BUKU MANUAL APLIKASI
IMPLEMENTASI *HONEYPOT* DAN IDPS PADA SISTEM KEAMANAN *SERVER* UNTUK MENDETEKSI SERANGAN YANG DIMUNCULKAN SEBAGAI PESAN PERINGATAN PADA *TELEGRAM BOT*

Oleh:

Vipkas Al Hadid Firdaus, ST,. MT

Yuri Ariyanto, S.Kom., M.Kom

Rizaldi Aryadana Anugrah Putra

POLITEKNIK NEGERI MALANG

OKTOBER 2022

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I. PENDAHULUAN

Terdapat pengujian penelitian yang sudah ada dengan mengimplementasikan simulasi sistem keamanan *server* dengan menggabungkan beberapa *tools* aplikasi yaitu *Cowrie* sebagai *Honeypot* dan *Snort* sebagai *Intrusion Detection and Prevention System* (IDPS) pada perangkat *Raspberry Pi* dan berhasil meminimalisir serangan, hanya saja perlu ada menjadi pengembangan di dalamnya.

Sebagai seorang *administrator* jaringan seharusnya bertanggung jawab atas lalu lintas jaringan di *server*, sehingga wajib untuk memantau dan menjaga keamanan jaringan *server* agar berjalan dengan aman tanpa gangguan. Dilihat dari permasalahan yang diperoleh, maka perlu dilakukan pengembangan yang terletak pada bagian pencatatan aktivitas atau *log* yang mendeteksi serangan pada *server*. Jika ada aktivitas serangan dan terdeteksi maka akan muncul pesan peringatan pada *Telegram* yang sedang berjalan berdasarkan aturan *Snort* yang akan dibuat. Alasan menggunakan *Telegram* adalah karena *Telegram* dapat digunakan di berbagai perangkat, sehingga *server administrator* jaringan dapat menggunakan perangkat apa saja yang dapat diinstal dengan *Telegram* untuk mendapatkan pesan peringatan jika terjadi serangan. Serta didalamnya terdapat fitur *Bot* yang dapat digunakan sebagai otomatisasi informasi yang tentunya akan memudahkan orang yang akan menggunakannya.

Fokus utama dalam pengembangan ini adalah melakukan integrasi untuk mendeteksi serangan yang akan muncul berupa pesan peringatan yang akan dikirimkan ke *Bot Telegram* yang dilakukan oleh *Snort*. Sehingga dapat disimpulkan bahwa tujuan dari penelitian ini adalah untuk memenuhi kebutuhan *server administrator* jaringan dalam menjaga sistem keamanan jaringan pada *server* dengan mengamankannya dengan konfigurasi *Cowrie Honeypot, Snort dan Fail2ban*, serta mengintegrasikan *Snort* untuk mendeteksi suatu menyerang dan mengirim pesan peringatan ke *Bot Telegram*.

BAB II. KEBUTUHAN SISTEM

Pada bagian ini akan dijelaskan bagaimana konsep pengembangan yang akan diterapkan dalam keamanan jaringan *server* yang ada sebelumnya.

Tabel 1. Deskripsi Sistem

Judul	IMPLEMENTASI *HONEYPOT* DAN IDPS PADA SISTEM KEAMANAN *SERVER* UNTUK MENDETEKSI SERANGAN YANG DIMUNCULKAN SEBAGAI PESAN PERINGATAN PADA *TELEGRAM BOT*
Jenis Aplikasi	*Monitoring* deteksi serangan pada sebuah *server* yang dimunculkan menggunakan aplikasi *Telegram*.
Pengguna	Seorang *network administrator* yang menangani sebuah *server*.
Konten	Pesan peringatan jika terjadi sebuah serangan tertentu pada *server*.
Aplikasi	Keluaran aplikasi berupa sebuah pesan peringatan dari *Snort* yang dikirimkan pada *Telegram Bot*.

Berdasarkan tabel 1 tentang konsep deskripsi sistem, maka kebutuhan aplikasi yang akan dikategorikan ke dalam kebutuhan fungsional dan kebutuhan *non*-fungsional.

A. Kebutuhan Fungsional

Sistem keamanan jaringan pada *server Raspberry Pi 3* pada awalnya sudah terpasang sebuah operasi sistem *Raspberry Pi* bawaan sebagai *mini server* yang menghubungkan antara penerapan dari kombinasi dari beberapa *software* yaitu instalasi *Cowrie* pada *Honeypot*, *Snort* dan *Fail2ban*. Pada pengujian penelitian ini, ketiga *software* tersebut akan berjalan bersamaan agar bisa memenuhi kebutuhan *server*.

Cowrie Honeypot akan bertindak sebagai *server* tiruan agar jika terdapat serangan, maka serangan tersebut tidak langsung menuju ke *server* yang asli, sehingga *server* yang asli akan lebih aman, dikarenakan pada *port server* SSH (*port* 22) akan melalui *server* tiruan milik *Cowrie Honeypot* sebelum menuju ke *server* yang asli.

Kemudian pada *Snort* akan dibuat sebuah *rules service* yang memiliki fungsi perintah untuk mendeteksi serangan yang akan dimunculkan berupa pesan peringatan yang akan dikirim ke *Telegram Bot* dan pesan peringatan tersebut akan berupa alamat IP *attacker* dan

waktu terjadinya serangan. Pesan peringatan tersebut berasal dari hasil *log Snort* yang telah diolah, sehingga akan memunculkan pesan peringatan sesuai dengan kebutuhan.

