BAB III

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Sesuai dengan struktur organisasi yang dijelaskan pada bab 2, praktik kerja magang dilakukan di bagian *Software Development* di departemen *Human resources Information System* dengan kedudukan sebagai programmer.

Koordinasi dilakukan bersama dengan bagian software development dan *system administrator* pada HRIS.

3.2 Tugas Yang Dilakukan

Tugas yang diberikan yaitu melakukan *setting* snort agar dapat digunakan di dalam server, membuat program yang dapat berjalan otomatis memblok paket sesuai dengan rekaman snort, serta membuat *web interface* agar admin mengetahui percobaan serangan apa saja yang pernah dilakukan.

3.3 Uraian Pelaksanaan Kerja Magang

Pelaksanaan magang yang dilaksanakan di Kompas bertugas untuk menambahkan fitur keamanan jaringan di dalam *server* kompas dengan menggunakan snort. Program snort sendiri perlu dilakukan tahap installasi dan *setting*, kemudian digunakan barnyard2 yang merupakan program yang membantu menerjemahkan hasil rekaman program snort, dan memasukkannya kedalam database MySQL. Barnyard2 juga perlu dilakukan tahap installasi dan *setting*.

3.3.1 User Requirements

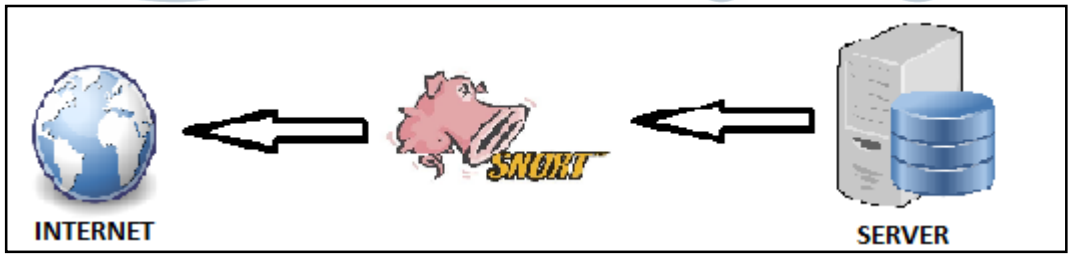
User meminta agar setiap paket yang masuk kedalam *database* snort dapat di blok secara otomatis, serta ingin mendapatkan informasi serangan, membuat user

baru, mengganti *password*, membuka kembali akses IP *address* yang di blok, menghapus *user* dalam bentuk web interface. Berdasarkan permintaan *user*, maka dirancang sebuah program pendukung dengan bantuan program iptables yang merupakan program yang sudah tersedia di sistem operasi linux. Adapun *web interface* yang membantu *server administrator* agar lebih mudah memantau jenis serangan apa saja yang masuk ke sistem dan melakukan kegiatan *administrator* lain seperti membuat user, menghapus *IP address*, dan menghapus user.

3.3.2 Installasi

Proses pelaksanaan dalam kerja magang pertama kali yang dilakukan adalah setting snort di server dan sistem operasi dari server itu sendiri adalah Ubuntu Server. Untuk dapat terkoneksi ke komputer server digunakan SSH. Digunakan SSH karena tidak diketahuinya keberadaan secara fisik komputer *server internal* perusahaan. SSH merupakan kependekan dari *Secure Socket Shell* yang merupakan protokol jaringan yang berfungsi untuk me-*remote* sebuah komputer melalui jaringan dengan transmisi data yang aman karena sebelum data dikirim, data sudah di-*encrypt* terlebih dahulu. *User* dan *password* masuk ke dalam SSH telah diberikan *privilege* sebagai *administrator*.

Dalam perancangannya, aplikasi snort ini ditempatkan pada sebuah server development kompas dengan skema logika :



Gambar 3.1 Topologi Tempat Snort Diinstall

Sebelum menginstall snort, ada beberapa library yang diperlukan agar snort dapat diinstall dan dapat berjalan dengan baik. Pertama pastikan, mendapatkan *privilege* sebagai admin dengan mengksekusi perintah "sudo su" pada *terminal* SSH/komputer server. Perintah ini akan membawa anda pada input sebuah *password,* setelah *password* terisi dan benar maka *privilege* sebagai admin telah didapatkan. Selanjutnya adalah menginstall "build-essential" *package* dalam ubuntu dengan mengeksekusi *command* "apt-get install build-essential" dan tunggu sampai proses *download* dan installasi selesai. Build-essential merupakan paket dari berbagai macam *library*.

Setelah *package* "build-essential" selesai terpasang selanjutnya adalah dengan menginstall "libpcap-dev", "libpcre3-dev", "libdumbnet-dev". Ketiga *packages* tersebut merupakan paket development untuk program-program tersebut. Untuk menginstall ketiga *packages* tersebut ketikkan perintah "apt-get install –y libpcap-dev libpcre3-dev libdumbnet-dev". Ketika dieksekusi perintah tersebut akan secara otomatis menginstall ketiga paket tersebut.

Masih ada tiga *library* yang diperlukan untuk menginstall snort, ketiga library tersebut adalah "zlib1g-dev" yang merupakan *compression library* untuk development, yang kedua dan ketiga adalah "flex" dan "bison". "flex" dan "bison" merupakan utilitas unix yang dapat membantu untuk menulis parser dengan sangat cepat untuk format *file arbitary*. Cara untuk menginstall ketiga *library* tersebut adalah, pertama eksekusi perintah "apt-get install zlib1g-dev" perintah ini untuk menginstall *library* "zlib1g-dev", tunggu sampai proses unduh dan install selesai. Selanjutnya adalah masukan perintah "apt-get install –y bison flex" ketika perintah

ini dieksekusi, maka “bison” dan ”flex” akan diunduh dan secara otomatis akan terinstall. Tunggu sampai proses selesai.

Setelah semua program dan library yang dibutuhkan telah terinstall maka selanjutnya adalah installasi snort.

Pertama membuat folder pada direktori mana saja dengan nama folder “snort” dengan menjalan kan perintah “mkdir snort” pada CLI. Lalu berpindah ke direktori tersebut dan download daq dan snort dengan menjalankan perintah “wget https://snort.org/downloads/snort/daq-2.0.6.tar.gz” (untuk mendownload *file* daq-2.0.6.tar.gz) dan “wget https://snort.org/downloads/snort/snort-2.9.8.3.tar.gz“ (untuk mendownload *file* snort-2.9.8.3.tar.gz).

Setelah kedua program telah selesai diunduh, selanjutnya adalah dengan mengekstrak *file* daq-2.0.6.tar.gz menggunakan program tar, perintahnya adalah “tar xvfz daq-2.0.6.tar.gz” tunggu sampai *file* selesai ter ekstrak, setelah selesai selanjutnya pindah ke direktory daq dan selanjutnya adalah pemasangan program dengan melakukan perintah “./configure && make && make install”. Setelah sebuah *file* selesai diekstrak, biasanya akan dibentuk suatu folder dengan nama sama dengan *file* kompresi dan jika ingin melakukan installasi tau penjalankan program yang telah terekstrak terlebih dahulu harus menuju direktori tersebut.

Program daq sudah selesai terinstall tahap installasi selanjutnya adalah installasi snort. Sama dengan installasi program daq hanya ada sedikit perbedaan, yaitu pada tahap konfigurasi yaitu dengan melakukan perintah “./configure --enable-sourcefire && make && sudo make install”. Setelah snort selesai terinstall selanjutnya adalah dengan mengetikkan perintah “sudo ldconfig” dan “sudo ln –s /usr/local/bin/snort /usr/sbin/snort” yang berfungsi untuk memudahkan

pemanggilan program snort. Tahap installasi terakhir adalah dengan pemasangan program barnyard2. Pertama adalah membuat direktory barnyard2 setelah membuat direktori menuju direktori yang telah dibuat dan mendownload barnyard2 via "git clone" dengan memasukkan perintah "Git clone https://github.com/firnsy/barnyard2.git". Selanjutnya masih pada direktori yang sama ketikkan perintah "autoreconf –fvi –I ./m4". Barnyard2 memerlukan akses ke *library* dnet.h dari paket libdumbnet dari installasi dengan ubuntu, maka dari itu masukkan perintah "sudo ln –s /usr/include/dumbnet.h /usr/include/dnet.h" dan "sudo ldconfig".

Setelah mendapatkan akses ke *library* yang diinginkan maka selanjutnya adalah installasi barnyard2 dengan memasukkan perintah "./configure --with-mysql --with-mysql-libraries=/usr/lib/x86_64-linux-gnu" untuk linux dengan arsitektur 64bit atau "./configure --with-mysql --with-mysql-libraries=/usr/lib/i386-linux-gnu" untuk linux dengan arsitektur 32bit. Dengan ini tahap installasi selesai dan dilanjutkan dengan tahap konfigurasi

3.3.3 Konfigurasi

Pertama program yang akan dikonfigurasi adalah program Snort dengan memaasukkan perintah ini secara berurutan.

Membuat direktori untuk program Snort :

1. sudo mkdir /etc/snort
2. sudo mkdir /etc/snort/rules
3. sudo mkdir /etc/snort/rules/iplists
4. sudo mkdir /etc/snort/preproc_rules sudo
5. mkdir /usr/local/lib/snort_dynamicrules

6. sudo mkdir /etc/snort/so_rules

Membuat *file* yang dapat menyimpan *rules* dan *ip lists* :

1. sudo touch /etc/snort/rules/iplists/black_list.rules
2. sudo touch /etc/snort/rules/iplists/white_list.rules
3. sudo touch /etc/snort/rules/local.rules
4. sudo touch /etc/snort/sid-msg.map

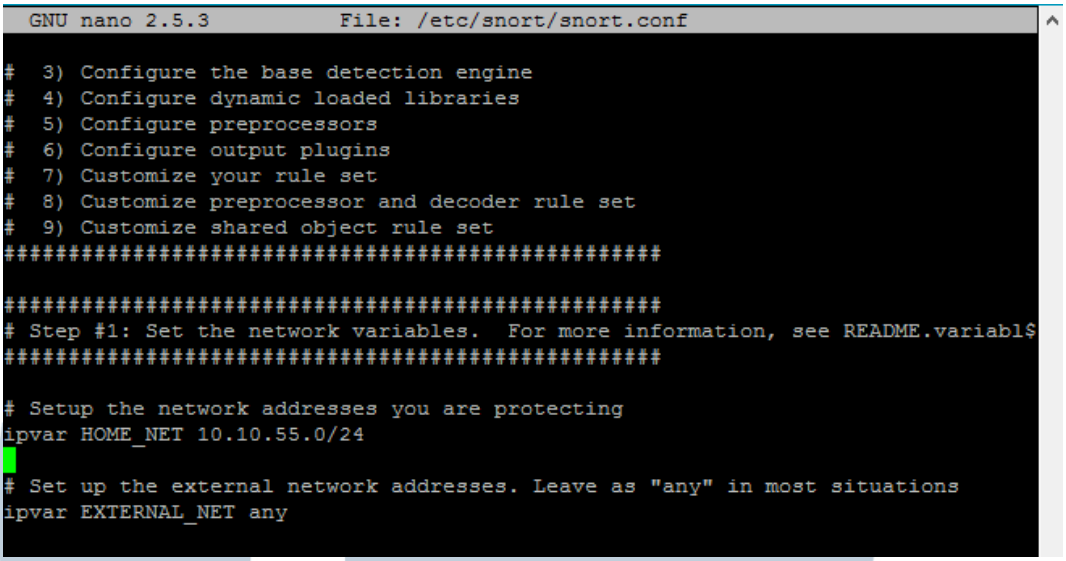
Membuat direktori untuk menyimpan hasil *log* dari program Snort

1. sudo mkdir /var/log/snort
2. sudo mkdir /var/log/snort/archived_logs

Setelah folder dan *file* terbuat selanjutnya adalah, *copy* semua isi *file* dari folder "etc" yang berasal dari folder snort yang telah diunduh pada tahap installasi menuju direktori dimana folder snort dibuat sebelumnya.

Setelah semua *file* selesai diduplikasi selanjutnya adalah dengan memasukkan perintah "sudo sed -i "s/include \$RULE_PATH/#include \$RULE_PATH/" /etc/snort/snort.conf" perintah tersebut akan meng-*comment* semua *ruleset* yang berada pada *file* 'snort.conf.'

Selanjutnya adalah dengan meng-*edit* isi dari *file* snort.conf yang berisi dari konfigurasi program snort.



Gambar 3.2 Hasil Setting HOME_NET

Pada tahap ini yang harus ditentukan adalah besaran *network address* yang ingin diproteksi. Dengan menuliskan script "ipvar HOME_NET 10.10.55.0/24" yang berarti dengan *network address* 10.10.55.0 dengan *prefix* 24 maka *range IP address* yang ingin diproteksi adalah 10.10.55.1 sampai dengan 10.10.55.254.

Selanjutnya adalah men-set variabel lokasi file rules diletakkan. Dalam kasus ini *file* rules terletak pada *file* "/etc/snort/rules" maka dari itu pada file snort.conf ketikkan perintah sebagai berikut :

var RULE_PATH /etc/snort/rules

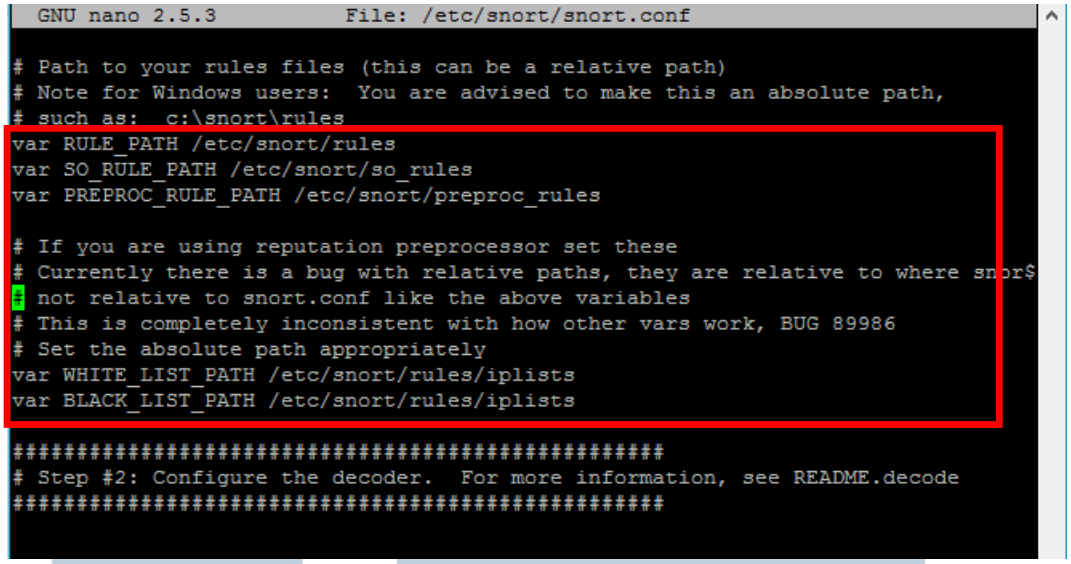
var SO_RULE_PATH /etc/snort/so_rules

var PREPROC_RULE_PATH /etc/snort/preproc_rules

var WHITE_LIST_PATH /etc/snort/rules/iplists

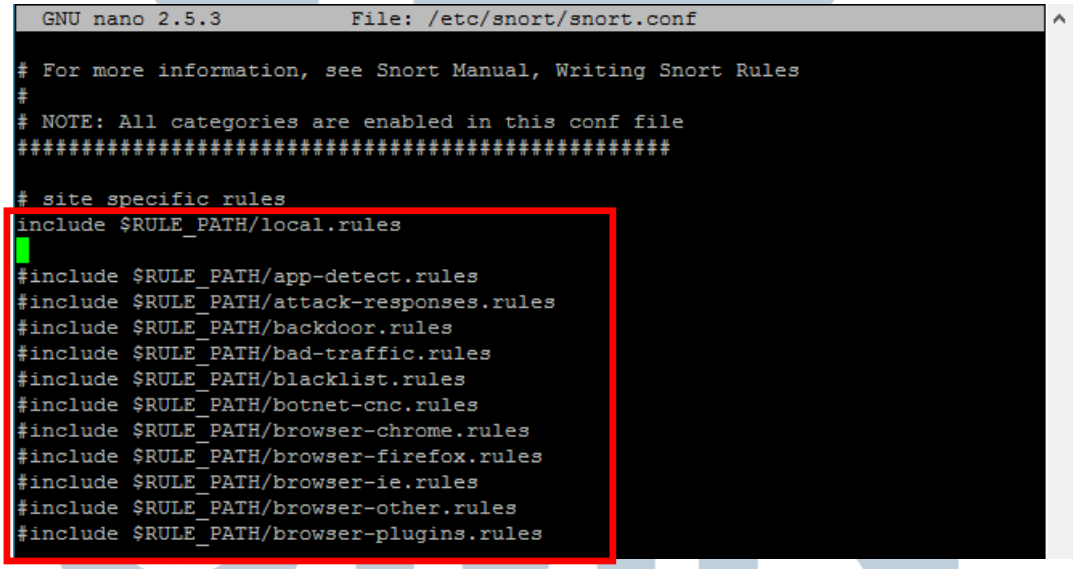
var BLACK_LIST_PATH /etc/snort/rules/iplist

sehingga didapat hasil seperti gambar 3.2



Gambar 3.3 hasil setting lokasi rules

Dan selanjutnya adalah melakukan *setting* lokasi *file* rule yang telah dibuat, dengan memanfaatkan variabel RULE yang telah dibuat sebelumnya.

