

BAB III

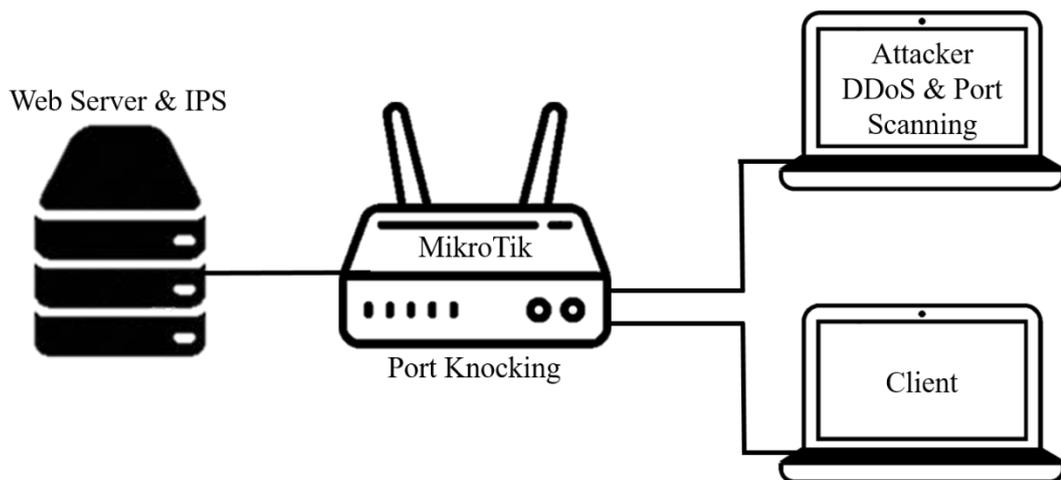
METODE PENELITIAN

3.1 Subjek dan Objek Penelitian

Subjek penelitian ini proses pengamatan dan penerapan *Port Knocking* dan IPS pada jaringan komputer yang akan diserang oleh serangan *cyber DDoS* dan *Port Scanning*. *Server* akan mengetahui dan memonitoring jaringan yang diserang. Objek penelitian ini adalah hasil pengamatan dan penerapan *Port Knocking* dan IPS pada jaringan komputer.

3.2 Topologi

Melakukan perancangan dan memberikan gambaran mengenai sistem keamanan yang akan dibangun.



Gambar 3.1 Topologi Jaringan

Berdasarkan Topologi Jaringan pada Gambar 3.1, *Port Ethernet 1* pada *Mikrotik* terhubung pada *Server*, *Port Ethernet 2* *Mikrotik* terhubung ke *Client* dan *Port Ethernet 3* *Mikrotik* terhubung ke *Attacker DDoS* dan *Port Scanning*.

3.3 Alat dan Bahan Penelitian yang Digunakan

3.3.1 Perangkat Keras

Perangkat keras yang dibutuhkan dalam proses penelitian ini adalah sebagai berikut :

1. Spesifikasi PC untuk *Web Server* dan *IPS* :
 - a. *Prosesor AMD Ryzen 5 2500 U.*
 - b. *Memory 4 GB DDR4.*
 - c. *Storage 128 GB SSD & 1TB HDD.*
 - d. *Display 14" FHD.*
 - e. *Graphic Radeon Vega Mobile GFX.*
2. Spesifikasi PC untuk *Attacker DDoS & Port Scanning* :
 - a. *Prosesor Intel Core i5-7200U.*
 - b. *Memory 4 GB DDR4.*
 - c. *Storage 1 TB HDD.*
 - d. *Display 14" FHD.*
 - e. *NVIDIA GeForce GT930MX.*
3. Spesifikasi PC untuk *Client* :
 - a. *Prosesor Intel Core i5-10210U.*
 - b. *Memory 8 GB DDR4.*
 - c. *Storage 512 GB SSD.*
 - d. *Display 14" FHD.*
 - e. *NVIDIA GeForce MX350.*
4. Spesifikasi *Mikrotik* :
 - a. *4 port Fast Ethernet.*
 - b. *Wireless 2.4Ghz (802.11b/g/n).*
 - c. *Antenna internal Dual-Chain 2x1.5dbi.*

3.2.2 Perangkat Lunak

Perangkat lunak yang dibutuhkan dalam proses penelitian ini adalah sebagai berikut :

1. Sistem Operasi *Ubuntu* sebagai sistem operasi *Web Server* dan *IPS*.
2. Sistem Operasi *Windows* sebagai sistem operasi *Client*.
3. Sistem Operasi *Kali Linux* sebagai sistem operasi *Attacker DDoS & Port Scanning*.
4. *Winbox* sebagai *tools* konfigurasi *Mikrotik*.
5. *Virtualbox* sebagai alat perangkat lunak virtualisasi yang dapat mengoperasikan beberapa *Operating System* pada sistem utama.
6. *Snort* sebagai *tools* yang digunakan *IPS*.
7. *Nmap* sebagai *tools* yang digunakan untuk melakukan penyerangan *Port Scanning*.
8. *Slowloris* sebagai *tools* yang digunakan untuk melakukan penyerangan *DDoS*.
9. *Apache2* sebagai *tools* yang digunakan untuk membangun *web server*.

3.4 Diagram Alur Penelitian

Alur penelitian yang digunakan oleh penulis dalam Implementasi *Port Knocking* pada *Mikrotik* dengan menerapkan *Intrusion Prevention System (IPS)* untuk sistem keamanan jaringan adalah metode studi literatur, perancangan sistem dan menyiapkan perangkat, konfigurasi sistem atau *tools* dan pengujian. Penjelasan lebih rinci dapat dilihat dalam diagram berikut:



Gambar 3.2 Diagram Alur Penelitian

3.4.1 Studi Literatur

Tahapan penelitian ini melakukan pengujian terhadap *server* yang telah disiapkan. Peneliti melakukan literatur mengenai *server* yang akan diserang menggunakan *tools Slowloris* dan *Nmap*. *Tools Snort* digunakan untuk mendeteksi intrusi-intrusi jaringan (penyusupan, penyerangan, pemindaian dan beragam bentuk ancaman lainnya) sekaligus melakukan pencegahan. Kemudian melakukan konfigurasi *Port Knocking* pada *Mikrotik*. Tujuan dari literatur adalah untuk memperkuat permasalahan yang diangkat pada penelitian ini serta menjadi dasar untuk melakukan pengembangan selanjutnya.