Di sisi lain, *Fail2ban* yang telah terpasang dengan konfigurasi *default* juga akan mendeteksi adanya tindakan yang tidak wajar tersebut. *Fail2ban* sendiri adalah sebuah *software* yang mendeteksi adanya sebuah tindakan kesalahan *login* yang tidak wajar dan dianggap sebagai serangan di dalam lalu lintas jaringan *server*.

Agar hasil deteksi yang didapatkan dari *Snort* bisa diterima oleh *network administrator server*, di dalam *Snort* nantinya akan terdapat kumpulan perintah dan fungsi untuk mengirimkan sebuah pesan peringatan. Tapi sebelumnya akan dilakukan sebuah pemanggilan *chat_id* yang didapatkan dengan melakukan permintaan *token* API menggunakan *bot* di dalam *Telegram* yang bernama *@BotFather*.

B. Kebutuhan *Non*-Fungsional

Kebutuhan non-fungsional dibagi menjadi dua bagian, yaitu:

a) Kebutuhan Perangkat Lunak

Jenis perangkat lunak atau *tools* yang digunakan untuk membantu proses pengembangan sistem ini adalah sebagai berikut:

Tabel 2 Kebutuhan Non-Fungsional

No	Perangkat Lunak	Deskripsi
1.	Sistem Operasi *Windows 10*	Digunakan untuk menjalankan aplikasi *Virtual*Box.
2.	*VirtualBox*	Digunakan untuk menjalankan Sistem Operasi *attacker* yaitu *Kali Linux* secara *virtual*.
3.	*Microsoft Office 2016*	Digunakan untuk melakukan pembuatan susunan laporan.
4.	*Cowrie Honeypot*	Sebuah *software* yang digunakan untuk menjebak *attacker*.
5.	*Snort*	Sebuah *software* untuk membuat *rules service* deteksi serangan dan mengirimkan data paket serangan tersebut ke *Telegram Bot*.
6.	*Fail2ban*	Sebuah *software* yang berperan sebagai

		tindakan pencegahan saat lalu lintas jaringan terdapat sebuah serangan.
7.	Sistem Operasi *Kali Linux*	Digunakan sebagai sistem operasi pembuatan serangan *brute-force* SSH yang ditujukan pada *server.*
8.	*Telegram*	Aplikasi yang digunakan sebagai media keluaran untuk memunculkan sebuah pesan peringatan jika terjadi serangan.
9.	API *Telegram*	Digunakan untuk integrasi antara IDPS *Snort* dan *Telegram Bot*.
10.	*Browser*	Digunakan untuk membaca *chat_id* dan *token* pada *Telegram*.
11.	*Hydra*	Untuk *software* serangan *brute-force* SSH yang ditujukan untuk menyerang *server* yang dijalankan pada *Kali Linux*.
12.	VIM	Digunakan sebagai *text editor* pada konfigurasi *Raspberry Pi*.

b) Kebutuhan Perangkat Keras

Adapun perangkat keras yang terlibat pada saat proses pengembangan sistem, yaitu:

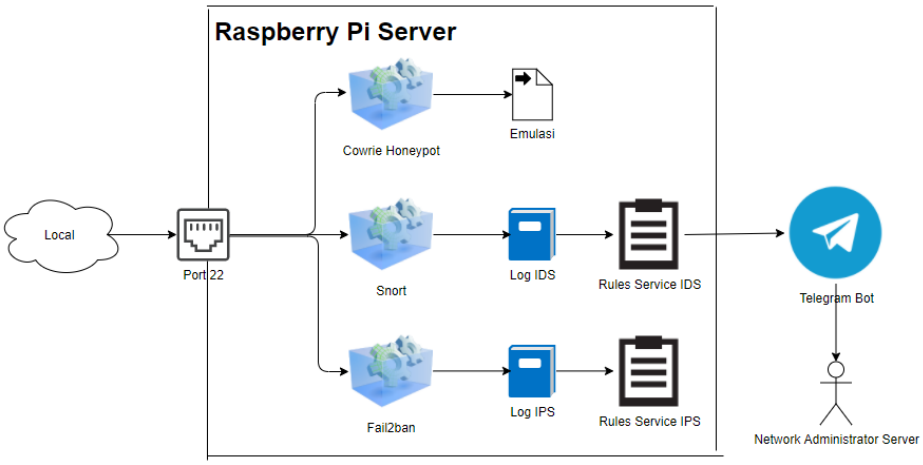
Tabel 3 Kebutuhan Perangkat Keras

No.	Perangkat Keras	Deskripsi
1.	Laptop	Menggunakan laptop merk HP dengan *processor Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz 2.40 GHz* dengan kapasitas RAM 8,00 GB (7,89 GB *usable*).
2.	*Raspberry Pi 3*	Menggunakan *Raspberry Pi 3* Model B+.
3.	*Mouse*	Digunakan untuk menggerakkan *cursor* yang ada pada *Raspberry Pi*.
4.	*Keyboard*	Digunakan sebagai alat untuk mengetik perintah dan konfigurasi pada *Raspberry Pi*.
5.	Monitor	Digunakan untuk menampilkan *interface*

		yang ada pada *Raspberry Pi*.
6.	HDMI	Sebagai perantara antarmuka untuk *Raspberry Pi* ke monitor
7.	*Power Adapter*	Digunakan sebagai penghubung daya listrik untuk *Raspberry Pi*.
8.	*Micro SD Card*	Digunakan untuk media penyimpanan yang ada pada *Raspberry Pi*.