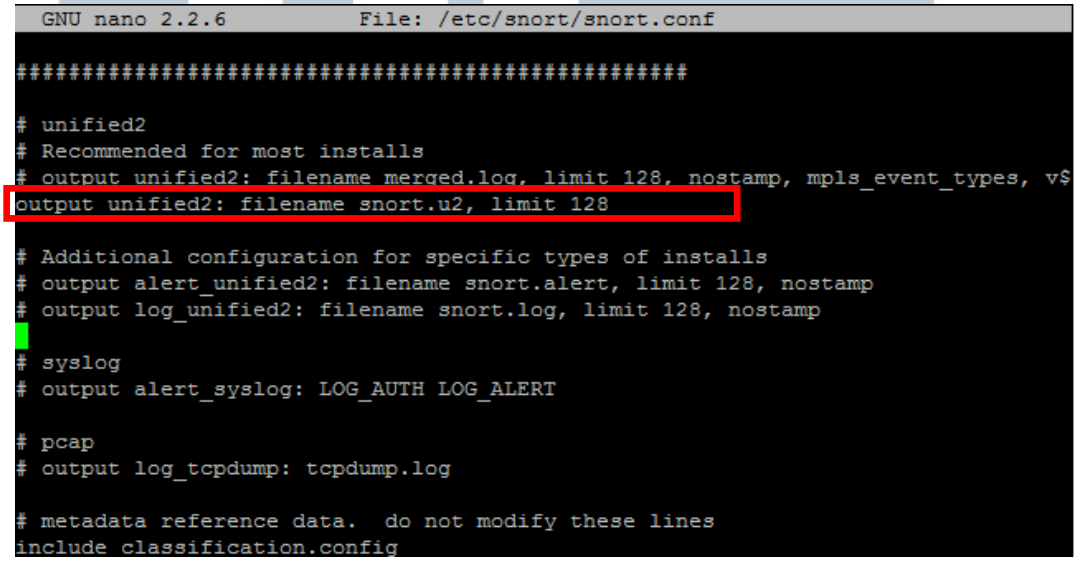


Gambar 3.4 hasil setting lokasi rules

Gambar diatas merupakan *file* ".rules" yang ingin digunakan. *Files* ".rules" berisi *rules* snort yang bertujuan untuk membaca paket jika *file* ingin digunakan maka hilangkan tanda "#" pada *line* yang berada dalam kotak merah. $RULE_PATH sendiri adalah variabel yang berisi direktori ke file .rules yang sebelumnya telah diset di atas. Untuk melakukan edit terhadap rules snort maka

dapat diedit *file* dengan ekstensi ".rules" pada *folder* /etc/snort/rules dan ketikkan *rules* sesuai dengan kebutuhan sistem. Konfigurasi snort selesai.

Selanjutnya adalah konfigurasi program Barnyard2 yang berfungsi sebagai menyimpan snort ke dalam database sehingga lebih mudah dibaca. Sebelum mengkonfigurasi Barnyard2 terlebih dahulu ubah isi dari *file* snort.conf dan tambahkan script pada baris kurang lebih baris ke 526 seperti gambar 3.5



Gambar 3.5 hasil setting output snort

Script di atas bertujuan untuk membuat *file output* yang sebelumnya berekstensi ".log" menjadi ".u2" ekstensi yang dapat dibaca oleh Barnyard2. Setelah itu buka program *database* MySQL melalui *console* dan masukkan perintah seperti gambar 3.6.

```
$ mysql -u root -p
mysql> create database snort;
mysql> use snort;
mysql> source ~/snort_src/barnyard2-2-1.14-336/schemas/create_mysql
mysql> exit
```

Gambar 3.6 query mysql untuk barnyard2

Setelah MySQL selesai dikonfigurasi selanjutnya masih ada *file* barnyard2.conf yang perlu di-*copy* dan dikonfigurasi. Pertama adalah dengan me-*copy file* barnyad2.conf yang terletak pada direktori tempat barnyard diunduh dan di-*copy* pada direktori /etc/snort yang merupakan direktori tempat me-*install* snort. Dan yang terakhir dari tahap konfigurasi adalah dengan mengetikkan script "output database: log, mysql, user=<username database> password=<password database> dbname=snort host=<alamat IP server Database Mysql berada>"

pada baris terakhir *file* barnyard2.conf.

```
  GNU nano 2.2.6                        File: /etc/snort/barnyard2.conf

# output alert_fwsam: <SnortSam Station>:<port>/<key>
#
#  <FW Mgmt Station>:  IP address or host name of the host running SnortSam.
#  <port>:         Port the remote SnortSam service listens on (default 898).
#  <key>:              Key used for authentication (encryption really)
#              of the communication to the remote service.
#
# Examples:
#
# output alert_fwsam: snortsambox/idspassword
# output alert_fwsam: fw1.domain.tld:898/mykey
# output alert_fwsam: 192.168.0.1/borderfw  192.168.1.254/wanfw
#

output database: log, mysql, user=root password=123456 dbname=snort host=10.10.55.3
```

Gambar 3.7 Script barnyard2.conf

Dalam praktik kerja magang kali ini, *script database config* yang harus dibuat adalah dua *script* pertama untuk *database* snort yang merupakan *database*

untuk merekam log dan apasaja jenis serangan yang telah dilakukan, IP asal, port tujuan dan waktu melakukan serangan.

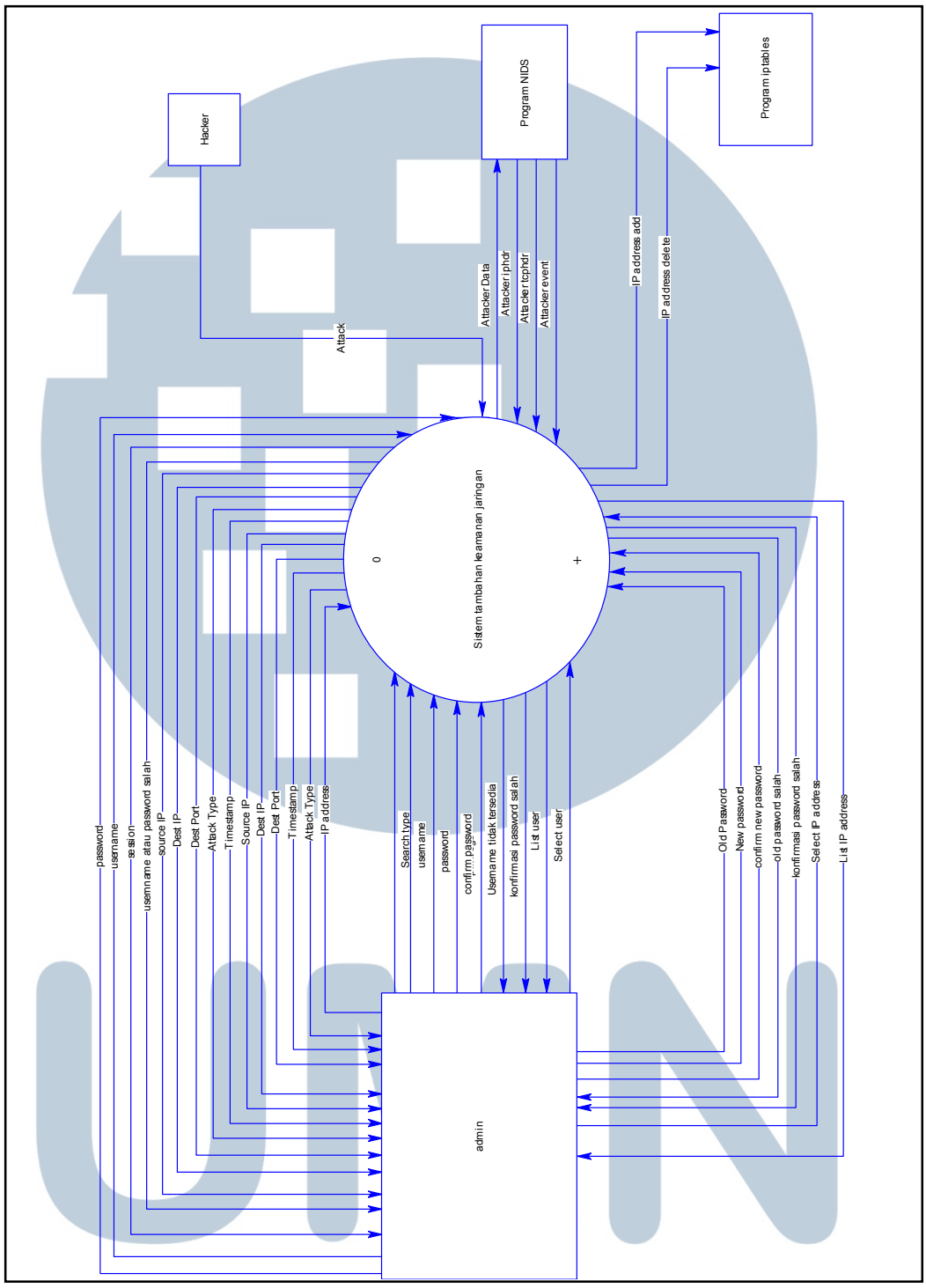
Dan *database* kedua adalah *database* iptables. *Database* iptables ini berfungsi untuk membantu program pendukung dan *web interface* dalam melakukan tugasnya seperti contoh melakukan *block* pada paket yang masuk dan membuka *block* jika IP tersebut telah terkena *block.*

3.3.4 Perancangan DFD Sistem

Dalam merancang sistem, dibutuhkan dua aplikasi yaitu, aplikasi web agar administrator dapat me-*manage* sistem dengan lebih mudah, dan aplikasi program pendukung dalam bahasa python untuk melakukan perintah sistem seperti, membuka blok dan melakukan blok secara otomatis.

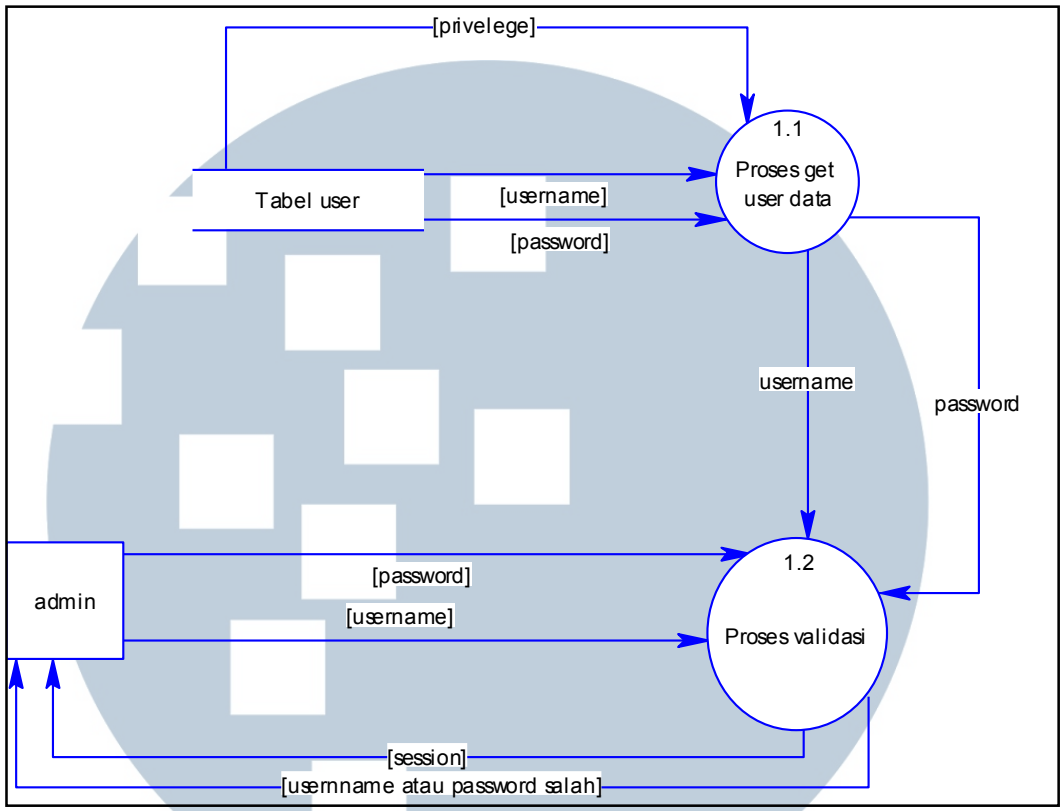
Kemampuan utama dari aplikasi web adalah, dapat melihat detail serangan seperti, IP asal, IP tujuan, jenis serangan, port yang dituju, dan waktu penyerangan, dan kemampuan pendukung seperti, menambah user, menghapus user, mengganti password, dan membuka kembali IP yang telah diblok. Kemampuan dari program pendukung adalah dapat mengambil data dari *database*, melakukan blok otomatis sesuai dengan isi dari *database* daftar IP penyerang, dapat membuka blok secara manual dan otomatis setiap jam 23.59.

Sesuai dengan kemampuan sistem yang dibutuhkan maka, dibuatlah DFD hingga *level* dua. Berikut adalah ilustrasi Context diagram dan *level* satu yang tertuang dalam diagram gambar 3.8 dan gambar 3.9.

