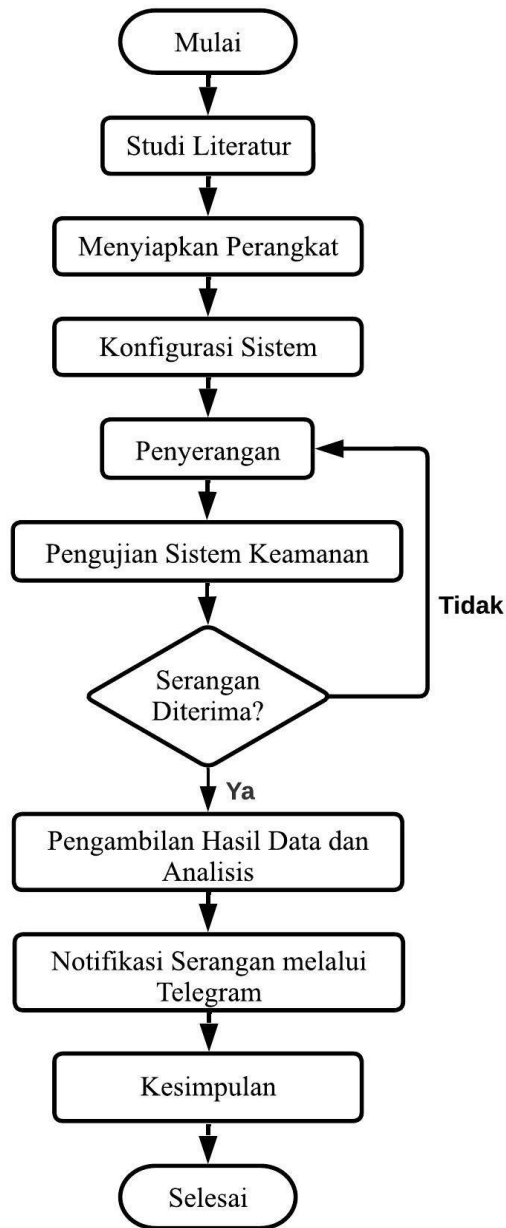


## BAB III METODE PENELITIAN

### 3.1 Alur Penelitian

Selama penelitian ini dilaksanakan, beberapa langkah penting dijalankan untuk memastikan hasil yang akurat dan konsisten. Runtutan tahapan yang diikuti oleh penulis dalam menyusun penelitian ini direpresentasikan dalam Gambar 3.1 yang menunjukkan *flowchart* dari proses tersebut.



**Gambar 3.1** *Flowchart* Alur Penelitian

### 3.1.1 Studi Literatur

Berdasarkan *flowchart* pada Gambar 3.1, tahap pertama dalam penelitian ini adalah melakukan studi literatur atau tinjauan pustaka, yang melibatkan pencarian sumber-sumber yang relevan. Pada tahap ini, penekanan diberikan pada kebutuhan studi literatur yang akan memberikan dukungan bagi pengetahuan teoritis dan praktis terkait penggunaan simulasi yang diinginkan. Studi literatur ini dilakukan berdasarkan tema yang dikembangkan dan mencakup literatur yang relevan dari penelitian sebelumnya yang berhubungan dengan masalah serupa, serta mencakup buku, jurnal, artikel, dan sumber *online* yang relevan mengenai bahasan *Intrusion Prevention System* berbasis Suricata dan berbagai jenis serangan jaringan.

### 3.1.2 Menyiapkan Perangkat

Dalam tahap persiapan perangkat, dibutuhkan alat dan komponen sistem yang disesuaikan dengan kebutuhan implementasi dari skema yang akan dikembangkan. Komponen tersebut terdiri dari perangkat keras (*hardware*) dan perangkat lunak (*software*). Setiap perangkat yang akan digunakan memiliki spesifikasi yang perlu dipertimbangkan, termasuk:

#### a. Spesifikasi *Hardware*

Berikut adalah spesifikasi perangkat keras yang akan diimplementasikan untuk menjaga keamanan jaringan dari serangan, sebagaimana tertera dalam Tabel 3.1 di bawah ini:

**Tabel 3.1 Spesifikasi *Hardware***

No	Perangkat	Spesifikasi	Fungsi	Jumlah
1	Laptop	Prosesor Intel <i>Core</i> i5- 6200U (2 CPUs), RAM 2 GB, <i>Hard disk</i> 80 GB	<i>Webserver</i> Suricata menggunakan sistem operasi Ubuntu 20.04 LTS.	1
2	Laptop	Prosesor Intel <i>Core</i> i3- 6100U (2 CPUs), RAM 2 GB, <i>Hard disk</i> 80 GB	<i>Attacker</i> menggunakan sistem operasi Ubuntu 20.04 LTS.	1

**Tabel 3.1 Spesifikasi Hardware (lanjutan)**

No	Perangkat	Spesifikasi	Fungsi	Jumlah
3	Laptop	Prosesor Intel <i>Core</i> i5- 6200U (2 CPUs), RAM 2 GB, <i>Hard disk</i> 80 GB	<i>Normal user</i> menggunakan sistem operasi Ubuntu 20.04 LTS.	1
4	<i>Smartphone</i>	Xiaomi Poco X3, RAM 6 GB, eMMC 64 GB	Mendapatkan <i>alert</i> melalui Telegram <i>bot</i> .	1

b. Spesifikasi *Software*

Di bawah ini terdapat spesifikasi perangkat lunak yang akan digunakan untuk mengamankan jaringan dari serangan, seperti yang ditunjukkan dalam Tabel 3.2 berikut:

**Tabel 3.2 Spesifikasi Software**

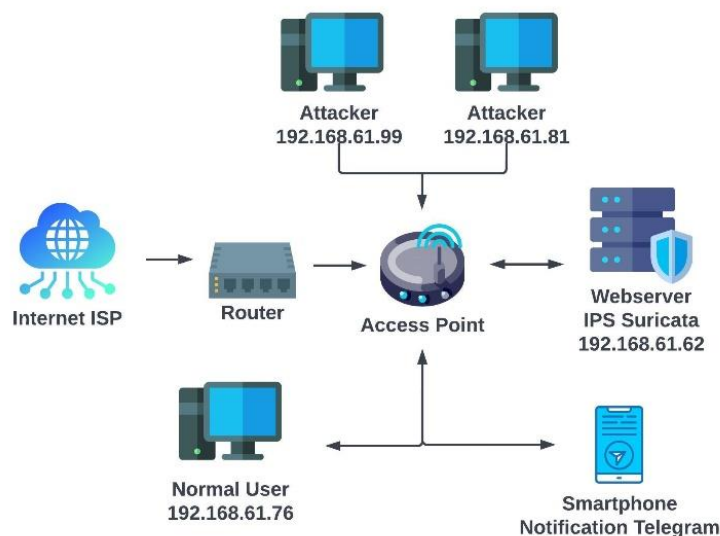
No.	<i>Software</i>	Fungsi
1.	Ubuntu 20.04 LTS	Sistem operasi penyerang, <i>webserver</i> dan <i>normal user</i>
2.	Suricata	<i>Instrusion Prevention System (IPS)</i>
3.	PHP 8.1	Bahasa pemograman DVWA
4.	DVWA	<i>Platform webserver</i>
5.	Hydra	<i>Tool</i> serangan <i>Bruteforce</i>
6.	SQLMap	<i>Tool</i> serangan <i>SQL Injection</i>
7.	Pyhton 2.7	Bahasa pemograman SQLMap dan program notifikasi Telegram <i>bot</i>
8.	Hping3	<i>Tool</i> serangan <i>ICMP flood</i>
9.	Httpperf	<i>Tool</i> mengukur perfomasi <i>webserver</i>
10.	API Telegram <i>bot</i>	Mengirimkan <i>alert</i> Suricata ke pesan Telegram <i>bot administrator</i>
11.	Telegram	Penerima <i>alert</i> Suricata

### 3.1.3 Konfigurasi Sistem

Tahap konfigurasi sistem dalam konteks ini mencakup beberapa aspek penting yang perlu diperhatikan. Pertama, perencanaan topologi sistem yang akan digunakan, termasuk pemilihan dan penempatan komponen-komponen yang relevan. Selanjutnya, langkah-langkah meng-*install* dan mengatur sistem IPS/IDS pada server komputer yang melibatkan proses instalasi perangkat lunak, konfigurasi jaringan, dan pengaturan keamanan yang sesuai. Selain itu, tahap konfigurasi juga mencakup pengoperasian sistem IPS/IDS secara efektif, termasuk pengawasan dan pemantauan kinerja sistem, pembaruan aturan-aturan yang digunakan, serta manajemen *log* aktivitas. Seluruh konfigurasi ini juga melibatkan pengaturan komponen pendukung lainnya, seperti *firewall*, *router* dan perangkat jaringan lainnya untuk memastikan integrasi yang baik dan kehandalan sistem secara keseluruhan.