3.4.2 Perancangan Sistem dan Menyiapkan Perangkat

Pada penelitian ini akan merancang sebuah sistem yang akan diteliti, yaitu menyiapkan *tools Snort* dan konfigurasi *Port Knocking* yang akan digunakan dalam melakukan pencegahan. Perangkat yang dibutuhkan untuk mengkonfigurasi sistem sampai pengujian sistem, antara lain :

1. Satu unit PC sebagai *Web server* dan *IPS*.
2. Satu unit PC sebagai *Attacker Port Scanning* dan *DDoS*.
3. Satu unit PC sebagai *Client*.
4. Satu unit *Mikrotik*.
5. Tiga Unit kabel *straight over*.

Lalu sistem akan diuji menggunakan serangan sesuai dengan skenario pengujian yang dirancang.

3.4.3 Konfigurasi Sistem

Pada tahap ini akan mengkonfigurasi sistem yang dimana melakukan konfigurasi *Web server*, instalasi *tools Snort*, konfigurasi *rules Snort*, instalasi *tools Slowloris*, instalasi *tools Nmap*, konfigurasi *Port Forwarding* dan *Port Knocking*. *Tools* tersebut akan digunakan untuk melakukan *defence* dan memberikan notifikasi paket serangan yang masuk, sedangkan *tools Slowloris* dan *tools Nmap* akan digunakan untuk melakukan *attack* pada *server*. Pada penelitian ini akan melakukan suatu percobaan serangan dalam jaringan. Serangan yang digunakan dalam penelitian ini adalah *Port Scanning* dan *DDoS*.

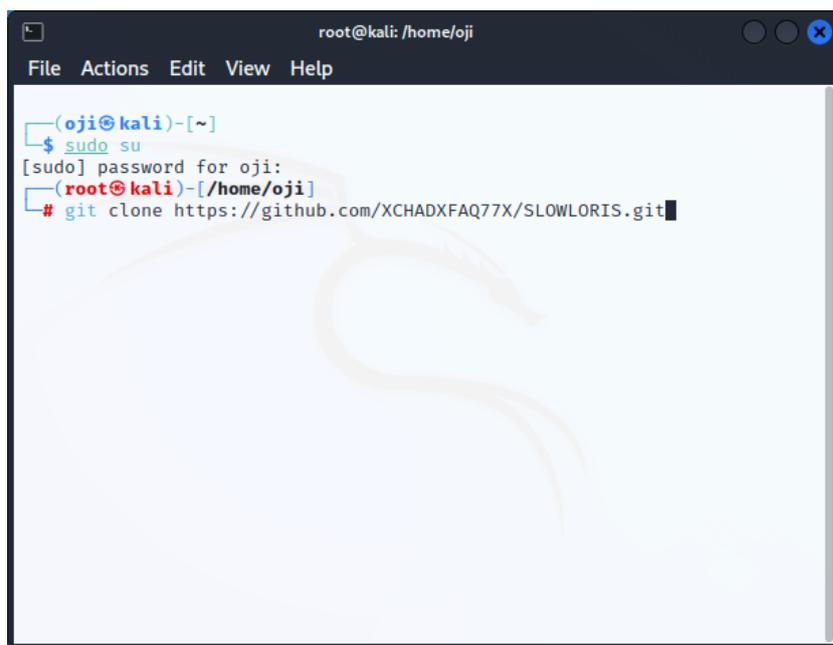
3.4.4 Pengujian

Pada tahap ini akan dilakukan pengujian terhadap *server* yang akan diserang. Kegiatan ini bertujuan untuk mengetahui tingkat keamanan *server* dari serangan *Port Scanning* dan *DDoS*. Pengujian yang akan dilakukan adalah pengujian serangan *Slowloris* dan *Nmap* untuk mengetahui serangan sukses dilakukan dan pengujian sistem *Snort* untuk mengetahui serangan berhasil dideteksi oleh sistem. Uji coba serangan masing – masing dilakukan selama 30 detik dengan

30 kali percobaan. Cara kerja dari serangan *Port Scanning* adalah dengan memasukkan *IP Address* target dan memilih aktivitas *Scan* yang dilakukan. Selain *Port Scanning*, uji coba serangan juga menggunakan serangan DDoS. Cara kerja dari serangan DDoS adalah dengan mengirimkan *packet* sebesar 4000 *packet*. Pengujian ini bertujuan untuk menguji *server* yang diserang, dan pada sistem *Snort* akan mendeteksi *packet* serangan *Slowloris* yang masuk, dan akan di *drop* oleh *Snort*.

3.5 Instalasi *Tools Slowloris*

Serangan DDoS dilakukan menggunakan *tools Slowloris*, yang nantinya serangan tersebut membuat *website* target tidak dapat diakses sebagaimana mestinya. Pengujian dilakukan dengan meng-*install* terlebih dahulu *Slowloris* yang akan digunakan.



```
root@kali: /home/oji
File Actions Edit View Help
(oji@kali)-[~]
└─$ sudo su
[sudo] password for oji:
(root@kali)-[~/home/oji]
└─# git clone https://github.com/XCHADXFAQ77X/SLOWLORIS.git
```

Gambar 3.3 Meng-*install Slowloris*

Pada gambar 3.3 kita bisa meng-*install Slowloris* dengan perintah “`git clone https://github.com/XCHADXFAQ77X/SLOWLORIS.git”`, setelah meng-*install Slowloris*, maka selanjutnya membuka *Slowloris* dengan perintah “`cd /opt/SLOWLORIS`” kemudian “`perl Slowloris.pl`” setelah kita membuka

`Slowloris.pl'` for help with options” digunakan untuk mengetahui fungsi – fungsi apa saja yang ada di *Slowloris*.

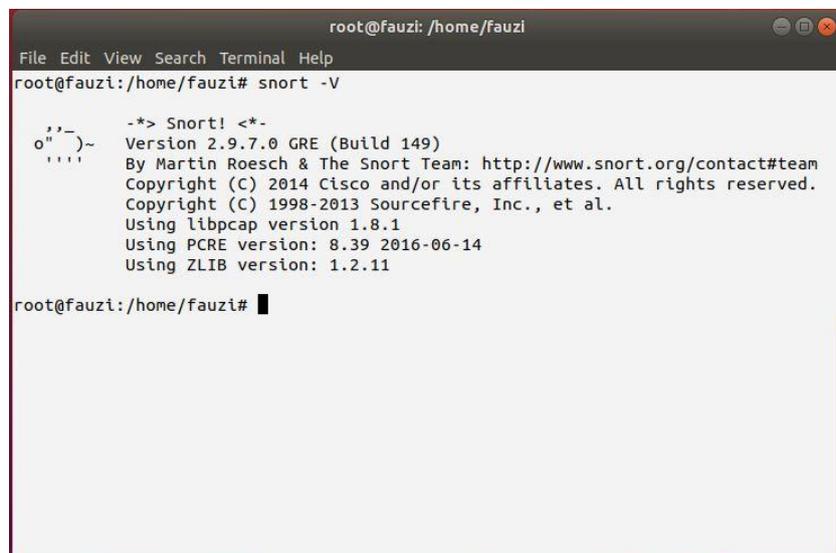
3.6 Instalasi *Intrusion Prevention System*

