BAB III
METODE PENELITIAN

Metode penelitian yang diterapkan pada penelitian ini merupakan *Design and Development* (D&D). Metode *Design and Development* (D&D) merupakan pendekatan penelitian yang digunakan untuk mengkaji sistem dengan tahapan proses desain, pengembangan, dan evaluasi. Menurut J. Ellis & Levy, (2010) menjelaskan bahwa D&D merupakan studi sistematis terkait proses desain, pengembangan, serta evaluasi yang bertujuan membangun pengalaman dalam pembuatan produk maupun perangkat, baik yang bersifat instruksional maupun non-instruksional, serta menghasilkan model baru atau penyempurnaan model yang sudah ada. Pemilihan metode D&D dalam penelitian ini didasarkan pada kelebihannya yang menyediakan kerangka kerja terstruktur untuk merancang, mengembangkan, sekaligus mengevaluasi solusi atau produk yang dikembangkan. Pada gambar 3.1 di bawah merupakan alur penelitian metode penelitian D&D.

Gambar 3.1 Alur Metode Penelitian D&D

Berdasarkan Gambar 3.1, tahapan penelitian diawali dengan analisis, yaitu proses mengidentifikasi kebutuhan serta permasalahan yang muncul dalam pengelolaan keamanan jaringan, khususnya pada aspek deteksi dan pemblokiran serangan siber pada server. Analisis ini mencakup potensi ancaman seperti DDoS (ICMP, SYN, UDP), *Brute Force*, IP *Spoofing*, SQL *Injection*, *Cross Site Scripting* (XSS), *Port Scanning*, dan *Slow* HTTP.

Tahap selanjutnya merupakan desain sistem, yaitu merancang solusi sistem keamanan dengan memanfaatkan Snort dan IPTables. Pada tahap ini juga disusun rancangan arsitektur jaringan, skenario pengujian serangan, serta integrasi notifikasi ke Telegram dan Grafana agar setiap serangan dapat dipantau dengan waktu yang cepat.

Selanjutnya, tahap pengembangan dilakukan dengan mengimplementasikan sistem yang telah dibuat ke dalam lingkungan nyata. Proses ini melibatkan instalasi

Snort serta konfigurasi *rules*, pembuatan skrip Bash untuk memicu IPTables dalam memblokir IP penyerang, instalasi dan konfgurasi Logstash, Elasticsearch, dan Grafana *dashboard* untuk memvisualisasikan log hasil dari serangan.

Tahap pengujian dilakukan dengan melakukan serangkaian simulasi serangan pada sistem, seperti brute force login, serangan web seperti SQL Injection dan XSS, serta port scanning. Hasil dari pengujian ini kemudian dievaluasi pada tahap evaluasi kerja sistem, guna menilai kinerja sistem dalam mendeteksi, memblokir, memberikan notifikasi, serta merekam log untuk analisis lebih lanjut.

Sebagai penutup, seluruh proses penelitian dan hasil implementasi didokumentasikan dalam tahap pelaporan. Dokumentasi ini disusun dalam bentuk laporan skripsi yang memuat hasil implementasi, hasil kinerja, serta analisis mendalam sebagai kontribusi ilmiah untuk pengembangan sistem keamanan server.

3.1. Analisis Kebutuhan Sistem

Pada tahapan ini, analisis kebutuhan sistem terbagi menjadi 3 bagian yaitu identifikasi masalah, studi literatur, dan spesifikasi alat. Berikut merupakan penjelasan tiap sub-bab nya:

3.1.1. Identifikasi Masalah

Pada tahapan ini, penulis mulai mengidentifikasi masalah yang muncul dari perlunya mengatasi ancaman serangan siber yang semakin kompleks. Peningkatan jumlah dan kompleksitas serangan, seperti DDoS (ICMP, SYN, UDP), *Brute Force*, IP *Spoofing*, SQL *Injection*, *Cross Site Scripting* (XSS), *Port Scanning*, dan *Slow* HTTP, menunjukkan kebutuhan mendesak untuk menerapkan solusi keamanan yang mumpuni. Rendahnya tingkat respons terhadap serangan serta dampak yang mungkin timbul juga menjadi sebab perhatian khusus dalam meningkatkan kecepatan dan efisiensi respons terhadap ancaman siber. Identifikasi masalah ini diharapkan menjadi dasar bagi penelitian untuk mengembangkan solusi keamanan yang lebih efektif dan responsif melalui implementasi Snort dan IPTables. Sistem ini didukung oleh notifikasi Telegram dan monitoring Grafana, yang bertujuan untuk menjaga keamanan server dari berbagai jenis serangan siber.

3.1.2. Studi Literatur

Pada tahapan ini, penulis mengumpulkan studi literatur yang berkaitan dengan implementasi Snort dan IPTables, notifikasi Telegram, serta monitoring Grafana. Informasi ini, yang bersumber dari jurnal ilmiah, skripsi, dan dokumentasi teknis, bertujuan untuk memberikan landasan teoritis yang kuat dan pemahaman mendalam mengenai konsep, metode, dan teknologi yang akan diterapkan pada penelitian ini. Kajian pustaka ini merupakan langkah esensial untuk memvalidasi pendekatan yang diambil dan memastikan bahwa sistem keamanan server yang diimplementasikan dapat bekerja secara efektif dan andal.