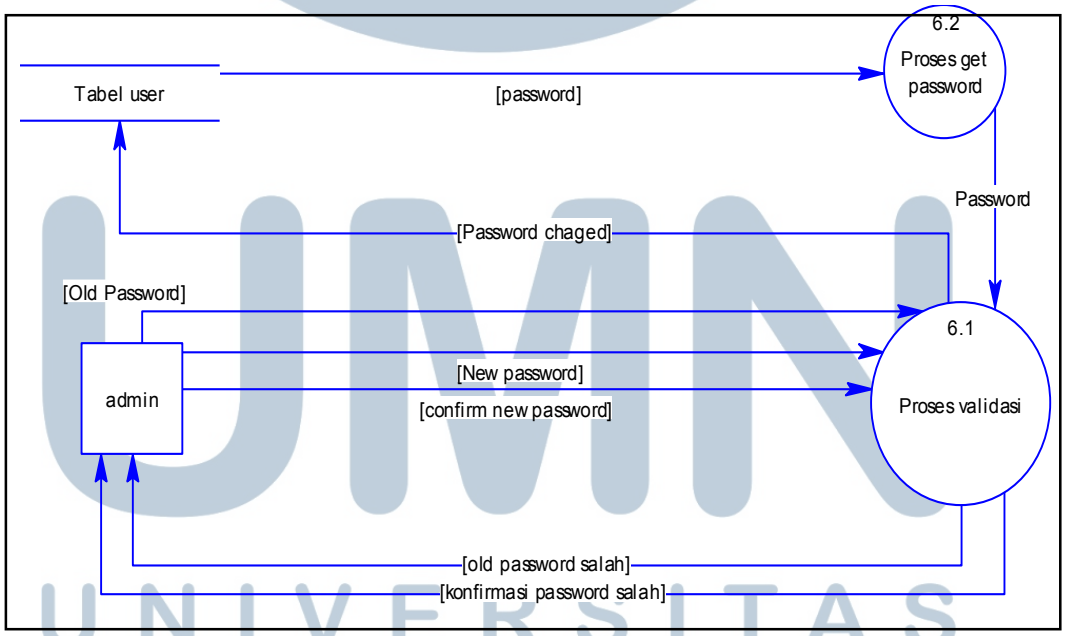


Gambar 3.8 Context Diagram Sistem

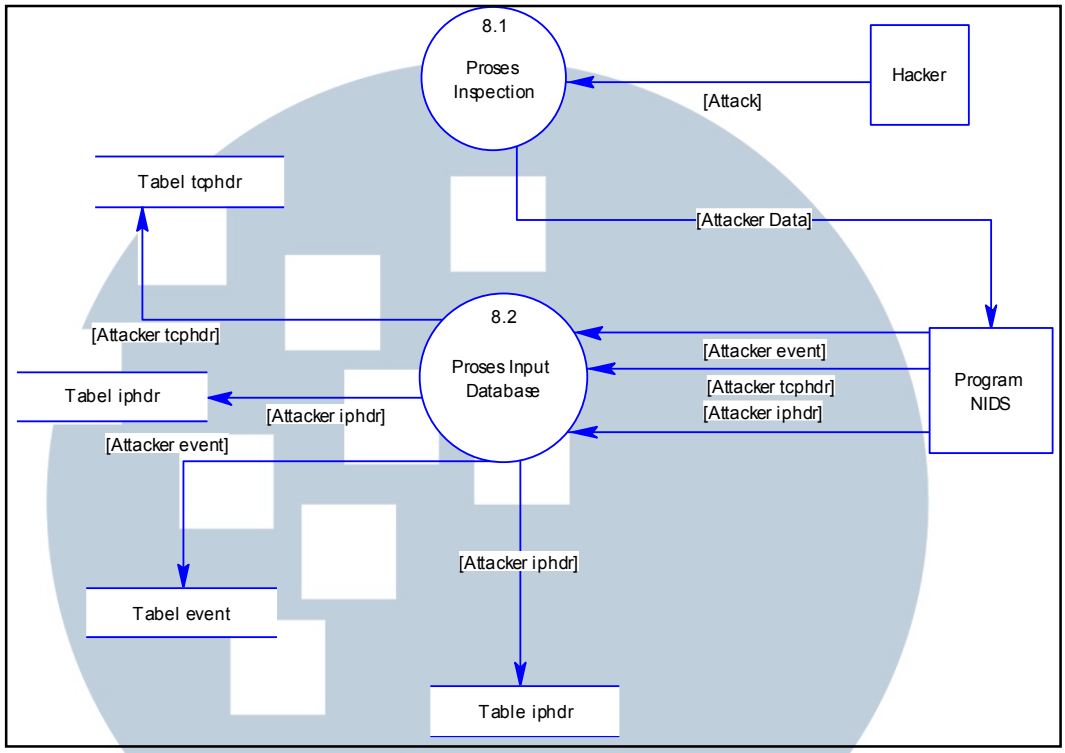
Pada gambar 3.8 merupakan konteks diagram dari sistem, dimana sistem memiliki entitas admin dan hacker yang merupakan individu dan sistem NIDS serta program iptables yang merupakan entitas sistem.



Gambar 3.10 DFD Level 2 Proses Login



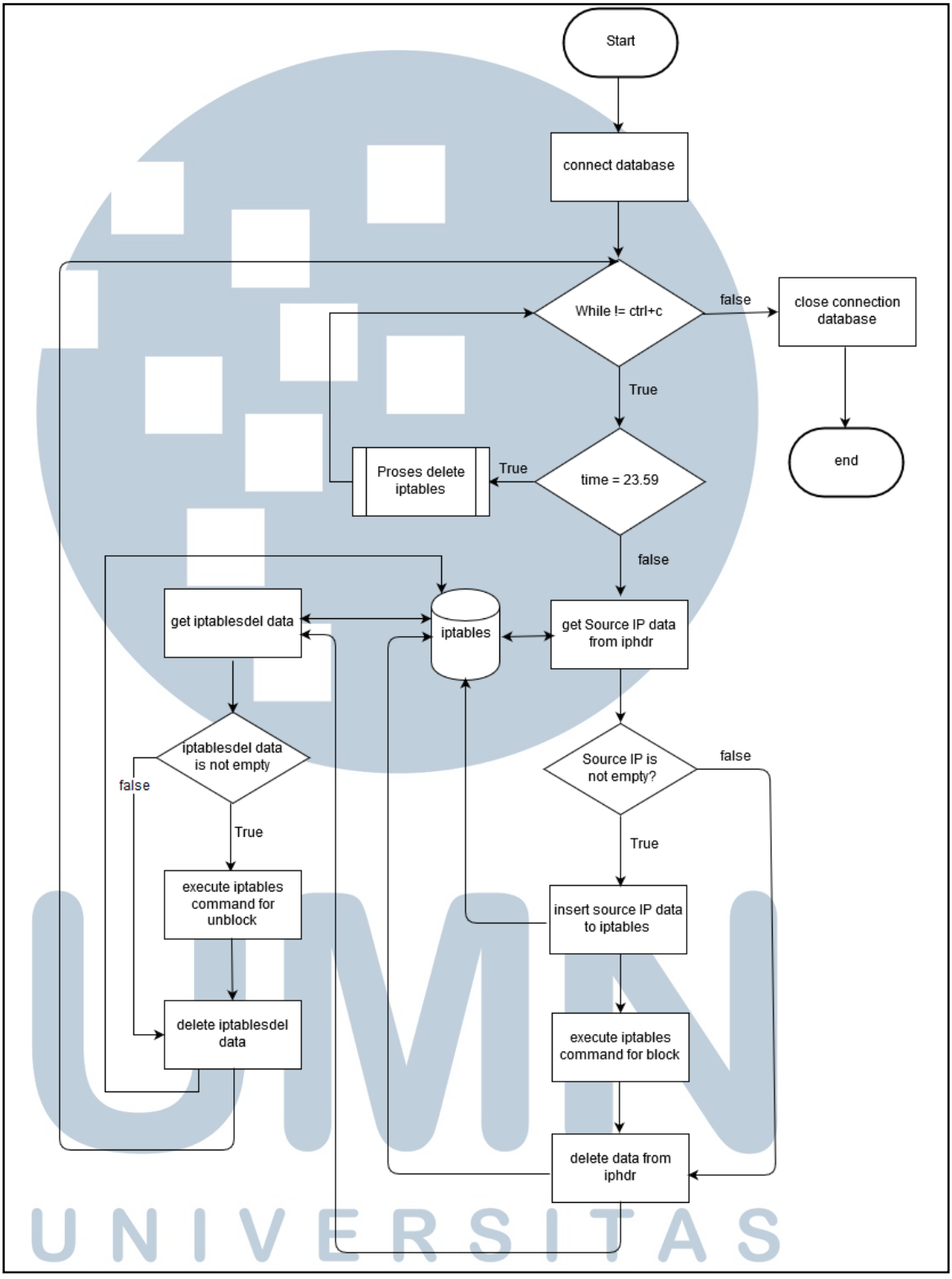
Gambar 3.11 DFD level 2 Proses Change Password



Gambar 3.12 DFD Level 2 Proses NIDS

3.3.5 Perancangan Program Pendukung

Program pendukung merupakan program yang dirancang dengan bahasa pemrograman python. Program pendukung ini berfungsi untung mengambil data dari database yang berupa *IP address* dan memasukkannya ke dalam iptables yang merupakan program *firewall* di sistem operasi linux. Dalam program ini ada juga *scheduller* yang berfungsi untuk mengatur agar isi rules dari iptables dikosongkan pada waktu yang telah ditentukan.



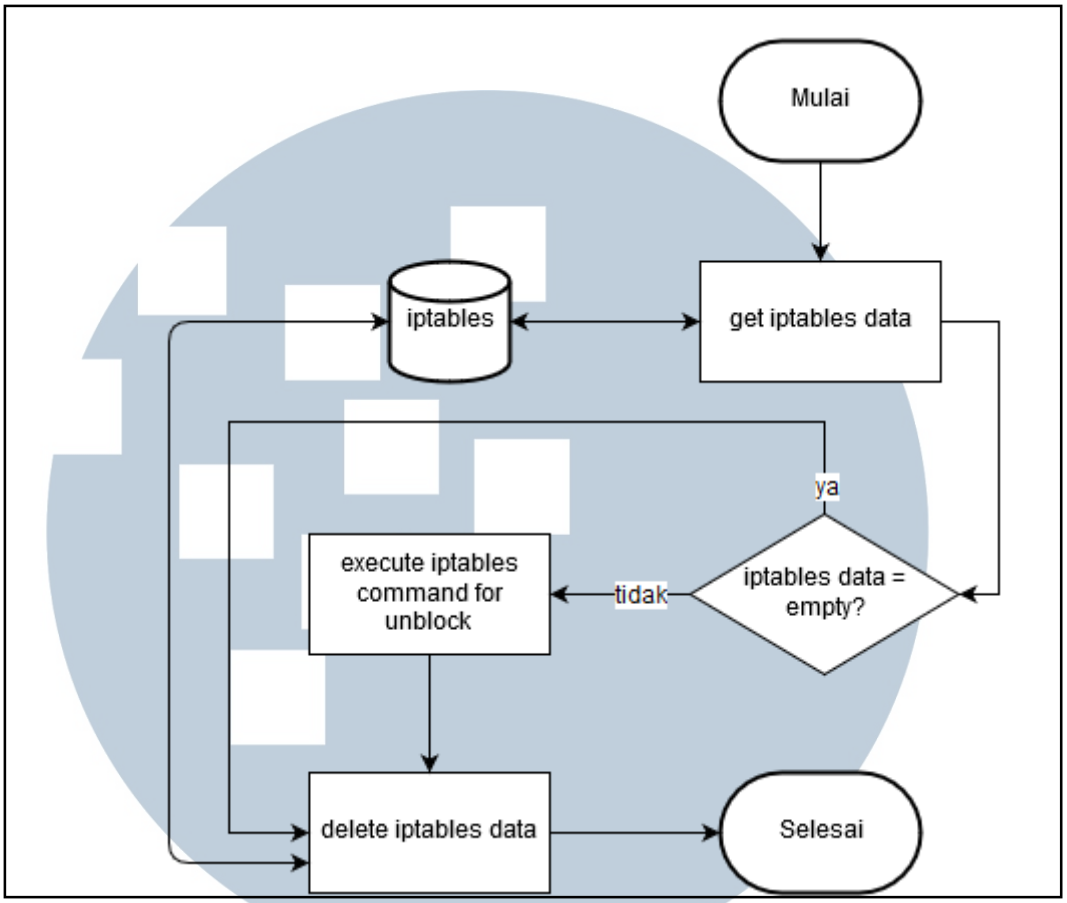
Gambar 3.13 flowchart program pendukung

Program pendukung ditulis dalam bahasa python dan merupakan program yang dijalankan secara *sequencial*. Sehingga terdapat *flowchart* yang dituang dalam gambar 3.13. penjelasan dari *flowchart* yang terdapat pada gambar 3.13 adalah sebagai berikut.

Pertama program akan membuat koneksi ke *database* dengan *connection string library* yang sudah disediakan oleh python. *Database* yang terkoneksi dengan program memiliki nama iptables, yang dimana *database* ini memiliki dua tabel yang memiliki fungsi berbeda. Tabel iptables berfungsi untuk merekam *IP adress* yang masuk ke dalam daftar program iptables, sedangkan iptablesdel berfungsi sebagai daftar *IP address* yang nantinya akan dihapus dari daftar program iptables.

Pada gambar 3.13, program utama akan meminta data dari *database* iptables. Sebelumnya ada suatu proses dalam barnyard2 yang akan mengirimkan data ke database iptables. Setelah program utama mendapatkan data dari database, selanjutnya adalah data tersebut dikirimkan menuju program iptables yang nantinya akan menjadi *rule* untuk *block* koneksi yang masuk melalui IP tersebut.

Program ini juga memiliki *scheduller* yang dimana setiap pukul 23.59 program akan melakukan proses *delete* IP *address* dari list daftar iptables.

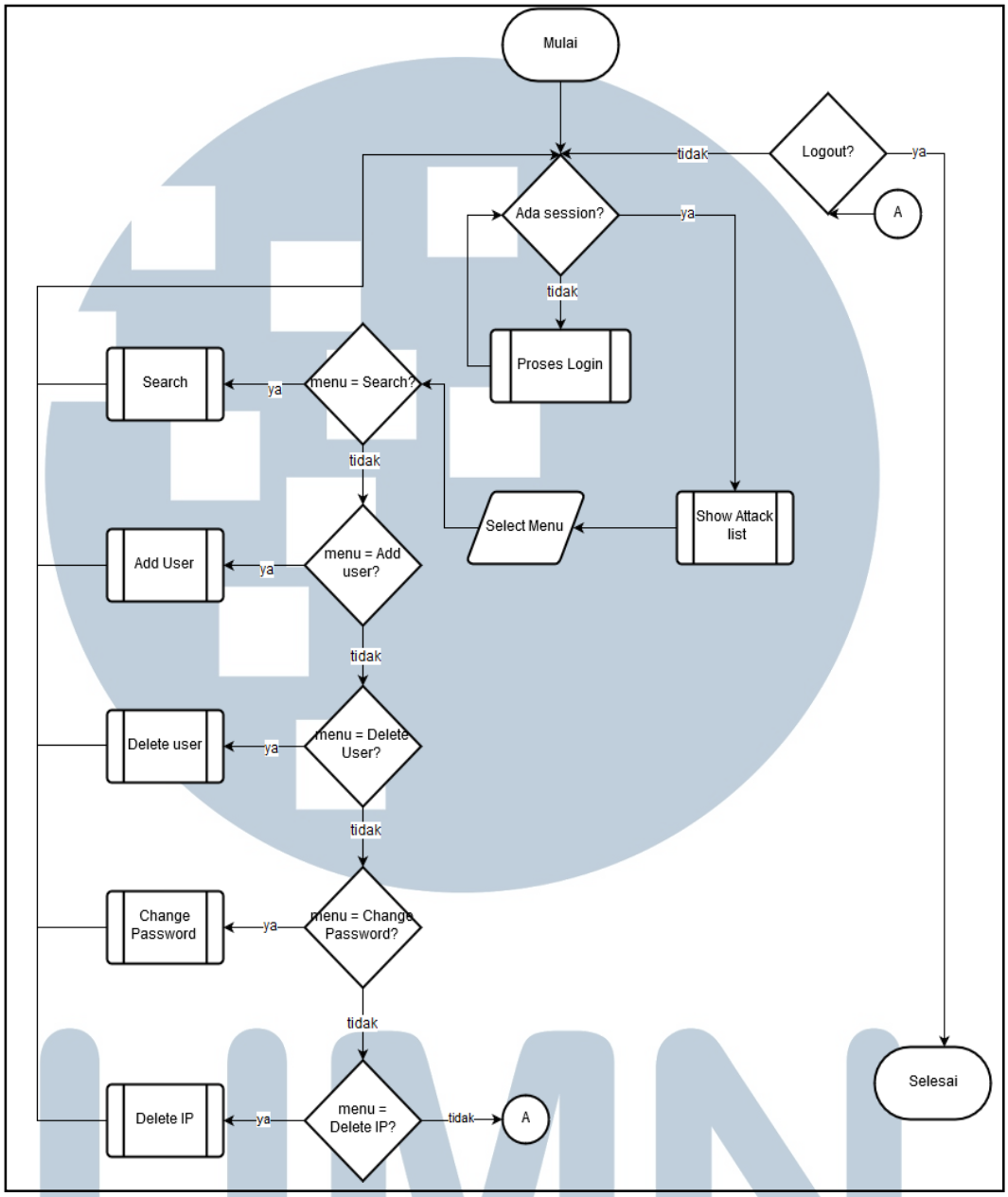


Gambar 3.14 Flowchart Proses Delete IP

Gambar 3.14 merupakan flowchart dari proses *delete* IP. Proses ini sendiri bertujuan untuk menghapus IP address dari daftar list program iptables, yang nantinya dari sebelumnya tidak memiliki akses menjadi mendapatkan akses kembali.

