesnya.

2.3 Pengertian Jaringan Intranet

intranet adalah jaringan pribadi atau private network yang menggunakan TCP dan IP protokol Internet dalam melakukan pembagian informasi atau data penting suatu instansi atau dalam perusahaan kepada pegawai yang diizinkan. Dapat disebut juga dengan web internal suatu instansi. dalam membangun jaringan intranet sebuah instansi haruslah memiliki beberapa komponen untuk mendukung jaringan intranet seperti protokol Internet dan protokol lainnya, klien dan juga server dan beberapa protokol Internet lainnya seperti FTP, POP3, atau SMTP dan umumnya merupakan komponen protokol yang sering digunakan.

2.4 Pengertian Snort

Snort adalah perangkat lunak yang gratis dan *open source* dalam melakukan *intrusion detection* dan *prevention system* yang dibuat oleh Martin Roesch di 1998. Snort memiliki kemampuan untuk melakukan lalu lintas *real-time* analisis dan pencatatan paket pada *Internet Protocol* (IP) jaringan. Ini melakukan analisis protokol, pencarian konten, dan pencocokan konten. Program ini juga dapat digunakan untuk mendeteksi probe atau serangan, tetapi tidak terbatas pada, upaya *operating system fingerprinting attempts*, *common gateway interface*, *buffer overflows*, *server message block probes*, dan *stealth port scans* (Mehra, 2012).

Snort adalah paket *sniffer* berbasis libpcap dan logger yang dapat digunakan sebagai *Network Intrusion Detection System* (NIDS). Aturan ini berdasarkan *logging* untuk melakukan pencocokan pola konten dan mendeteksi berbagai serangan, seperti *buffer overflow*s, *stealth port scans*, *CGI attack*, *SMB probes*, dan banyak lagi. Snort

memiliki peringatan secara *real-time* yang mampu memberitahukan peringatan yang dikirim ke syslog, *Server Message Block* (SMB) pesan WinPopup, atau file alert yang terpisah (Roesch, 1990).

Dalam penggunaan Snort masih di basiskan *command-lin*e dapat memberikan cukup kesulitan bagi pengguna yang terbiasa akan lingkungan *Graphical User Interface* (GUI) sehingga ada beberapa pihak pendukung yang mengembangkan seperti Acid, Mircrosoft Windows, dan IDScenter dengan basis php yang dapat di akses menggunakan web browser berbasis GUI.

Terdapat 4 mode dalam menjalankan Snort sebagai analisa keamanan jaringan yaitu :

1. Mode *Logger*

   Jenis mode ini bekerja dalam mencatat semua data yang dilalui pada jaringan untuk dapat di analisa di lain hari.

2. Mode *Sniffer*

   Jenis mode yang bekerja dengan menangkap dan melihat data yang melewati pada jaringan

3. Mode *Intrusion Detection*

   Jenis ini membuat Snort bekerja sebagai pendeteksi kegiatan terhadap berbagai serangan pada jaringan yang mengikuti berdasarkan aturan (*Rules*).

4. Mode *Inline*

   Jenis yang bekerja aktif sebagai pengaman pada jaringan dengan memblokir upaya penyerangan dan memberikan respon terhadap serangan

menggunakan *Rules* yang di kombinasikan dengan *Firewall* untuk di izinkan atau tidak data pada jaringan berdasarkan *Rules* yang telah di tentukan.

2.5 Rules Snort

Rules Snort merupakan database yang berisi pola-pola serangan yang berupa *signature* jenis serangan, *Rules* Snort ini harus secara rutin diupdate. sehingga ketika terjadi pola serangan baru, Snort dapat mendeteksi pola tersebut sebagai sebuah serangan. Penulisan *rules* Snort mempunyai aturan yaitu *rules* harus ditulis dalam satu baris (*single line*) (Affandi dan Setyowibowo, 2013).

Untuk membaca dan membuat *rules* pahami bagian yang ada pada *rules* terdiri dari *Rules Header* dan *Rules Options*. *Rules Header* berisikan tindakan, protokol, alamat IP, nomor port, destinasi host. *Rules Options* berisikan pesan yang akan di tampilkan nantinya, dapat di jelaskan dari 1 perintah *Rules* seperti berikut :

```
Alert tcp any any -> $HOME_NET (msg:”ICMP DETECT”;)
```

Dari *Rules* di atas dapat dilihat untuk *Rules Header* di tandai dari Alert tcp any any -> $HOME_NET dan untuk *Rules Options* isi pesan yang ada pada di antara (msg:”ICMP DETECT”;).

2.6 Data AcQuisition (DAQ)

Data AcQuisition (DAQ) adalah modul tambahan untuk paket *Input* / *Output* pada Snort yang digunakan untuk mengaktifkan fitur *Pervention* yang ada pada Snort.

Berikut beberapa modul DAQ untuk mendukung Snort agar dapat berjalan dengan fitur *Prevention* :

1. Packet CAPture (PCAP) dapat menjadikan Snort bekerja secara default dengan *Sniffer* dan *Intrusion Detection System* (IDS).
2. AFPACKET menjadikan Snort bekerja pada mode *inline* dengan menggunakan 2 *interface* yang saling menjembatani tanpa ada tambahan Firewall.
3. NetFilter Queue (NFQ) menjadikan Snort bekerja secara *inline* dengan menggunakan *Queue* dan *netfilter* atau Firewall.
4. IPFW membuat Snort bekerja dengan mode *inline* untuk OpenBSD dan FreeBSD menggunakan socket pf dan ipfw.
5. DUMP memungkinkan Snort melakukan pengujian mekanisme *inline* dan normalisasi.

2.7 Intrusion Detection System (IDS)

Intrusion Detection System (IDS) adalah perangkat lunak yang digunakan untuk memonitor jaringan untuk apa pun aktivitas yang tidak menyenangkan menjembatani fungsi normal sistem sehingga menyebabkan beberapa pelanggaran kebijakan. Ini meninjau beberapa sistem deteksi intrusi dan perangkat lunak yang menyoroti klasifikasi utama mereka, evaluasi kinerja dan pengukuran (Jacob *et al*, 2017).

Intrusion detection system dapat di kalsifikasi menjadi 3 bagaian yaitu :

1. *Host Intrusion Detection System* (HIDS)

Jenis ini ditempatkan pada satu perangkat seperti server atau workstation, dimana data dianalisis secara lokal ke mesin dan mengumpulkan data ini dari berbagai sumber. HIDS dapat menggunakan sistem deteksi anomali dan penyalahgunaan.

2. *Network Intrusion Detection System* (NIDS)

   NIDS dikerahkan pada titik strategis dalam jaringan infrastruktur. NIDS dapat menangkap dan menganalisis data mendeteksi serangan yang diketahui dengan membandingkan pola atau tanda dari database atau deteksi aktivitas ilegal dengan memindai lalu lintas untuk aktivitas anomali. NIDS juga disebut sebagai "*Packet-sniffer*", Karena ini menangkap paket yang lewat melalui media komunikasi.

3. *Hybrid Intrusion Detection System*

   Manajemen dan memperingatkan dari kedua perangkat deteksi intrusi jaringan dan berbasis host, dan menyediakan pelengkap logis untuk NID dan HID - manajemen deteksi intrusi pusat (Ashoor dan Gore, 2011).

Dalam kebanyakan IDS adalah sistem yg bersifat pasif yang mana tugas dari IDS ini hanyalah mendeteksi intrusi yang bila terjadinya penyerangan dan memberikan peringatan kepada admin jaringan bahwa terjadinya penyerangan.

2.8 Intrusion Prevention System (IPS)

Intrusion Prevention System (IPS) atau *inline* adalah sebuah aplikasi yang bekerja untuk *monitoring traffic* jaringan, mendeteksi aktivitas yang mencurigakan, dan melakukan pencegahan dini terhadap intrusi atau kejadian yang dapat membuat

jaringan menjadi berjalan tidak seperti sebagaimana mestinya. Bisa jadi karena adanya serangan dari luar, dan sebagainya (Ariyadi *et al,* 2012).