3.6.1 Instalasi *Snort*

Pada *tool Snort*, terdapat paket – paket yang dibutuhkan untuk menjalankan fungsi *Intrusion Prevention System* (IPS). Paket – paket tersebut dapat di *install* melalui terminal *Ubuntu* 18.04. Script yang dijalankan untuk meng*install* paket – paket tersebut adalah :

```
sudo apt install -y gcc libpcre3-dev zlib1g-dev liblua5.1-dev \
libpcap-dev openssl libssl-dev libnghttp2-dev libdumbnet-dev \
bison flex libdnet autoconf libtool.
```

Setelah paket – paket berhasil di *install*, kemudian jalankan perintah “*apt install Snort*” pada terminal *ubuntu*. Setelah *Snort* berhasil di *install* kemudian lakukan pengecekan. Pengecekan pada *Snort* dapat dilihat pada Gambar 3.5.



```
root@fauzi: /home/fauzi
File Edit View Search Terminal Help
root@fauzi:/home/fauzi# snort -V

o''-
  )~
  ''

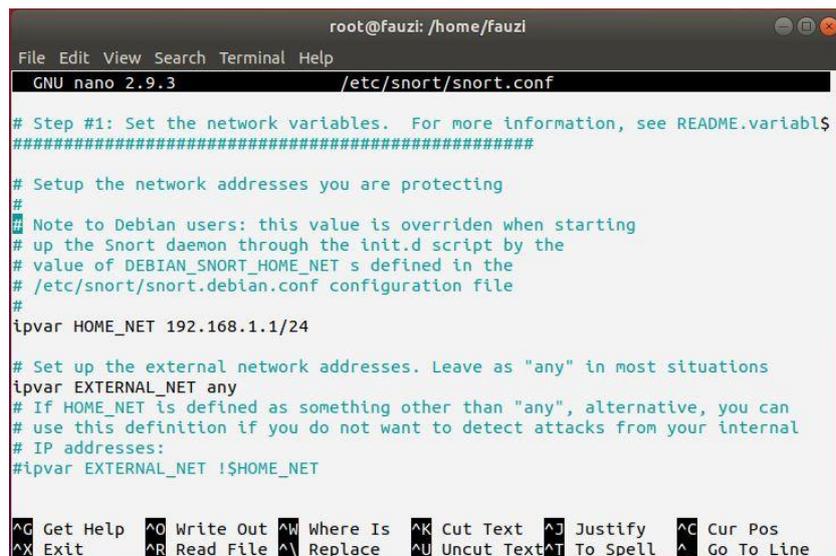
  -*> Snort! <*-
  Version 2.9.7.0 GRE (Build 149)
  By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
  Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
  Copyright (C) 1998-2013 Sourcefire, Inc., et al.
  Using libpcap version 1.8.1
  Using PCRE version: 8.39 2016-06-14
  Using ZLIB version: 1.2.11

root@fauzi:/home/fauzi# █
```

Gambar 3.5 *Snort* Berhasil di *Install*

3.6.2 Konfigurasi Snort

Pada konfigurasi *Snort*, hal yang dilakukan pertama kali adalah masuk ke file “*Snort.conf*” dengan mengetikkan “`nano /etc/Snort/Snort.conf`”. File “*Snort.conf*” berfungsi sebagai penyimpanan *rules* yang akan diberikan pada konfigurasi *Snort*. Pada file “*Snort.conf*”, dapat mengatur pemberian alamat network, menentukan *interface* yang akan digunakan, membuat *rules* atau perintah yang berfungsi untuk menjalankan fungsi IPS. Langkah pertama yang dilakukan untuk mengisi file “*Snort.conf*” adalah mengatur alamat IP yang akan dilindungi oleh *Snort*. Alamat IP yang digunakan adalah alamat IP *server* yaitu 192.168.1.1/24 dan HOME_NET adalah pendeklarasian nama *address group* yang diatur secara *default* oleh *Snort* seperti yang ditunjukkan pada Gambar 3.6.



```
root@fauzi: /home/fauzi
File Edit View Search Terminal Help
GNU nano 2.9.3 /etc/snort/snort.conf

# Step #1: Set the network variables. For more information, see README.variables
#####

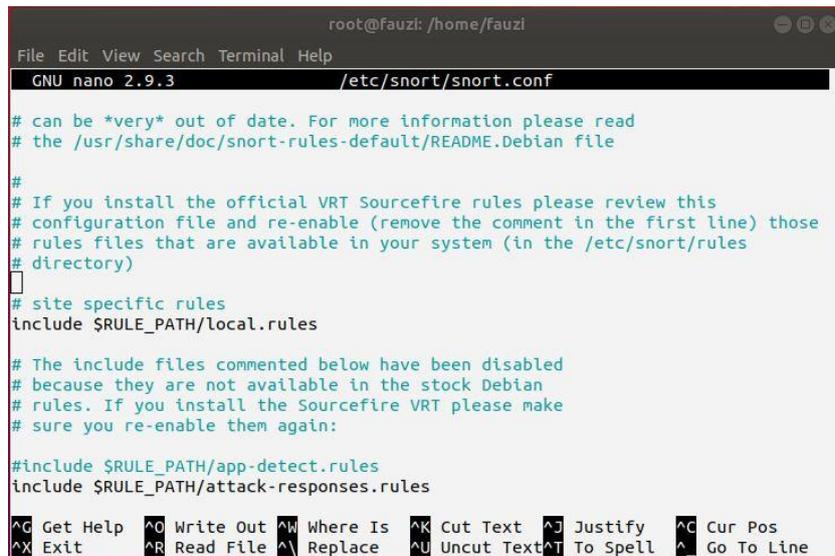
# Setup the network addresses you are protecting
#
## Note to Debian users: this value is overridden when starting
# up the Snort daemon through the init.d script by the
# value of DEBIAN_SNORT_HOME_NET s defined in the
# /etc/snort/snort.debian.conf configuration file
#
ipvar HOME_NET 192.168.1.1/24

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
# If HOME_NET is defined as something other than "any", alternative, you can
# use this definition if you do not want to detect attacks from your internal
# IP addresses:
#ipvar EXTERNAL_NET !$HOME_NET

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

Gambar 3.6 Address yang digunakan

Rules pada *Snort* merupakan aturan – aturan berupa perintah yang dibuat oleh para pengguna *tools Snort* untuk mengatur fungsi apa saja yang dijalankan untuk *Snort* yang telah di-*install*. Pada gambar 3.7, letak *rules* akan disimpan di dalam direktori *RULE_PATH*. Kemudian setelah menentukan direktori, langkah selanjutnya adalah memberikan nama *file* untuk *rules*. Dalam pengujian ini penulis menggunakan nama *default* dari *Snort* yaitu *local.rules*. sehingga, semua *rules* yang diatur oleh penulis merupakan isi dari *local.rules*.