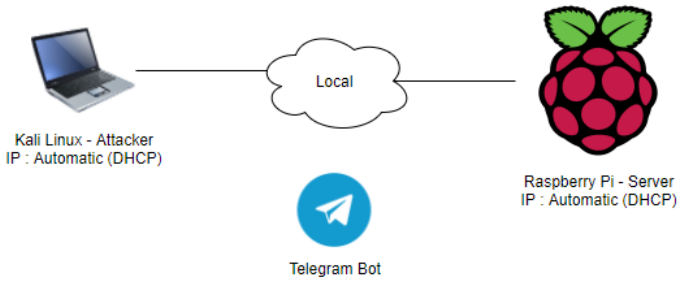
BAB III. DESAIN SISTEM

Tahap ini akan dilakukan pembuatan sebuah pengembangan arsitektur yang akan dibangun seperti apa. Diharapkan pada desain yang akan dibuat akan memberikan gambaran seutuhnya dari kebutuhan yang ada. Desain yang digambarkan adalah sebagai berikut:



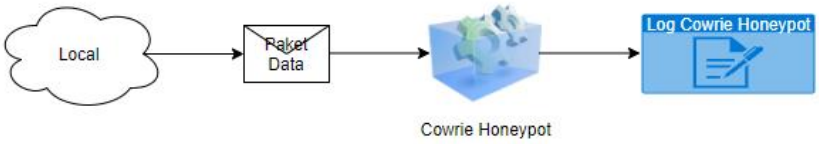
Gambar 1 Arsitektur Sistem Secara Keseluruhan

Dari Gambar 1 diatas dijelaskan bahwa gambar tersebut merupakan penjelasan dari arsitektur sistem yang akan dibangun secara keseluruhan. Dimulai dengan menjelaskan alur mulai dari serangan yang ada di *cloud* atau diluar dari sistem sampai dengan seorang *network administrator server* mendapatkan sebuah pesan peringatan ketika terjadi serangan di *server* berupa IP *address* dari *attacker* dan waktu terjadinya serangan yang telah diproses sebelumnya oleh *Snort*.



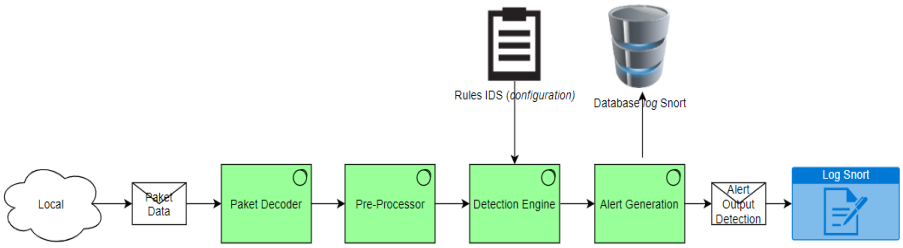
Gambar 2 Topologi Jaringan Secara Keseluruhan

Pada Gambar 2 diatas merupakan desain topologi jaringan yang akan diterapkan dalam penelitian ini untuk dilakukan sebuah simulasi.



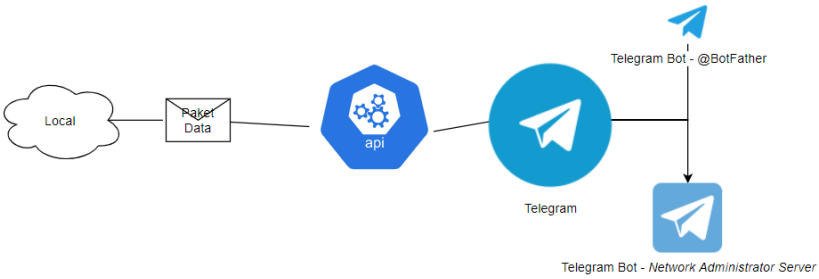
Gambar 3 Arsitektur Sistem *Cowrie Honeypot*

Dari Gambar 3 dijelaskan saat serangan masuk pada *port 22* yang sebenarnya *port 22* tersebut merupakan penerapan dari *Cowrie Honeypot* yang sudah dikonfigurasi sebagai wadah atau untuk menjebak *attacker* agar tidak langsung masuk pada *server* asli. *Cowrie Honeypot* disini berfungsi sebagai *server* yang menyerupai *server* asli *Raspberry Pi.*



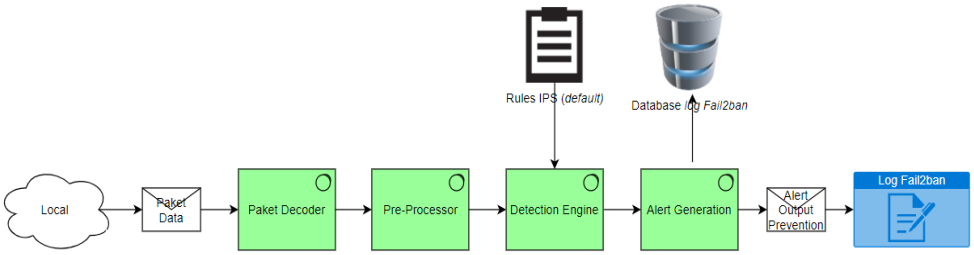
Gambar 4 Arsitektur Sistem *Snort*

Pada Gambar 4 dijelaskan bahwa bagaimana proses olah data yang diterima oleh *Snort* pada lalu lintas jaringan yang ada. Setiap paket yang terindikasi memiliki kecocokan dengan *signature* dari suatu serangan, *Snort* akan menghasilkan keluaran berupa pencatatan aktifitas intrusi atau *alert output* yang sudah berhasil dideteksi.