3.1.3. Spesifikasi Alat

Pada tahapan ini, peneliti akan menyiapkan perangkat untuk mendukung penelitian ini dengan menggunakan perangkat keras dan perangkat lunak. Berikut merupakan spesifikasi yang peneliti gunakan pada tabel 3.1 dan 3.2 di bawah ini:

Tabel 3.1 Spesifikasi Perangkat Keras

No	Perangkat Keras	Spesifikasi	Catatan
1	PC *server*	AMD Ryzen 7 5800H with Radeon Graphics, Nvidia RTX 3060 Laptop GPU, 32 GB RAM	Spesifikasi yang digunakan untuk menjalankan ubuntu server
2	PC *attacker*	AMD Ryzen 7 5800H with Radeon Graphics, Nvidia RTX 3060 Laptop GPU, 32 GB RAM	Spesifikasi yang digunakan untuk menjalankan kali linux

Tabel 3.2 Spesifikasi Perangkat Lunak

No	Perangkat Lunak	Versi	Deskripsi
1	Sistem Operasi Server	Ubuntu Live Server 24.04.2 LTS	Sistem operasi ini merupakan sistem operasi yang digunakan server pada penelitian ini
2	Sistem Operasi Penyerang	Kali Linux	Sistem operasi ini merupakan sistem operasi

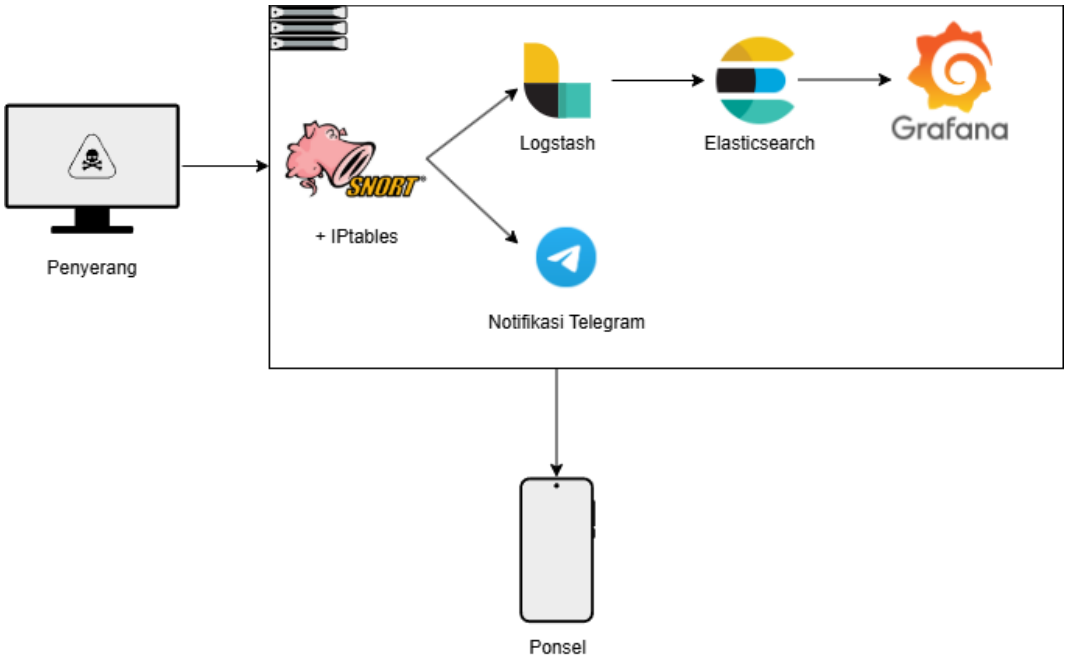
No	Perangkat Lunak	Versi	Deskripsi
			yang digunakan penyerang pada penelitian ini
3	Snort	*Version* 3.7.3.0	Aplikasi yang digunakan untuk mendeteksi serangan.
4	Logstash	logstash 7.17.28	*Tools open-source* dari Elastic Stack yang berfungsi sebagai *data pipeline* untuk mengumpulkan, memproses, dan mentransformasi data log dari Snort sebelum dikirim ke Elasticsearch.
5	Elasticsearch	Elasticsearch 7.17.28	Elasticsearch merupakan *Tools open-source* yang digunakan sebagai mesin pencari dan analitik dalam jumlah besar (big data) secara cepat.
6	Grafana	*Version* 12.0.0	*Platform* visualisasi dan monitoring *open-source* yang digunakan untuk membuat *dashboard* interaktif, menampilkan metrik dan tren data log Snort dari Elasticsearch secara cepat.
7	Telegram	Telegram *for* Android v11.14.0 (6102)	Aplikasi pesan instan yang diintegrasikan melalui *bot* API untuk mengirimkan notifikasi *alert* Snort secara cepat kepada administrator jaringan.
8	*Web* Server	Apache 2.4.58	Layanan *server* yang akan diinstal pada *server* target untuk mensimulasikan layanan *website* dan menjadi target serangan seperti SQL *Injection*, *Cross Site Scripting* (XSS), dan *Slow* HTTP.

No	Perangkat Lunak	Versi	Deskripsi
9	FTP Server	Vsftpd *version* 3.0.5	Layanan *server* yang akan diinstal pada *server* target untuk mensimulasikan layanan transfer *file* dan menjadi target serangan seperti *Brute Force*.
10.	GNOME *Desktop*	GNOME *Shell* 46.0	Fitur yang digunakan pada Ubuntu Server (jika *server* diakses secara GUI) atau pada *host* lain yang memerlukan antarmuka grafis untuk konfigurasi dan monitoring.
11	*Hping3*	hping3 *version* 3.0.0-alpha-2	Tool *Command* yang digunakan untuk membuat dan mengirimkan paket TCP/IP kustom, sangat berguna untuk mensimulasikan serangan seperti SYN *Flood* dan IP *Spoofing*.
12	Nmap	Nmap 7.95	*Network scanner open-source* yang digunakan untuk melakukan penemuan host dan *Port Scanning* terhadap server target untuk menguji kemampuan deteksi Snort.
13	Hydra	Hydra V9.5	Tool *Command* password cracking yang digunakan untuk mensimulasikan serangan *Brute Force* pada berbagai protokol seperti SSH dan FTP.
14	SQL*map*	SQL*map* 1.9.4 *stable*	SQLmap merupakan sebuah *penetration testing tool* yang dirancang untuk secara otomatis mendeteksi dan mengeksploitasi kerentanan SQL injection pada sebuah aplikasi web.
15	DVWA	-	Aplikasi *web* digunakan sebagai target yang aman

No	Perangkat Lunak	Versi	Deskripsi
			untuk mensimulasikan serangan *web* seperti SQL *Injection* dan XSS.
16	ngrok	ngrok *version* 3.24.0	Layanan server tunneling yang digunakan untuk membuat server lokal dapat diakses publik.
17	L2TP VPN	-	Layer 2 Tunneling Protocol merupakan protokol yang digunakan untuk membuat "tunnel" secara virtual untuk mengirimkan paket data.