```
root@fauzi: /home/fauzi
File Edit View Search Terminal Help
GNU nano 2.9.3 /etc/snort/snort.conf

# can be *very* out of date. For more information please read
# the /usr/share/doc/snort-rules-default/README.Debian file

#
# If you install the official VRT Sourcefire rules please review this
# configuration file and re-enable (remove the comment in the first line) those
# rules files that are available in your system (in the /etc/snort/rules
# directory)
#
# site specific rules
include $RULE_PATH/local.rules

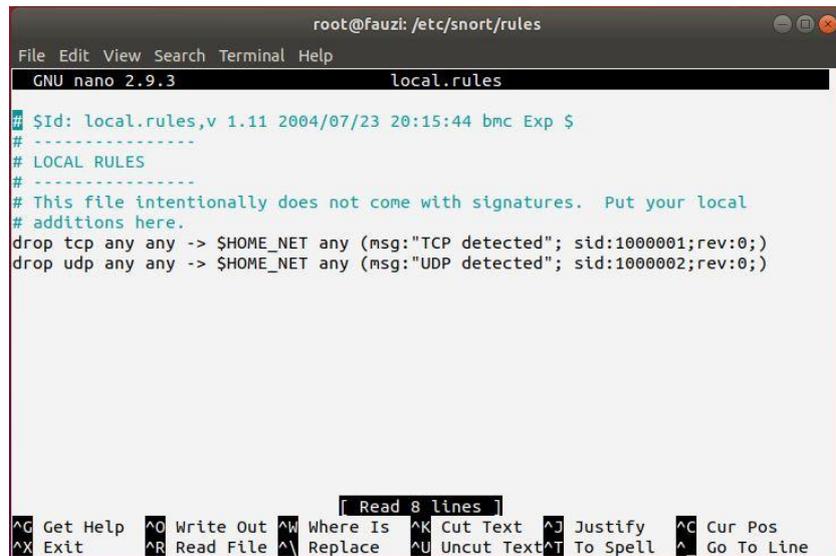
# The include files commented below have been disabled
# because they are not available in the stock Debian
# rules. If you install the Sourcefire VRT please make
# sure you re-enable them again:

#include $RULE_PATH/app-detect.rules
include $RULE_PATH/attack-responses.rules

^G Get Help  ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text ^T To Spell  ^_ Go To Line
```

Gambar 3.7 Setting letak file rules

Pada penelitian ini, penulis menjalankan fungsi *Intrusion Detection System* dan *Intrusion Prevention System*. Sehingga jika ada aktivitas mencurigakan atau penyusup yang berusaha masuk kedalam *web server* yang dibangun, maka *Snort* akan melakukan pendeteksian sehingga memunculkan *alert* untuk pemberitahuan dan juga melakukan *prevention* yaitu *Block* pada aktivitas yang mencurigakan. Sebuah *rule* pada *Snort* terdiri atas enam *field* utama, antara lain *Action*, *protocol*, *Source and Destination*, *ports (Source and Destination)*, *direction*, dan *rule option*. *Rules* yang digunakan pada penelitian ini, yaitu *rules* yang berfungsi untuk mendeteksi dan mencegah serangan DDoS berkarakteristik *TCP* atau *UDP* seperti yang ditunjukkan pada Gambar 3.8.



```
root@fauzi: /etc/snort/rules
File Edit View Search Terminal Help
GNU nano 2.9.3 local.rules
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# -----
# LOCAL RULES
# -----
# This file intentionally does not come with signatures.  Put your local
# additions here.
drop tcp any any -> $HOME_NET any (msg:"TCP detected"; sid:1000001;rev:0;)
drop udp any any -> $HOME_NET any (msg:"UDP detected"; sid:1000002;rev:0;)

[ Read 8 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace  ^U Uncut Text ^T To Spell  ^_ Go To Line
```

Gambar 3.8 Rules yang digunakan

Snort memiliki cara kerja untuk mendeteksi sebuah serangan atau aktivitas mencurigakan yang masuk kedalam suatu *interface* jaringan. Namun, selain memiliki pendeteksian, *Snort* juga memiliki fungsi untuk melakukan pencegahan atau yang dikenal sebagai IPS. Untuk mengaktifkan fungsi IPS tersebut, perlu melakukan *setting* konfigurasi pada sistem *Snort* yaitu dengan menjalankan mode *inline* dengan data *acquisition* (daq). Daq memiliki banyak tipe diantaranya adalah NFQ, *Afpacket*, IPQ dan IPW. Dari masing – masing tipe daq, memiliki skema penangkapan paket yang berbeda-beda. Pada penelitian, penulis menggunakan tipe *afpacket* yang cara kerjanya dengan menggunakan skema *forward* paket dari satu *interface* ke *interface* yang lain. Maka, untuk menjalankan fungsi *afpacket*, penulis membuat dua *interface*. Kelebihan dari menggunakan *afpacket* adalah tidak memerlukan konfigurasi tambahan pada *iptables*. Konfigurasi *afpacket* pada *Snort* dapat dilihat pada Gambar 3.9.

```

root@fauzi: /home/fauzi
File Edit View Search Terminal Help
GNU nano 2.9.3 /etc/snort/snort.conf
# config ignore_ports: tcp 21 6667:6671 1356
# config ignore_ports: udp 1:17 53
#
# Configure active response for non inline operation. For more information, see$
# config response: eth0 attempts 2
#
# Configure DAQ related options for inline operation. For more information, see$
#
config daq: afpacket
config daq_dir: /usr/local/lib/daq
config daq_mode: inline
config daq_var: buffer_size_mb=512
#
# <type> ::= pcap | afpacket | dump | nfq | ipq | ipfw
# <mode> ::= read-file | passive | inline
# <var> ::= arbitrary <name>=<value passed to DAQ
# <dir> ::= path as to where to look for DAQ module so's
#
# Configure specific UID and GID to run snort as after dropping privs. For more$
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line

```

Gambar 3.9 Mengaktifkan *afpacket*

Setelah mengaktifkan mode *inline* dan tipe *afpacket*, langkah selanjutnya adalah mengaktifkan mode IPS pada *Snort* agar tipe *afpacket* dapat berjalan sesuai dengan fungsi IPS. Pengaktifan mode IPS dapat dilihat pada gambar 3.10.