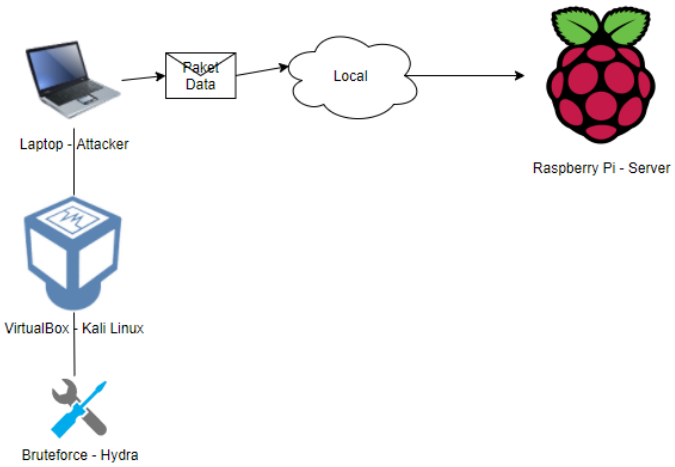
Gambar 5 Arsitektur Sistem integrasi *Snort* ke *Telegram Bot*

Dari Gambar 5 diatas menjelaskan bahwa bagaimana proses integrasi dari hasil deteksi yang telah dilakukan oleh *Snort* yang nantinya dikirimkan sebagai pesan peringatan yang diterima oleh *Telegram Bot* milik *network administrator server*. Proses awalnya yaitu dimulai dengan meminta pada akun *bot* resmi *Telegram* yaitu *@BotFather* dengan cara memasukan perintah untuk meminta *bot* baru hingga mendapatkan *token* dari *bot* yang akan digunakan sebagai media untuk mengirimkan notifikasi kepada *network administrator server*.



Gambar 6 Arsitektur Sistem *Fail2ban*

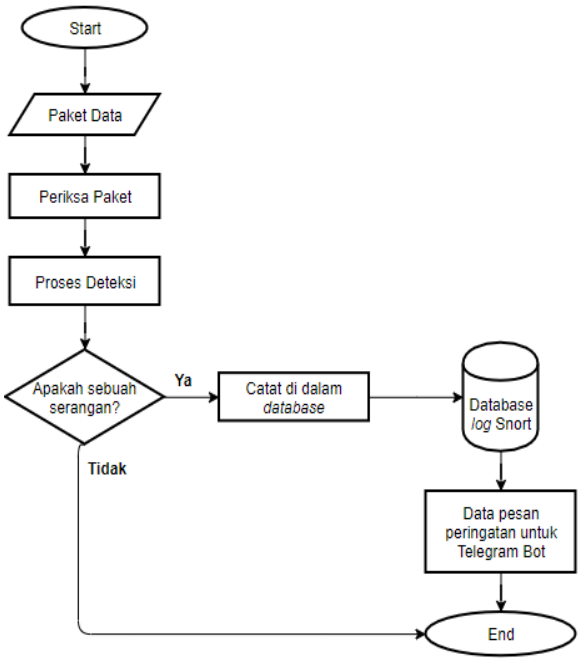
Secara arsitektur sistem yang bisa dilihat pada Gambar 6, *Fail2ban* sendiri konsepnya hampir sama dengan arsitektur yang dimiliki oleh *Snort*, hanya saja fungsi dari *Fail2ban* sendiri yaitu melakukan tindakan pencegahan yang berada pada lalu lintas jaringan yang dianggap tidak wajar dan akan dilakukan tindakan berupa pemblokiran (*ban)* IP *Address* untuk menghentikan serangan yang dilancarkan oleh *attacker*.



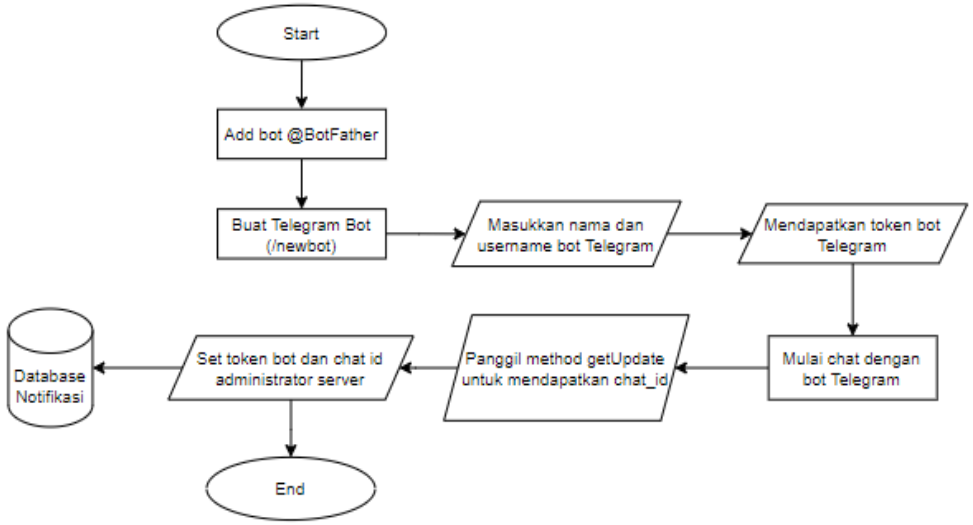
Gambar 7 Arsitektur Sistem & Simulasi Serangan yang dilakukan oleh *Attacker*