#### a. Perancangan Topologi Jaringan

Topologi jaringan yang dipilih adalah langkah penentuan struktur jaringan yang sesuai dengan sistem yang akan dikembangkan, sehingga memberikan gambaran yang jelas tentang sistem yang akan dibangun. Pada Gambar 3.2, topologi yang diterapkan dalam penelitian ini adalah topologi star yang membentuk jaringan LAN. Dalam topologi ini, implementasi Suricata diterapkan pada *webservice*, atau biasa disebut sebagai *Host-based Intrusion Prevention System (HIPS)*.

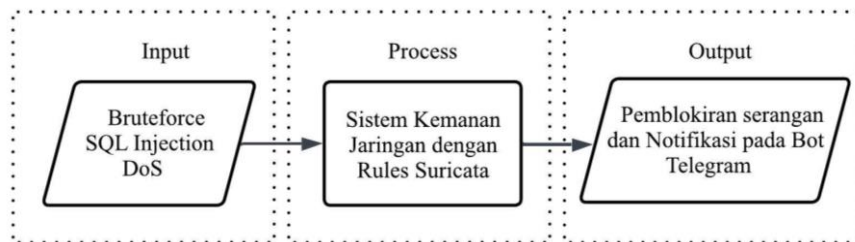


**Gambar 3.2 Topologi Jaringan**

Dengan mempertimbangkan topologi jaringan yang telah direncanakan sebelumnya, sebuah diagram blok dibuat untuk menggambarkan interaksi sistem, seperti yang terlihat pada Gambar 3.3. Bagian *input* dari diagram blok ini mencerminkan serangan yang dilakukan oleh penyerang terhadap *webserver*. Implementasi sistem IPS Suricata pada *webserver* memungkinkan IPS untuk secara otomatis mengambil tindakan sesuai dengan aturan (*rules*) yang telah dikonfigurasi sebelumnya di Suricata.

*Output* dari sistem ini mencakup pemblokiran paket data yang dikirimkan oleh penyerang, serta notifikasi yang dikirimkan kepada *administrator* jaringan melalui Telegram *bot*. Dengan adanya notifikasi tersebut, *administrator* jaringan dapat dengan cepat mengetahui tentang serangan yang terjadi dan mengambil tindakan yang diperlukan.

Melalui diagram blok ini, dapat dengan jelas dipahami bagaimana sistem IPS Suricata berperan dalam mendeteksi dan merespons serangan yang ditujukan pada *webserver*, serta memberikan perlindungan dan pemantauan yang efektif terhadap jaringan.



**Gambar 3.3 Blok Diagram**


#### b. Konfigurasi IPS

Dalam konteks ini, server menggunakan sistem IPS Suricata pada jaringan LAN dan akan dikonfigurasi sebelum digunakan sebagai sistem keamanan untuk melindungi dari serangan *Bruteforce*, *SQL Injection*, dan *DoS*. Berikut adalah tahapan konfigurasi IPS Suricata yang akan dilakukan:

##### 1) Instalasi Suricata

Sebelum melakukan instalasi, pastikan komputer terhubung ke internet dan sudah berada dalam mode *superuser* dengan menggunakan perintah "sudo su". Mode *superuser* memberikan hak akses yang tidak terbatas untuk melakukan perubahan sistem. Proses instalasi dimulai dengan mengunduh

*file* Suricata dari situs resmi Suricata di <https://suricata.io/>. Kemudian, gunakan perintah "*apt-get install suricata*" pada sistem operasi Linux Ubuntu untuk menginstal Suricata, seperti Gambar 3.4.

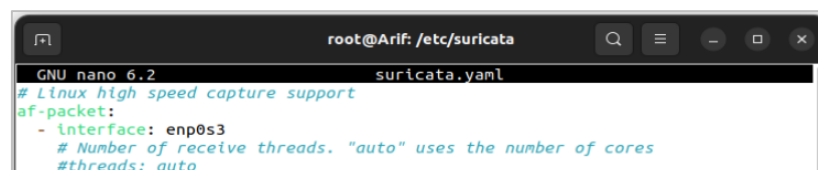


```
root@Arif:~/home/arif# apt-get install suricata
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
suricata is already the newest version (1:7.0.0-0ubuntu6).
The following packages were automatically installed and are no longer required:
  libnetfilter-log1 linux-image-5.19.0-35-generic
  linux-modules-5.19.0-35-generic linux-modules-extra-5.19.0-35-generic
  oinkmaster python3-simplejson snort-rules-default suricata-update
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 125 not upgraded.
root@Arif:~/home/arif#
```

**Gambar 3.4 Install Suricata**

## 2) Konfigurasi Jaringan Suricata

Dalam proses penggunaan Suricata pada jaringan wireless LAN, diperlukan penyesuaian kondisi *interface* yang awalnya menggunakan eth0 untuk diubah menjadi enp0s3. Penyesuaian ini dilakukan melalui pengeditan *file* "suricata.yaml" yang terletak dalam direktori "etc/suricata/". Dengan melakukan penyesuaian ini, Suricata dapat beroperasi secara optimal dalam jaringan wireless LAN. Gambar 3.5 merupakan gambaran visual yang memperlihatkan lokasi *file* dan proses penyesuaian yang dilakukan. Penyesuaian ini penting agar Suricata dapat berfungsi dengan baik dalam lingkungan jaringan nirkabel yang dituju.

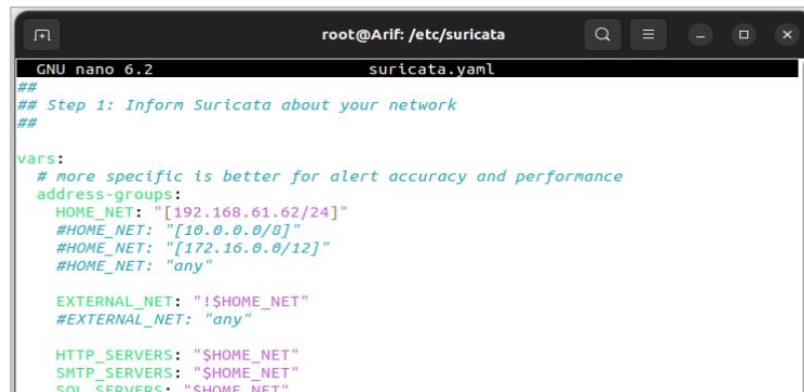


```
root@Arif: /etc/suricata
GNU nano 6.2 suricata.yaml
# Linux high speed capture support
af-packet:
- interface: enp0s3
# Number of receive threads. "auto" uses the number of cores
#threads: auto
root@Arif: /etc/suricata
```

**Gambar 3.5 Penyesuaian Jaringan Suricata**

Selain itu, konfigurasi lain yang dilakukan melibatkan penyesuaian pada beberapa variabel penting. Salah satu penyesuaian ini terjadi pada variabel IP *address* HOME\_NET, yang mengacu pada alamat jaringan yang akan dilindungi oleh IPS Suricata. Dalam konteks ini, IP *address* HOME\_NET diatur agar sesuai dengan lingkungan jaringan yang sedang dianalisis, pada penelitian ini, IP *address* HOME\_NET diatur sebagai 192.168.61.62 seperti yang terlihat pada Gambar 3.6. Penyesuaian ini memungkinkan Suricata

untuk memahami bahwa jaringan dengan alamat tersebut adalah jaringan yang perlu diawasi dan dilindungi dari potensi ancaman.



```
root@Arif: /etc/suricata
GNU nano 6.2 suricata.yaml
##
## Step 1: Inform Suricata about your network
##
vars:
# more specific is better for alert accuracy and performance
address-groups:
HOME_NET: "[192.168.61.62/24]"
#HOME_NET: "[10.0.0.0/8]"
#HOME_NET: "[172.16.0.0/12]"
#HOME_NET: "any"

EXTERNAL_NET: "!$HOME_NET"
#EXTERNAL_NET: "any"

HTTP_SERVERS: "$HOME_NET"
SMTP_SERVERS: "$HOME_NET"
SQL_SERVERS: "$HOME_NET"
```