```

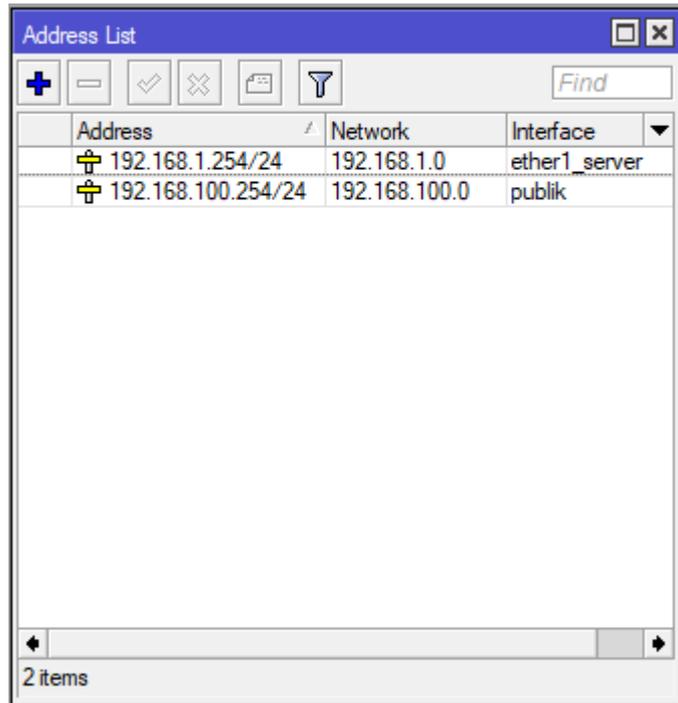
root@fauzi: /home/fauzi
File Edit View Search Terminal Help
GNU nano 2.9.3 /etc/snort/snort.conf
# preprocessor gtp: ports { 2123 3386 2152 }
#
# Inline packet normalization. For more information, see README.normalize
# Does nothing in IDS mode
preprocessor normalize_ip4
preprocessor normalize_tcp: ips ecn stream
preprocessor normalize_icmp4
preprocessor normalize_ip6
preprocessor normalize_icmp6
#
# Target-based IP defragmentation. For more information, see README.frag3
preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy windows detect_anomalies overlap_limit 10 min$
#
# Target-Based stateful inspection/stream reassembly. For more information, see$
preprocessor stream5_global: track_tcp yes, \
  track_udp yes, \
  track_icmp no, \
  max_tcp 262144, \

```

Gambar 3.10 Mengaktifkan mode IPS

3.7 Konfigurasi Mikrotik

Pada konfigurasi *Mikrotik* ini akan dibuat sebuah alamat IP Address *ethernet* 1 yang terhubung ke *Web server* dan *IPS* dan alamat IP address yang telah di *bridge* untuk *ethernet* 2 dan *ethernet* 3 yang terhubung ke *Attacker* dan *Client*.

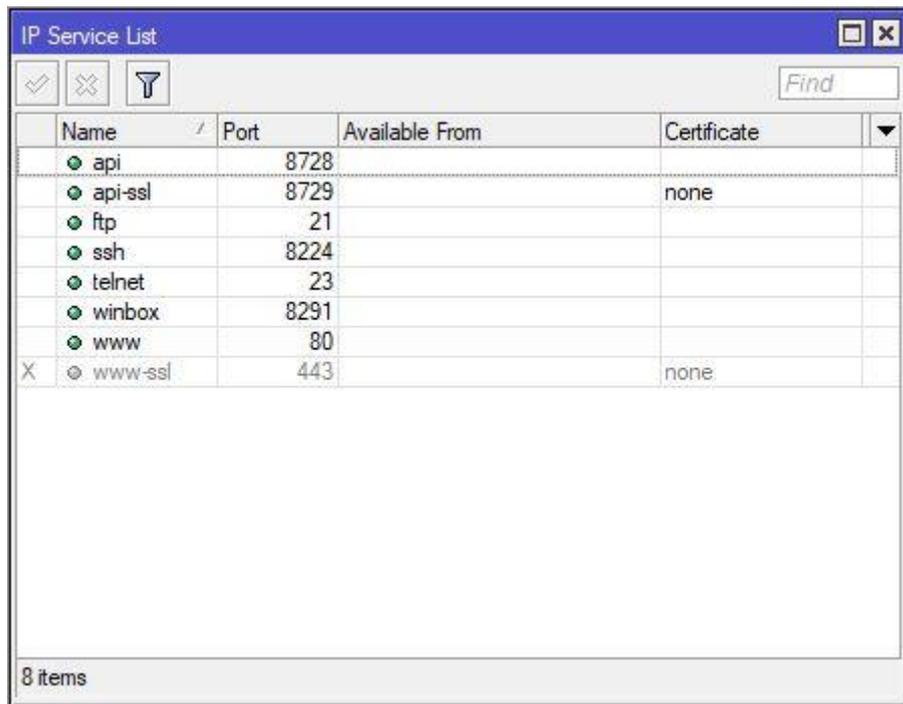


Address	Network	Interface
192.168.1.254/24	192.168.1.0	ether1_server
192.168.100.254/24	192.168.100.0	publik

Gambar 3.11 IP Address Mikrotik

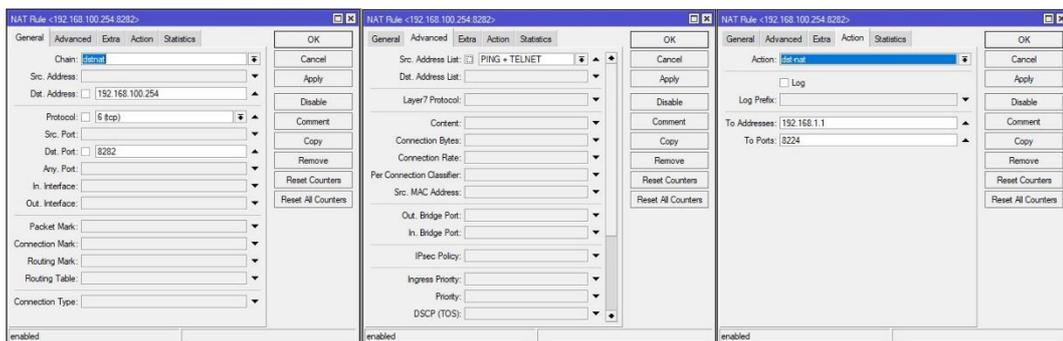
Pada Gambar 3.11 alamat IP Address telah ditambahkan ke *ethernet* 1 dengan alamat IP Address 192.168.1.254/24 dan 192.168.100.254/24 untuk *ethernet* 2 dan *ethernet* 3 yang telah di *bridge* dan diberi nama *interface* publik.

Port SSH pada penelitian ini menggunakan *port* unik agar upaya penyerang mengakses *Mikrotik* dapat dihentikan sedini mungkin. Perubahan *port* dikonfigurasi pada menu *IP Service* kemudian mengubah *port default* 22 menjadi *port* 8224. Hasil perubahan *port SSH Mikrotik* dapat dilihat pada Gambar 3.12



Gambar 3.12 Mengubah Port Default SSH

Port forwarding berfungsi untuk membuka akses terhadap perangkat pada jaringan lokal untuk bisa diakses melalui jaringan publik, sehingga penulis menerapkan metode ini agar *ethernet 1* yang dijadikan jaringan lokal dapat terhubung dengan *ethernet 2* dan *ethernet 3* yang dijadikan sebagai jaringan publik. Proses konfigurasi *Port forwarding* dilihat pada gambar 3.13.



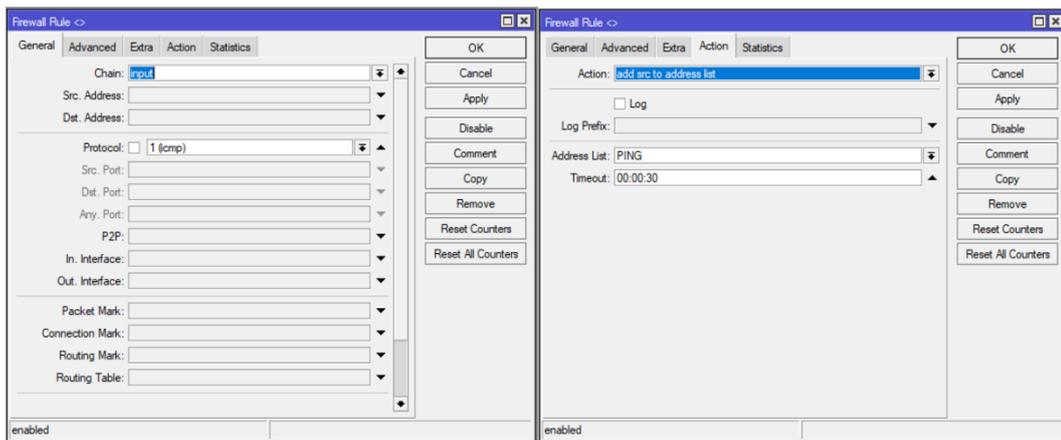
Gambar 3.13 Konfigurasi Port forwarding