Pada Gambar 7 dijelaskan bahwa bagaimana arsitektur sistem dan simulasi yang dilakukan *attacker* untuk melakukan serangan terhadap *Raspberry Pi* sebagai *server* yang akan diserang. Dalam melancarkan serangannya, *attacker* akan mencoba menggunakan *tools* bernama *Hydra* untuk melakukan penyerangan dengan metode *brute-force attack* SSH.

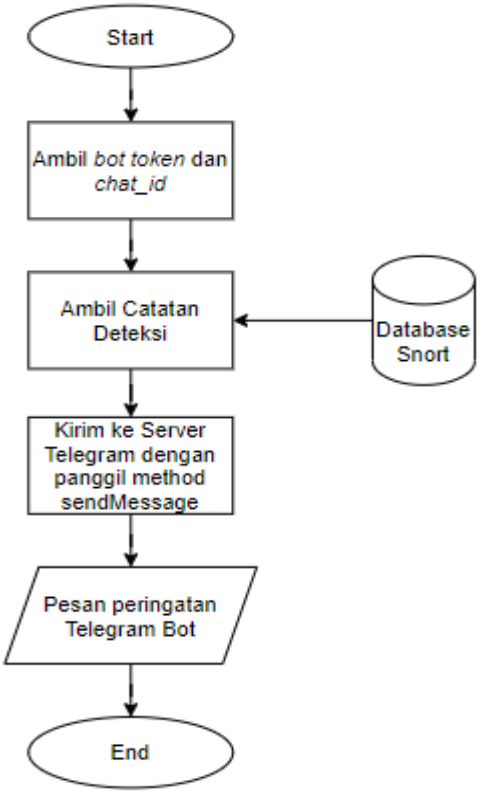
Dibawah ini adalah alur jalannya proses pembuatan dan konfigurasi sistem pada *server* dari mulai desain sistem *Snort*, *Fail2ban*, pembuatan *bot* di *Telegram*, pengiriman pesan peringatan ke *Telegram Bot*, hingga konsep pesan keluaran yang akan dimunculkan dengan menggunakan desain *flowchart*. Untuk desainnya adalah sebagai berikut:



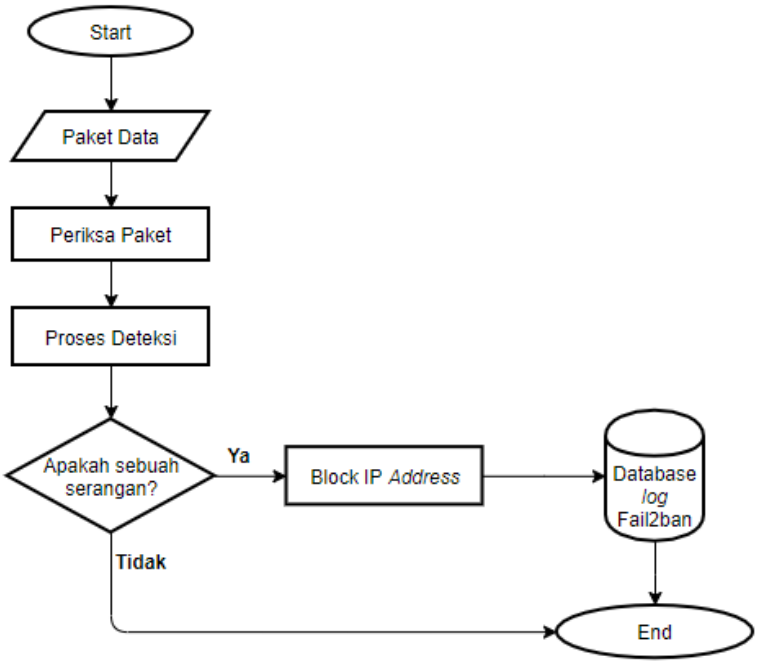
Gambar 8 *Flowchart Snort* sebagai *Intrusion Detection System*