Secara umumnya terdapat 2 jenis yang ada pada IPS yaitu :

1. *Network-based intrusion prevention system* (NIPS)

   NIPS adalah jenis sistem yang dapat menahan seluruh lalu lintas jaringan dan memerikasa penyerangan dan kode yang dicurigai. Dalam NIPS ini menggunakan in-line mode yang membutuhkan performansi tinggi yaitu sebuah elemen krusial dari perangkat IPS dalam mencegah tidak seimbangannya pemrosessan pada jaringan. Biasanya NIPS didesain menggunakan beberapa komponen dalam mengakselerasikan performa bandwidth yaitu NIPS melakukan pemantauan dan pertahanan dalam satu jaringan secara global. NIPS menggabungkan fitur IPS dengan firewall dan kadang disebut sebagai In-Line IDS.

2. *Host-based intrusion prevention system* (HIPS)

   HIPS adalah sistem pencegah yang terdiri dari banyaknya lapisan dan menggunakan packet filtering, inspeksi status juga metode pencegahan yang digunakan bersifat real-time dalam menjaga host berada dibawah keadaan efisiensi performansi yang layak. Mekanisme kerjanya adalah dengan mencegah kode-kode berbahaya yang memasuk melalui host agar tidak dieksekusi tanpa perlu mengecek *threat signature*.

2.9 Jenis Serangan

Denial of Service (DoS) adalah serangan yang menyerang server dalam jaringan internet atau intranet dengan cara mengirimkan permintaan informasi dengan membanjiri sumber daya yang dimiliki oleh server tersebut sampai server tidak dapat menjalankan kinerjanya dengan benar dan secara tidak langsung menghambat kinerja server dalam pengoperasiannya secara normal.

Serangan DoS memiliki beberapa jenis penyerangan dengan beberapa cara yaitu :

1. Syn Flood

   Pada saat keadaan normal penyerang akan mengirimkan paket TCP SYN dalam melakukan hubungan dengan server , dengan melalui cara ini penyerang akan membanjiri server berupa banyaknya paket TCP SYN

2. ICMP flood

   Penyerangan yang bertujuan untuk membuat server menjadi crash, yang di sebabkan banyaknya pengiriman paket ke arach target. Penyerangan ini dilakukan dengan mengirimkan suatu perintah ping dengan jumlah yang besar. Hal ini yang membuat server menjadi *crash* dan menurunkan kinerja dari server.

3. *Remote controled attack*

   Penyerangan dengan mengendalikan beberapa jaringan lain untuk menyerang target. Penyerangan dengan tipe ini biasanya akan berpengaruh besar, karena biasanya server- server untuk menyerang mempunyai bandwith yang besar.

4. *Buffer Overflow*

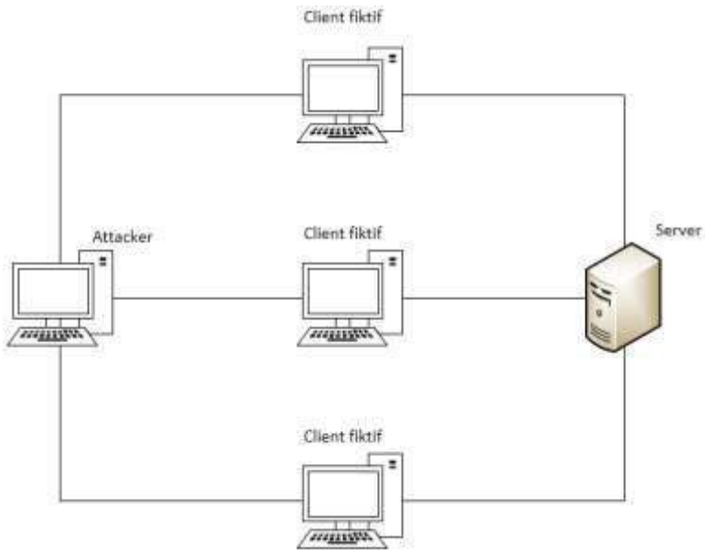
   Penyerang melakukan serangan dengan mengirimkan data yang melebihi kapasitas sistem. TCP Flood dan UDP Flood

5. Teardrop

   Penyerang mengirimkan paket terfragmentasi ke server dengan memanfaatkan fitur yang ada pada TCP/IP yaitu paket fragmentation. Hal ini menyebabkan pecahan-pecahan yang terkirim tidak dapat dikumpulkan kembali oleh mesin target.

Serangan DoS umumnya memberikan peket yang berlebih kepada server sehingga server mengalami Flood atau kebanjiran paket yang mengakibatkan server tidak dapat bekerja dengan baik.

Untuk melihat bagaimana DoS bekerja dapat dilihat seperti berikut:



Gambar 2.1 Proses Penyerangan DoS

2.10 Paket Filtering Iptables

Iptables mengizinkan user untuk mengontrol sepenuhnya jaringan melalui paket IP dengan system LINUX yang diimplementasikan pada kernel Linux. Sebuah kebijakan atau Policy dapat dibuat dengan iptables sebagai polisi lalulintas jaringan (Sondakh *et al,* 2014),.

Paket filtering Iptables adalah *tools* perangkat lunak yang biasanya di pergunakan oleh linux sebagai *filter traffic* atau lalulintas data di dalam server. sistem yang dirancang khusus untuk mencegah akses serangan yang mencurigakan masuk ke dalam jaringan.

Paket filter Iptables memiliki aturan dasar, yaitu:

1. Input

   Semua paket data yang masuk dari firewall dari intranet atau internet.

2. Output

   Semua paket data yang keluar dari firewall melalui intranet atau internet

3. Forward

   Paket data yang melalui firewall dari intranet atau internet.

4. Prerouting

   Digunakan sebagai mentranslasikan paket sebelum proses routing terjadi, yaitu merubah alamat tujuan dari paket data yang biasanya disebut dengan *Destination* NAT atau DNAT.

5. Postrouting

Digunakan sebagai mentraslasikan alamat setelah proses routing terjadi, yaitu merubah alamat dari paket data biasanya disebut dengan *Source* NAT atau SNAT.

2.11 MySQL

MySQL merupakan DBMS (*Database Management System*) yang di pergunakan secara gratis berlisensi dari General Public License(GPL), dimana setiap orang bebas untuk menggunakannya akan tetapi tidak untuk dijadikan program untuk komersial.

MySQL merupakan generasi lanjut dari SQL (Structured Query Language). SQL adalah sebuah konsep pengoperasian basis data terutama dalam proses seleksi, pemasukan, pengubahan, dan penghapusan data yang mungkin dapat di kerjakan dengan mudah dan otomatis.

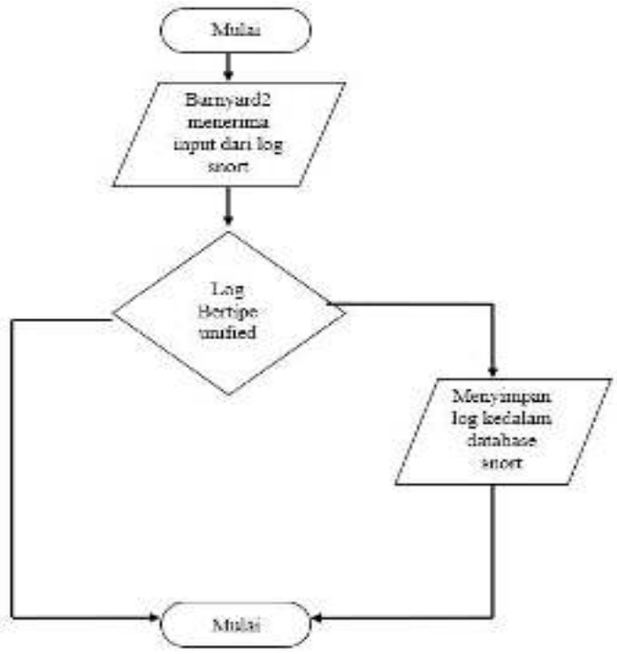
Beberapa fitur yang terdapat pada MySQL yaitu:

1. RDBMS (*Relational Database Management System*).
2. Arsitektur dengan Client-Server.
3. Perintah SQL standar mudah di kenal. Hampir segala software database menggunakan SQL.
4. Mendukung foreign key.
5. Bebas dalam penggunaan atau Gratis.
6. Fleksibel dengan berbagai bahasa pemrograman.
7. Stabil dan keamanan yang baik.