Gambar 3.6 Konfigurasi *HOME\_NET*

### 3) Konfigurasi *Rules* Suricata

Selanjutnya, langkah berikutnya adalah melakukan konfigurasi pada sistem Suricata, yang bertujuan untuk mengatur dan menambahkan aturan-aturan yang akan digunakan dalam operasional Suricata. Setiap aturan akan ditulis dalam beberapa *file* yang disimpan di direktori `/etc/suricata/rules`. Untuk serangan *Bruteforce*, aturan-aturan akan dibuat dalam *file* `"Bruteforce.rules"` dengan aturan berikut:

```
drop http any any -> $HOME_NET any (msg:"DVWA Brute Force Attempt"; flow:established,to_server; content:"GET"; http_method; content:"/DVWA/vulnerabilities/brute/index.php"; http_uri; threshold:type threshold, track by_src, count 10, seconds 20; sid:100002; rev:1;)
```

Kolom diatas adalah aturan yang digunakan untuk mendeteksi dan memblokir serangan *Bruteforce*, dengan penjelasan sebagai berikut:

- *drop* : Tindakan yang diambil IPS berupa ditolak
- *http* : Protokol yang digunakan untuk *rules* ini
- *any (Source IP)* : Semua IP sumber
- *any (Source Port)* : Semua *port* sumber
- *\$HOME\_NET* : IP *address* tujuan
- *msg* : Pesan yang disampaikan oleh Suricata

- *flow* : Aliran paket yang telah dibangun (*established*) dan bergerak ke server (*to\_server*).
- *Content GET* : Mencari paket dengan metode HTTP "GET"
- *Content URI* : Mencari URI HTTP
- *Threshold* : Aktivasi *rules* ketika ada 10 permintaan yang cocok dari alamat IP sumber tertentu dalam waktu 20 detik
- *sid* : *Signature ID*
- *rev* : Nomor revisi aturan ini adalah 1

Aturan untuk serangan *SQL Injection* akan ditulis dalam file "*SQLInjection.rules*" dengan aturan berikut:

```
drop http any any -> $HOME_NET $HTTP_PORTS (msg:"ET
WEBSERVER Possible SQL Injection Attempt UNION SELECT";
flow:established,to_server; content:"UNION"; http_uri; nocase;
content:"SELECT"; http_uri; nocase; pcre:"/UNION.+SELECT/U";
reference:url,doc.emergingthreats.net/2006446; classtype:web-
application-attack; sid:2006446; rev:13;)

drop tcp any any -> $HOME_NET $HTTP_PORTS (msg:"SQL Injection
Detected"; flow:established,to_server; content:"id"; nocase; http_uri;
pcre:"/(and\W+select)|(union.*select)|(or|and\d+=\d+)|(\'.\-\-)/Ui";
classtype:web-application-attack; sid:1000005; rev:1;)
```

Kolom di atas adalah aturan yang digunakan untuk mendeteksi dan memblokir serangan *Bruteforce*, dengan penjelasan sebagai berikut:

- *drop* : Tindakan yang diambil IPS berupa ditolak
- *tcp* : Protokol yang digunakan untuk *rules* ini
- *http* : Protokol yang digunakan untuk *rules* ini
- *any (Source IP)* : Semua IP sumber
- *any (Source Port)* : Semua *port* sumber
- *\$HOME\_NET* : IP tujuan



- *\$HTTP\_PORTS* : *Port* tujuan HTTP
- *msg* : Pesan yang disampaikan oleh Suricata
- *flow* : Aliran paket yang telah dibangun (*established*) dan bergerak ke server (*to\_server*).
- *Content UNION* : Mencari kata *UNION* dengan pencocokan yang tidak memperhatikan besar kecil huruf (*nocase*)
- *Content SELECT* : Mencari kata *SELECT* dengan pencocokan yang tidak memperhatikan besar kecil huruf (*nocase*)
- *Content URI* : Mencari kata id dengan pencocokan yang tidak memperhatikan besar kecil huruf (*nocase*)
- *PCRE* : Mencocokkan urutan "*UNION*" diikuti oleh "*SELECT*" secara bersamaan tanpa memperhatikan besar kecil huruf.
- *Threshold* : Aktivasi *rules* ketika ada 10 permintaan yang cocok dari alamat IP sumber tertentu dalam waktu 20 detik
- *Refrence* : Menyertakan referensi URL yang berkaitan dengan *rule* ini
- *Classtype* : Menandakan jenis serangan,
- *SID* : *Signature ID*
- *rev* : Nomor revisi aturan ini adalah 13

Aturan untuk serangan ICMP *flood* akan diimplementasikan dalam *file* "*DoS.rules*" dengan aturan berikut:

<pre>drop icmp 192.168.61.81 any -&gt; 192.168.61.62 any (msg:"ICMP Ping Detected"; itype:8; sid:100001; rev:1;)</pre>
<pre>drop icmp 192.168.61.99 any -&gt; 192.168.61.62 any (msg:"ICMP Ping Detected"; itype:8; sid:100001; rev:1;)</pre>

Kolom diatas adalah aturan yang digunakan untuk mendeteksi dan memblokir serangan *Bruteforce*, dengan penjelasan sebagai berikut:

- *drop* : Tindakan yang diambil IPS berupa ditolak
- *icmp* : Protokol yang digunakan untuk *rules* ini
- 192.168.61.81 : *Source IP*
- *any (Source Port)* : Semua *port* sumber
- 192.168.61.62 : IP tujuan
- *msg* : Pesan yang disampaikan oleh Suricata
- *ICMP Type* : Tipe pesan *Echo Request*, yaitu permintaan untuk ping
- *sid* : *Signature ID*
- *rev* : Nomor revisi aturan ini adalah 1

#### 4) Konfigurasi IPTables

Dengan melakukan konfigurasi IPTables, manfaatnya adalah untuk mengontrol paket yang masuk ke sistem operasi dari luar, contohnya saat seorang *administrator* menggunakan SSH untuk terhubung ke mesin Linux dari mesin lain. Konfigurasi IPTables dapat memberikan izin koneksi untuk *port* SSH, HTTP, dan HTTPS dengan aturan berikut:

```
sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT
sudo iptables -A INPUT -p tcp -m tcp --dport 80 -j ACCEPT
sudo iptables -A INPUT -s 192.168.61.81/32 -p icmp -m icmp -j DROP
```

#### 5) Instalasi Apache2

Untuk menyajikan konten *website*, sebuah server memerlukan aplikasi *webserver*, seperti Apache2. Proses instalasi Apache2 dapat dilakukan seperti yang ditunjukkan pada Gambar 3.7. Setelah instalasi berhasil, gunakan alamat IP *webserver* (192.168.61.62) atau *localhost* (127.0.0.1) untuk memeriksa keberhasilan instalasi Apache2 yang telah dilakukan, seperti yang terlihat pada Gambar 3.8 yang menampilkan halaman *default* Apache2.

```
root@Arif:/home/arif# apt-get install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apache2 is already the newest version (2.4.52-1ubuntu4.6).
The following packages were automatically installed and are no longer required:
 libnetfilter-log1 linux-image-5.19.0-35-generic
 linux-modules-5.19.0-35-generic linux-modules-extra-5.19.0-35-generic
 oinkmaster python3-simplejson snort-rules-default suricata-update
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 125 not upgraded.
root@Arif:/home/arif#
```