3.2. Desain Sistem

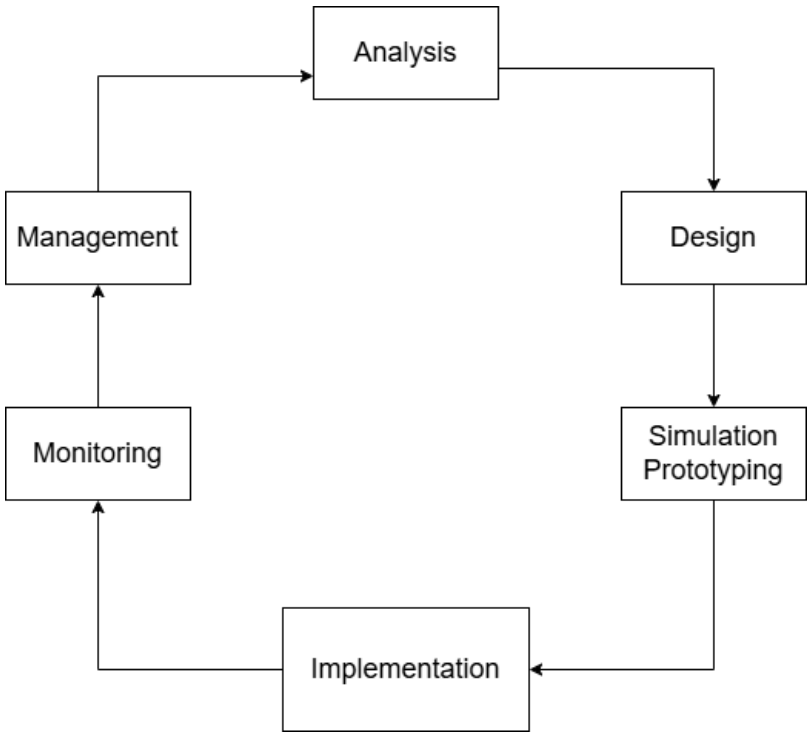
Pada tahapan ini, secara definisi desain sistem merupakan perancangan terhadap arsitektur dan alur kerja sistem yang akan dibuat yang sesuai dengan kebutuhan penelitian ini. Tujuannya adalah untuk memberikan gambaran yang jelas mengenai interaksi atau alur antara komponen perangkat keras, perangkat lunak, dan jaringan dalam mendukung mendukung implementasi sistem deteksi serangan menggunakan Snort dan IPTables dan notifikasi Telegram serta monitoring Grafana. Pada Gambar 3.2 Di bawah merupakan arsitektur sistem pada implementasi penelitian ini.



Gambar 3.2 Arsitektur Sistem

3.3. Metode Pengembangan Sistem

Jenis metode pengembangan sistem yang digunakan dalam penelitian ini adalah *Network Development Life Cycle* (NDLC). *Network Development Life Cycle* (NDLC) merupakan sebuah metode pengumpulan data yang dimulai dengan *analysis*, *design*, *simulation prototype*, *implementation*, monitoring, dan *management* (Nurdadyansyah & Hasibuan, 2021). Pada Gambar 3.3 terdapat alur penelitian menggunakan metode *Network Development Life Cycle* (NDLC).



Gambar 3.3 Alur Metode Pengembangan *Network Development Life Cycle* (NDLC)

1. *Analysis*

   Pada tahapan analisis, penelitian dilakukan secara menyeluruh untuk mengidentifikasi dan memahami pola serangan siber yang mungkin terjadi, khususnya jenis serangan yang menjadi fokus penelitian ini seperti serangan DDoS (ICMP, SYN, UDP), *Brute Force*, IP *Spoofing*, SQL *Injection*, *Cross Site Scripting* (XSS), *Port Scanning*, dan *Slow* HTTP. Tahap analisis melibatkan evaluasi data lalu lintas jaringan yang relevan dengan Snort, pengenalan pola perilaku mencurigakan yang akan dideteksi oleh Snort, dan penilaian terhadap kebutuhan serta respons notifikasi melalui Telegram, serta data apa saja yang perlu ditampilkan di Grafana. Analisis ini memungkinkan untuk mengidentifikasi karakteristik spesifik setiap serangan, memahami cara kerjanya, dan mengevaluasi kebutuhan dari sistem keamanan yang akan dibangun. Dengan pendekatan ini, penelitian dapat menghasilkan pengetahuan mendalam tentang taktik serangan yang digunakan dan memperkuat pertahanan sistem melalui pemahaman yang lebih baik terhadap ancaman yang muncul.

2. *Design*

   Pada tahapan ini dilakukan desain perancangan strategi keamanan yang mencakup implementasi Snort serta IPtables dan integrasi notifikasi melalui Telegram, serta sistem monitoring dengan Elasticsearch dan Logstash yang divisualisasikan oleh Grafana. Desain sistem mencakup perancangan topologi jaringan untuk pengujian, penetapan kebijakan keamanan dan penyesuaian *rules* Snort untuk mendeteksi dan mencegah serangan, konfigurasi Logstash untuk memproses log Snort, skema penyimpanan data di Elasticsearch, serta pengaturan notifikasi Telegram untuk memberikan notifikasi cepat kepada administrator dan perancangan *dashboard* Grafana untuk visualisasi data.