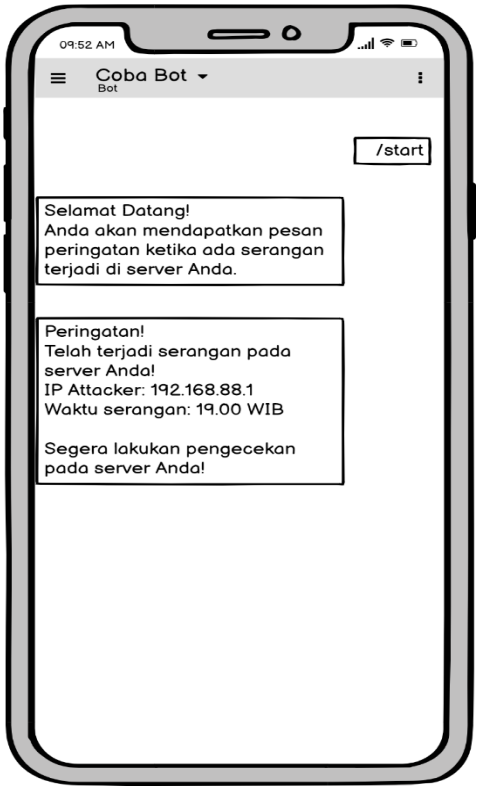
Gambar 9 *Flowchart* Pembuatan *Telegram Bot*



Gambar 10 *Flowchar*t Konfigurasi *Snort* Untuk Melakukan Pengiriman Pesan Peringatan ke *Telegram Bot*



Gambar 11 *Flowchart Fail2ban* sebagai *Intrusion Prevention System*

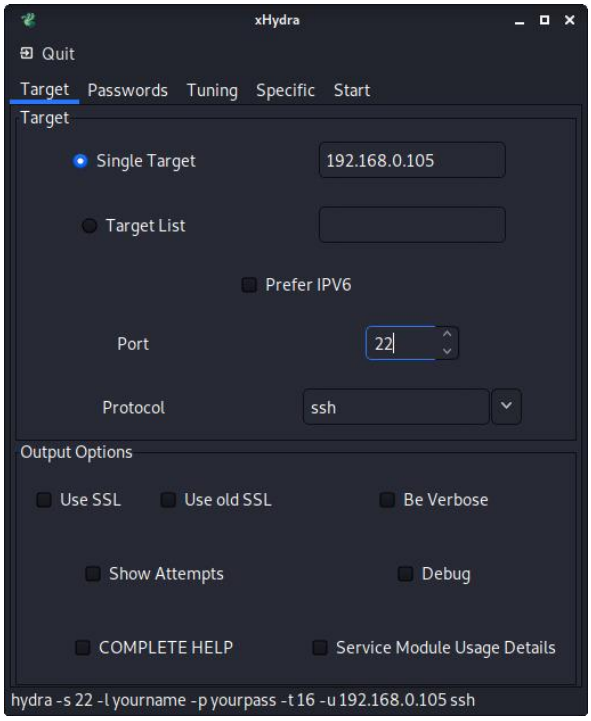


Gambar 12 *Mockup* Konsep Keluaran Pesan Peringatan *Telegram Bot*

BAB IV. PENGGUNAAN FITUR DAN SISTEM

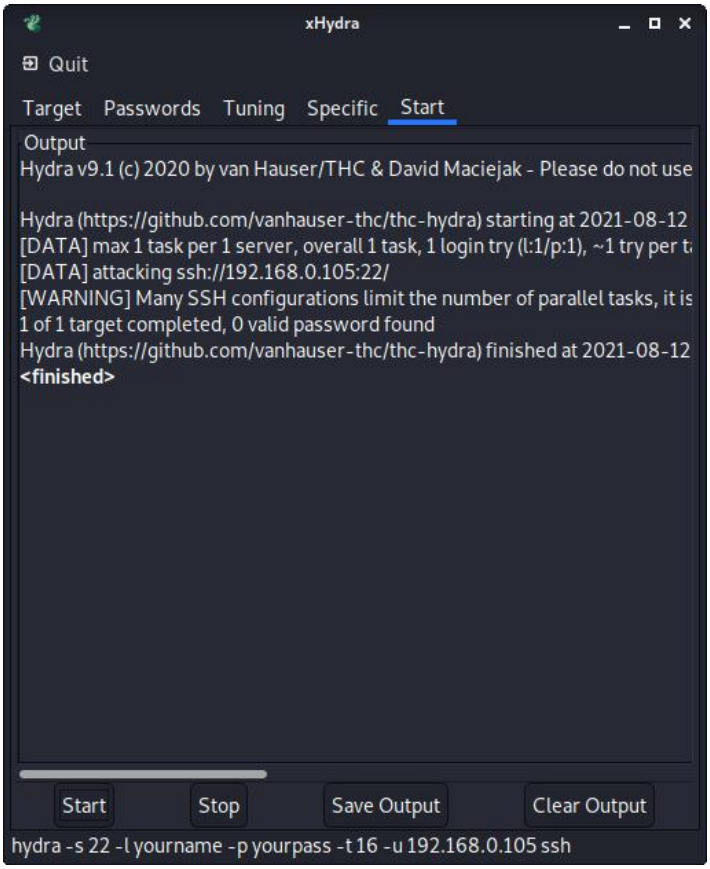
Pada bagian ini menjelaskan tentang alur pengujian secara lengkap dimulai dari proses penyerangan oleh *attacker* sampai hasil jadi dari pengujian *Hydra* berikut tahapan dalam pengujian *Hydra*. Sebelum pengujian menggunakan *Hydra*, dipastikan bahwa *Cowrie Honeypot*, *Snort* dan *Fail2ban* sudah berhasil diterapkan pada *server Raspberry Pi*.