2.12 Barnyard2

Barnyard adalah sebagai penghasil *output system* dari Snort. Barnyard2 bekerja memproses kemudian menyimpan proses output biner dari Snort kedalam database MySql. Barnyard2 akan membaca file logging Snort dan memasukkannya ke dalam database.

Untuk proses penulisan log penyerangan ke dalam database menggunakan barnyard2 dapat dilihat dengan flowchart berikut:



Gambar 2.2 Proses Penulisan Log Masuk ke Database

2.13 Gammu

Gammu adalah perangkat lunak yang digunakan dalam mengelola berbagai fungsi pada handphone, modem dan perangkat lainnya. Fungsi yang dapat di kelola oleh Gammu adalah nomor kontak dan SMS (*Short Message Service*).

Gammu SMS *deamon* merupakan perangkat lunak yang terdapat pada Gammu bekerja agar dapat mendeteksi modem GSM dengan mengecek apakah terdapat pesan atau tidak suatu pesan yang dikirim ke penyimpanan. Gammu dapat menyimpan pesan yang diterima dalam bentuk file atau dalam berbagai jenis basis data lainnya seperti MySQL dan PostgreSQL. Dalam paket Gammu termasuk juga gammu-smsd-inject alat yang bekerja dalam membantu membuat pesan teks panjang dan gammu-smsd-monitor untuk memonitor status *Daemon* SMS atau telepon.

2.14 Linux

Linux adalah sistem operasi open source yang paling terkenal dan paling banyak digunakan. Sebagai sistem operasi, merupakan bagian dari unix yang bekerja pada berbagai macam perangkat keras komputer mulai dari inter x86 sampai dengan RISC. Dengan berlisensikan GNU (*Gnu Not Unix*) dengan memperoleh program dan dilengkapi dengan kode sumbernya (source code). Tidak hanya itu, dapat pula hak untuk menyalin sebanyak yang dimau, atau dapat mengubah kode sumbernya hingga hal tersebut legal dilakukan dibawah lisensi. Lisensi GNU memperbolehkan pihak yang ingin mendapatkan biaya untuk penyalinan maupun pengiriman program. Yang paling penitng pada Linus yaitu kebebasan, terutama untuk programmer dan administrator jaringan, yaitu kebebasan untuk memperoleh kode sumber (source code) dan kebebasan untuk mengubahnya.

2.15 Sistem Operasi Ubuntu

Ubuntu adalah berkumpulnya dari individu-individu yang umum dalam menciptakan sebuah sistem operasi yang bebas. Sistem operasi open source itu dapat disebut juga dengan Ubuntu. Sebuah sistem operasi dengan beberapa program dasar dan utilitas yang membuat komputer dapat berjalan. Inti dari sebuah sistem operasi tersebut dapat di sebut juga dengan kernel. Pengertian dari kernel adalah program yang paling dasar yang ada pada komputer dan memungkinkan kita untuk memulai program lain.

Kernel yang di gunakan Ubuntu saat ini menggunakan kernel Linux atau kernel FreeBSD. Linux yang dimulai dari Linus Torvalds dan didukung ribuan programmer di seluruh dunia. FreeBSD adalah sebuah sistem operasi termasuk kernel dan perangkat lunak lainnya.

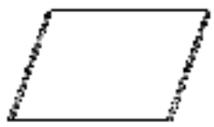
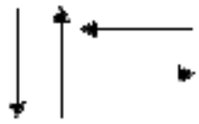
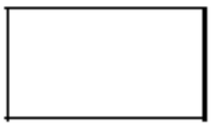
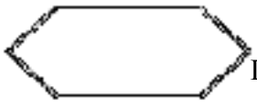
Program perangkat lunak tersebut dapat membantu brainware dalam mendapatkan apa yang ingin mereka lakukan dan diperbuat, dari memodifikasi dokumen untuk menjalankan bisnis, bermain game dan menulis perangkat lunak yang lebih banyak lagi. Ubuntu datang dengan lebih dari 45.000 paket (software precompiled yang terbungkus dalam format yang bagus untuk memudahkan dalam installasi dan konfigurasi), manajer paket (APT) dan beberapa utilitas yang mungkin dapat untuk mengelola ribuan paket yang ada pada ribuan komputer dengan mudah untuk menginstal sebuah aplikasi. Ubuntu sangat mengatur segala sesuatu pekerjaan di sistem operasinya sehingga semua bekerja dengan sangat baik.

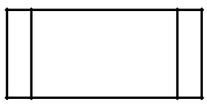
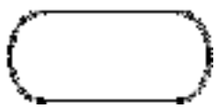
2.16 Flowchart

Flowchart adalah bagian-bagian yang memiliki arus dan menggambarkan langkah-langkah penyelesaian suatu masalah. Flowchart merupakan suatu cara penyajian dari suatu Algoritma

Flowchart disusun dengan simbol. Simbol ini dipakai sebagai alat bantu menggambarkan proses didalam program. Simbol-simbol yang digunakan dapat dibagi, yakni sebagai berikut:

Tabel 2.1 Simbol Flowchart

Simbol	Keterangan
	Input/Output Digunakan untuk mewakili data input/output
	Arus/Flow Digunkana untuk menunjukkan arah/alir dari suatu proses.
	Proses Digunakan untuk mewakili suatu proses.
	Keputusan/*Decision* Digunakan untuk suatu penyelesaian kondisi dalam program.
	Persiapan/ *pendefined* Proses Digunakan untuk memberikan nilai awal dari proses.
	Penghubung/*Connector* Digunakan untuk menunjukkan sambungan dari aliran yang terputus dihalaman yang sama.

	*Predefined* proses Digunakan untuk proses yang detilnya terpisah.
	Awal/akhir (Terminal) Digunakan untuk menunjukkan awal dan akhir dari proses.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Kebutuhan Spesifikasi Hardware

Perangkat keras yang digunakan dalam mendukung pembuatan Snort IDPS dapat di lihat dengan tabel sebagai berikut:

Tabel 4.1 Komponen Perangkat Keras

NO	Perangkat Keras	Keterangan	Jumlah
1	Server	Acer Aspire E1-431-10002G32mn Spesifikasi: processor Type Intel Celeron Processor (1.80 GHz, 2MB Cache) RAM 2 GB DDR3 HDD 320GB	1 unit
2	Attacker	Intel Celeron B877 Speed 1.4GHz Cache 2MB RAM 4Gb DDR3 HDD 320 GB	1 unit
3	Modem Telkomsel Flash	USB Modem untuk Gammu	1 unit

4.2 Kebutuhan Spesifikasi Software

Dalam membangun sistem Snort IDPS dengan notifikasi SMS *Gateway* dibutuhkan tool-tool perangkat lunak agar sistem bekerja dengan baik, adapun perangkat lunak yang dibutuhkan dapat dilihat dengan tabel berikut:

Tabel 4.2 Komponen Perangkat Lunak

NO	Perangkat Lunak	Keterangan
1	Sistem Operasi Linux Ubuntu 8.11	Bekerja sebagai Server yang akan menjadi target penyerangan
2	Snort 2.9.9.0	Bekerja sebagai *tools* dari IDPS
3	Barnyard2, BASE, dan Firewall	Bekerja sebagai tools tambahan untuk SNORT dalam mengoptimalisasikan IDPS
4	Gammu	Perangkat lunak yang di gunakan sebagai penghubung komputer dengan perangkat handphone melalui notifikasi
5	Operasi Sistem Windows 7	Dipergunakan pada komputer *Attacker* dan *Client*
6	LOIC (*Low Orbit Ion Cannon*)	Perangkat lunak *Attacker*

4.3 Implementasi Sistem

Dalam hal ini sistem yang telah dianalisa dan dikonfigurasi dilanjutkan dengan sistem dioperasikan dan melakukan pengujian untuk melihat hingga sampai mana sistem yang dibuat dapat berjalan dengan baik hingga tujuan.