3. *Simulation & Prototyping*

   Pada tahapan ini, dilakukan pembuatan model prototipe sistem keamanan menggunakan Snort, IPtables, notifikasi Telegram, serta konfigurasi awal Logstash, Elasticsearch, dan Grafana. Simulasi dilakukan untuk menguji desain dan fungsionalitas sistem secara parsial sebelum diimplementasikan secara penuh dalam lingkungan yang lebih luas. Prototipe sistem memungkinkan untuk mensimulasikan skenario serangan siber dan mengamati responsnya (deteksi oleh Snort, dan pengiriman notifikasi), serta memvalidasi metode keamanan yang diusulkan dalam skala kecil.

4. *Implementation*

   Pada tahap implementasi ini, solusi keamanan yang telah dirancang, yaitu Snort yang terintegrasi dengan notifikasi menggunakan Telegram serta sistem *monitoring* berbasis Elasticsearch, Logstash, dan Grafana, akan diterapkan ke dalam lingkungan jaringan yang relevan. Proses implementasi mencakup instalasi dan konfigurasi Snort, Logstash, Elasticsearch, dan Grafana, penyesuaian kebijakan keamanan dan *rules* Snort, serta pengaturan skrip untuk pengiriman notifikasi Telegram. Dengan langkah ini, penelitian dapat mewujudkan solusi keamanan secara praktis dalam lingkungan nyata, sehingga sistem dapat aktif mendeteksi, mencegah, dan merespons serangan siber dengan menggunakan metode NDLC.

5. *Monitoring*

   Tahapan *monitoring* dilakukan secara kontinu terhadap lalu lintas jaringan dan kejadian keamanan yang terdeteksi oleh Snort. Data log yang dihasilkan akan dipantau melalui *dashboard* Grafana yang terhubung ke Elasticsearch. Proses *monitoring* ini memungkinkan identifikasi cepat terhadap anomali atau serangan yang terdeteksi, serta validasi kecepatan dan akurasi notifikasi yang dikirimkan ke Telegram.

6. *Management*

   Tahapan *management* adalah proses pemeliharaan kebijakan keamanan Snort, pembaruan perangkat lunak Snort dan Grafana yang digunakan. Ini juga mencakup pengelolaan notifikasi yang dihasilkan oleh Snort serta optimalisasi sistem secara berkala untuk memastikan relevansi dan kinerjanya dalam menghadapi ancaman baru dalam jangka panjang.

3.3.1. Analisis

Pada tahapan ini, peneliti melakukan analisis dengan tujuan untuk mengidentifikasi kebutuhan untuk implementasi keamanan yang digunakan guna melindungi sistem server dari serangan siber. Analisis disini mencakup identifikasi jenis-jenis serangan siber yang akan diuji, kebutuhan integrasi Snort, IPtables, notifikasi Telegram dan *monitoring* Grafana. Spesifikasi alat yang digunakan pada penelitian ini juga dilakukan analisis dengan tujuan guna melakukan implementasi yang optimal.

a) Jenis dan Sumber Data

Jenis dan sumber data yang didapatkan dalam penelitian ini akan menggunakan data utama. Data utama ini akan diperoleh langsung dari penerapan Snort dan IPtables serta penggunaan notifikasi Telegram dan sistem monitoring berbasis Elasticsearch, Logstash, dan Grafana pada sistem yang relevan. Ini mencakup hasil pengamatan langsung selama eksperimen atau implementasi sistem keamanan. Data utama ini akan memberikan informasi langsung tentang respons dan kinerja sistem selama pengujian simulasi serangan siber.

b) Teknik Pengumpulan Data

Teknik pengumpulan data pada penelitian ini melibatkan beberapa tahapan untuk mendapatkan data yang dibutuhkan. Berikut adalah penjelasan rinci mengenai teknik pengumpulan data yang akan digunakan:

1. Observasi

   Tahapan ini melibatkan pengamatan langsung terhadap implementasi Snort dan IPtables serta notifikasi Telegram dan *dashboard monitoring* Grafana pada sistem server yang diuji. Observasi ini mencakup cara kerja Snort dalam mendeteksi, mencegah atau memblokir serangan yang masuk oleh IPtables, respons notifikasi Telegram terhadap jenis serangan yang diuji, serta kesesuaian visualisasi data di Grafana.

2. Dokumentasi Teknis

   Tahapan ini mengumpulkan informasi dari dokumentasi teknis terkait Snort, Telegram API, Skrip IPtables, Elasticsearch, Logstash, dan Grafana. Ini mencakup instalasi dan tata cara konfigurasi untuk memahami implementasi dengan lebih mendalam dan memastikan konfigurasi yang optimal.

3. Literatur Terkait

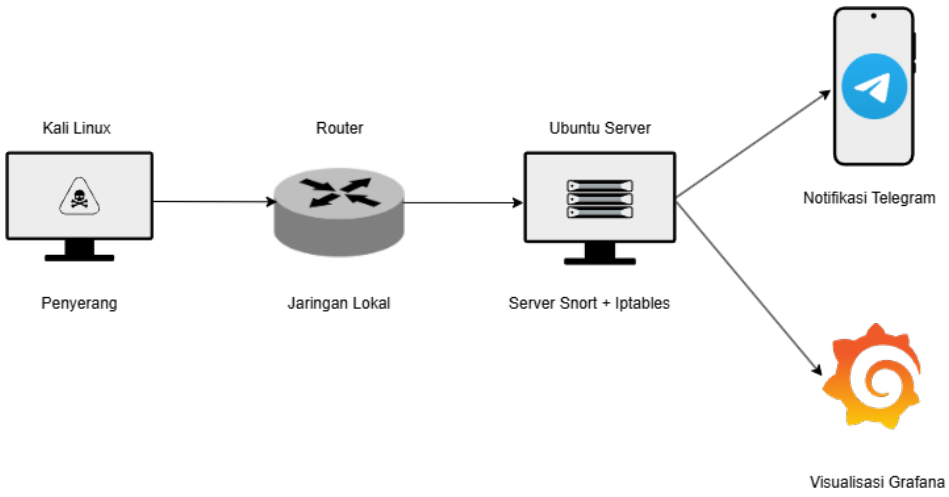
   Menggunakan jurnal ilmiah, buku, dan artikel terkait implementasi Snort, IPtables, deteksi serangan siber, notifikasi menggunakan Telegram, dan juga terkait sistem pengelolaan log terpusat seperti Elasticsearch, Logstash dan visualisasi data dengan Grafana. Literatur ini akan memberikan dasar yang kuat untuk penelitian.