Pada Gambar 3.13 langkah awal yang dilakukan untuk melakukan konfigurasi *Port forwarding* adalah menambahkan *NAT Rule*. Pada tab *General* penulis menambahkan *dstnat* pada kolom *Chain* dan menambahkan IP Address 192.168.100.254. Protokol yang digunakan pada *rule* ini adalah protokol *TCP* dan

menggunakan *port* 8282 pada *Destination Port*. Pada tab *Advanced* di kolom *Source Address List* penulis menambahkan *rules PING + TELNET* dan *Action* yang digunakan pada *rule* ini adalah *dst-nat*. Pada kolom *To Addresses* kita menambahkan IP dari *Web server* dan IPS dengan alamat IP Address 192.168.1.1 dengan *port* 8224.

Penulis juga menerapkan metode *Port forwarding* untuk *Client* bisa mengakses *server* melalui *SSH*. Namun, sebelum mengakses *server* melalui *SSH*, *Client* harus melakukan *PING + TELNET* ke *port* 8223 dan masuk melalui *port* 8282. Sehingga dengan menerapkan *rule* ini, *server* akan lebih aman dari *attacker* yang ingin masuk kedalam sistem *server*.

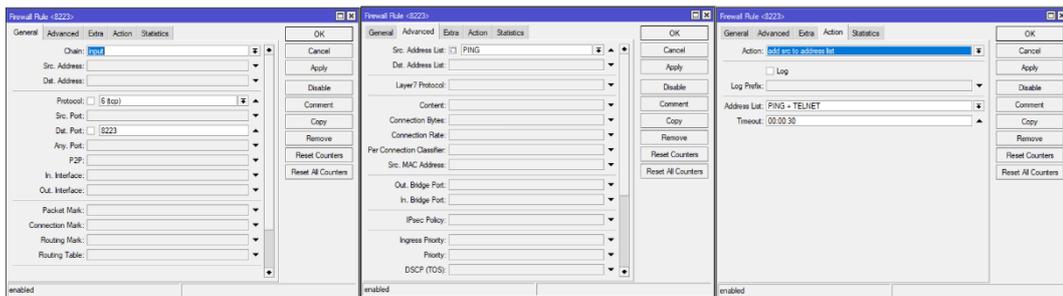
Port Knocking adalah metode yang digunakan untuk membuka akses ke *port* tertentu yang telah di-*block* oleh *firewall* dengan cara mengirimkan paket tertentu. Paket bisa berupa *protocol ICMP, TCP* dan *UDP*. Jika paket yang dikirimkan oleh *host* telah sesuai *rule* autentikasi yang digunakan, maka secara dinamis *firewall* akan memberikan akses ke *port* yang sudah di-*block*. Proses konfigurasi *Port Knocking rule* pertama dapat dilihat pada Gambar 3.14.



Gambar 3.14 Konfigurasi *Port Knocking rule* pertama

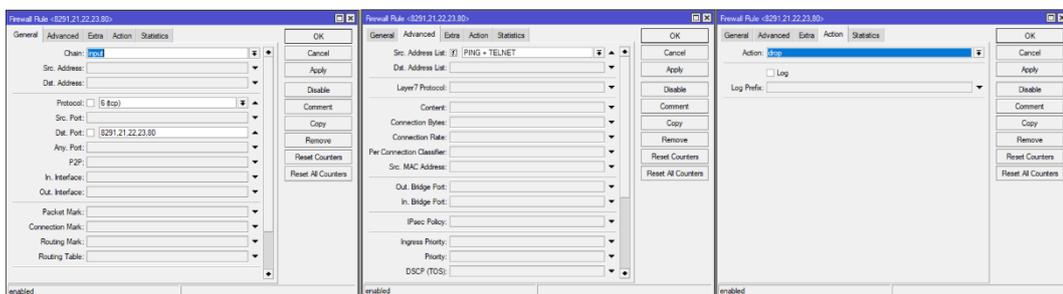
Pada gambar 3.14 adalah *rule* pertama yang diterapkan untuk *Port Knocking*. *Rule* pertama yang diterapkan untuk menggunakan *Port Knocking* adalah protokol *ICMP*. Pada tab *General* penulis menambahkan *input* pada kolom *Chain*. *Action* yang digunakan pada *rule* ini adalah *add src to address list* agar

Mikrotik bisa merekam IP Address yang menjalankan rule ini. Pada kolom Address List penulis menamakan rules ini dengan kata PING dan memberikan Timeout sebanyak 30 detik. Setelah menerapkan rule pertama, penulis menerapkan rule kedua yang dapat dilihat pada Gambar 3.15.



Gambar 3.15 Konfigurasi Port Knocking rule kedua

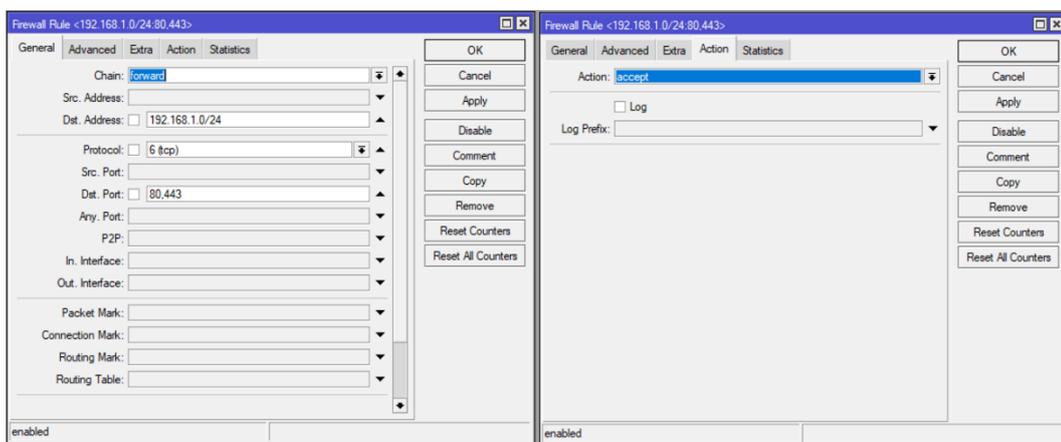