Dalam proses Implementasi *Security System* menggunakan Snort *Instrusion Detection Prevention System* (IDPS) terdapat bagian utama yang akan berperan yaitu:

1. Implementasi Snort

   Bekerja dalam *memonitoring* jalur paket data

2. Implementasi Rule Snort

   Rule mendeteksi dan mengolah paket data yang melewati Snort apakah sebuah *Attacker* atau paket data tanpa ancaman

3. Implementasi Output

   Rule Snort mengirim data ke Firewall Iptables agar paket *Attacker* di *Block*, kemudian barnyard2 mengimputkan data alert kedalam database dan BASE dapat menampilkan Web *interface*, Gammu berfungsi mengirimkan alert dalam bentuk SMS ke *administrator*. Ganglia berfungsi dalam memonitoring kinerja server dalam bentuk web

Implementasi *Security System* menggunakan Snort *Instrusion Detection Prevention System* (IDPS) nantinya akan mendapatkan hasil dan batasan, hasil yang akan didapat yaitu pencegahan sebuah serangan yang terjadi pada server dan menampilkan *Output* Web *interface* pada database, Web *monitoring* kinerja server serta notifikasi SMS ke *administrator*, untuk batasannya pengujian hanya menggunakan serangan Denial of Service (DoS) TCP Flood menggunakan perangkat lunak LOIC.

4.4 Pengujian Serangan *Denial of Service* (DoS)

Pengujian ini akan melakukan penyerangan dari serangan DoS (*Denial of Service*) yaitu TCP (*Transmission Control* Protocol) Flood yang bekerja dengan mengirimkan paket TCP dengan jumlah yang sangat besar ke server sehingga dapat mampu membuat sebuah server *crash* atau server tidak bekerja dengan baik. Hasil ini akan menunjukkan serangan TCP Flood dengan LOIC dalam menyerang server yang terdapat Snort IDPS dengan server tanpa Snort IDPS.

4.4.1 Penggunaan perangkat lunak *Low Orbit Ion Cannon* (LOIC)

Low Orbit Ion Cannon (LOIC) adalah *tools* yang digunakan dalam melakukan serangan DoS, LOIC dapat dipergunakan secara bebas tanpa biaya. Tools ini bekerja dengan menyerang URL atau alamat IP server dengan 3 metode yaitu TCP Flood,UDP Flood, dan HTTP Flood.

LOIC memiliki ukuran tidak begitu besar dan mudah dalam pengoperasiannya sehingga banyaknya oknum atau orang tertentu secara iseng menggunakan LOIC hanya untuk hiburan saja. Padahal dampak yang dihasilkan dari LOIC yaitu dapat mengacaukan kinerja server dan merugikan pihak tertentu.

Dalam penelitian ini LOIC dipergunakan dalam menguji keamanan sebuah sistem server dari ancaman serangan DoS. Untuk tampilan LOIC dapat dilihat dari gambar berikut:

Gambar 4.1 Tampilan *Low Orbit Ion Cannon* (LOIC)

4.4.2 Kondisi serangan tanpa menggunakan Snort IDPS

Dalam pengujian ini akan menggunakan serangan DoS yang ada pada LOIC yaitu serangan TCP Flood, sebuah serangan yang mengarah pada protokol *Transmission Control Protocol* (TCP) memanfaatkan koneksi TCP yang terhubung dengan server dengan membanjirinya (*Flooding*), percobaan dalam membuat koneksi TCP yang ditampung oleh server dalam *buffer* memang berbeda-beda tetapi daya tampungnya tidak lebih dari beberapa ratus koneksi, dengan mengirimkan banyaknya paket kedalam TCP, lokasi buffer akan mengalami kepenuhan dan mengakibatkan server menjadi tidak bekerja dengan baik.

Dalam menjalankan serangan menggunakan alamat IP server untuk dijadikan sebuah target penyerangan terhilat pada kolom IP, dapat dilihat pada tampilan berikut:

Gambar 4.2 Pengujian Serangan LOIC Tanpa Snort *IDPS*

Dalam pengujian diatas dapat dilihat bahwa pengujian menggunakan teknik serangan TCP Flood yaitu membanjiri *Transmission Control Protocol* dengan mengirim banyaknya paket sehingga server menjadi penuh akan paket tersebut, gambar 4.2 dapat dilihat bagian Requested menandakan bahwa serangan TCP Flood bekerja dengan meminta permintaan paket informasi Server.

Dari pengujian penyerangan TCP Flood dengan perangkat lunak LOIC, kinerja server dapat diukur menggunakan *system monitor* bawaan linux ubuntu dalam bentuk *graphic*, dengan *system monitor* server dapat dimonitor kinerja jaringan, CPU, dan memori yang berjalan pada server, dapat dilihat dengan tampilan berikut:

Gambar 4.3 Monitoring Kinerja Server Keadaan Normal

Pada gambar 4.3 kinerja CPU, Memory, dan Network pada *system monitorin*g server ubuntu hanya menunjukkan kinerja CPU yang bekerja menjalankan sistem server dan tak menunjukkan adanya tanda serangan apapun dalam menjalankan kinerja server, sehingga dapat dikatakan kondisi server berjalan dalam keadaan normal, sebab server bekerja dengan normal dan belum adanya serangan LOIC pada server.

Ketika terjadinya sebuah serangan DoS pada server, *system monitoring* server ubuntu akan menunjukkan perubahan seperti gambar berikut:

Gambar 4.4 Monitoring Kinerja Server Terjadi Serangan

Pada gambar 4.4 dapat dilihat pengujian serangan pada server dapat mempengaruhi kinerja server dengan meningkatnya kinerja pada *Network history* nantinya juga berpengaruh pada kinerja CPU server sebab *Network History* terus menerus dibebani oleh serangan TCP Flood. *Network History* mampu menyentuh 598.1KB/s di karenakan LOIC melakukan serangan fiktif kepada server sebanyak 99 serangan fiktif atau sama dengan 99 client yang mengirimkan paket TCP secara bersamaan terus menerus.

Setelah mendapatkan hasil dilakukan pengujian penyerangan oleh 99 host fiktif yang menggunakan LOIC dalam menyerang server. Dilakukan kembali pengujian simulasi serangan sebanyak 99 host fiktif menggunakan 1 komputer penyerang dengan

LOIC dalam menyerang server untuk menentukan berapa nilai yang dapat dikirim tiap host fiktif ke server.

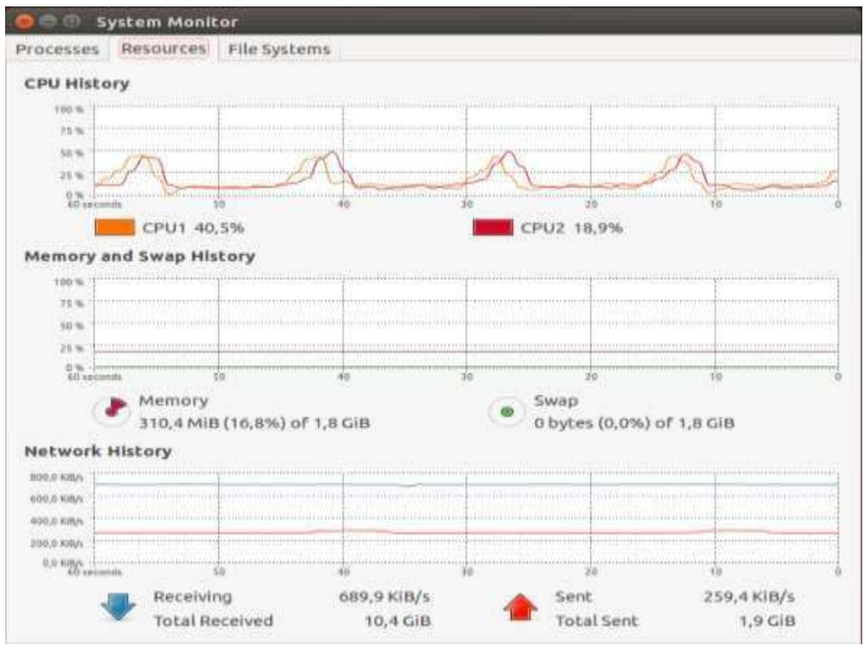
Dalam melihat nilai bagaimana sebuah serangan DoS yang dilakukan nenggunakan perangkat lunak LOIC dalam kurun waktu 30 menit dengan jumlah serangan fikif yang ditentukan, dapat dibuat berupa tabel berikut:

Tabel 4.3 Nilai Pengujian

Jenis Srangan	Client Fiktif	Times	CPU 1	CPU 2	RAM	Network
Keadaan Normal	-	-	5,0%	3.0%	16.7%	-
TCP Flood	10	30	3%-40.0%	5%-37%	16.8%	690 KiB
TCP Flood	25	30	10%-44%	10-39%	17.1%	1.4 MiB
TCP Flood	50	30	30%-70%	24%-50%	17.6%	3.1 MiB
TCP Flood	75	30	35%-55%	43%-64%	17.8%	3.6 MiB
TCP Flood	99	30	36%-60%	44%-61%	18.0%	3.8 MiB

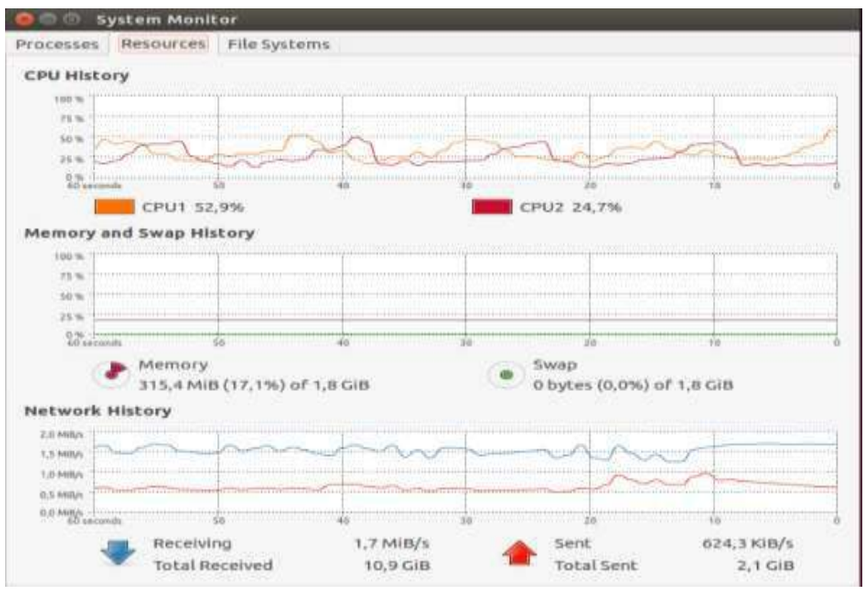
Dalam tabel 4 menghasilkan nilai maksimal yang telah diuji dengan kurun waktu 30 menit dengan serangan TCP Flood, pengujian penyerangan dilakukan dengan 10,25,50,75, dan 99 client fiktif dapat menjelaskan serangan DoS jenis TCP Flood yang dilakukan dengan menggunakan LOIC besarnya nilai yang ditentukan berdasarkan jumlah client fiktif hingga dapat mempengaruhi kinerja server.

System monitor dalam serangan yang dilakukan dengan 10 host akan menunjukkan hasil berikut:



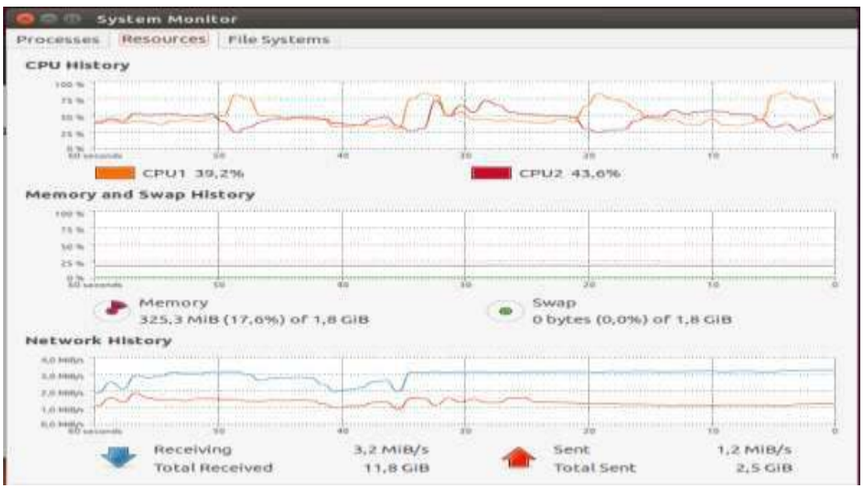
Gambar 4.5 Monitoring Network History Serangan 10 Host Fiktif

Pada gambar 4.5 dapat dijelaskan bahwa paket yang di dikirim dari serangan DoS menggunakan LOIC dengan 10 host fiktif yaitu bernilai CPU 1 maksimal 30.0%, CPU 2 maksimal 26.6% dan RAM 25.5%



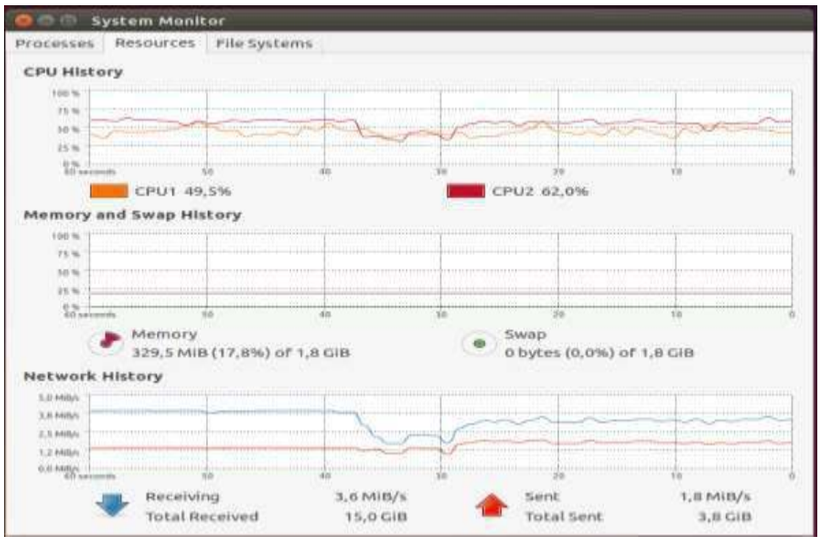
Gambar 4.6 Monitoring Network History Serangan 25 Host Fiktif

Pada gambar 4.6 dapat dijelaskan bahwa paket yang di dikirim dari serangan DoS menggunakan LOIC dengan 25 host fiktif yaitu bernilai CPU 1 maksimal 44.8%, CPU 2 maksimal 39.8% dan RAM 25.9%.



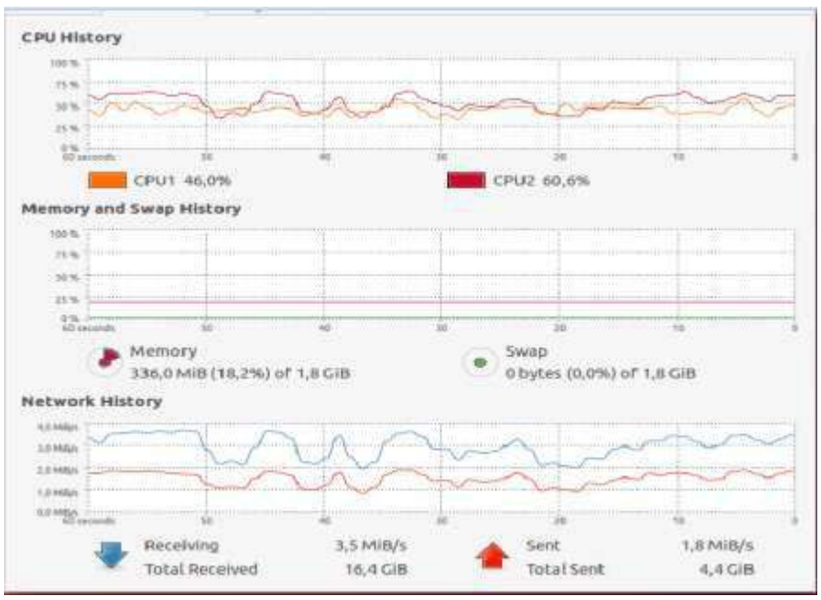
Gambar 4.7 Monitoring Network History Serangan 50 Host Fiktif

Pada gambar 4.7 dapat dijelaskan bahwa paket yang di dikirim dari serangan DoS menggunakan LOIC dengan 50 host fiktif yaitu bernilai CPU 1 maksimal 56.1%, CPU 2 maksimal 47.0% dan RAM 26.7%.