Gambar 3.7 Instalasi Apache2



Gambar 3.8 Apache2 Default Page

## 6) Konfigurasi DVWA

*Damn Vulnerable Web App (DVWA)* adalah sebuah *platform* yang sengaja dibuat untuk membantu pengembang *web* memahami proses keamanan aplikasi *web*. DVWA menggunakan bahasa pemrograman PHP dan MySQL sebagai *database*. Untuk menginstal DVWA, perlu dilakukan instalasi *software* PHP terlebih dahulu dengan menggunakan perintah "apt-get install php8.1" seperti yang ditunjukkan Gambar 3.9.

```
root@Arif:/home/arif# apt-get install php8.1
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
php8.1 is already the newest version (8.1.2-1ubuntu2.13).
The following packages were automatically installed and are no longer required:
 libnetfilter-log1 linux-image-5.19.0-35-generic
 linux-modules-5.19.0-35-generic linux-modules-extra-5.19.0-35-generic
 oinkmaster python3-simplejson snort-rules-default suricata-update
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 125 not upgraded.
root@Arif:/home/arif#
```

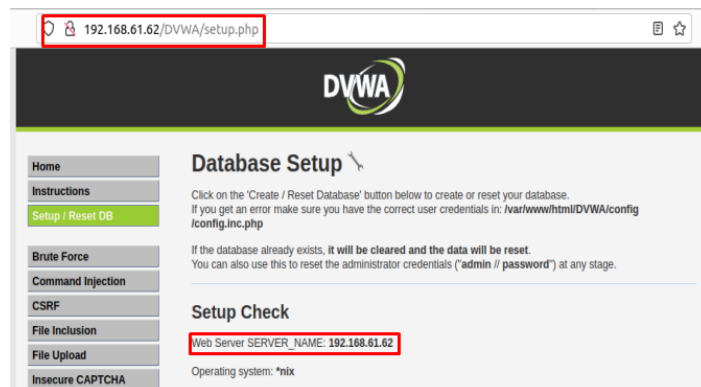
Gambar 3.9 Instalasi PHP 8.1

Dalam proses instalasi DVWA sebagai *webserver*, perlu dilakukan penyesuaian lokasi direktori yang sesuai, yaitu di "/var/www/". Untuk *download* dan menginstal DVWA, digunakan forum resmi Github dengan menggunakan perintah "git clone https://github.com/digininja/DVWA.git" seperti yang terlihat pada Gambar 3.10.

```
root@Arif: /var/www
root@Arif:/home/arif# cd /var/www/
root@Arif:/var/www# git clone https://github.com/digininja/DVWA.git
Cloning into 'DVWA'...
remote: Enumerating objects: 4318, done.
remote: Counting objects: 100% (96/96), done.
remote: Compressing objects: 100% (78/78), done.
remote: Total 4318 (delta 31), reused 65 (delta 15), pack-reused 4222
Receiving objects: 100% (4318/4318), 2.14 MiB | 3.13 MiB/s, done.
Resolving deltas: 100% (2031/2031), done.
root@Arif:/var/www#
```

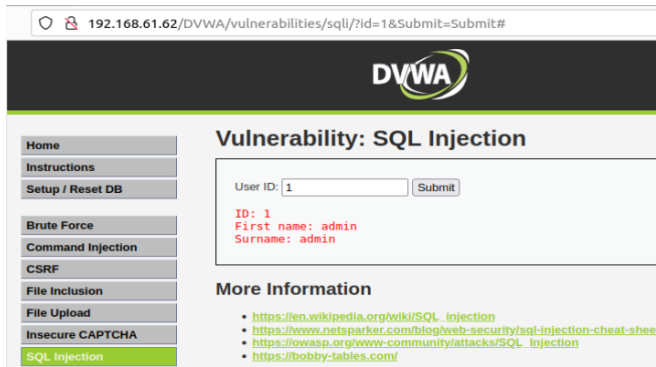
**Gambar 3.10 Instalasi DVWA**

Setelah serangkaian proses instalasi dan konfigurasi DVWA berhasil dilakukan, tampilan DVWA akan muncul dan terlihat seperti yang ditampilkan pada Gambar 3.11. Penting untuk memastikan bahwa alamat IP yang digunakan oleh *webserver* DVWA sesuai dengan alamat IP yang telah dikonfigurasi sebelumnya pada Suricata. Dengan melakukan pencocokan antara kedua alamat IP ini, memastikan bahwa *webserver* DVWA dan Suricata terhubung dengan benar dan dapat berkomunikasi secara efektif.

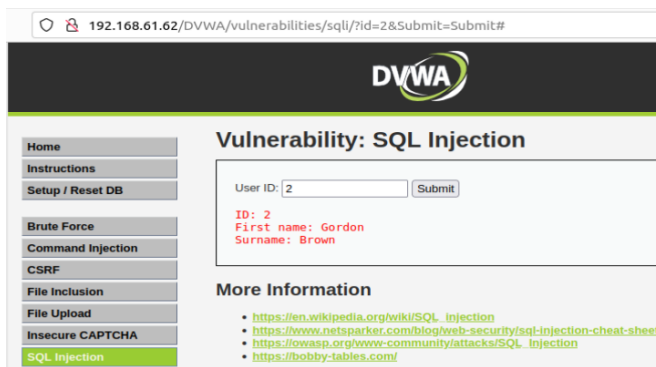


**Gambar 3.11 Tampilan DVWA**

Pada tahap ini, dilakukan injeksi data yang dirancang secara khusus ke dalam *database* yang digunakan oleh *web* DVWA (*Damn Vulnerable Web Application*) yang menjadi subjek penelitian. Proses injeksi ini melibatkan berbagai *query* yang dirancang untuk mengeksplorasi dan mengeksploitasi potensi kerentanan pada aplikasi. Proses injeksi tersebut terbagi dalam beberapa tahap, dan dalam konteks pengujian ini, fokusnya adalah pada *User ID*. Beberapa contoh *query* diaplikasikan pada tahap pengujian, dimulai dari *query* 1 hingga *query* 5. *Query-query* ini diformulasikan secara khusus untuk menguji sejauh mana keamanan dan kerentanan sistem terhadap serangan *SQL Injection*. Gambar 3.12 sampai Gambar 3.16 memvisualisasikan contoh-contoh *query* yang diaplikasikan.