3.3.6 Perancangan Web Interface

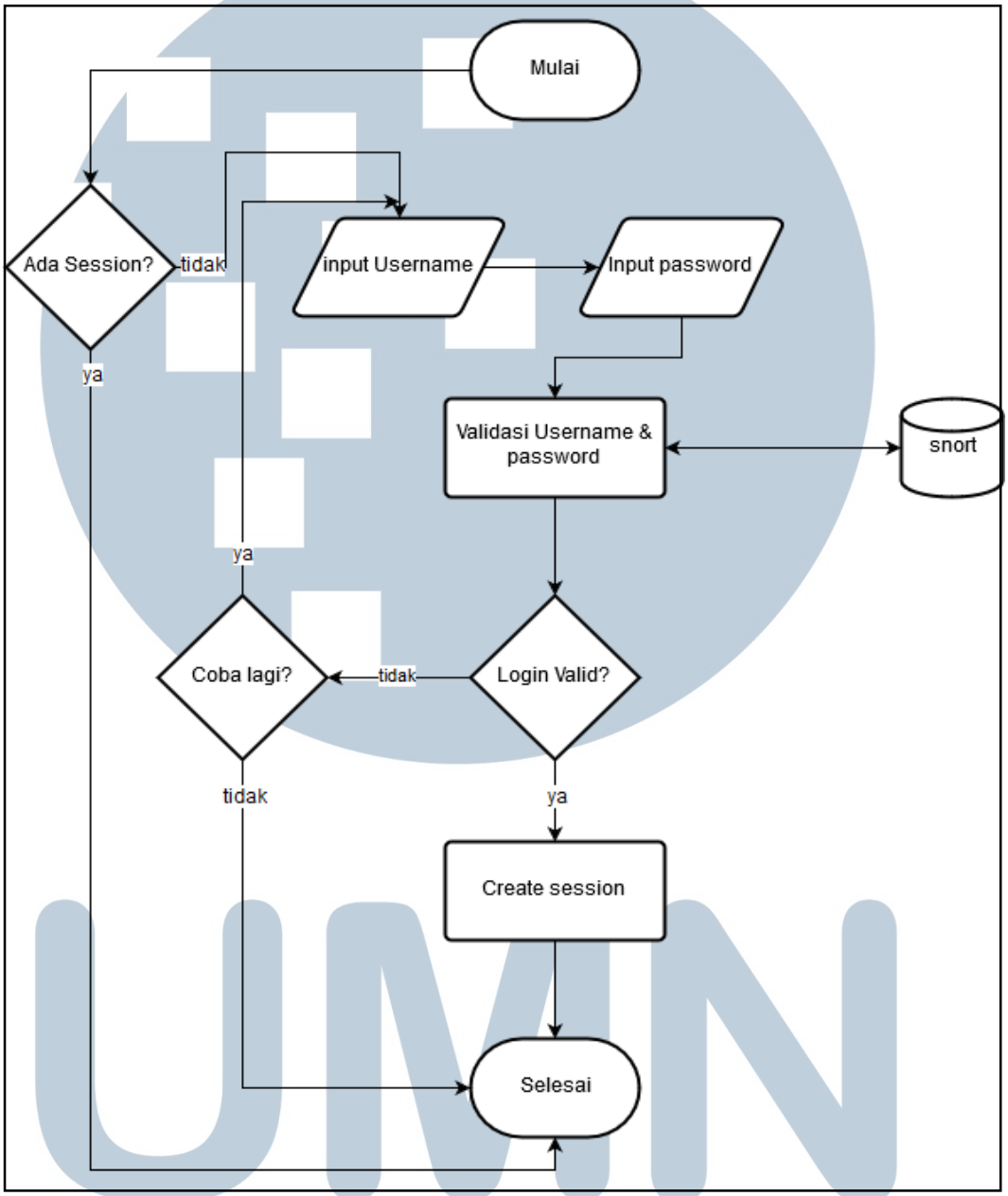
Web interface bertujuan untuk mempermudah admin untuk membaca *log* yang dikeluarkan snort. Pada awalnya log snort hanya dapat dibaca dengan bantuan program lain seperti program pcap, akan tetapi dengan adanya bantuan barnyard2, *log* yang dikeluarkan snort dapat disimpan di *database* yang nantinya datanya akan lebih mudah ditampilkan dan lebih mudah untuk dibaca.



Gambar 3.15 Flowchart Home

Flowchart home menggambarkan alur program beserta beberapa prosesnya. Pada *flowchart* ini program akan melakukan cek terhadap *session*. Jika tidak ada *session* maka akan dilanjutkan ke proses *login*, dan jika ada *session* maka berlanjut ke proses *show attack list*. *User* juga dapat melakukan *input menu*, dan setiap *menu* memiliki proses sendiri. Terdapat beberapa *menu* yang terdiri dari *menu search* yang memiliki proses *search, add user* memiliki proses *add user, delete user* yang

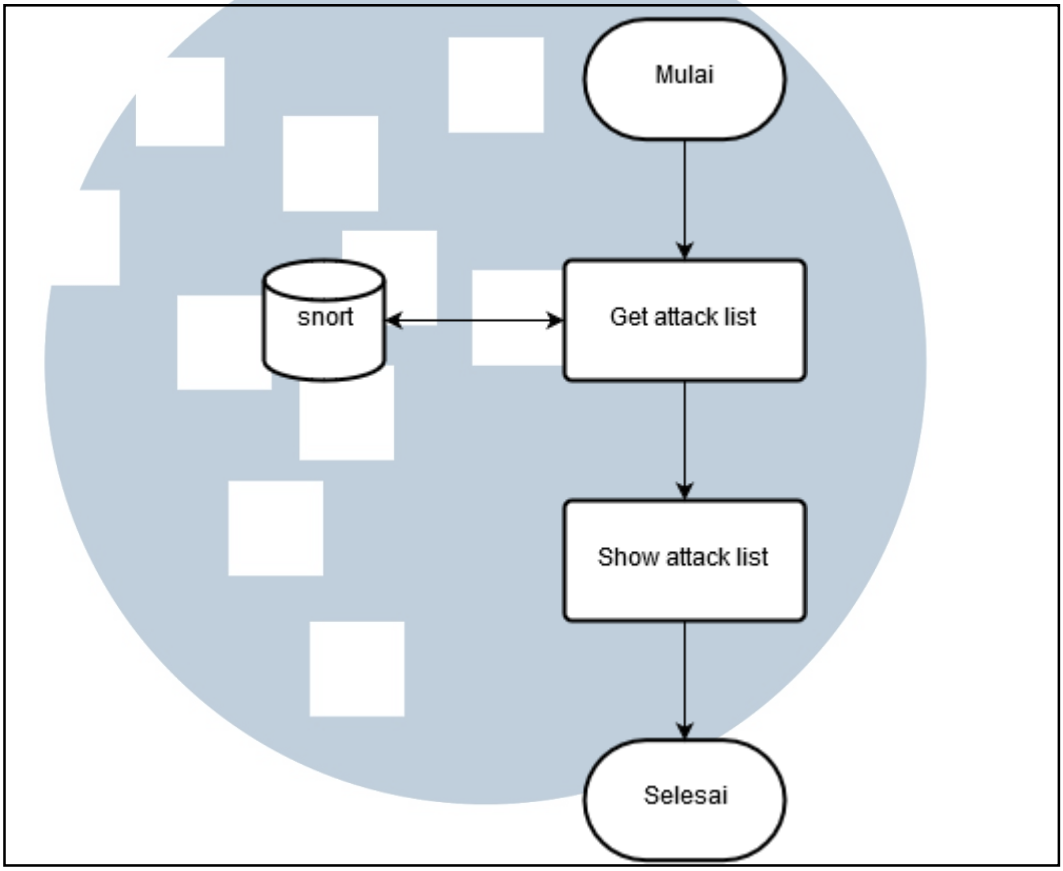
memiliki proses *delete user*, *change password* yang memiliki proses *change password*, dan terakhir menu *delete* IP yang memiliki proses *delete* IP.



Gambar 3.16 Flowchart Login

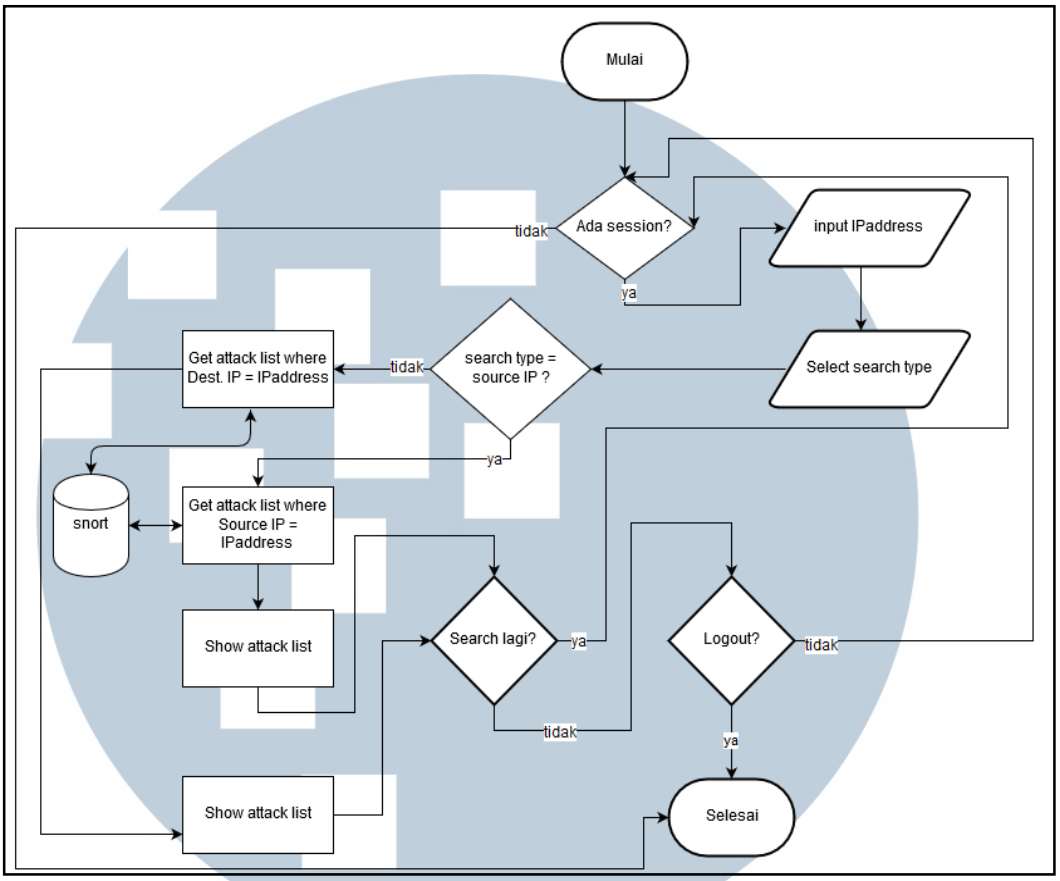
Dalam gambar 3.16 program akan melakukan pengecekan *session*. Jika terdapat *session* maka proses akan berakhir, dan jika tidak terdapat *session* maka selanjutnya *user* mengisi *username* dan *password*. Setelah *user* mengisi *username* dan *password* maka ada proses validasi *login*. Jika validasi salah maka *user* harus

mencoba mengisi *username* dan *password* kembali, dan jika *validasi* benar maka proses selanjutnya adalah *user* akan mendapatkan *session*.



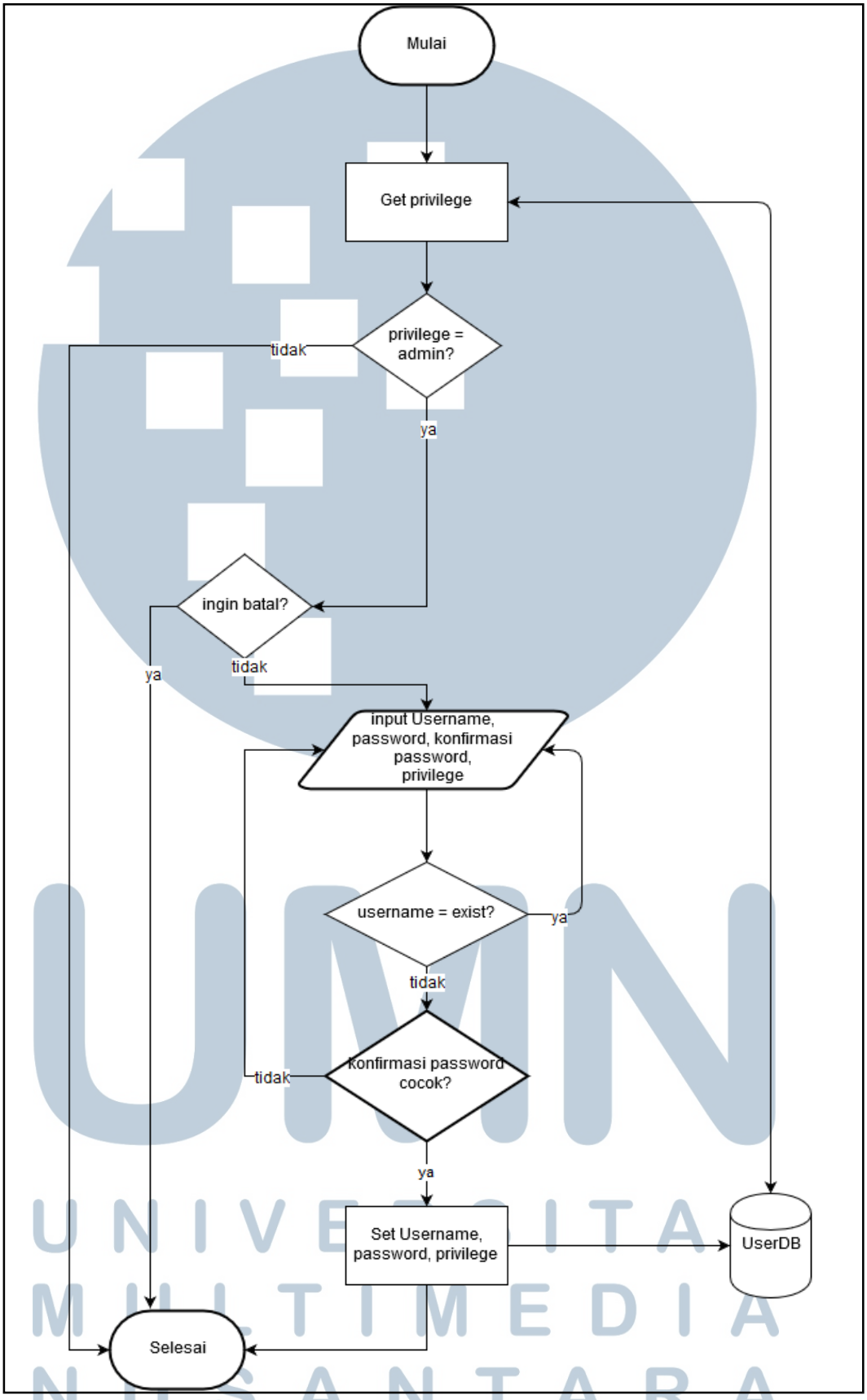
Gambar 3.17 Flowchart Show Attack List

Flowchart pada gambar 3.17 berfungsi sebagai menampilkan *attack list* yang terekam oleh *database*.