Gambar 4.8 Monitoring Network History Serangan 75 Host Fiktif

Pada gambar 4.8 dapat dijelaskan bahwa paket yang di dikirim dari serangan DoS menggunakan LOIC dengan 75 host fiktif yaitu bernilai CPU 1 maksimal 71.0%, CPU 2 maksimal 56.7% dan RAM 27.9%.



Gambar 4.9 Monitoring Network History Serangan 99 Host Fiktif

Pada gambar 4.9 dapat dijelaskan bahwa paket yang di dikirim dari serangan DoS menggunakan LOIC dengan 99 host fiktif yaitu bernilai CPU 1 maksimal 100.0%, CPU 2 maksimal 57.0% dan RAM 29.0%.

Dengan begitu dari kegiatan yang telah dilakukan dari pengujian 99 host fiktif dan dengan uji coba satu persatu host fiktif didapatkan hasil bila semakin banyak host fiktif yang melakukan serangan semakin besar pengiriman paket yang akan diterima server dan semakin mengarahkan server pada beban berat yang akan diterimanya sehingga mengakibatkan server terjadi *crash*.

4.4.3 Kondisi serangan menggunakan Snort IDPS

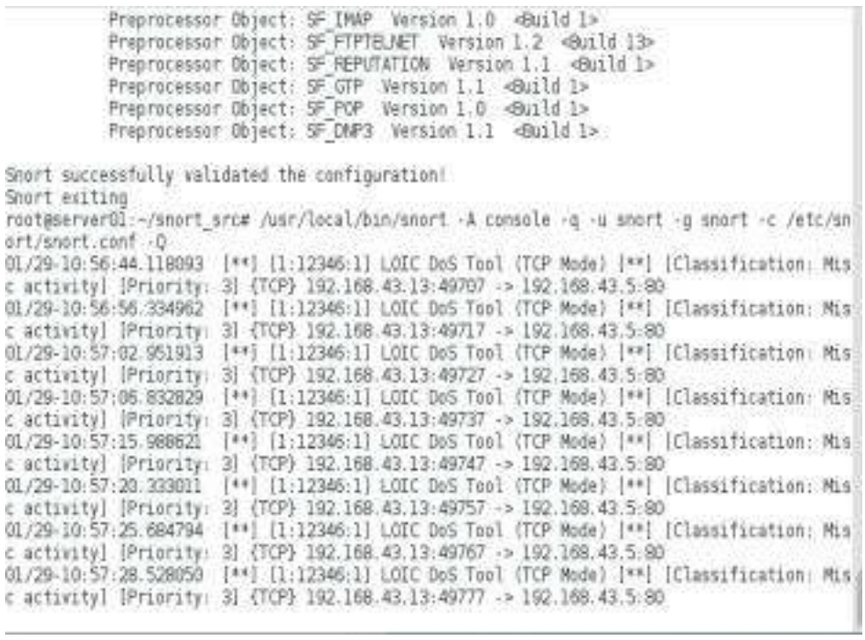
Snort IDPS bekerja bila terjadinya serangan yang nantinya akan melewati netfilter dan diproses oleh Snort apakah itu berupa ancaman atau tidak, bila itu sebuah ancaman paket serangan tersebut akan terdeteksi oleh alert dan netfilter akan memblocking paket tersebut.

Gambar 4.10 Pengujian Serangan LOIC Dengan Snort *IDPS*

Dalam pengujian penyerangan TCP Flood terhadap server dengan menggunakan Snort IDPS mendapatkan hasil, pada LOIC tidak adanya tanda-tanda serangan terhadap server dan bagian permintaan (*Requested*) menunjukkan 0. Dengan begitu pengujian pengamanan server dengan Snort IDPS dapat dinyatakan berhasil.

Kemudian setelah hasil yang didapat dari LOIC, Snort menerima alert dan dengan alert yang telah ditentukan dan dibuat nantinya akan menjadi pemicu untuk mengintrusikan netfilter Iptables untuk memblok paket tersebut. Untuk melihat kinerja

Snort memonitoring keadaan server dari serangan secara real time dapat dilihat dengan tampilan berikut:



Gambar 4.11 Hasil Real Time Mode Console Adanya Serangan

Pada gambar 4.11 dapat dilihat Snort mendeteksi adanya sebuah serangan DoS TCP Flood menggunakan tools LOIC secara real time dalam mode console. Mode console hanya memantau suatu serangan dengan menghasilkan output log file yang masih belum dapat dimengerti bila di buka yaitu berupa kode-kode, untuk itu dibutuhkan tools pendukung dalam membantu kita membaca log file Snort yaitu dibutuhkannya barnyard2 yang nantinya sebagai tools pendukung Snort dalam menerjemahkan output Snort dari berupa kode ke Bahasa yang dapat dimengerti dan mengimputkan log file Snort kedalam database MySQL untuk dibaca oleh BASE. Hasil output Snort mode console seperti berikut:

Gambar 4.12 Log File Snort Mode Console

4.4.4 Pengamatan Database Snort IDPS menggunakan BASE

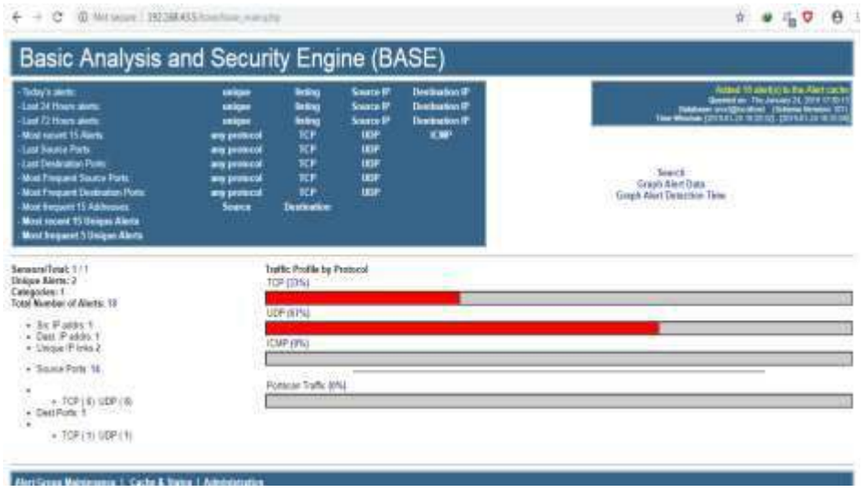
Berbeda dengan mode console Snort yang memantau kejadian yang ada pada Snort dengan menampilkannya secara real time dan menghasilkan output yang sulit dimengerti. Untuk melihat apakah hasil atau output dari alert yang telah terdeteksi, Snort menggunakan Barnyard2 sebagai toolsnya, dengan menjalankan barnyard2 nantinya log output yang dihasilkan akan dapat diterjemahkan hingga dapat dimengerti dan log file diolah kedalam database MySQL hasil output dari barnyard2 seperti berikut:

Gambar 4.13 Pembacaan Alert Dari Log File Dengan Barnyard2

Pada gambar 4.13 untuk melihat alert yang telah ada sebelumnya kita dapat menggunaan tools barnyard2, Setelah alert output Snort diproses oleh barnyard2 dan diimput kedalam database MySQL.

Setelah didapati hasil alert yang telah diolah barnyard2 kedalam database, alert dapat pula dilihat dengan menggunakan web interface BASE, BASE bekerja dengan memanfaatkan acid_event database untuk menampilkan ke web interface dan dengan BASE kita dapat melihat terperinci mengenai jenis serangan, waktu dan alamat mana yang menyerang server.