1. Pada gambar 13 akan dijelaskan aplikasi *Hydra* yang ada di sistem operasi *Kali Linux* (*VirtualBox windows* atau *attacker*) untuk melakukan penyerangan. Disini IP *address* tujuan penyerangan *brute-force* SSH oleh *Hydra* adalah 192.168.0.105 (IP *server*) dan menggunakan *port* 22 sebagai protokol yang akan dilakukan penyerangan.



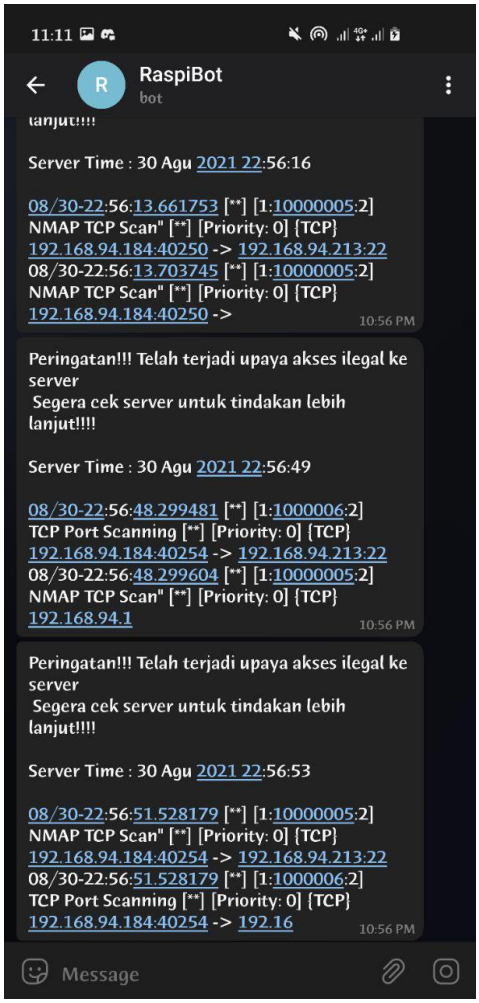
Gambar 13 Tampilan *interface Hydra* pada *Kali Linux*

2. Pada gambar 14 proses *brute-force* SSH pada IP *server* menggunakan *address* 192.168.0.105, sedangkan *Hydra* mendapatkan IP *address* 192.168.0.106. Metode yang digunakan *Hydra* adalah dengan cara melakukan upaya *login* terhadap sistem sasaran sebanyak mungkin sampai menemukan kombinasi *username* dan *password* yang tepat pada target, jika tidak mendapat kombinasi *username* dan *password* tersebut, maka proses *brute-force* akan berjalan dan berhenti dengan tanpa menghasilkan kombinasi *username* dan *password* dari target. Hal ini dipengaruhi oleh seberapa lengkap referensi *username* dan *password* pada *database Hydra*.



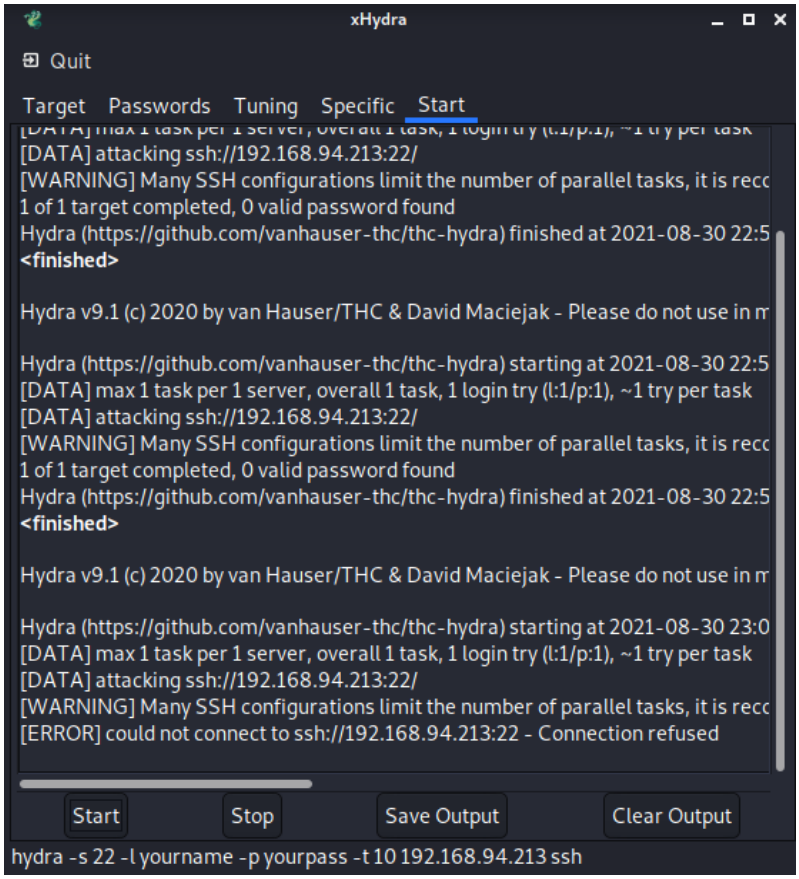
Gambar 14 Proses *Brute-force* SSH IP *Server*

3. Pada gambar 15 *Snort* yang ada pada *server Raspberry Pi* mendeteksi adanya serangan, maka *rules* yang telah dibuat akan dijalankan dan mengirim pesan peringatan ke *Telegram Bot* secara otomatis berdasarkan konfigurasi yang sudah dibuat sebelumnya pada *server Raspberry Pi* dengan dilakukannya pemanggilan *token* dan *chat id* pada *Telegram Bot*. Informasi notifikasi yang dikirim dapat berupa himbauan, IP *address* dan waktu terjadinya serangan.