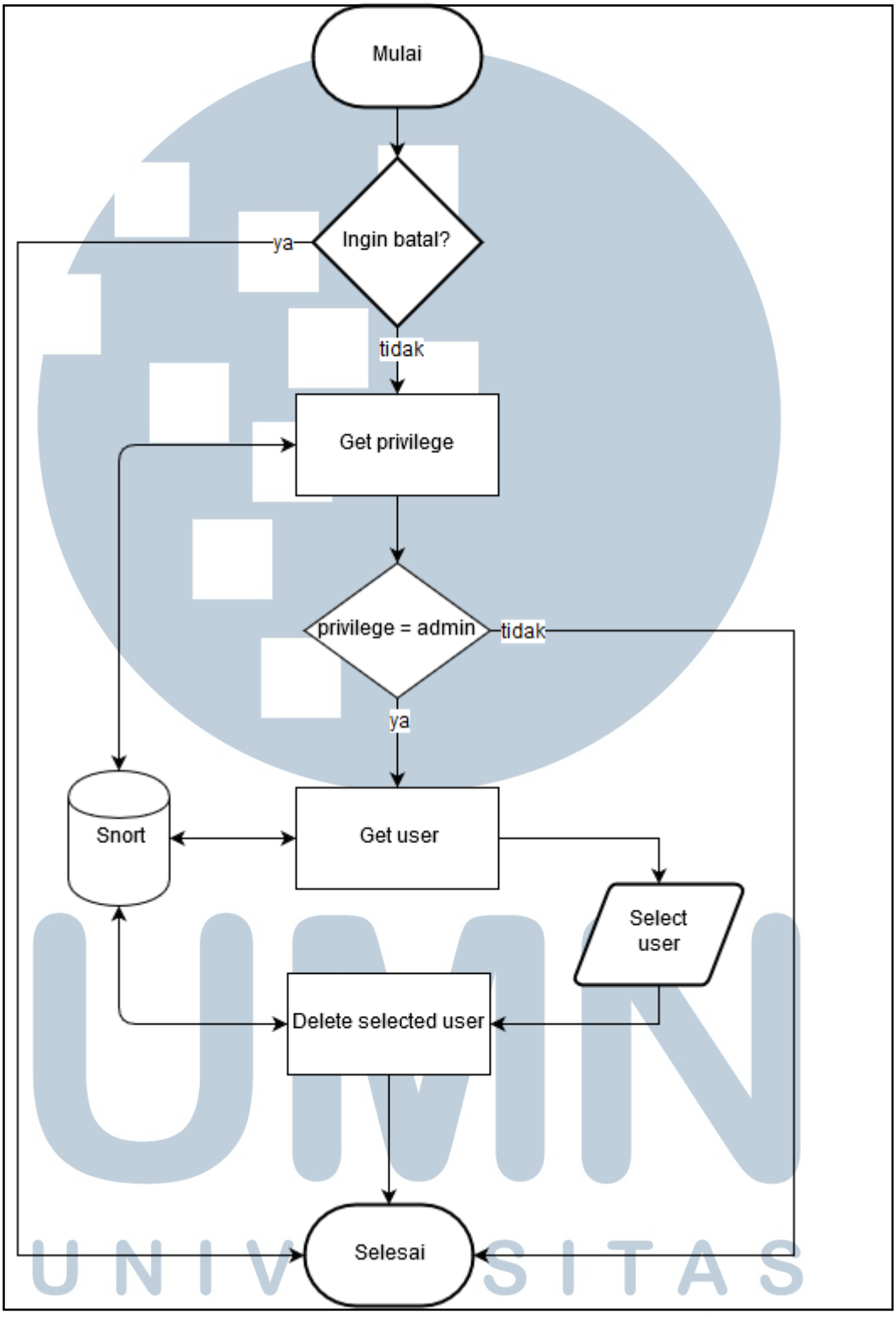
Gambar 3.18 Flowchart Search

Proses *search* berfungsi sebagai jika ingin mencari sebuah data yang dapat berdasarkan *source* IP dan *destination* IP. Flowchart pada gambar 3.18 merupakan *flowchart* untuk proses *search*.



Gambar 3.19 Flowchart Add User

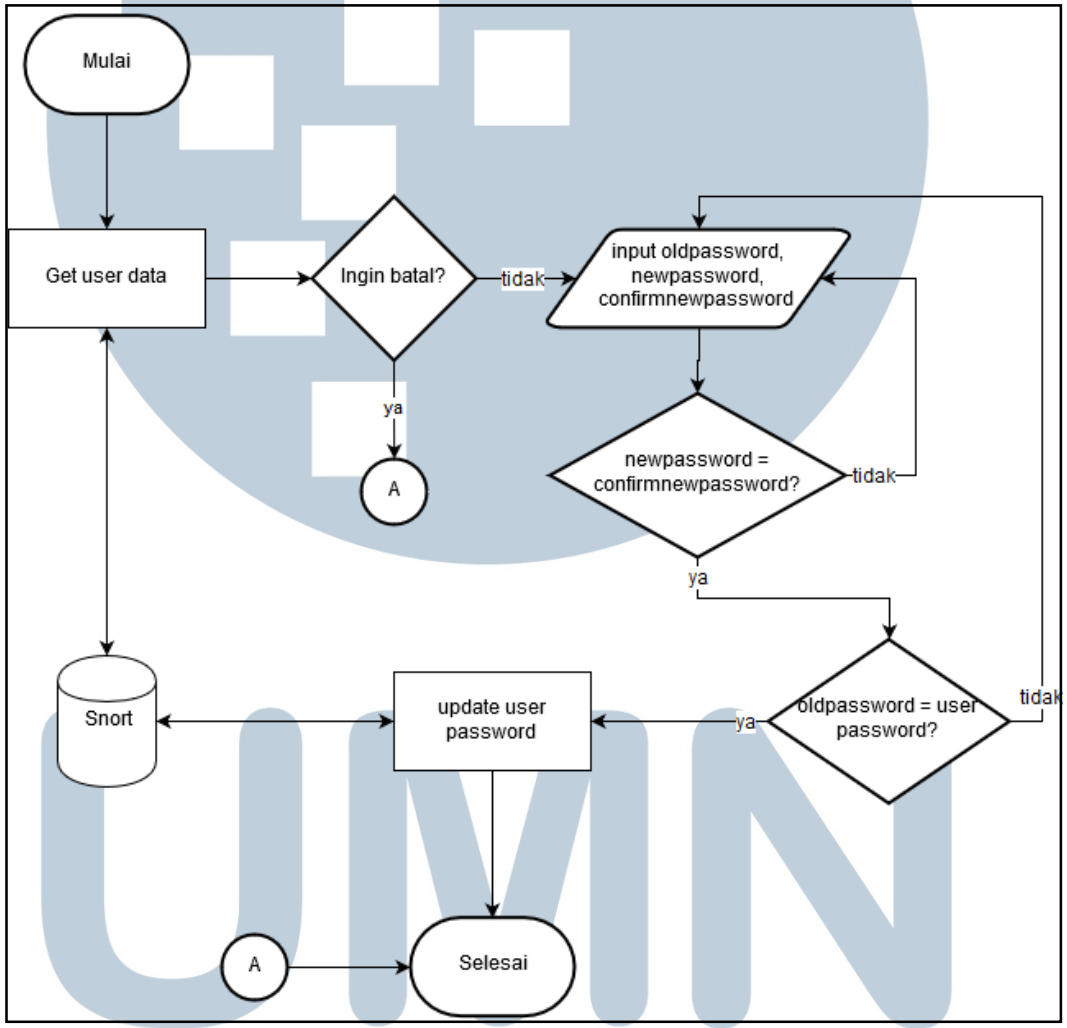
Flowchart add user bertujuan untuk menambah user baru yang dilakukan oleh *administrator*. Pertama proses akan melakukan cek *privelege.* Jika *privilege* bukan admin maka proses akan selesai, dan jika iya maka proses akan dilanjutkan dengan *user* akan memasukkan *username, password,* dan *privilege*. Selanjutnya akan dilakukan cek pada ketersediaan *username.* Jika *username* tidak tersedia maka *user* harus melakukan input *username, password,* dan *privilege* kembali, dan jika tersedia maka ketiga data tersebut akan dimasukkan ke database. Dan jika user ingin batal maka proses akan selesai.



Gambar 3.20 Flowchart Delete User

Flowchart delete user merupakan proses yang bertujuan untuk menghapus *user* yang sudah tidak diperlukan lagi. Proses akan melakukan cek *privilege*. Jika

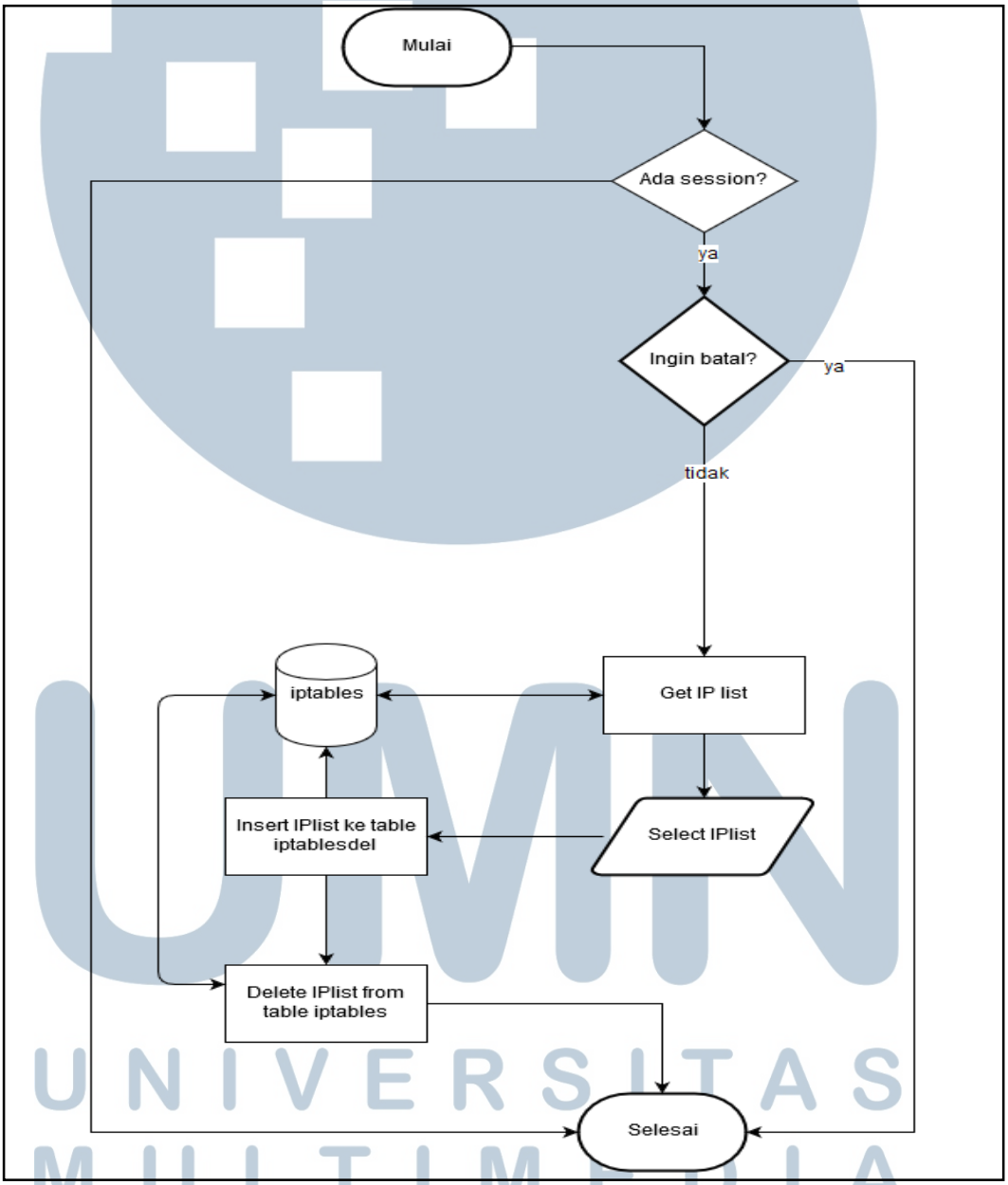
privelege bukan admin maka proses akan selesai dan jika iya maka proses selanjutnya adalah dengan mendapatkan *list user* dari *database.* Kemudian *user* akan memilih *user* mana yang ingin dihapus. Setelah *user* melakukan penghapusan user maka proses juga akan menghapus data *user* dari *database* dan proses akan selesai.



Gambar 3.21 Flowchart Proses Password

Proses *update password* bertujuan untuk melakukan *update password* yang dimiliki *user*. Pada proses ini pertama akan dilakukan pengambilan *data user* dari *database*. Jika *user* ingin membatalkan proses ini maka proses ini akan selesai. Jika tidak batal maka *user* akan disuruh memasukan *password* baru dan akan dicocokan dengan

password lama yang tadi telah diambil dari *database*. Sebelumnya proses akan melakukan validasi konfirmasi *password* baru. Jika cocok maka proses akan dilanjutkan ke validasi *password* lama dan *password* baru Jika cocok maka *password* akan diperbarui, jika tidak maka, user akan disuruh memasukkan kembali *password* lama.



Gambar 3.22 Flowchart Delete Iptables

Pada proses *delete* iptables pertama akan dilakukan cek *session*. Jika terdapat *session* proses dapat dilanjutkan, jika tidak ada maka proses akan selesai. Jika terdapat *session* dan *user* ingin batal maka proses juga akan selesai. Pada proses ini akan dilakukan pengambilan data IP *address* yang masuk ke dalam daftar *block* program iptables dari *database*. Setelah itu user akan memilih IP mana yang ingin di-*unblock*. Setelah *user* memilih IP selanjutnya IP tersebut akan dimasukkan ke *database list* IP *address* yang akan dibuka kembali aksesnya. Pembukaan akses dilakukan oleh program pendukung.

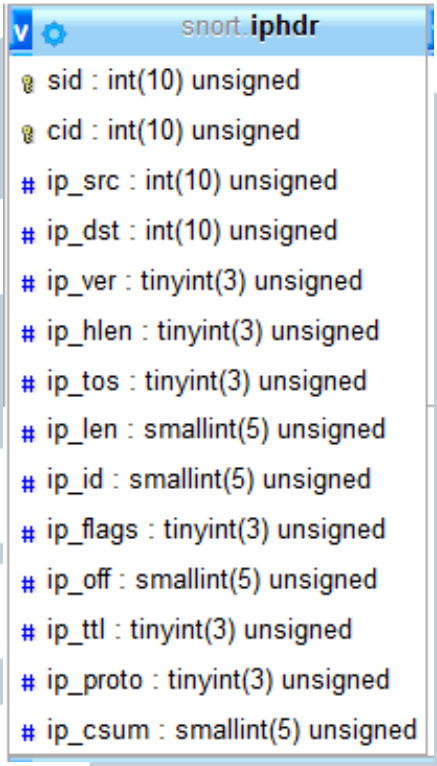
3.3.7 Penggunaan Database

Pada dasarnya skema *database* telah disediakan oleh program barnyard2 sendiri. Sehingga hanya melakukan *import* tabel ke dalam *database* yang telah dibuat. Akan tetapi untuk mendukung dan memaksimalkan kerja program pendukung maka dilakukan sedikit perubahan pada struktur tabel. Tidak semua tabel pada *database* digunakan. Berikut *database* beserta tabel yang digunakan.

Ada dua *database* yang dibuat dalam perancangan sistem ini. Pertama adalah *database* snort yang berfungsi hanya untuk menampilkan data, dan yang terakhir adalah *database* iptables yang hanya berfungsi untuk keperluan program pendukung dan program iptables.

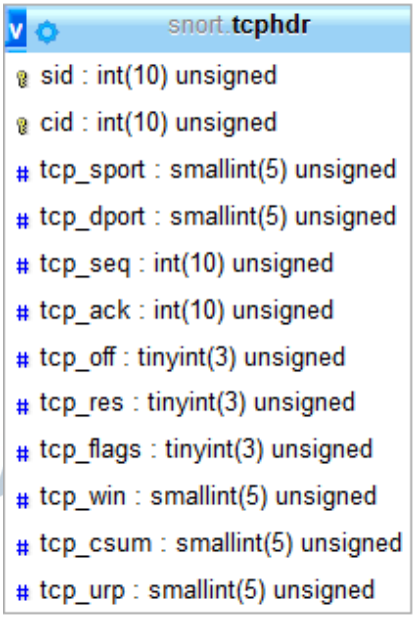
Berikut merupakan *list* tabel yang digunakan pada *database* snort.