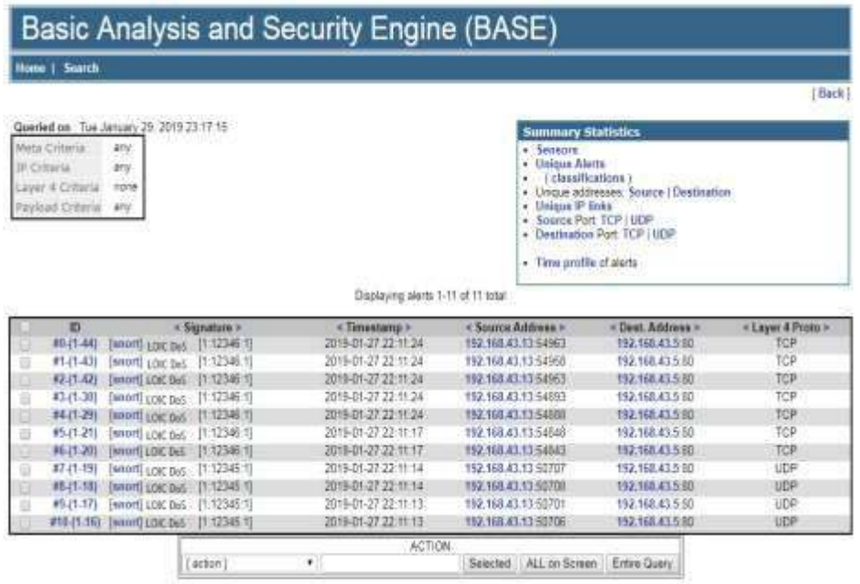
Berikut adalah tampilan awal pada BASE dapat dilihat pada tampilan berikut:



Gambar 4.14 Tampilan *Basic Analysis Security Engine* (BASE)

Pada gambar 4.14 dapat dilihat Output dari serangan DoS LOIC yang di simpan kedalam database di tampilkan ke web *interface* BASE.

Menu web interface BASE yang menampilkan alert bila terjadi serangan pada server dengan rinci waktu dan tanggal serta jenis serangan dari alamat mana ke tujuan mana.



Gambar 4.15 Output Rinci Dari BASE

4.4.5 Notifikasi SMS Gateway dari Output Snort IDPS

Notifikasi SMS Gateway dibutuhkan untuk mendukung administrator dalam mengetahui kondisi pada server bila server terjadi serangan. Administrator akan menerima SMS dan segera dapat mengantisipasi bila terjadinya serangan yang kemungkinan akan berlanjut.

Gambar 4.16 Notifikasi SMS *Gateway*

Pada gambar 4.16 diatas dapat dilihat Snort yang berbasis SMS gateway memberikan kemudahan bagi para *administrator* karena mereka dapat memonitoring secara mobile.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dapat menikmati hiburan, *browsing*, *streaming* , dan tukar menukar data merupakan kelebihan dalam penggunaan jaringan komputer, jaringan komputer tidak hanya memiliki kelebihan untuk pengguna yang menikmatinya saja akan tetapi banyak pula sebuah instansi seperti perusahaan, pemerintahan, pendidikan, kesehatan, dan lainnya. Dari kelebihan itu semua tentu jaringan komputer memiliki kelemahan, kelemahan itu tentunya akan merugikan beberapa pihak penyedia layanan (*Server*) kelemahan yang berhubungan dengan jaringan komputer yaitu pada segi keamanan Server itu sendiri.

Salah satu tindak kejahatan pada jaringan komputer adalah serangan *Denial of Service* (DoS). Serangan *Denial of Service* (DoS) adalah serangan yang mungkin bisa sering kita jumpai diantara serangan-serangan lainnya. Serangan DoS menggunakan sejumlah host untuk membanjiri komputer korban dengan *request* informasi sehingga sistem tidak dapat bekerja dengan normal yang berakibat sistem tidak dapat memberikan layanan sebagaimana mestinya (Ditaprawira *et al,* 2013). Salah satu berntuk dari serangan *Denial of Service* (DoS) adalah TCP *Flooding* dimana penyerangan akan memanfaatkan komunikasi TCP untuk dibanjiri oleh paket data yang diminta.

Solusi yang dapat digunakan adalah dengan dibutuhkannya rancangan sistem yang dapat menjaga jaringan itu sendiri. Sistem jaringan komputer harus dilengkapi

dengan sistem yang dapat mendeteksi adanya penyusupan atau intrusi (*intrusion*) (Harjono *et al*, 2014). Sistem tersebut yang dikenal dengan istilah Snort. Snort merupakan tool yang berbasis *Intrusion Detection System* (IDS) yang dapat memonitor jaringan yang berdampak serangan dan menyimpan serangan pada log (dewi dan patmi, 2017). *Intrusion Detection System* (IDS) diterapkan karena mampu mendeteksi paket-paket berbahaya pada jaringan, bekerja sebagai pendeteksi aktivitas yang mencurigakan pada jaringan (A.Masse *et al*, 2015). Dengan menggunakan *Intrusion Detection System* (IDS) sebagai proteksi jaringan masih belum mangkus/mumpuni dalam menjaga jaringan tersebut.

Penambahan *mode inline* atau *Intrusion Prevention System* menjadi salah satu cara yang mampu mendeteksi serangan dan melakukan *Drop* pada serangan. Melakukan penerapan pada sistem operasi Linux menggunakan Snort dalam mode *inline* mampu mencegah dari serangan yang mengancam (Monoarfa *et al*, 2016),.

Dengan begitu pada penelitian ini akan di implementasikan *Intrusion Detection Prevention System* (IDPS) yang nantinya diharapkan sistem ini dapat menjadi solusi dalam keamanan jaringan. Dengan adanya notifikasi SMS *Gateway* dalam sistem tersebut maka akan memberikan kemudahan bagi admin dalam *monitoring* jaringan secara *mobile*, dan memungkinkan admin mendapatkan notifikasi bila terjadi *attacker*.

Berdasarkan latar belakang yang telah di uraikan di atas maka peneliti mengambil topik penelitian ini dengan judul "**Implementasi Security System Menggunakan Snort IDPS (*Instrusion Detection Prevention System*) Dengan Notifikasi SMS Gateway**".

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, maka dapat di susun rumusan masalah sebagai berikut :

1. Bagaimana Snort *Intrusion Detection Prevention System* (IDPS) mendeteksi dan melakukan drop bila adanya serangan TCP Flooding ?
2. Bagaimana Snort *Intrusion Detection Prevention System* (IDPS) yang dibangun dapat dimonitor dengan berbasiskan web *interface*?
3. Bagaimana Snort dapat memberikan notifikasi menggunakan SMS bila adanya serangan?
4. Bagaimana gambaran sederhana efek dari serangan dari TCP Flooding?

1.3 Batasan Masalah

Berdasarkan batasan masalah yang ada, maka dibuat batasan masalah sebagai berikut :

1. Menggunakan jenis penyerangan TCP Flooding yang menunjukkan gambaran sederhana efek dari serangan terhadap sebuah server.
2. Menggunakan Snort *Intrusion Detection Prevention System* (IDPS) dalam melakukan deteksi dan *drop* bila adanya serangan TCP Flooding.
3. Menggunakan *Basic Analysis and Security Engine* (BASE) dalam memonitor serangan berbasiskan web *interface*.
4. Menggunakan Gammu dalam mendukung Snort untuk memberikan notifikasi berupa SMS bila adanya serangan.

1.4 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah :

1. Snort *Intrusion Detection Prevention System* (IDPS) diharapkan mampu melakukan deteksi dan drop bila adanya serangan TCP Flooding.
2. Snort *Intrusion Detection Prevention System* (IDPS) diharapkan mampu memanfaatkan sistem monitoring menggunakan *Basic Analysis and Security Engine* (BASE) dengan berbasiskan web interface.
3. Dapat memanfaatkan Gammu yang terhubung dengan Snort *Intrusion Detection Prevention System* (IDPS) diharapkan mampu memberikan notifikasi bila adanya serangan TCP Flooding.
4. Dapat menggambarkan secara sederhana bagaimana efek dari serangan TCP Flooding.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan pada bab-bab pembahasan sebelumnya dari proses konfigurasi dan implementasi, dapat diambil beberapa kesimpulan yaitu:

1. Sistem Snort *Intrusion Detection Prevention System* (IDPS) dapat bekerja dengan efektif sebagai pemberi peringatan dini adanya kegiatan penyerangan terhadap jaringan dan server yang diterapkan telah berhasil dan dikembangkan.
2. Sistem Snort *Intrusion Detection Prevention System* (IDPS) bekerja dengan menggunakan Snort *engine* yang memonitor paket data dan mencocokkannya pada *Rules* yang telah ada. Bila paket data teridentifikasi dengan serangan, *Rules* akan meneruskan serangan ke Iptables untuk dipisahkan dan di *drop*.
3. Mekanisme Sistem Snort *Intrusion Detection Prevention System* (IDPS) dari Snort, BASE dan Gammu dalam pengujiannya yaitu penyerangan dilakukan menggunakan tools *Low Orbit Ion Canon* (LOIC) berupa tipe serangan *Denial of Servic*e (DoS) dan SMS *Gateway* sebagai notifikasi bila terjadinya serangan.