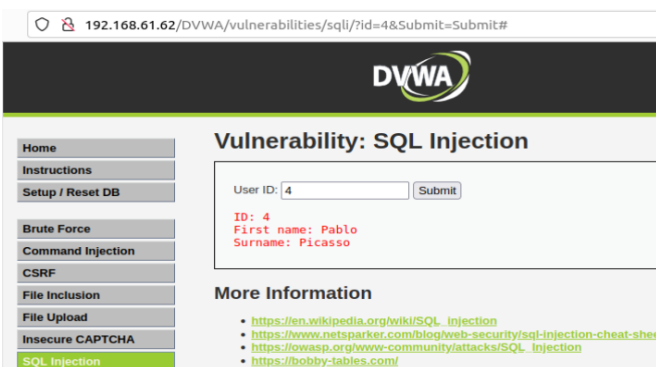
Gambar 3.12 *Input Query '1'*



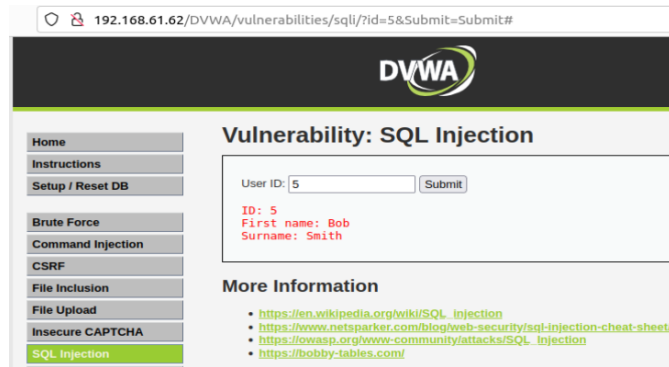
Gambar 3.13 *Input Query '2'*



Gambar 3.14 *Input Query '3'*



Gambar 3.15 *Input Query '4'*



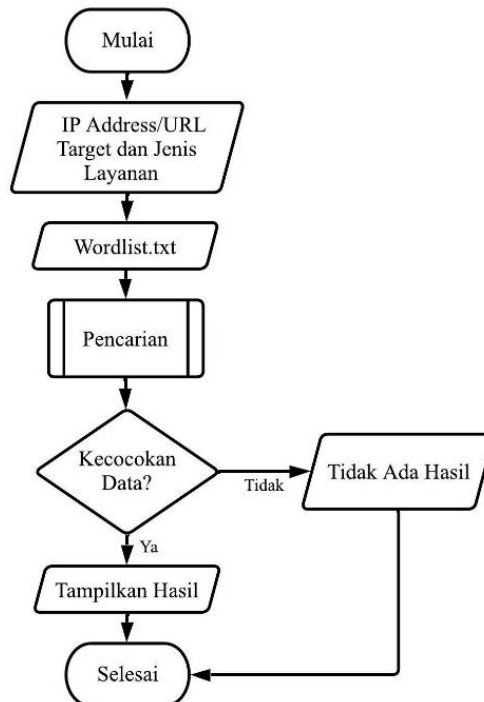
Gambar 3.16 *Input Query '5'*

### 3.1.4 Penyerangan

Sebelum melakukan tahap pengujian, perangkat penyerang harus dipersiapkan dengan melakukan instalasi dan konfigurasi beberapa komponen, antara lain:

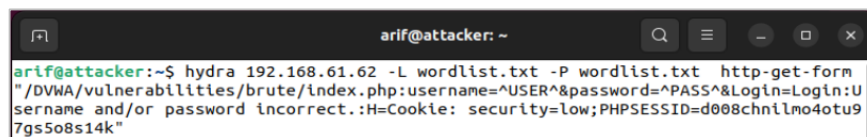
#### a. Instalasi Hydra

*Tool* Hydra dimaksudkan sebagai uji coba penyerangan HTTP *Bruteforce*, dan dapat diunduh menggunakan perintah "*apt-get install hydra*". Sebelum digunakan sebagai alat penyerang, penting untuk memahami alur kerja dari serangan *Bruteforce* yang akan diterapkan, yaitu jenis *dictionary attacks*.



Gambar 3.17 Alur *Dictionary Attacks*

Gambar 3.17 menggambarkan alur proses dalam serangan *Bruteforce* dengan metode *dictionary attacks*. Proses dimulai dengan penyerang menyiapkan *IP address* target dan sebuah *file wordlist* yang berisi kumpulan *username* dan *password* dalam format *.txt*. Selanjutnya, aplikasi yang digunakan untuk serangan melakukan pencarian *username* dan *password* dengan melakukan pemindaian pada *website* target. Jika pencarian berhasil, aplikasi akan menampilkan hasil yang sesuai. Namun, jika tidak berhasil, aplikasi tidak akan memberikan hasil. Proses serangan ini dilakukan melalui terminal Linux dengan konfigurasi yang telah disiapkan untuk menguji coba penyerangan terhadap *webserver* DVWA, seperti yang ditunjukkan pada Gambar 3.18.



```
arif@attacker: ~  
arif@attacker:~$ hydra 192.168.61.62 -L wordlist.txt -P wordlist.txt http-get-form  
"/DVWA/vulnerabilities/brute/index.php:username=^USER^&password=^PASS^&Login=Login:U  
sername and/or password incorrect.:H=Cookie: security=low;PHPSESSID=d008chnlno4otu9  
7gs5o8s14k"
```

**Gambar 3.18 Perintah HTTP Bruteforce**

Keterangan:

- **-L** : Kumpulan *username* unik yang digunakan *Bruteforce*
- **-P** : Kumpulan *password* unik yang digunakan *Bruteforce*
- **Service** : Jenis layanan yang akan diserang
- **Host** : Tipe target serangan
- **Target URI** : Target URL
- **Security** : Tingkat keamanan DVWA
- **PHPSESID** : Identifikasi unik yang digunakan pengguna

#### b. Instalasi SQLMap

Untuk menggunakan SQLMap pada sistem operasi Ubuntu, perlu adanya pengunduhan perangkat lunaknya dengan perintah "*apt-get install sqlmap*" seperti Gambar 3.19. Setelah berhasil diinstal, pengguna dapat memeriksa apakah SQLMap berfungsi dengan baik atau tidak dengan menjalankan perintah "*SQLmap -h*". Jika *command prompt* menampilkan informasi seperti yang ditunjukkan pada Gambar 3.20, itu berarti instalasi dan konfigurasi SQLMap telah berhasil dilakukan.





c. Instalasi Hping3

Untuk melakukan pengujian serangan *Denial of Service* (DoS) dapat menggunakan alat Hping3 yang berfungsi untuk melumpuhkan server. Pertama, lakukan pengunduhan Hping3 dengan mengetikkan perintah "*apt-get install hping3*" seperti Gambar 3.22. Setelah diinstal, lakukan konfigurasi dengan menentukan target *IP address* yang akan digunakan sebagai *webserver*, memilih mode DoS yang akan dijalankan, dan menentukan jumlah packet data yang akan dikirimkan sebagai serangan. Dengan melakukan konfigurasi ini, dapat menguji sejauh mana tingkat ketahanan *webserver* terhadap serangan DoS seperti Gambar 3.23.

```
root@attacker:/home/arif# apt-get install hping3
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
hping3 is already the newest version (3.a2.ds2-10).
0 upgraded, 0 newly installed, 0 to remove and 141 not upgraded.
root@attacker:/home/arif#
```

Gambar 3.22 Instalasi Hping3

```
root@attacker:/home/arif# hping3 192.168.61.62 --icmp --flood
HPING 192.168.61.62 (enp0s3 192.168.61.62): icmp mode set, 28 headers + 0 data
bytes
hping in flood mode, no repltes wll be shown
```

Gambar 3.23 Perintah ICMP Flood

Keterangan:

- --icmp : Jenis serangan DoS
- --flood : Mengirimkan serangan sebanyak mungkin

### 3.1.5 Pengujian Sistem Keamanan

Penelitian ini bertujuan untuk mengimplementasikan skema pengujian arsitektur keamanan *webserver*. Skema pengujian *webserver* dibagi menjadi dua bagian.

Tabel 3.3 Skenario Serangan Pengujian

Pengirim	Skenario Serangan	Kondisi IPS	Parameter
Attacker (192.168.61.81)	Bruteforce	Tidak Aktif	Efektivitas <i>Rule</i> ,
		Aktif	Penggunaan CPU dan
	SQL Injection	Tidak Aktif	<i>memory</i> serta <i>response</i>
		Aktif	<i>time</i>

Pengirim	Skenario Serangan	Kondisi IPS	Parameter
	ICMP <i>Flood</i>	Tidak Aktif	
		Aktif	
<i>Attacker</i> (192.168.61.99)	ICMP <i>Flood</i>	Tidak Aktif	Penggunaan CPU
		Aktif	

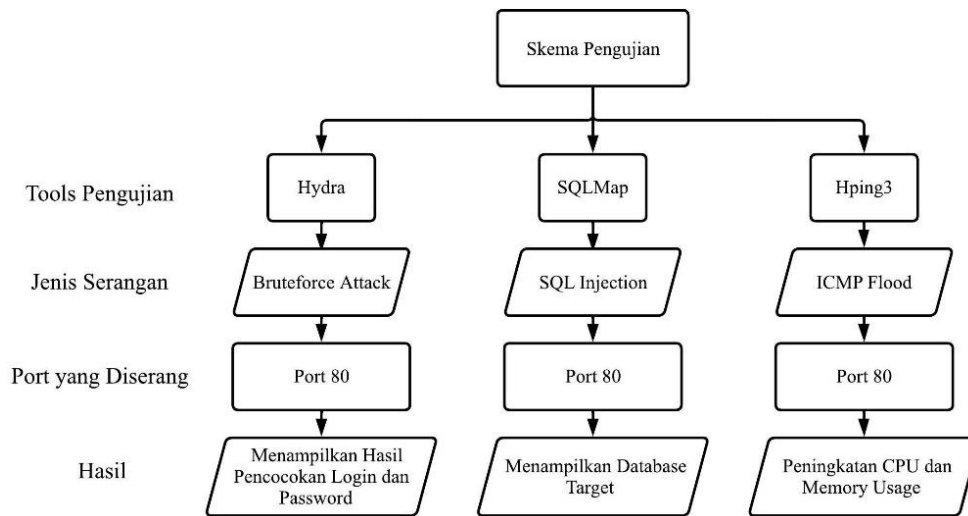
Berdasarkan Tabel 3.3 dan Gambar 3.24, bagian pertama mengasumsikan bahwa *Intrusion Prevention System (IPS)* Suricata belum diinstal atau belum diaktifkan pada sistem. Bagian kedua, di sisi lain, mengasumsikan bahwa IPS Suricata telah diinstal. Pada fase ini, sistem *webserver* yang sudah dikonfigurasi dengan IPS Suricata dan aturan-aturan tertentu akan diuji untuk mengevaluasi seberapa efektifnya dalam mendeteksi dan mencegah serangan HTTP *Bruteforce*, SQL *Injection*, dan serangan DoS (ICMP *flood*). Pengujian ini bertujuan untuk memberikan penilaian terhadap sejauh mana Suricata mampu melindungi *webserver* dari jenis-jenis serangan tersebut.