Tabel 3.1 Struktur Tabel iphdr



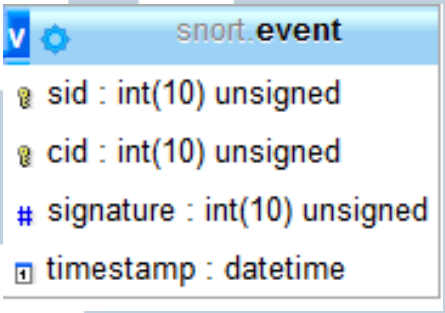
Tabel iphdr berfungsi untuk menyimpan data tentang IP *address* dan *header*-nya terutama untuk menyimpan data IP asal dan IP tujuan.

Tabel 3.2 Struktur Tabel tcphdr



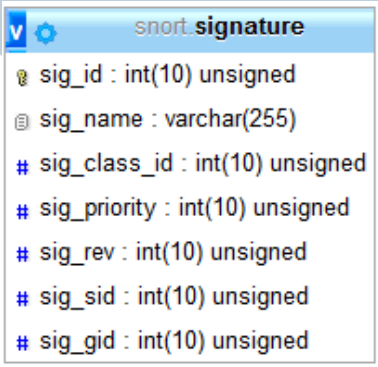
Tabel tcphdr berfungsi untuk menyimpan data berupa *header* dari protokol TCP. Tabel ini digunakan terutama untuk menyimpan dan mengambil data tentang *port* tujuan dari paket

Tabel 3.3 Struktur Tabel event



Tabel event hanya berfungsi untuk menangkap dan membaca *timestamp* dari serangan dan signature pada tabel event berguna untuk menentukan jenis serangan yang nantinya akan dibandingkan dengan tabel signature.

Tabel 3.4 Struktur Tabel event



Tabel signature berfungsi untuk mendefinisikan jenis serangan yang terjadi, dengan data kode signature yang berasal dari tabel event.

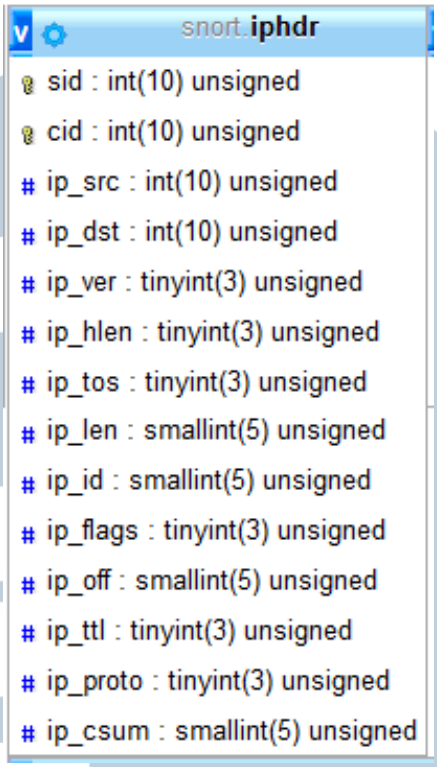
Tabel 3.5 Struktur tabel logins

No.	Nama Kolom	Tipe Data	Keterangan
1.	Usernama	VARCHAR(150)	Untuk menampung username dari pengguna
2.	Passkata	VARCHAR(150)	Untuk menampung password dari pengguna
3.	Privilege	VARCHAR(10)	Untuk menentukan kedudukan pengguna

Tabel logins berfungsi untuk menyimpan data user. Struktur tabel terdiri dari usernama yang berfungsi untuk menyimpan *username* dengan format data VARCHAR dengan panjang 150 karakter, passkata yang berfungsi untuk menyimpan *password* dari pengguna dengan format data VARCHAR dengan panjang 150 karakter, dan privilege yang berfungsi untuk menentukan kedudukan dari pengguna dengan tipe data VARCHAR dengan panjang 10 karakter.

Kemudian berikut *list* tabel yang digunakan di dalam database iptables.

Tabel 3.6 Struktur Tabel iphdr



Tabel iphdr pada database ini sama fungsinya dengan tabel iphdr pada database snort dan memiliki struktur yang sama.

Tabel 3.7 Struktur tabel iptables

No.	Nama Kolom	Tipe Data	Keterangan
1.	Ip	VARCHAR(30)	Untuk menyimpan IP *address* yang masuk ke daftar blok program iptables

Tabel iptables merupakan tabel yang berfungsi untuk menyimpan data daftar IP address berapa saja yang sudah di blok oleh program iptables.

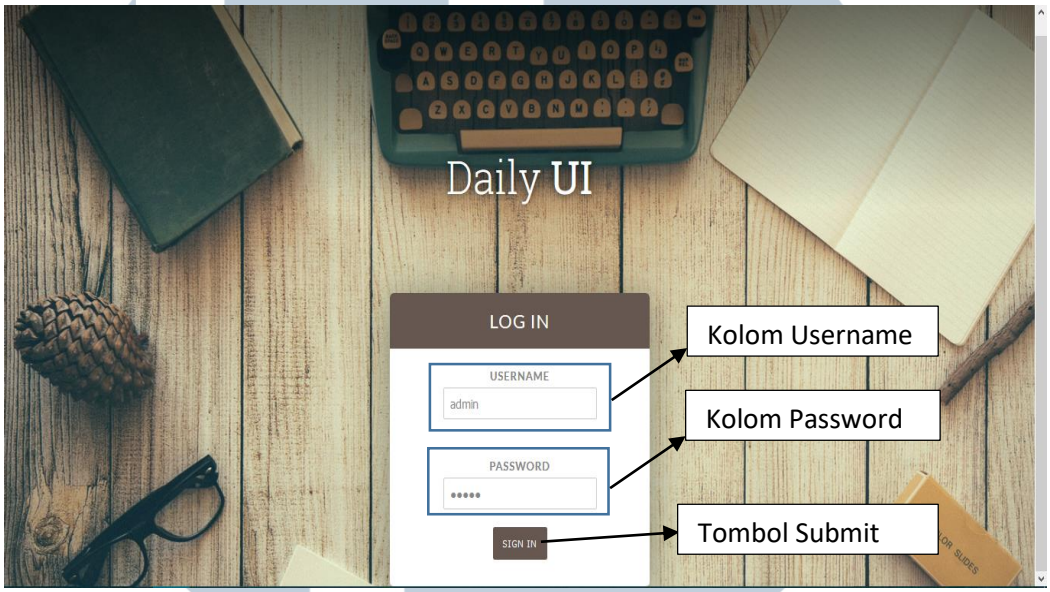
Tabel 3.8 Struktur tabel iptablesdel

No.	Nama Kolom	Tipe Data	Keterangan
1.	Ip	VARCHAR(30)	menyimpan *list* daftar IP *address* mana saja yang ingin dibukakan kembali aksesnya yang sebelumnya terblok oleh program iptables.

Tabel iptablesdel memiliki struktur yang sama tetapi memiliki fungsi yang berbeda. Fungsi dari tabel ini adalah untuk menyimpan *list* daftar IP *address* mana

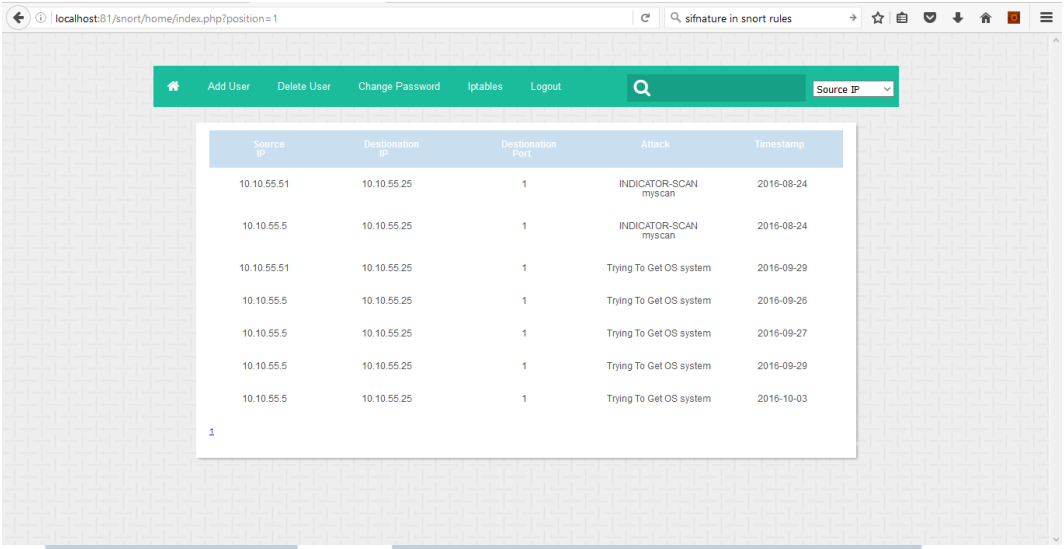
saja yang ingin dibukakan kembali aksesnya yang sebelumnya terblok oleh program iptables.

3.3.8 User Interface Web Snort



Gambar 3.23 Tampilan login

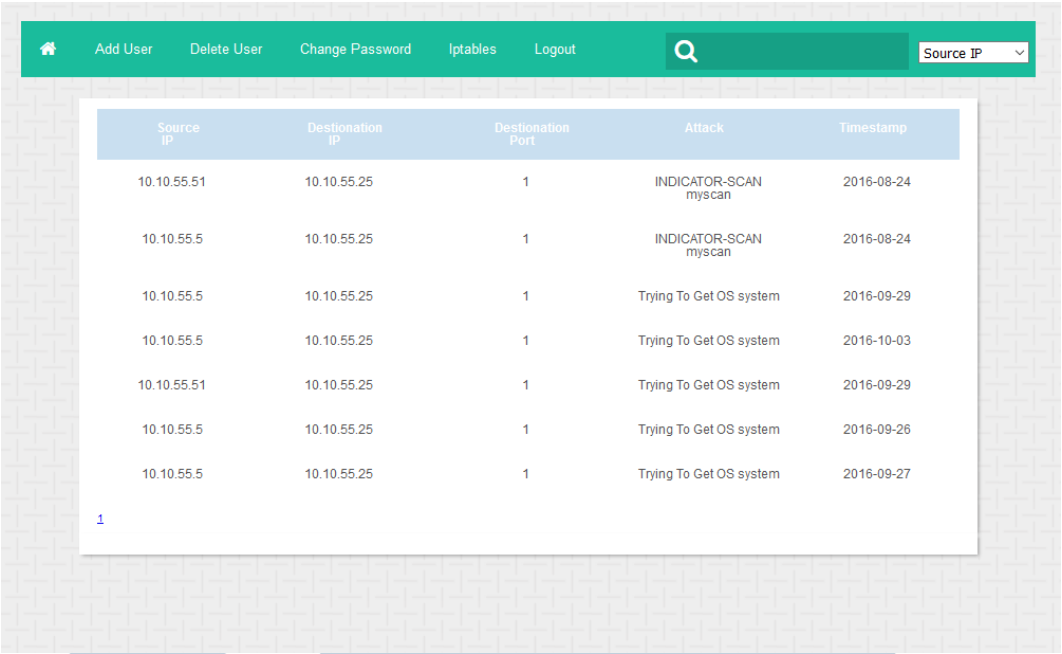
Pada Gambar 3.23 terdapat halaman *login* yang merupakan antar muka pertama jika pengguna ingin masuk kedalam sistem. Pada halaman ini terdapat dua kolom, yang pertama adalah kolom *username* dan yang kedua adalah kolom *password* dan tombol *submit* untuk melakukan *login* jika *username* dan *password* benar.



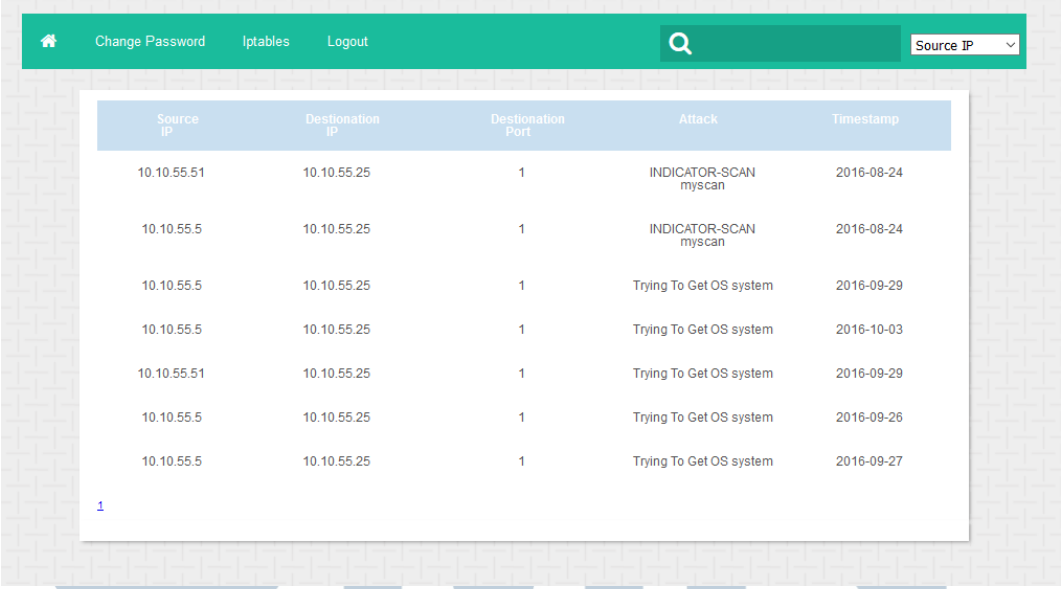
Gambar 3.24 Tampilan Home Web

Pada halaman *home*, terdapat list *history* dari jenis serangan yang perdah terjadi. List sendiri terdiri dari *Source IP* yang merupakan alamat IP penyerang, *Destination IP* adalah tujuan alamat IP korban, *Destination Port* adalah *port* yang diserang oleh penyerang, *Attack* merupakan jenis serangan yang dilakukan oleh penyerang, dan terakhir adalah *Timestamp* yang merupakan kapan terjadinya serangan.

Pada bagian atas terdapat *navigation bar* dan *search bar*. Fungsi dari *navigation bar* adalah merupakan navigasi kita jika kita ingin melakukan kegiatan lain seperti menambah *user* maupun mengganti *password*. Terdapat perbedaan antara halaman *home* dari *user* biasa dengan *administrator*. *Navigation bar* terdiri dari gambar rumah yang merupakan navigasi menuju halaman utama, *delete user* untuk menghapus pengguna, *add user* untuk menambah pengguna, *change password* untuk merubah kata sandi, iptables merupakan fitur untuk membuka kembali akses IP yang terkena blok oleh program iptables, dan yang terakhir adalah *logout* yang berfungsi untuk keluar dari sistem.