4. Data dan log dari Snort disajikan dalam bentuk *Graphical User Interface* (GUI) menggunakan BASE sebagai sistem *monitoring*. Dan Gammu sebagai mesin SMS *Gateway* yang memberikan *administrator* notifikasi SMS

5.2 Saran

Pada penelitian ini penulis menemukan saran-saran yang perlu untuk pengembangan selanjutnya adalah:

1. Melakukan pengujian yang melibatkan lebih banyak lagi serangan dan *Rules* yang dapat mempengaruhi kinerja server dan merugikan server.
2. Menambahkan Pulledpork untuk melakukan update secara otomatis *Rules* dan penambahan *Rules* yang lebih lengkap lagi dengan menggunakan Pulledpork.
3. Penambahan modul lain yang mendukung kinerja IDPS untuk membantu efisiensi kerja seperti menambahakn DNS Server, DHCP Server, dan lainnya.

DAFTAR PUSTAKA

A.Masse, Fitriyanti, Andi Nurul Hidayat, Badrianto. (2015). Penerapan Nework Intrusion Detection System Menggunakan Snort Berbasis Database Mysql Pada Hotspot Kota. Palu: Jurnal Electronik Sistem Informasi dan Komputer.

Affandi, Mohammad, Sigit Setyowibowo. (2013). Impelementasi Snort Sebagai Alat Pendeteksi Intrusi Menggunakan Linux. Malang: Jurnal Teknologi Informasi.

Ariyadi, Tamsir, Yesi Novaria Kunang, Rusmala Santi. (2012). Implementasi Intrusion Prevention System (Ips) Pada Jaringan Komputer Kampus B Universitas Bina Darma. Palembang: Jurnal Ilmiah Teknik Informatika Ilmu Komputer.

Ashoor, Asma Shaker, Prof. Sharad Gore. (2011). Importance of Instrusion Detection System (IDS). India: International Journal of Scientific & Engineering Research.

Batubara, S., Wahyuni, S., & Hariyanto, E. (2018, September). Penerapan Metode Certainty Factor Pada Sistem Pakar Diagnosa Penyakit Dalam. In Seminar Nasional Royal (SENAR) (Vol. 1, No. 1, pp. 81-86).

Dewi, Ervin Kusuma, Patmi Kasih. (2017). Analisis Log Snort Menggunakan Network Forensic. Kediri: Jurnal Ilmiah Penelitian dan Pembelajran Infromatika (JIPI).

Ditaprawira, Gusti N.G.Bayu, Ary Mazharuddin Shiddiqi, Baskoro Adi Pratomo. (2013). Penerapan Congestion Participation Rate (CPR) untuk Pendeteksian Serangan Low-Rate. Surabaya: Jurnal Teknik Pomits.

Hardi, Richki, zaini. (2017). Implementasi Sistem Keamanan Komputer Menggunakan Sistem Terintegrasi Client Server Metode Service Oriented Architecture (SOA). Bontang: Jurnal Teknologi Terpadu.

Harjono, Agung Purwo Wicaksono, Debian. (2014). Sistem Deteksi Intrusi dengan Snort (Intrusion Detection System with Snort). Purwokerto: JUITA.

Hariyanto, E., & Rahim, R. (2016). Arnold's cat map algorithm in digital image encryption. International Journal of Science and Research (IJSR), 5(10), 1363-1365.

Hartanto, S. (2017). Implementasi fuzzy rule based system untuk klasifikasi buah mangga. TECHSI-Jurnal Teknik Informatika, 9(2), 103-122.

Harumy, T. H. F., & Sulistianingsih, I. (2016). Sistem penunjang keputusan penentuan jabatan manager menggunakan metode mfep pada cv. Sapo durin. In Seminar Nasional Teknologi Informasi dan Multimedia (pp. 6-7).

Hendrawan, J. (2018). Rancang Bangun Aplikasi Mobile Learning Tuntunan Shalat. INTECOMS: Journal of Information Technology and Computer Science, 1(1), 44-59.

Jacob, Neyole Misiko, Muchelule Yusuf Wanjala. (2017). A Review of Intrusion Detection Systems. Kenya: International Journal of Computer Science and Information Tecnology Research.

Khairul, K., Haryati, S., & Yusman, Y. (2018). Aplikasi Kamus Bahasa Jawa Indonesia dengan Algoritma Raita Berbasis Android. Jurnal Teknologi Informasi dan Pendidikan, 11(1), 1-6.

Kurnia, D., Dafitri, H., & Siahaan, A. P. U. (2017). RSA 32-bit Implementation Technique. Int. J. Recent Trends Eng. Res, 3(7), 279-284.

Mariance, U. C. (2018). Analisa dan Perancangan Media Promosi dan Pemasaran Berbasis Web Menggunakan Work System Framework (Studi Kasus di Toko Mandiri Prabot Kota Medan). Jurnal Ilmiah Core IT: Community Research Information Technology, 6(1).

Marlina, L., Muslim, M., Siahaan, A. U., & Utama, P. (2016). Data Mining Classification Comparison (Naïve Bayes and C4. 5 Algorithms). Int. J. Eng. Trends Technol, 38(7), 380-383.

Marlina, L., Putera, A., Siahaan, U., Kurniawan, H., & Sulistianingsih, I. (2017). Data Compression Using Elias Delta Code. Int. J. Recent Trends Eng. Res, 3(8), 210-217.

Mehra, Pritika. (2012). A brief study and comparison of Snort and Bro Open Source Network Intrusion Detection Syste. India: IJARCCE.

Monoarfa, Mohamad Nurul Huda, Xaverius B.N Najoan, ST.,MT,Alicia A.E Sinsuw, ST.,MT. (2016). Analisa dan Implementasi Network Intrusion Prevention System di Jaringan Universitas Sam Ratulangi. Manado: E-Journal Teknik Elektro dan Komputer.

Prihasmoro, Sukma Ageng, Yuliana Rachmawati, Erfanti Fatkhiyah. (2014). Simulasi Sistem Deteksi Penyusup Dalam Jaringan Komputer Berbasis Web Interface Serta Pencegahan Untuk Meningkatkan Keamanan. Yogyakarta: Jurnal JARKOM.

Putri, N. A. (2018). Sistem Pakar untuk Mengidentifikasi Kepribadian Siswa Menggunakan Metode Certainty Factor dalam Mendukung Pendekatan Guru. INTECOMS: Journal of Information Technology and Computer Science, 1(1), 78-90.

Putri, R. E., & Siahaan, A. (2017). Examination of document similarity using Rabin-Karp algorithm. International Journal of Recent Trends in Engineering & Research, 3(8), 196-201.

Rahim, R., Aryza, S., Wibowo, P., Harahap, A. K. Z., Suleman, A. R., Sihombing, E. E., ... & Agustina, I. (2018). Prototype file transfer protocol application for LAN and Wi-Fi communication. Int. J. Eng. Technol., 7(2.13), 345-347.

Rouesch, Martin. (1999). Snort – Lightweight Intrusion Detection for Networks. USA: USENIX Assocation.

Ruwaida, D., & Kurnia, D. (2018). Rancang Bangun File Transfer Protocol (FTP) dengan Pengamanan Open SSL pada Jaringan VPN Mikrotik di SMK Dwiwarna. CESS (Journal of Computer Engineering, System and Science), 3(1), 45-49.

Sarif, M. I. (2017). Penemuan Aturan yang Berkaitan dengan Pola dalam Deret Berkala (Time Series).

Sondakh, Glend, Meicsy E. I. Najoan, ST., MT, Arie S. Lumenta, ST, MT. (2014). Perancangan Filtering Firewall Menggunakan Iptables di Jaringan Pusat Teknologi Informasi Unsrat. Manado: E-Journal Teknik Elektro dan Komputer.