Pada Gambar 3.14 adalah rule kedua yang diterapkan untuk Port Knocking. Rule kedua yang diterapkan untuk menggunakan Port Knocking adalah protokol TCP dengan port 8223. Pada tab General penulis menambahkan input pada kolom Chain. Pada tab Advanced di kolom Source Address List penulis menambahkan PING yang sebelumnya telah dibuat pada rule pertama. Action yang digunakan pada rule ini adalah add src to address list agar Mikrotik bisa merekam IP Address yang menjalankan rule ini. Pada kolom Address List penulis menamakan rules ini dengan kata PING + TELNET dan memberikan Timeout sebanyak 30 detik. Setelah menerapkan rule kedua, penulis menerapkan rule ketiga yang dapat dilihat pada Gambar 3.16.



Gambar 3.16 Konfigurasi Port Knocking rule ketiga

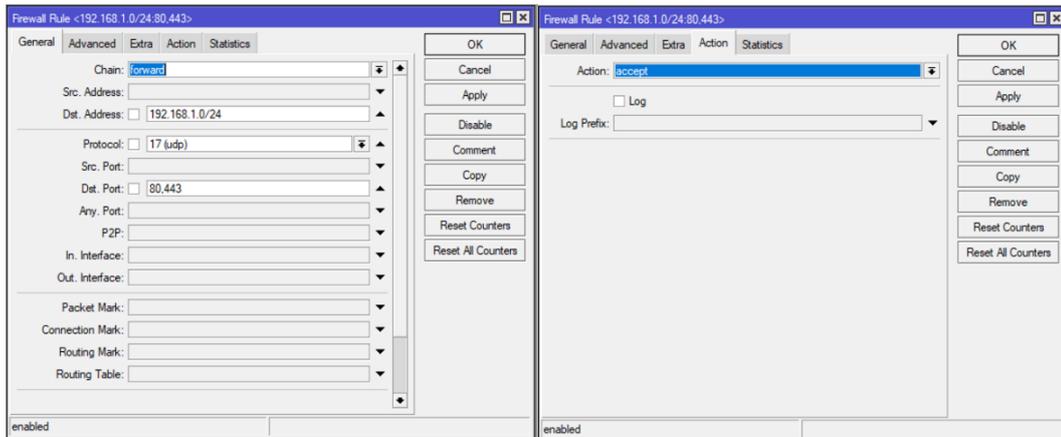
Pada gambar 3.16 adalah rule ketiga yang diterapkan untuk Port Knocking. Rule ketiga yang diterapkan untuk menggunakan Port Knocking adalah

protokol *TCP* dengan *port* 8291, 21, 22, 23 dan 80. Pada tab *General* penulis menambahkan *input* pada kolom *Chain*. Pada tab *Advanced* di kolom *Source Address List* penulis menambahkan *PING + TELNET* yang sebelumnya telah dibuat pada *rule* kedua. *Action* yang digunakan pada *rule* ini adalah *drop* agar *Mikrotik* melakukan penutupan *port* 8291, 21, 22, 23, dan 80 sebelum *host* melakukan *rule* pertama dan *rule* kedua. Setelah menerapkan *rule* ketiga, penulis menerapkan *rule* keempat yang dapat dilihat pada Gambar 3.17.



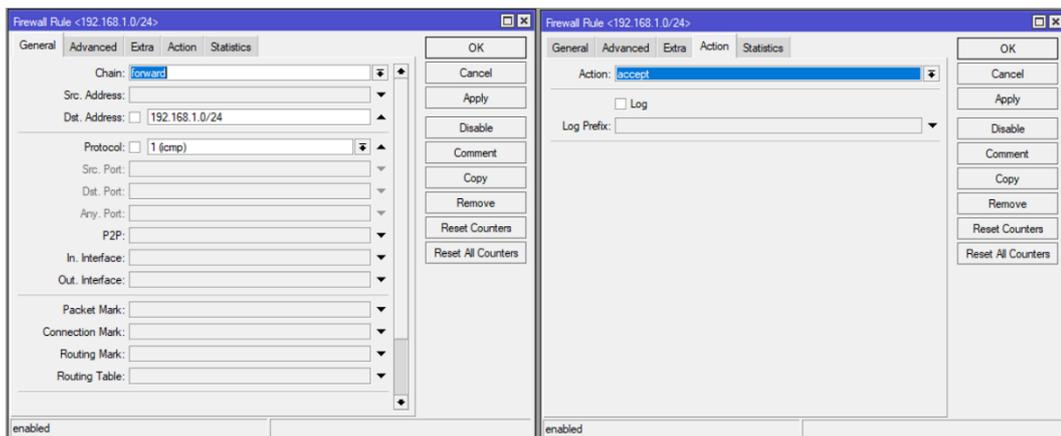
Gambar 3.17 Konfigurasi *Port Knocking rule* keempat

Pada gambar 3.17 adalah *rule* keempat yang diterapkan untuk *Port Knocking*. *Rule* keempat yang diterapkan untuk menggunakan *Port Knocking* adalah protokol *TCP* dengan *port* 80 dan 443. Pada tab *General* penulis menambahkan *forward* pada kolom *Chain* dan *Network Address server* pada kolom *Dst. Address*. *Action* yang digunakan pada *rule* ini adalah *accept* agar *Mikrotik* mengizinkan semua *host* untuk mengakses *Web server* melalui *port* 80 dan 443. Setelah menerapkan *rule* keempat, penulis menerapkan *rule* kelima yang dapat dilihat pada Gambar 3.18.



Gambar 3.18 Konfigurasi *Port Knocking* rule kelima

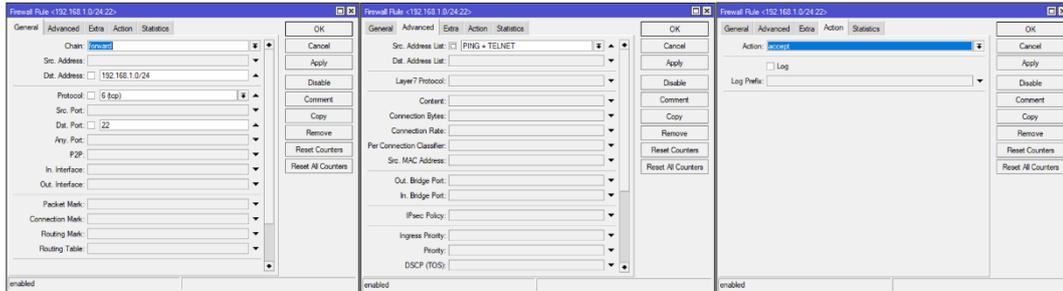
Pada gambar 3.18 adalah *rule* kelima yang diterapkan untuk *Port Knocking*. *Rule* kelima yang diterapkan untuk menggunakan *Port Knocking* adalah protokol *UDP* dengan *port* 80 dan 443. Pada tab *General* penulis menambahkan *forward* pada kolom *Chain* dan *Network Address server* pada kolom *Dst. Address*. *Action* yang digunakan pada *rule* ini adalah *accept* agar *Mikrotik* mengizinkan semua *host* untuk mengakses *Web server* melalui *port* 80 dan 443. Setelah menerapkan *rule* kelima, penulis menerapkan *rule* keenam yang dapat dilihat pada Gambar 3.19.