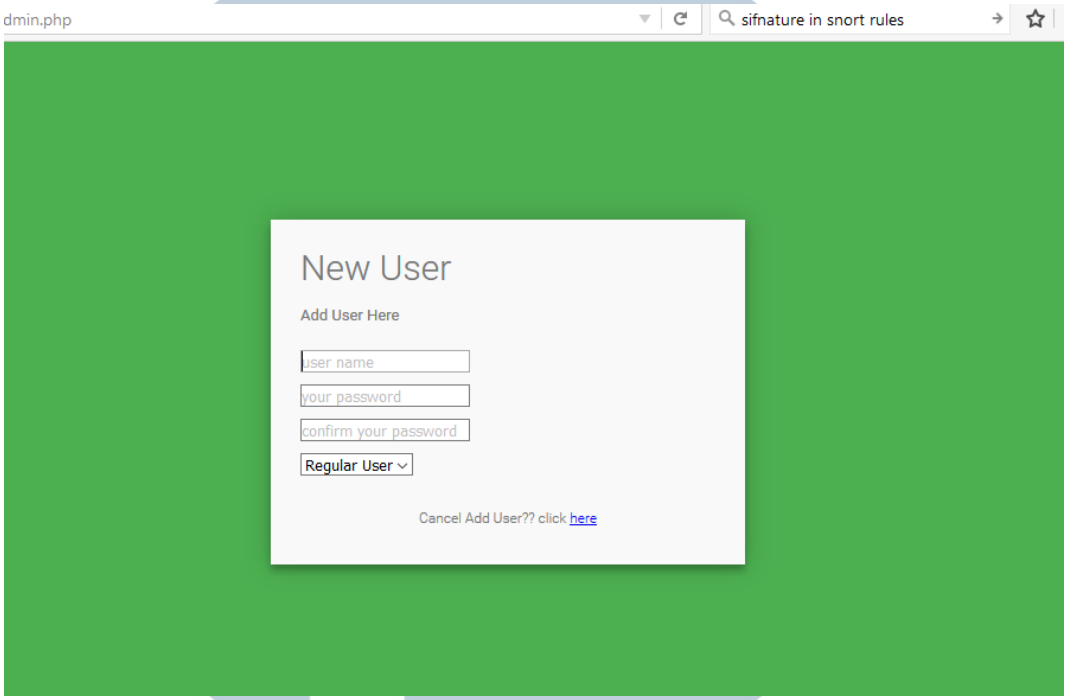
Gambar 3.25 Navigation Bar Administrator



Gambar 3.26 Navigation Bar User Biasa

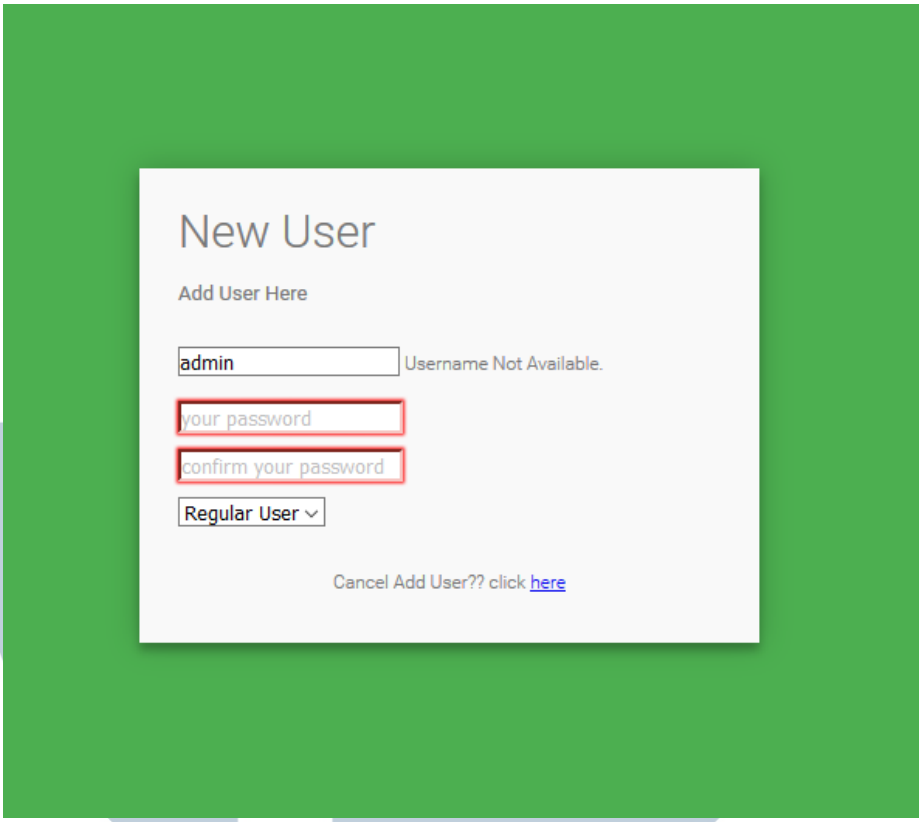
Terlihat pada gambar 3.26 *navigation bar* pada admin memiliki kelebihan yaitu, dapat menambah pengguna, dan dapat menghapus pengguna. Sedangkan *user* biasa hanya terdapat *change password* dan iptables.

Dan terakhir pada halaman utama adalah *search bar* yang berfungsi untuk mencari alamat IP, apakah IP tujuan atau IP sumber.

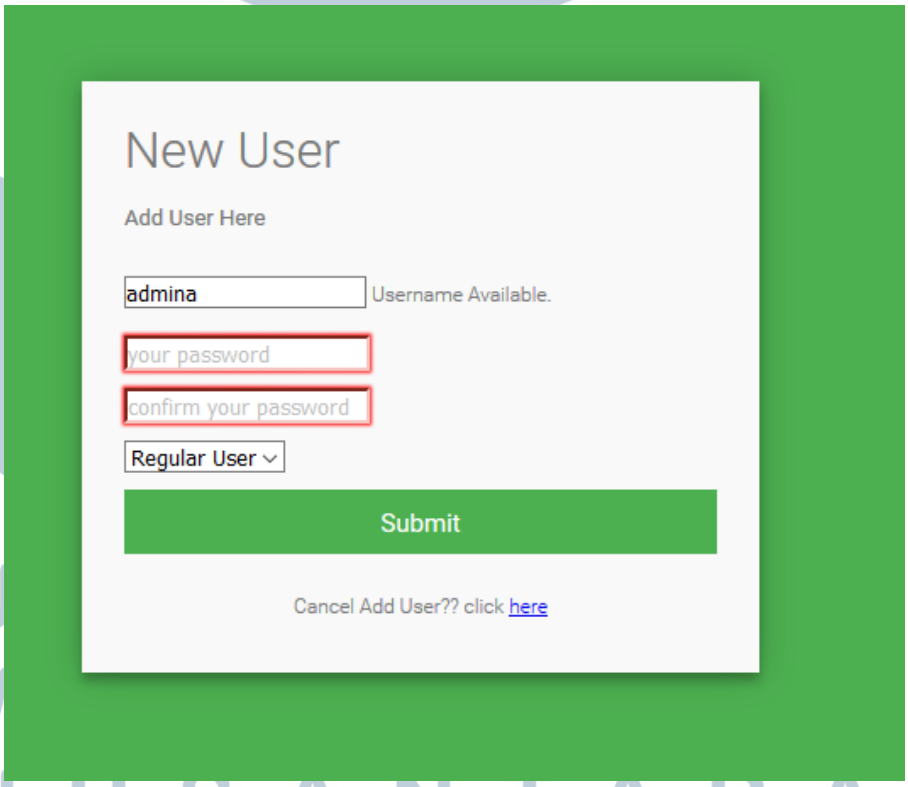


Gambar 3.27 Halaman Add User

Halaman ini bertujuan untuk menambah pengguna. Terdapat tiga kolom *text box* dan satu kolom *dropdown list*. Untuk menambah pengguna pertama isi *username* yang diinginkan pada kolom *username*, pastikan *username* masih ada jika *username* sudah digunakan maka akan muncul error "*Username Not Available*" di sebelah kolom *username* seperti gambar 3.27 di bawah, dan jika benar maka akan keluar tulisan *username available* disamping kolom *username* seperti gambar 3.28 di bawah. Kolom *password*, dan *confirm password* harus diisi dan jika tidak diisi maka kotak akan diberi garis berwarna merah pada *text box*.

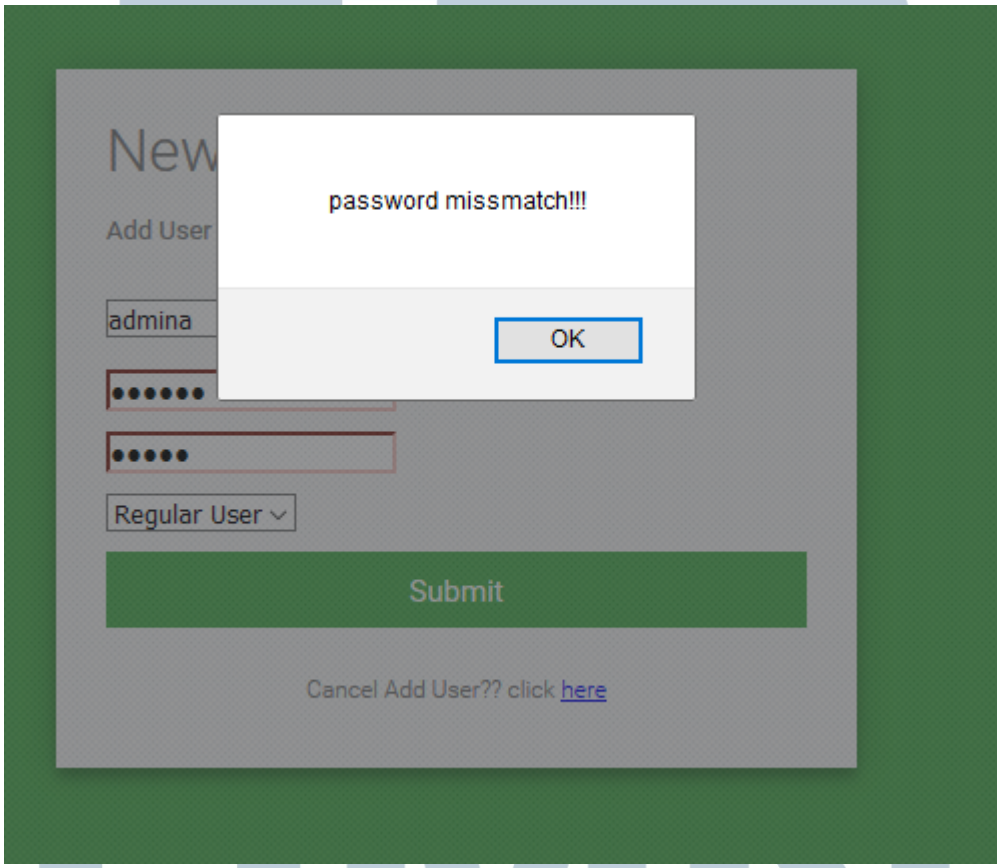


Gambar 3.28 Username Not Available

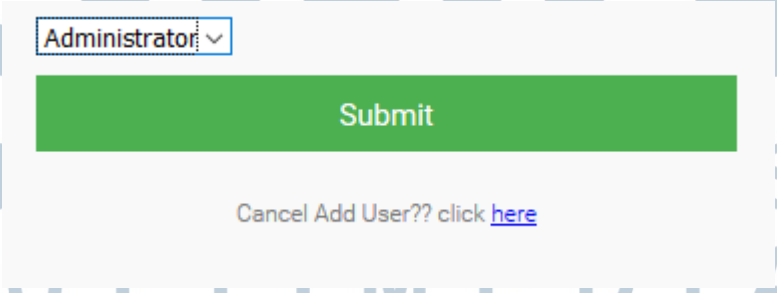


Gambar 3.29 Username Available

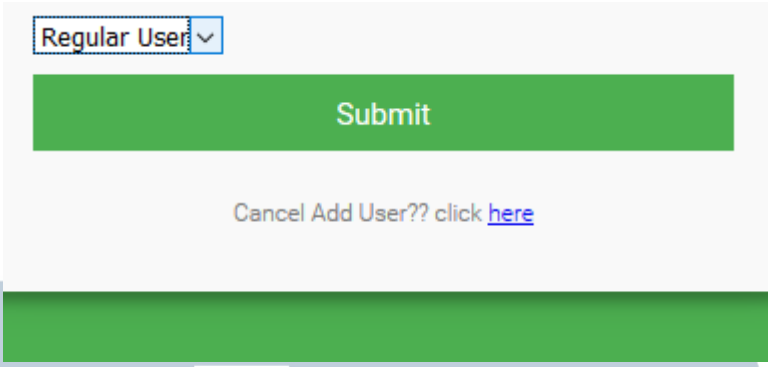
Pada halaman ini juga ada fungsi pengecekan apakah *password* yang diketikkan sudah sama dengan *confirm password* atau tidak. Jika tidak maka akan muncul pesan *pop up* yang berisi "*password missmatch*". Serta yang terakhir pada halaman ini adalah admin dapat memilih membuat *user* memiliki *privilege* admin atau *user* biasa.



Gambar 3.30 Password Tidak Sama

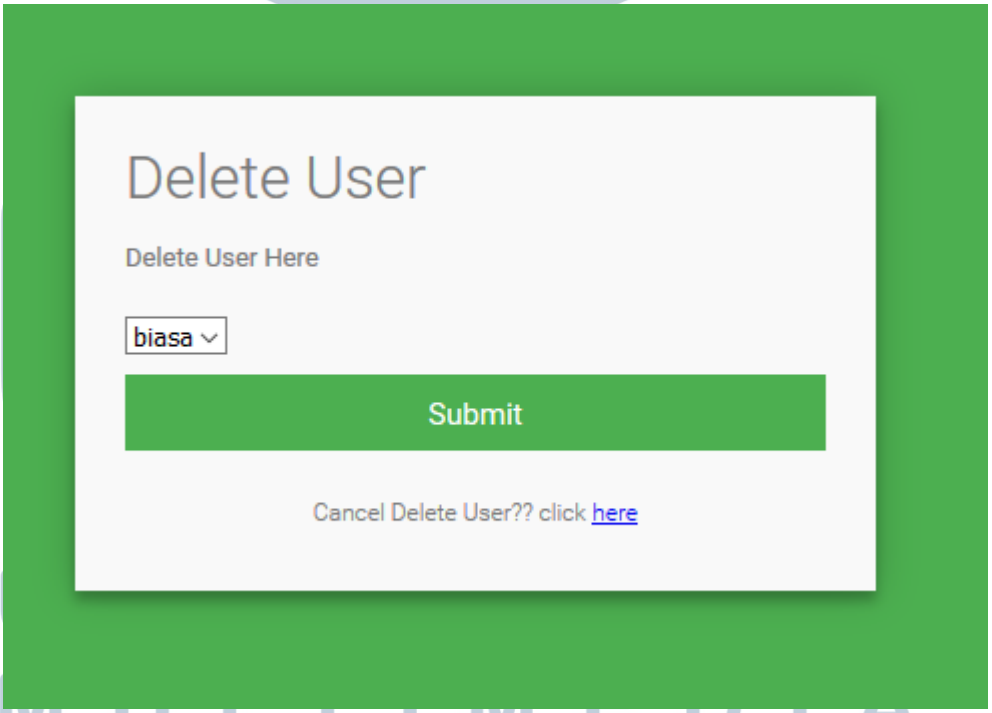


Gambar 3.31 User Administrator

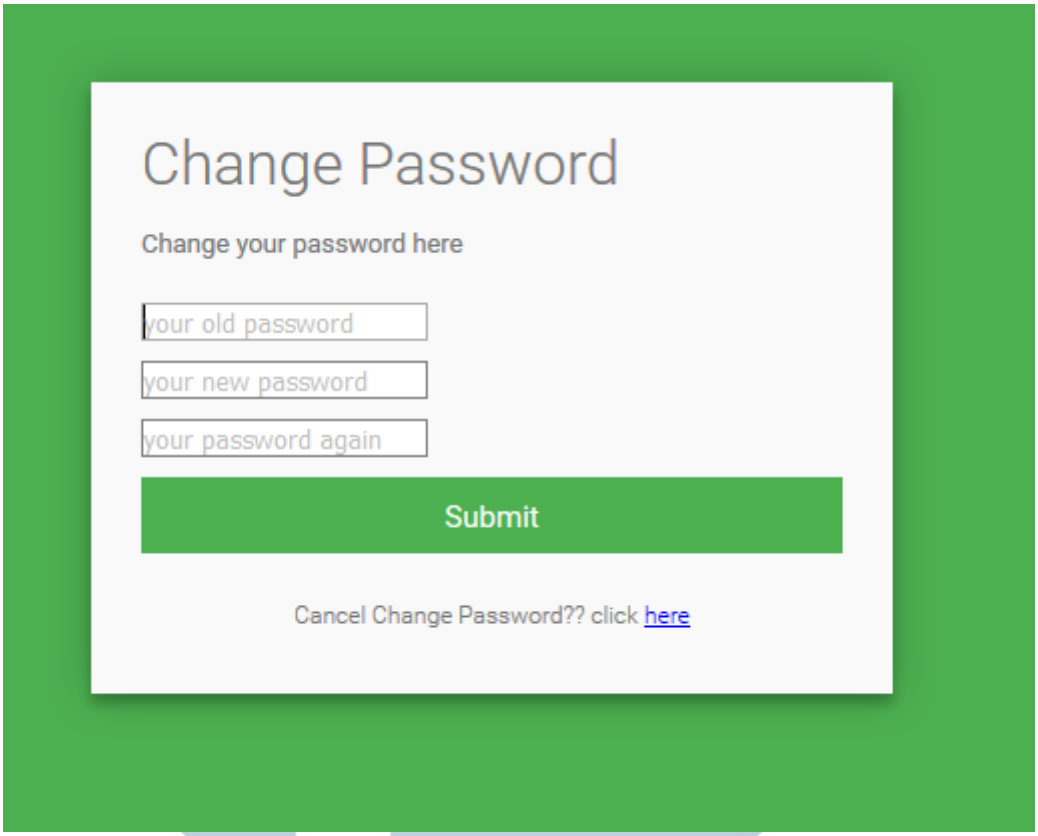


Gambar 3.32 User Biasa

Halaman selanjutnya adalah halaman *Delete User* yang memiliki fungsi sebagai menghapus *user* jika sudah tidak aktif lagi. Halaman ini hanya dapat dibuka oleh *user* yang memiliki *privilege administrator*. Dalam halaman ini terdapat list *user* yang sudah terdaftar dan jika *user* tersebut terpilih dan jika ditekan tombol *submit* maka, *user* teresebut akan terhapus. Contoh halaman *delete user* seperti seperti gambar 3.33