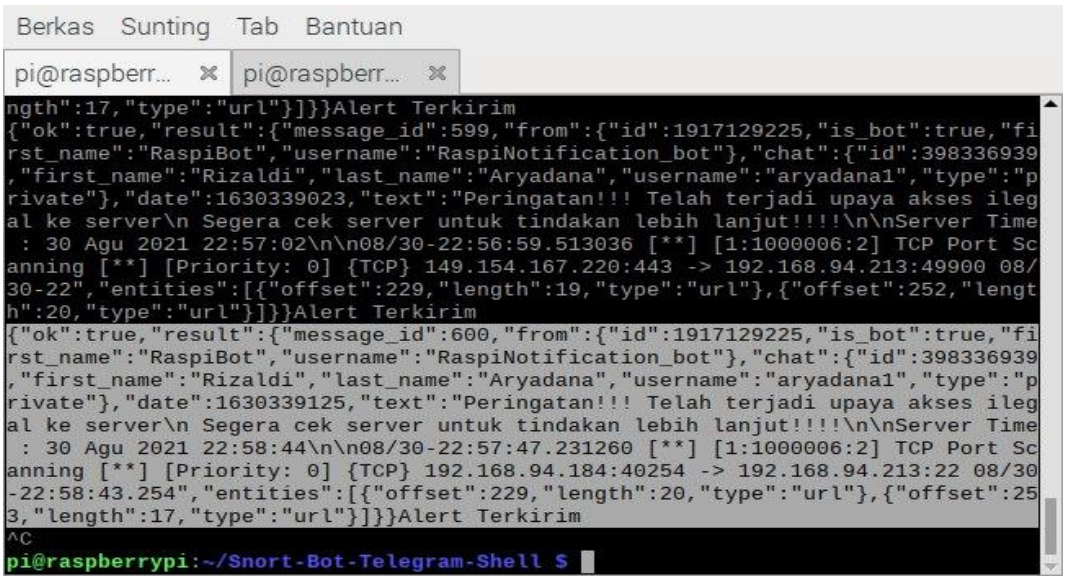
Gambar 15 Pesan Peringatan yang Dimunculkan *Snort* pada *Telegram Bot*

4. Pada gambar 16 akan menjelaskan hasil dari penerapan konfigurasi *Fail2ban* pada *server Raspberry Pi* jika *attacker* mencoba serangan yang berlebihan dan dianggap serangan yang tidak wajar. Jika hal itu terjadi, maka IP *address attacker* akan otomatis dilakukan pemblokiran (*ban*) untuk mencegah adanya percobaan serangan serupa oleh *attacker*.



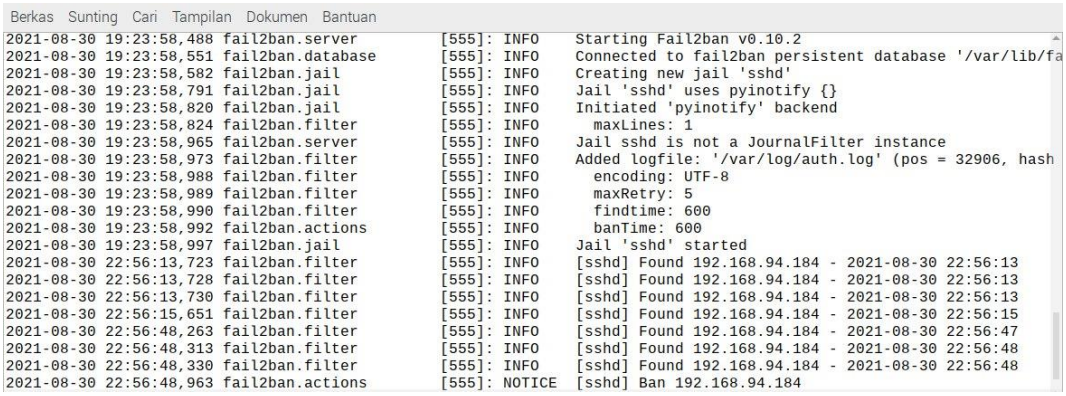
Gambar 16 Hasil Tindakan Pencegahan Yang Dilakukan oleh *Fail2ban*

5. Pada gambar 17 ditunjukkan saat terjadi penyerangan, *terminal* yang dijalankan sebagai bentuk integrasi *Snort* dan *Telegram Bot* memunculkan *log* atau pencatatan aktifitas paket data yang telah dideteksi.



Gambar 17 *Log* dari konfigurasi Integrasi Antara *Snort* dan *Telegram Bot*

6. Pada gambar 18 akan ditunjukkan *log* pada *Fail2ban* jika berhasil melakukan tindakan pencegahan berupa pemblokiran (*ban*) IP *address attacker* yang melakukan penyerangan.



Gambar 18 *Log Fail2ban* Untuk Melihat Pemblokiran IP *Address Attacker*