Ketiga tahapan di atas diharapkan dapat memberikan analisis yang mendalam terhadap kinerja Snort dan IPtables dalam menghadapi serangan siber dengan dukungan notifikasi melalui Telegram dan monitoring melalui Grafana.

3.3.2. Desain

Pada tahapan desain, peneliti membuat perancangan topologi jaringan yang akan digunakan untuk implementasi Snort dan IPtables sebagai sistem keamanan server. Tahapan ini juga mencakup perancangan *rules* pada Snort yang spesifik untuk mendeteksi serangan yang telah dilakukan identifikasi pada tahapan analisis, seperti aturan untuk mendeteksi serangan DDoS (ICMP, SYN, UDP), *Brute Force*,

IP *Spoofing*, SQL *Injection*, *Cross Site Scripting* (XSS), *Port Scanning*, dan *Slow* HTTP. Deteksi Snort ini kemudian akan memicu skrip *Bash* untuk membaca log dari snort dan memblokir IP penyerang melalui IPtables. Tahap desain juga mencakup integrasi notifikasi Telegram, yang akan mengirimkan peringatan adanya serangan kepada administrator jaringan segera setelah serangan terdeteksi dan diblokir, serta perancangan integrasi dengan Elasticsearch dan Logstash sebagai fondasi data, dan Grafana untuk visualisasi log Snort. Berikut merupakan gambaran dari topologi jaringan yang digunakan pada penelitian ini, Pada gambar 3.4 di bawah, merupakan gambaran dari perancangan topologi jaringan di ip lokal yang akan dibuat pada penelitian ini:

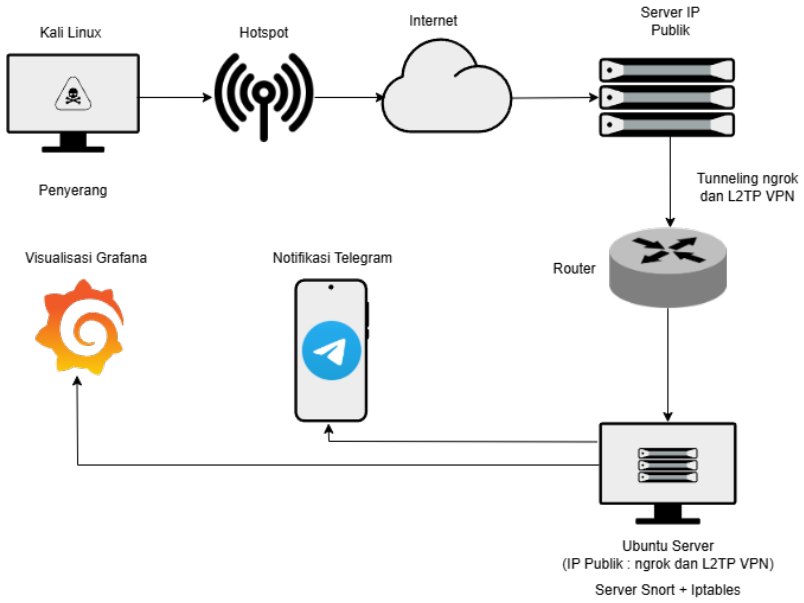


Gambar 3.4 Topologi IP Lokal

Pada gambar 3.3 penyerang melakukan serangan dari *device* yang terkoneksi dalam satu jaringan dengan server. Selanjutnya, penyerang melancarkan aksi serangan kepada server dengan tujuan untuk mengeksploitasi jaringan server dan melumpuhkan layanan yang tersedia pada server. Eksploitasi server ini dilakukan dengan cara mengirimkan beberapa jenis serangan siber menggunakan *tools* bantuan yang dibutuhkan untuk mengganggu kinerja dari server. Setelah serangan dikirimkan, maka *rules* Snort yang telah dibuat akan disesuaikan agar bisa mendeteksi serangan siber. Setelah aturan disesuaikan dengan jenis serangan yang ingin dideteksi, maka aturan Snort akan terpicu dan skrip *Bash* akan membaca log dari snort tersebut untuk memblokir IP penyerang menggunakan IPtables. Selanjutnya, setelah *rules* pada Snort terpicu oleh serangan yang masuk kepada

server dan mengindikasikan ada serangan yang masuk, sistem akan mengirimkan notifikasi Telegram yang memberitahukan adanya serangan kepada administrator jaringan agar segera melakukan pengecekan terhadap server. Data log dari juga akan dikirimkan ke Logstash dan disimpan di Elasticsearch untuk selanjutnya divisualisasikan pada Grafana.