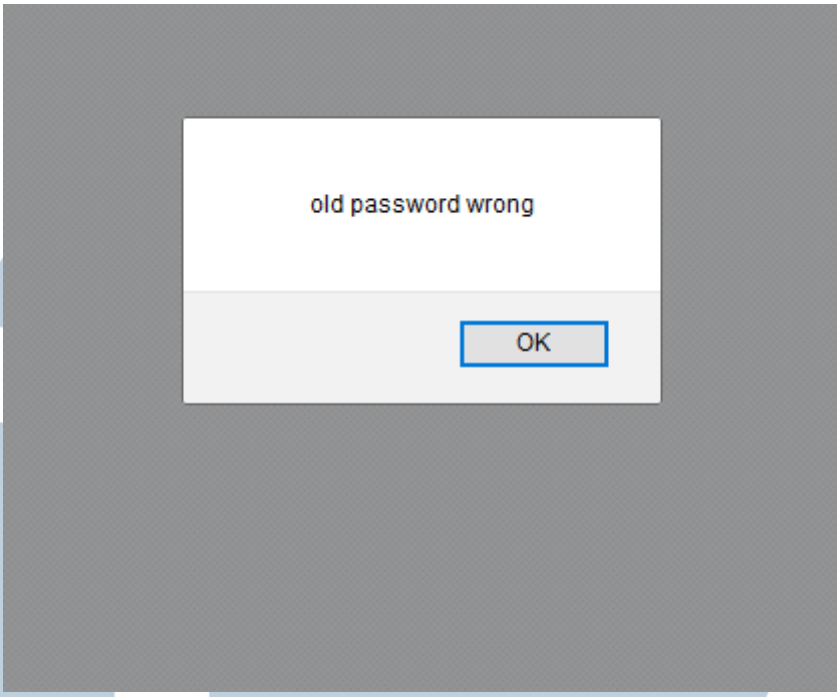


Gambar 3.33 Halaman Delete User



Gambar 3.34 Halaman Change Password

Pada gambar 3.34 terdapat suatu halaman yang bernama halaman *change password*. Halaman ini dapat diakses oleh semua *user* yang berfungsi untuk merubah *password user*. Pada halaman ini terdapat tiga buah *text* box yang masing-masing berfungsi sebagai validasi *password* lama, *input password* baru, dan validasi input *password* baru. Validasi *password* lama, harus sama dengan *password* kita sebelumnya. Jika *password* lama salah maka akan keluar pesan seperti gambar 3.35.



Gambar 3.35 Pesan Password Lama Salah

Gambar 3.36 Halaman Iptables

Pada gambar 3.36 merupakan halaman iptables yang bertujuan untuk menghapus secara manual IP address yang telah terdeteksi sebagai serangan atau *threat* dan telah masuk kedalam iptables / daftar IP yang *diblock*. Pada halaman ini terdapat *drop down list* yang berisi alamat IP yang masuk ke daftar IP yang *diblok*

dan tidak dapat mengakses *server*. Untuk membuka kembali IP yang diblok dengan cara memilih IP dari *drop down list*, kemudian klik *submit*, dan IP tersebut telah kembali mendapatkan akses ke *server*.

3.3.9 Testing Sistem Keamanan Jaringan Tambahan

```
root@ubuntu:/etc/snort# ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:49:e5:d8
          inet addr:192.168.1.107  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe49:e5d8/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:31959 errors:0 dropped:0 overruns:0 frame:0
          TX packets:14590 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:42739617 (42.7 MB)  TX bytes:6334132 (6.3 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:160 errors:0 dropped:0 overruns:0 frame:0
          TX packets:160 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:11840 (11.8 KB)  TX bytes:11840 (11.8 KB)
```

Gambar 3.37 IP Address Server

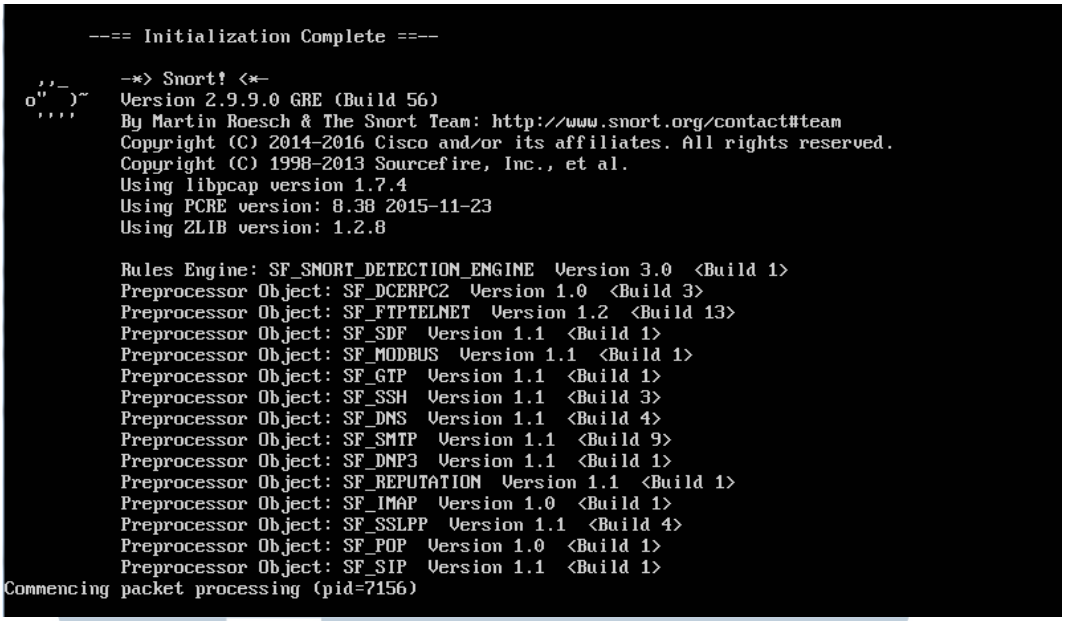
```
Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::6447:83a5:bae5:36a7%22
   IPv4 Address. . . . . . . . . . . : 192.168.1.104
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1
```

Gambar 3.38 IP Address Penyerang

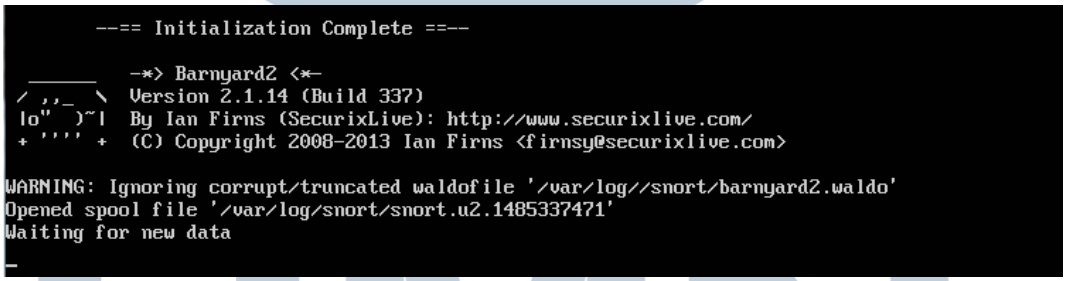
```
root@ubuntu:/etc/snort# snort -i ens33 -c /etc/snort/snort.conf -l /var/log/snort/
```

Gambar 3.39 Perintah Untuk Menjalankan Snort



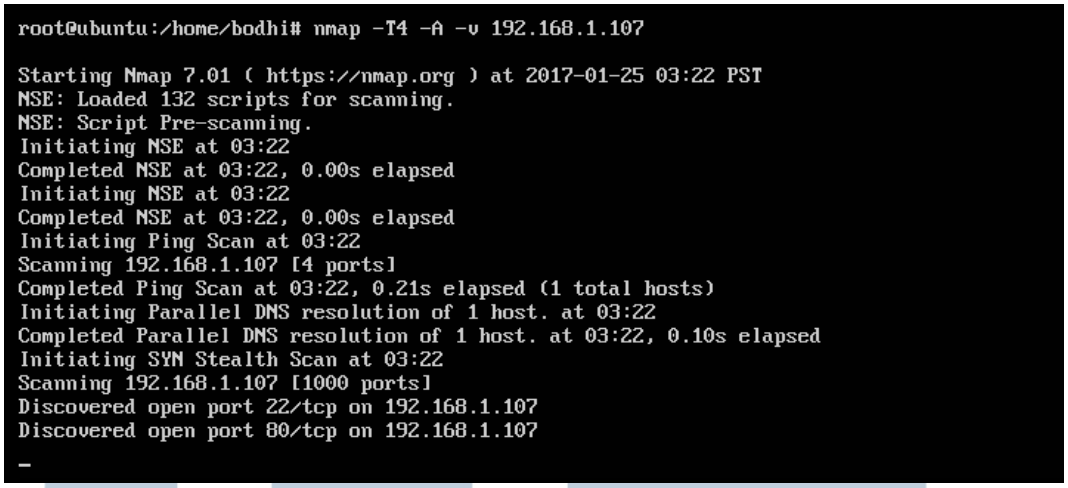
Gambar 3.40 Snort Berhasil Dijalankan

Gambar 3 41 Perintah Untuk Mengeksekusi Barnyard2

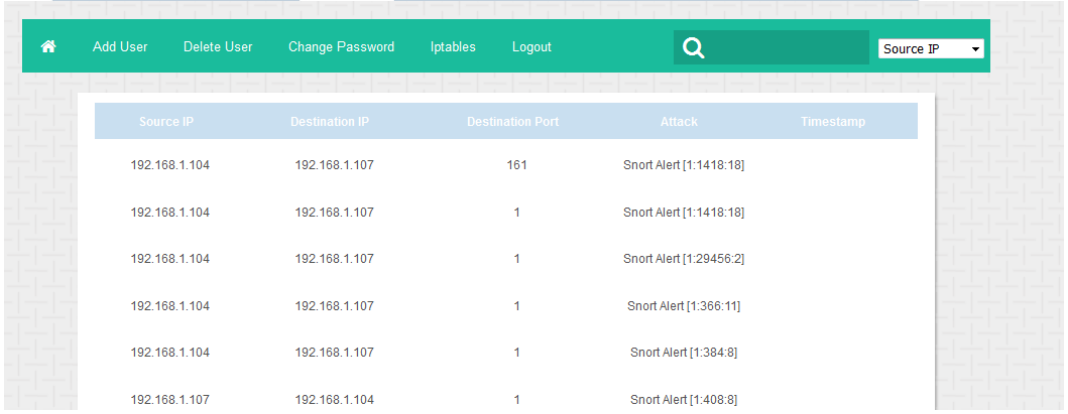


Gambar 3 42 Barnyard2 Berhasil Dijalankan

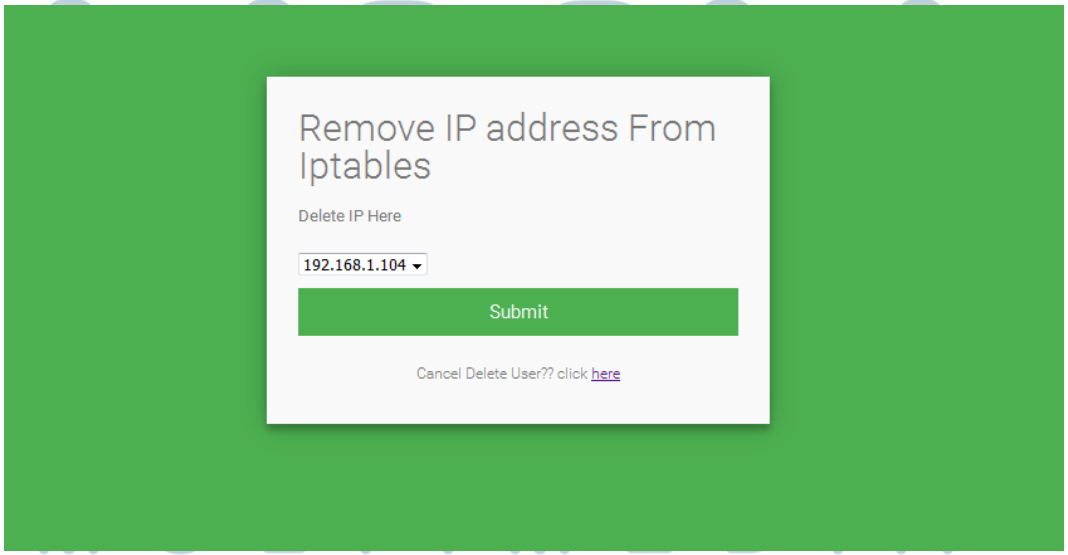
Gambar 3.43 Program Pembantu Berhasil Dijalankan



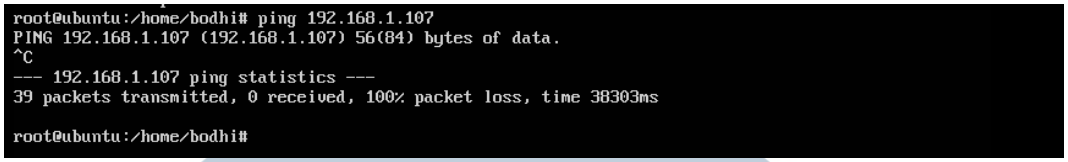
Gambar 3.44 Attacker Melakukan Port Scanning Dengan Nmap



Gambar 3.45 IP Attacker Masuk Ke Web Interface



Gambar 3.46 IP Attacker Masuk Ke Daftar IP Yang Diblock



Gambar 3.47 IP Attacker Sudah Tidak Dapat Berkomunikasi Dengan Server

3.3 Kendala Yang Ditemukan

Kendala yang ditemukan dalam pelaksanaan kerja magang adalah dalam konfigurasi snort. Dalam konfigurasi snort sering kali mendapatkan banyak pesan error yang menandakan kita salah dalam konfigurasi atau kekurangan program depedency, dan juga minimnya pengetahuan dalam menggunakkan sistem operasi ubuntu server yang merupakan sistem operasi dengan full command line operation. Serta minimnya pengetahuan tentang program snort dan program-program pendukung lainnya seperti barnyard2.

3.4 Solusi Atas Kendala Yang Ditemukan

Salah satu cara untuk mengatasi kendala yang ditemukan adalah dengan membaca kembali dokumentasi yang telah diterbitkan snort, dan untuk mengatasi salah konfigurasi dan belajar perintah command line baru adalah dengan banyak belajar dan mencari di google.