Selain itu, dalam pengujian ini, akan diuji sebuah paket data baru dengan aturan-aturan yang belum pernah diuji sebelumnya. Langkah ini diambil untuk melihat bagaimana Suricata merespons terhadap paket data yang belum pernah dikenali sebelumnya. Lebih jauh lagi, serangan ICMP *flood* berikutnya menggunakan alamat IP yang belum terdeteksi. Dalam kasus ini, notifikasi tentang penggunaan CPU yang berlebihan akan dikirim melalui bot Telegram, memberikan peringatan kepada *administrator* jaringan.

Melalui hasil pengujian ini, aturan-aturan IPS Suricata dapat diperbarui dan ditingkatkan untuk meningkatkan efektivitasnya dalam mendeteksi serangan-serangan yang lebih baru dan kompleks. Pengujian sistem akan difokuskan pada setiap jenis serangan yang telah ditetapkan sebelumnya, yaitu HTTP *Bruteforce*, SQL *Injection*, dan ICMP *flood*. Hasil pengujian ini akan memberikan pandangan tentang sejauh mana tingkat keamanan *webserver* dan seberapa efektif IPS Suricata dalam melindungi sistem dari potensi serangan.

Melalui pendekatan pengujian yang komprehensif dan mendalam, diharapkan bahwa penelitian ini dapat memberikan pemahaman yang lebih baik

tentang keamanan *webserver* dan manfaat yang diperoleh dari penerapan IPS dalam melindungi jaringan dari berbagai jenis serangan yang mungkin muncul.



**Gambar 3.24 Skema Pengujian**

### 3.1.6 Serangan Diterima

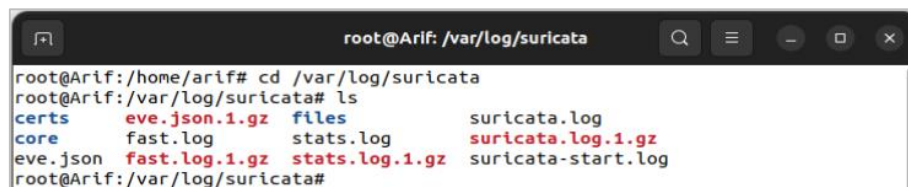
Setelah penyerangan oleh *attacker* terhadap *webserver* telah dilakukan, penelitian ini akan melakukan pengamatan dan validasi terhadap serangan tersebut. Dalam proses ini, digunakan dua skenario untuk mensimulasikan potensi serangan selama pengujian. Jika pengujian gagal, maka prosedur konfigurasi sistem akan dimulai kembali. Namun, jika pengujian berhasil, maka data *log* Suricata akan diambil sebagai hasilnya. Dalam konteks ini, pengujian dikatakan gagal apabila *webserver* tidak mampu menerima serangan jaringan.

### 3.1.7 Pengambilan Hasil Data dan Analisis

Proses pengumpulan data dalam penelitian ini dilakukan dengan mengacu pada parameter-parameter yang telah sebelumnya ditetapkan untuk memberikan jawaban terhadap permasalahan yang diajukan dalam penelitian. Hasil yang diperoleh akan dianalisis secara komprehensif dengan mempertimbangkan Suricata *logs*, penggunaan CPU dan *memory*, serta *response time* sebagai indikator kinerja sistem.

Pada umumnya, *log* Suricata dapat diakses melalui direktori *log* yang terletak di bawah jalur `"/var/log/suricata/fast.log"`. Ini berarti bahwa informasi *log* yang dihasilkan oleh Suricata disimpan dalam folder khusus dengan alamat tersebut

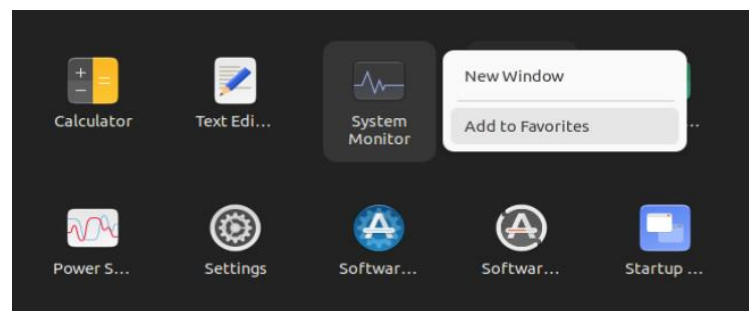
di sistem operasi yang relevan. Gambar 3.25 menunjukkan perintah yang digunakan untuk mengakses Suricata logs pada sistem operasi Ubuntu.



```
root@Arif: /var/log/suricata
root@Arif:/home/arif# cd /var/log/suricata
root@Arif:/var/log/suricata# ls
certs      eve.json.1.gz  files          suricata.log
core      fast.log       stats.log      suricata.log.1.gz
eve.json   fast.log.1.gz  stats.log.1.gz suricata-start.log
root@Arif:/var/log/suricata#
```

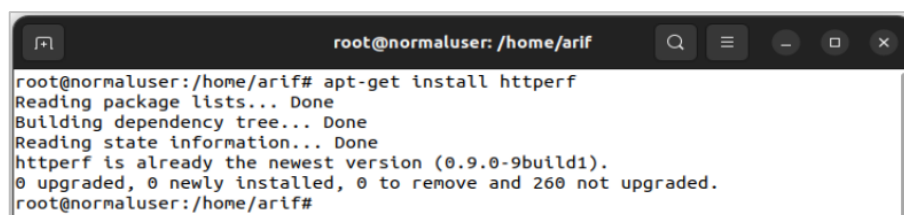
**Gambar 3.25 Suricata Logs**

Untuk mengumpulkan data hasil penggunaan CPU dan *memory* pada *webserver*, digunakan perangkat lunak pemantau sistem yang telah tersedia di sistem operasi Ubuntu seperti Gambar 3.26. Dengan memanfaatkan perangkat lunak ini, data terkait penggunaan sumber daya CPU dan *memory* dapat dikumpulkan secara terperinci dan akurat, memberikan wawasan mendalam tentang bagaimana *webserver* merespons dan menangani beban kerja yang diberikan.