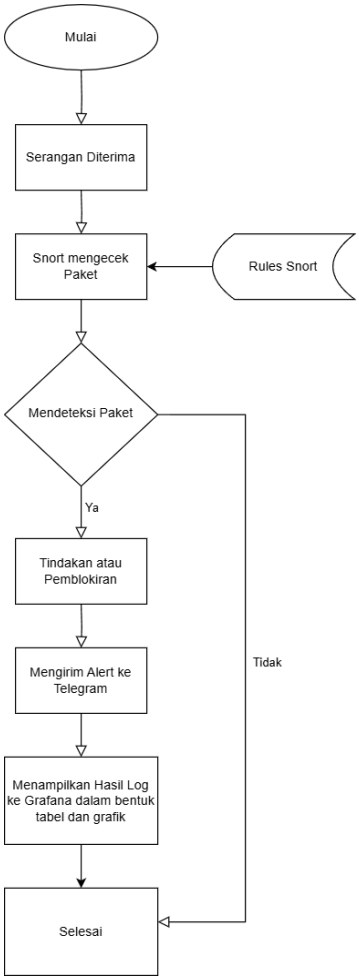
Disini peneliti selain menggunakan ip lokal, ip publik akan digunakan dalam rangka pengujian sistem. Ip publik yang digunakan merupakan Ngrok dan L2TP VPN. Berikut juga merupakan rancangan topologi untuk pengujian serangan dari IP publik menggunakan ngrok dan L2TP VPN, Pada gambar 3.5 di bawah, merupakan gambaran dari perancangan topologi jaringan ip lokal yang akan dibuat pada penelitian ini. Dari gambaran topologi tersebut, penyerang melakukan serangan dari *device* yang terkoneksi dalam jaringan yang berbeda dengan server snort tapi akan terhubung melalui ip publik lewat dua jenis *tunneling* yaitu ngrok dan L2TP VPN



Gambar 3.5 Topologi IP Publik

3.3.3. Simulasi

Pada tahapan simulasi, peneliti disini akan menjelaskan bagaimana alur sistem server yang telah terintegrasi oleh Snort, IPtables, Telegram, dan Grafana. Gambar 3.6 di bawah menjelaskan alur kerja sistem secara keseluruhan dan bertahap.



Gambar 3.6 Alur Sistem Server Snort

1. Proses dimulai ketika sebuah serangan diterima oleh jaringan yang sedang dipantau. Setelah itu, Snort akan mengecek setiap paket yang masuk.
2. Tahapan selanjutnya adalah Mendeteksi Paket. Pada titik ini, Snort tidak hanya memeriksa paket secara umum, tetapi secara aktif membandingkannya dengan "*Rules* Snort" yang telah dikonfigurasi sebelumnya. Aturan-aturan ini adalah inti dari Snort, berisi definisi atau pola-pola spesifik dari berbagai jenis serangan siber yang ingin dideteksi dan dicegah.
3. Jika Snort menemukan kecocokan antara paket yang masuk dan salah satu aturan dalam "*Rules* Snort" (mengindikasikan adanya serangan), maka serangan tersebut akan terdeteksi. Segera setelah deteksi, skrip Bash untuk IPtables akan mengambil tindakan atau pemblokiran terhadap aktivitas

mencurigakan tersebut, ini adalah fungsi utama Snort sebagai *Intrusion Detection System* (IDS) dan IPtables sebagai *Intrusion Prevention System* (IPS).

4. Selanjutnya, sebagai bentuk peringatan dan pencatatan, sistem akan Mengirim *Alert* ke Telegram secara cepat kepada administrator jaringan. Selain notifikasi instan, semua hasil log dari deteksi dan tindakan Snort juga akan Menampilkan Hasil Log lewat Elasticsearch dan Logstash ke Grafana dalam bentuk tabel dan grafik. Ini memungkinkan administrator untuk memantau, menganalisis, dan melacak seluruh aktivitas keamanan jaringan secara komprehensif.
5. Sebaliknya, jika paket yang dicek oleh Snort tidak sesuai dengan *rules* Snort (tidak terdeteksi sebagai ancaman), maka paket tersebut dianggap sebagai lalu lintas jaringan biasa dan tidak akan memicu *alert* atau tindakan pemblokiran. Proses ini berlangsung secara kontinu hingga mencapai titik selesai

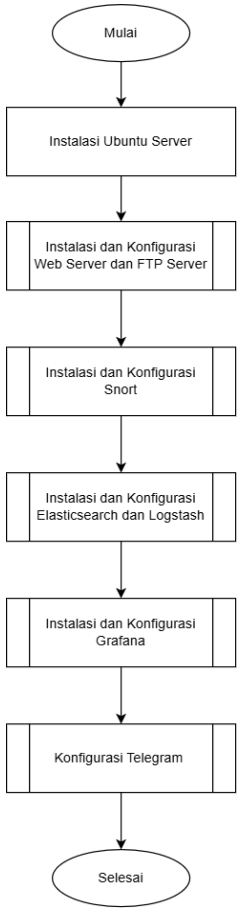
3.3.4. Implementasi

Pada tahapan implementasi, Snort diinstal dan dikonfigurasi pada server utama, yaitu Ubuntu Server, sesuai dengan spesifikasi dan desain arsitektur yang telah direncanakan sebelumnya. Implementasi ini mencakup penerapan *rules* Snort yang telah dirancang untuk mendeteksi jenis serangan siber, seperti DDoS, *Brute Force*, IP *Spoofing*, SQL *Injection*, XSS, *Port Scanning*, dan *Slow* HTTP.

Setelah Snort berhasil dikonfigurasi dengan *rules* tersebut, langkah selanjutnya adalah menambahkan skrip *bash* untuk IPtables untuk mencegah serangan masuk ke dalam sistem server. Langkah selanjutnya adalah mengintegrasikan sistem notifikasi. *Bot* Telegram diatur dan diintegrasikan untuk memastikan bahwa setiap pendeteksian serangan siber yang terjadi oleh Snort, notifikasi peringatan akan segera dikirimkan secara cepat kepada administrator jaringan.

Selain itu, pada tahap implementasi ini, Logstash juga diatur untuk mengambil dan memproses log dari Snort, kemudian mengirimkannya ke Elasticsearch untuk penyimpanan terpusat. Grafana dikonfigurasi untuk terhubung

dengan Elasticsearch, memungkinkan pembuatan *dashboard monitoring* untuk visualisasi data log dan metrik keamanan. Untuk lebih jelasnya, visualisasi implementasi sistem dapat dilihat pada gambar 3.6 di bawah ini:



Gambar 3.7 Alur Implementasi Sistem

1. Instalasi Ubuntu Server

   Langkah awal dari alur implementasi sistem adalah mempersiapkan server utama yang akan digunakan. Ini dimulai dengan menginstal Ubuntu Server dengan versi 24.02.2 LTS yang akan berfungsi sebagai *platform* utama untuk seluruh komponen sistem keamanan yang diusulkan.