Gambar 3.19 Konfigurasi *Port Knocking* rule keenam

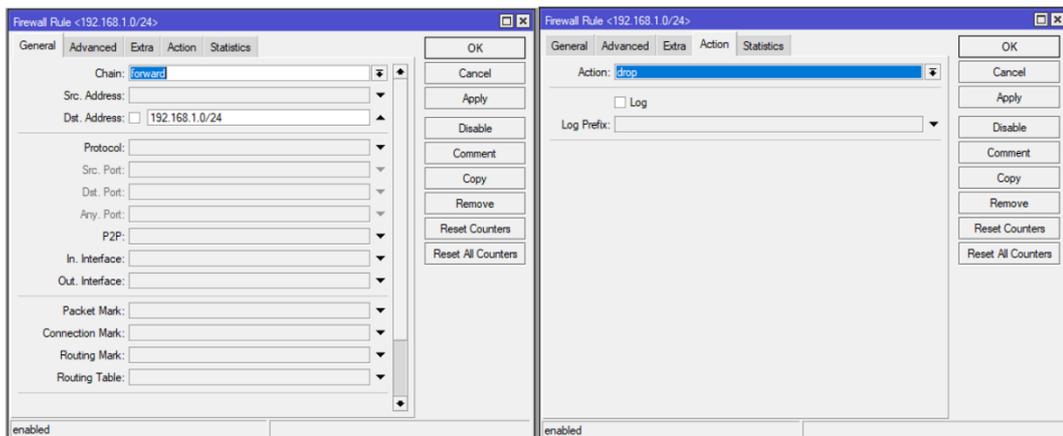
Pada gambar 3.19 adalah *rule* keenam yang diterapkan untuk *Port Knocking*. *Rule* keenam yang diterapkan untuk menggunakan *Port Knocking* adalah protokol *ICMP*. Pada tab *General* penulis menambahkan *forward* pada kolom *Chain* dan *Network Address server* pada kolom *Dst. Address*. *Action* yang

digunakan pada *rule* ini adalah *accept* agar *Mikrotik* mengizinkan semua *host* untuk mengakses *server*. Setelah menerapkan *rule* keenam, penulis menerapkan *rule* ketujuh yang dapat dilihat pada Gambar 3.20.



Gambar 3.20 Konfigurasi *Port Knocking rule* ketujuh

Pada gambar 3.20 adalah *rule* ketujuh yang diterapkan untuk *Port Knocking*. *Rule* ketujuh yang diterapkan untuk menggunakan *Port Knocking* adalah protokol *TCP* dengan *port* 22. Pada tab *General* penulis menambahkan *forward* pada kolom *Chain* dan *Network Address server* pada kolom *Dst. Address*. Pada tab *Advanced* di kolom *Source Address List* penulis menambahkan *PING + TELNET* yang sebelumnya telah dibuat pada *rule* kedua. *Action* yang digunakan pada *rule* ini adalah *accept* agar *Mikrotik* mengizinkan semua *host* untuk mengakses *server*. Setelah menerapkan *rule* ketujuh, penulis menerapkan *rule* terakhir yang dapat dilihat pada Gambar 3.21.



Gambar 3.21 Konfigurasi *Port Knocking rule* terakhir

Pada gambar 3.21 adalah *rule* terakhir yang diterapkan untuk *Port Knocking*. Pada tab *General* penulis menambahkan *forward* pada kolom *Chain* dan

Network Address server pada kolom *Dst. Address*. *Action* yang digunakan pada *rule* ini adalah *accept* agar *Mikrotik* mengizinkan semua *host* untuk mengakses *server*. Setelah menerapkan *rule* terakhir, hasil seluruh konfigurasi dapat dilihat pada Gambar 3.22.

#	Action	Chain	Src. Address	Dst. Address	Proto...	Src. Port	Dst. Port	In.
0	add src to address list	input			1 (c...			
1	add src to address list	input			6 (tcp)		8223	
2	drop	input			6 (tcp)		8291,21,...	
3	accept	forward		192.168.1....	6 (tcp)		80,443	
4	accept	forward		192.168.1....	17 (u...		80,443	
5	accept	forward		192.168.1....	1 (c...			
6	accept	forward		192.168.1....	6 (tcp)		22	
7	drop	forward		192.168.1....				

Gambar 3.22 Hasil konfigurasi *Port Knocking*