**Gambar 3.26 System Monitor**

Selain itu, untuk pengambilan data *response time*, digunakan *tool* *Httpperf* yang dijalankan melalui akses pengguna normal. *Tool* ini memberikan kemampuan untuk mengirimkan permintaan HTTP *GET* dan secara akurat mengukur waktu *respons* dari server, sehingga memberikan data yang relevan untuk menganalisis kinerja sistem dengan lebih rinci. Untuk instalasi dapat dilakukan dengan mengetikkan perintah “*apt-get install httpperf*” serta perintah yang digunakan seperti Gambar 3.27. Sedangkan perintahnya terdapat pada gambar 3.28



```
root@normaluser: /home/arif
root@normaluser:/home/arif# apt-get install httpperf
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
httpperf is already the newest version (0.9.0-9build1).
0 upgraded, 0 newly installed, 0 to remove and 260 not upgraded.
root@normaluser:/home/arif#
```

**Gambar 3.27 Instalasi Httpperf**

```
arif@normaluser: ~  
arif@normaluser:~$ httpperf --server 192.168.61.62 --num-conns 10000  
httpperf --client=0/1 --server=192.168.61.62 --port=80 --uri=/ --send-buffer=4096  
--recv-buffer=16384 --num-conns=10000 --num-calls=1  
httpperf: warning: open file limit > FD_SETSIZE; limiting max. # of open files to  
FD_SETSIZE
```

**Gambar 3.28** Perintah *Response Time*

Keterangan:

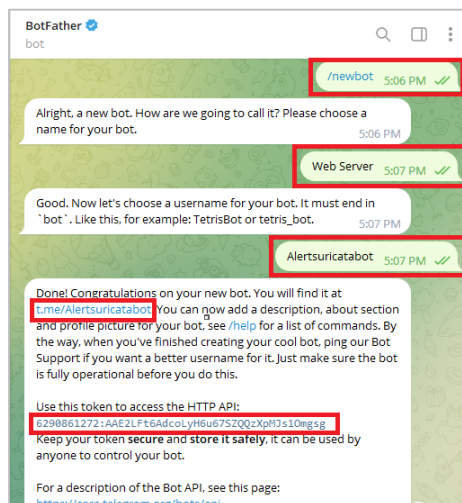
- --server : Alamat server target
- --num-conns : Jumlah permintaan koneksi

### 3.1.8 Notifikasi Melalui Telegram

Telegram berperan berfungsi sebagai platform notifikasi yang mengirimkan pesan dari *alert* IPS Suricata kepada admin jaringan. Untuk mencapai ini, API Telegram digunakan sebagai antarmuka menghubungkan sistem *webserver* dengan layanan Telegram melalui *channel* khusus. Beberapa konfigurasi diperlukan untuk mengaktifkan pemantauan server sebagai berikut:

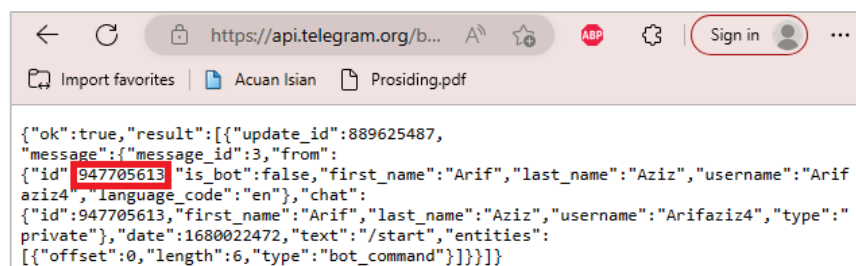
#### a. Pembuatan Telegram *bot*

Gunakan aplikasi Telegram untuk membuat *bot*. Caranya adalah dengan berinteraksi dengan "*Botfather*" *channel bot* utama yang membuat bot lain. Seperti yang terlihat pada Gambar 3.29 dengan perintah `"/newbot"` pada chat, bot baru dapat dibuat. Misalnya, penulis memberi nama bot "*Webserver*" dengan username "*Alertsuricatabot*" Setelah pembuatan *bot*, *Botfather* akan memberikan tautan *channel bot* dan *token* unik untuk akses HTTP API. Akhiri dengan perintah `"/start"` pada *channel bot* yang baru untuk mengaktifkannya.



**Gambar 3.29** *New Bot*

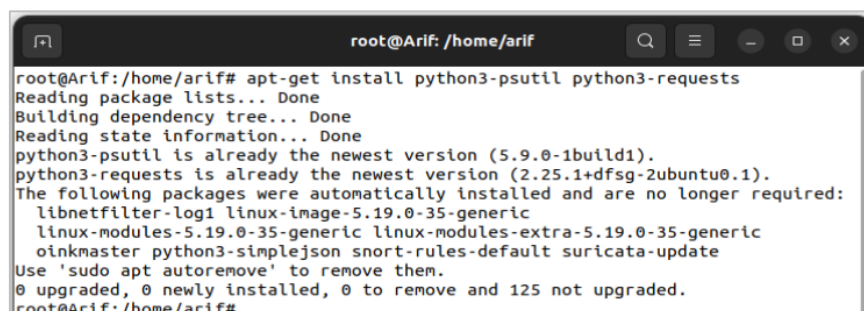
Gambar 3.29 menunjukkan tahap memperoleh *chat\_id* pada *bot Alertsuricatabot* untuk memastikan bahwa pesan yang dikirim oleh sistem terkirim dengan benar sesuai dengan *bot* yang telah dibuat. Untuk mendapatkan *chat\_id* tersebut, dapat dilakukan melalui *web browser* dengan mengakses URL `https://api.telegram.org/bot<token>/getUpdates`, di mana bagian "`<token>`" diganti dengan *token* unik yang telah diperoleh dari *Botfather*, yaitu `6290861272:AAE2LFt6AdcoLyH6u67SZQQzXpMJs1Omsgg`. Setelah melakukan akses tersebut, *web browser* secara otomatis akan menampilkan *chat\_id* yang diperlukan untuk keperluan selanjutnya.



Gambar 3.30 *Chat\_id*

#### b. Konfigurasi Sistem *Monitoring Server*

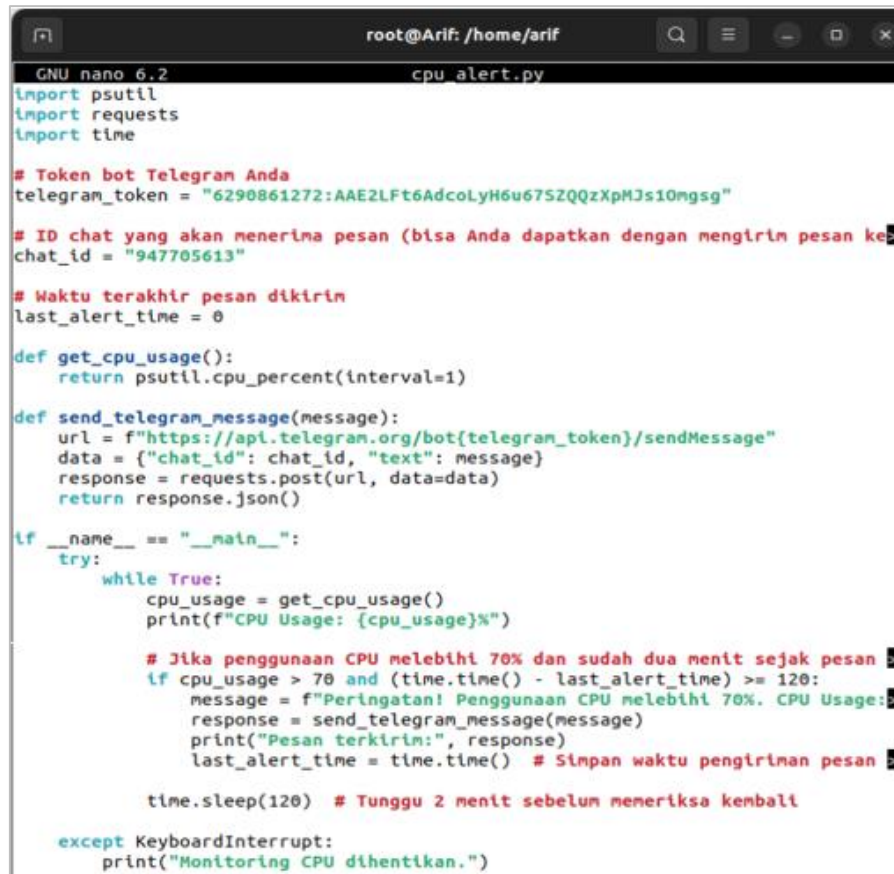
Pada tahap ini, Python digunakan sebagai program pengirim notifikasi ke Telegram *bot*. Dengan memanfaatkan pustaka *psutil* dan *requests* seperti Gambar 3.31, serta mengakses API Telegram, program dapat mengirimkan pesan notifikasi ke grup atau pengguna tertentu ketika terdeteksi aktivitas mencurigakan dalam jaringan.