2. Instalasi dan Konfigurasi *Web* Server dan FTP Server

   Setelah server dasar siap, peneliti akan melanjutkan dengan menginstal dan mengkonfigurasi *Web* Server dan FTP Server. Kedua layanan ini akan berfungsi sebagai target simulasi serangan siber, tempat Snort akan memantau dan melindungi layanan yang berjalan di atasnya. Peneliti juga

akan mengatur konfigurasi IP *address* pada server tersebut agar dapat terhubung ke jaringan dan dijadikan sebagai IP target serangan.

3. Instalasi dan Konfigurasi Snort dan IPtables

   Peneliti selanjutnya akan menginstal dan mengkonfigurasi Snort dan IPtables, yang berperan untuk mendeteksi dan memitigasi serangan siber yang nantinya akan diujikan pada penelitian ini. Untuk alur kerja dari server snort, penulis sudah jelaskan pada Gambar 3.5 di atas, serta dilakukan konfigurasi *rules* di dalamnya untuk setiap masing-masing jenis serangan yang telah diidentifikasi (DDoS, *Brute Force*, IP *Spoofing*, SQL *Injection*, XSS, *Port Scanning*, dan *Slow* HTTP).

4. Instalasi dan Konfigurasi Elasticsearch dan Logstash

   Pada tahapan ini, peneliti akan menginstal dan mengkonfigurasi komponen Elasticsearch dan Logstash. Logstash akan diatur untuk mengambil data log peringatan dari Snort, memprosesnya, dan kemudian mengirimkannya ke Elasticsearch untuk penyimpanan dan pengindeksan data log keamanan secara terpusat.

5. Instalasi dan Konfigurasi Grafana

   Tahapan selanjutnya, peneliti akan menginstal dan melakukan konfigurasi Grafana. Grafana akan diintegrasikan dengan Elasticsearch sebagai *data source* untuk membuat *dashboard* visualisasi yang interaktif, menampilkan metrik dan tren serangan siber yang dideteksi oleh Snort, serta status keamanan sistem secara cepat.

6. Konfigurasi Telegram

   Tahap terakhir dalam alur implementasi adalah konfigurasi Telegram. Ini melibatkan pengaturan *bot* Telegram dan skrip *bash* yang akan terintegrasi dengan Snort melalui pemantauan log untuk memastikan bahwa setiap kali ada deteksi serangan yang signifikan, notifikasi peringatan akan segera dikirimkan secara cepat kepada administrator jaringan melalui aplikasi Telegram.

3.3.5. Monitoring

Monitoring merupakan langkah penting dalam NDLC keamanan server, terutama untuk memastikan bahwa sistem deteksi dan pencegahan intrusi seperti Snort dan IPtables berfungsi dengan optimal dan seluruh alur data berjalan dengan sesuai. Dalam penelitian ini, proses *monitoring* dilakukan untuk mengawasi lalu lintas jaringan secara langsung dan untuk memastikan bahwa Snort dapat mendeteksi serangan dengan cepat dan akurat dan IPtables dapat mencegah atau memblokir IP *address* penyerang. *Monitoring* juga mencakup evaluasi notifikasi yang dikirimkan oleh *bot* Telegram kepada administrator jaringan, serta visualisasi data log keamanan yang komprehensif melalui Grafana yang terhubung dengan Elasticsearch dan Logstash.

a. Proses Monitoring

Monitoring dilakukan dengan memanfaatkan fitur-fitur yang ada dalam Snort untuk menganalisis lalu lintas jaringan secara terus-menerus. Snort diinstal pada server utama yaitu Ubuntu Server dan dikonfigurasi untuk memantau semua lalu lintas yang masuk dan keluar dari jaringan yang dilindungi. Proses ini melibatkan beberapa langkah penting:

1. Pengaturan *Logging* dan *Pipeline* Data Logstash & Elasticsearch

   Snort dikonfigurasi untuk mencatat semua kejadian yang terdeteksi, baik berupa potensi serangan maupun lalu lintas yang mencurigakan. Log ini kemudian disalurkan dan diproses oleh Logstash, yang akan mengubahnya menjadi format terstruktur (JSON). Data yang telah diproses ini selanjutnya dikirim dan disimpan di Elasticsearch, sebuah *database* yang dirancang untuk pengindeksan dan pencarian data log secara cepat. Pengaturan ini memastikan bahwa semua *alert* dan informasi relevan dari Snort tersimpan dengan rapi dan mudah diakses untuk analisis lebih lanjut.

2. Visualisasi Data Grafana

   Untuk mempermudah pemantauan dan analisis, Grafana diintegrasikan dengan Elasticsearch sebagai sumber data. Berbagai *dashboard* interaktif dibuat di Grafana untuk menampilkan visualisasi dari data log Snort. Ini mencakup grafik tren serangan per waktu, distribusi jenis serangan (DDoS

(ICMP, SYN, UDP), *Brute Force*, SQL *Injection*, *Cross Site Scripting* (XSS), *Port Scanning*, IP *Spoofing*, *Slow* HTTP), daftar IP sumber penyerang teratas, dan detail *alert* terbaru dalam bentuk tabel. Visualisasi ini memberikan gambaran umum yang jelas dan cepat mengenai status keamanan jaringan.

3. Integrasi Notifikasi Telegram

   Untuk mempercepat respons terhadap ancaman, sistem ini terintegrasi dengan *bot* Telegram. Ketika Snort mendeteksi ancaman berdasarkan *rules* yang telah ditetapkan dan mengirimkan *alert*, sebuah skrip *bash* akan memicu notifikasi otomatis yang dikirimkan melalui Telegram kepada administrator jaringan. Hal ini memungkinkan tindakan segera dilakukan, bahkan ketika administrator tidak berada di depan sistem monitoring Grafana.