Gambar 3.31 Instalasi Pustaka Python

Pembuatan *script* memanfaatkan *nano* atau *vi* yang akan digunakan untuk membuat *script* Python dengan nama *file* "`CPU_alert.py`" sesuai dengan yang tercantum pada Gambar 3.32. *Script* tersebut akan berfungsi untuk melakukan *monitoring server* dan akan dipicu oleh batasan minimal penggunaan CPU

sebesar 70%. Dengan adanya *bot token* dan *chat\_id*, nantinya *script* akan dapat mengirimkan notifikasi ke Telegram *bot* apabila terjadi penggunaan CPU yang melebihi batasan minimal yang telah ditentukan. Dengan demikian, tahap ini akan membantu memastikan keamanan dan kinerja server dalam menghadapi potensi permasalahan terkait penggunaan CPU yang tinggi.



```
root@Arif: /home/arif
GNU nano 6.2                               cpu_alert.py
import psutil
import requests
import time

# Token bot Telegram Anda
telegram_token = "6290861272:AAE2Lft6AdcoLyH6u67SZQzXpMJs10mgsg"

# ID chat yang akan menerima pesan (bisa Anda dapatkan dengan mengirim pesan ke
chat_id = "947705613"

# Waktu terakhir pesan dikirim
last_alert_time = 0

def get_cpu_usage():
    return psutil.cpu_percent(interval=1)

def send_telegram_message(message):
    url = f"https://api.telegram.org/bot({telegram_token})/sendMessage"
    data = {"chat_id": chat_id, "text": message}
    response = requests.post(url, data=data)
    return response.json()

if __name__ == "__main__":
    try:
        while True:
            cpu_usage = get_cpu_usage()
            print(f"CPU Usage: {cpu_usage}%")

            # Jika penggunaan CPU melebihi 70% dan sudah dua menit sejak pesan
            if cpu_usage > 70 and (time.time() - last_alert_time) >= 120:
                message = f"Peringatan! Penggunaan CPU melebihi 70%. CPU Usage:
                response = send_telegram_message(message)
                print("Pesan terkirim:", response)
                last_alert_time = time.time() # Simpan waktu pengiriman pesan

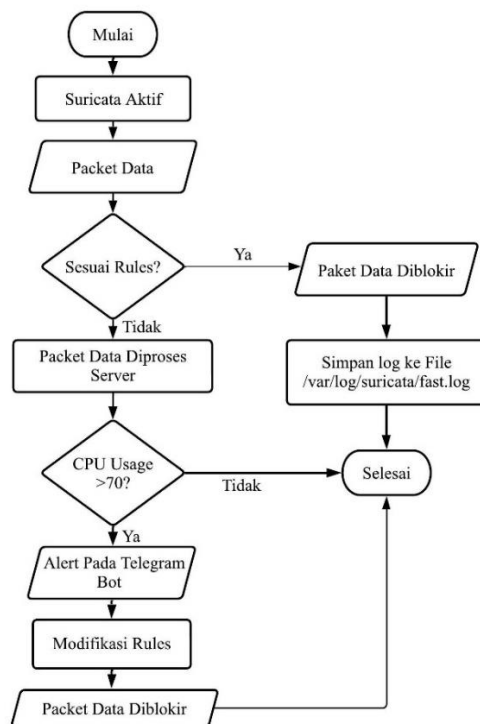
            time.sleep(120) # Tunggu 2 menit sebelum memeriksa kembali

    except KeyboardInterrupt:
        print("Monitoring CPU dihentikan.")
```

Gambar 3.32 Konfigurasi Script

### 3.2 Alur Sistem

Pada bagian ini, akan diuraikan pandangan keseluruhan mengenai alur kerja sistem keamanan jaringan yang diimplementasikan melalui *Intrusion Prevention System* (IPS) berbasis Suricata yang akan direncanakan. Ilustrasinya dijelaskan pada Gambar 3.33. Awal dari rangkaian proses ini adalah memulai eksekusi Suricata, yang telah dikonfigurasi dengan sekumpulan aturan untuk menyaring paket data yang ditangkap oleh sistem. Setiap kali sebuah paket data sesuai dengan aturan yang telah ditetapkan, informasi terkait akan direkam dalam *file* */var/log/Suricata/fast-log.txt*. *File* ini memuat detail mengenai jenis aturan yang relevan.



**Gambar 3.33 Flowchart Alur Kerja**

Secara berkala, setiap 2 menit, sebuah program Python akan dijalankan untuk mengawasi perubahan-perubahan terhadap tingkat penggunaan CPU. Ketika tingkat CPU mencapai batas minimal 70%, tindakan akan diambil. Dalam hal ini, jika sebuah paket data tidak memenuhi kriteria aturan yang telah ditetapkan oleh IPS Suricata, paket tersebut akan diteruskan ke server. Namun, bila paket data tersebut menyebabkan penggunaan CPU *webservice* melewati batas normal, yaitu 70%, sistem akan segera mengirimkan notifikasi peringatan melalui aplikasi Telegram *bot*. Tindakan selanjutnya akan diambil oleh *administrator*, yang kemungkinan akan memodifikasi aturan-aturan untuk mengatasi serangan yang sedang terjadi.

Melalui proses ini, sistem bekerja terkoordinasi untuk mendeteksi dan tanggap terhadap potensi serangan. Informasi tentang ancaman diambil oleh IPS Suricata dan dicatat untuk analisis lanjutan. Ketika situasi menunjukkan penggunaan CPU yang mencurigakan, sistem memberikan peringatan kepada *administrator* via Telegram *bot*. Ini memungkinkan admin untuk segera bertindak sesuai dengan keadaan, melawan serangan saat ini, dan menyesuaikan aturan guna cegah serangan serupa di masa mendatang.



### 3.3 Parameter

Penelitian ini dilakukan untuk menguji tiga jenis serangan, yaitu HTTP *Bruteforce*, *SQL Injection*, dan *Denial of Service* (DoS). Dalam proses pengujian serangan ini, parameter-parameter yang digunakan didasarkan pada prinsip CIA *Triad*. Dengan demikian, serangan HTTP *Bruteforce* dan *SQL Injection* termasuk dalam parameter kerahasiaan (*Confidentiality*). Alasannya adalah karena serangan-serangan tersebut dapat mengancam keamanan informasi dan data penting yang disimpan dalam *database*, sehingga informasi sensitif dapat jatuh ke tangan orang yang tidak berwenang. Di sisi lain, serangan *ICMP flood* termasuk dalam parameter ketersediaan (*Availability*). Serangan ini menyebabkan sumber daya server menjadi terlalu terbebani dan tidak responsif, sehingga pihak yang memiliki hak akses yang sah akan menghadapi kesulitan dalam menerima layanan yang memadai dari sistem. Tujuan dari penelitian ini adalah untuk mengevaluasi dampak serangan-serangan tersebut terhadap prinsip-prinsip keamanan yang menjadi dasar dalam keamanan sistem informasi, yakni kerahasiaan dan ketersediaan. Dengan pemahaman ini, diharapkan langkah-langkah pengamanan yang efektif dapat diambil untuk melindungi sistem dari serangan-serangan tersebut.

**Tabel 3.4 Parameter Serangan**

Jenis Serangan	CIA <i>Triad</i>		
	<i>Confidentiality</i>	<i>Availability</i>	<i>Integrity</i>
<i>Bruteforce</i>	✓	✗	✓
<i>SQL Injection</i>	✓	✗	✓
DoS	✗	✓	✗

Tabel 3.4 menunjukkan keberhasilan mekanisme sistem ini akan ditentukan oleh seberapa efektifnya aturan-aturan dalam menghadapi serangan yang berpotensi mempengaruhi penggunaan CPU dan *memory* pada server, dan hal ini juga akan dievaluasi melalui parameter *response time*. Untuk memantau *webserver*, digunakan Telegram *bot* sebagai media pemantauan. Efektivitasnya diukur melalui notifikasi yang diberikan saat terdeteksi adanya aktivitas yang mencurigakan. Dengan demikian, penggunaan Telegram *bot* sebagai alat pemantauan memberikan kemampuan untuk mendapatkan informasi secara *realtime* tentang aktivitas yang berpotensi membahayakan keamanan dan kinerja sistem.