4. Analisis Log

   Tahapan ini melibatkan analisis log yang dihasilkan oleh Snort terkait deteksi. Log ini akan disalurkan, diproses, dan disimpan di Elasticsearch melalui Logstash. Analisis log akan memberikan informasi detail tentang jenis serangan yang terdeteksi (DDoS, *Brute Force*, SQL *Injection*, XSS, *Slow* HTTP, *Port Scanning*, IP *Spoofing*), IP sumber/tujuan, waktu kejadian, protokol, dan *action* yang diambil oleh Snort. Data dari log inilah yang kemudian akan divisualisasikan di Grafana.

3.4. Pengujian Sistem

Dalam penelitian ini peneliti akan menggunakan tiga skema pengujian yang dimana skema pengujian ini bertujuan agar implementasi sistem bisa berjalan sesuai fungsinya. Berikut merupakan skema yang akan dilakukan untuk implementasi:

3.4.1. Pengujian Fungsionalitas Sistem Keseluruhan

Pengujian ini dilakukan sebagai validasi awal untuk memastikan semua komponen sistem berfungsi sebagaimana mestinya dan saling terhubung dengan baik, baik dalam lingkungan IP Lokal maupun IP Publik. Berikut beberapa hal yang menjadi fokus pada pengujian fungsionalitas sistem:

1. Fungsionalitas Dasar Snort dan IPtables

   Pada pengujian ini, peneliti menguji apakah Snort dapat berjalan dan mendeteksi lalu lintas jaringan, dan apakah skrip *Bash* berhasil memicu IPtables untuk memblokir lalu lintas berdasarkan log dari snort. Pengujian ini akan dilakukan untuk *traffic* yang berasal dari IP Lokal maupun IP Publik.

2. Integrasi Logstash-Elasticsearch-Grafana

   Pada pengujian ini, peneliti memastikan log dari Snort dari *alert*_json.txt berhasil diproses oleh Logstash, disimpan di Elasticsearch, dan divisualisasikan dengan benar di *dashboard* Grafana. Ini akan diverifikasi dengan memicu *alert* Snort dari IP Lokal dan IP Publik dan memeriksa kemunculannya di Grafana.

3. Integrasi Notifikasi Telegram

   Pada pengujian ini, peneliti menguji apakah *bot* Telegram berhasil mengirimkan notifikasi adanya serangan ke administrator jaringan segera setelah *alert* Snort terpicu, baik untuk serangan yang berasal dari IP Lokal maupun IP Publik.

4. Akurasi Deteksi oleh Snort

   Pada pengujian ini, peneliti menguji bahwasanya *rules* pada snort bisa bekerja semestinya dengan mendeteksi serangan yang diujikan.

5. Efisiensi Pemblokiran oleh IPtables

   Pada pengujian ini, peneliti menguji seberapa cepat skrip *Bash* memicu IPtables untuk memblokir *traffic* melalui log snort.

6. Penggunaan CPU Server

   Pada pengujian ini, peneliti memantau penggunaan CPU server selama pengujian berlangsung berdasarkan jenis serangan yang diuji.

7. Monitoring Grafana

   Pada pengujian ini, peneliti Memantau akurasi data serangan yang ditampilkan *dashboard* Grafana sesuai berdasarkan log Snort

3.4.2. Pengujian Serangan Siber Spesifik

Berikut penjelasan secara detail untuk setiap jenis serangan spesifik yang akan diuji. Setiap pengujian akan dilakukan dalam skenario serangan dari IP Lokal dan IP Publik untuk mengevaluasi kemampuan sistem (Snort dan IPtables) dalam berbagai kondisi jaringan. Adapun beberapa pengujian yang dilakukan sebagai berikut:

1. Pengujian pada Serangan ICMP *Flood*:

   Penyerang membanjiri server target dengan paket ICMP dalam jumlah besar untuk menilai kemampuan Snort dalam mendeteksi dan IPtables dalam memitigasi serangan tersebut.

2. Pengujian pada Serangan UDP *Flood*:

   Penyerang menggunakan beberapa sistem penyerang (atau *tool* yang dapat mensimulasikan *distributed attack*) untuk membanjiri server target dengan *traffic*, bertujuan menghabiskan sumber daya dan mengganggu layanan. Kinerja Snort dalam mendeteksi dan IPtables dalam mencegah serangan DDoS terdistribusi akan dievaluasi.

3. Pengujian pada Serangan SYN *Flood*:

   Penyerang mengirimkan banyak paket SYN palsu ke server target, bertujuan untuk menghabiskan sumber daya server dengan koneksi TCP yang tidak lengkap.

4. Pengujian pada Serangan *Port Scanning*:

   Penyerang menyelidiki server target untuk mengetahui *port* dan layanan yang terbuka, menggunakan teknik pemindaian pada setiap *port*. Hal tersebut menguji kemampuan Snort dalam mengidentifikasi dan IPtables dalam memblokir aktivitas pengintaian.

5. Pengujian pada Serangan IP *Spoofing*:

   Penyerang mengirimkan paket-paket berbahaya menggunakan Alamat IP sumber palsu ke server target untuk menguji kemampuan Snort dalam mendeteksi dan IPtables dalam memblokir paket-paket yang berusaha menyamarkan asal usulnya.

6. Pengujian pada Serangan *Brute Force*:

Penyerang mengirimkan berulang kali permintaan login ke server target dengan menggunakan berbagai kombinasi *username* dan *password*. Tujuannya adalah untuk menguji kemampuan Snort dalam mendeteksi percobaan login yang gagal secara beruntun dan IPtables dalam memblokir alamat IP yang melakukan serangan tersebut.

7. Pengujian pada Serangan *Cross Site Scripting* (XSS):

   Penyerang memasukkan *script* berbahaya ke dalam halaman *web* yang dilayani oleh server target. Bertujuan untuk menilai kinerja Snort dalam mendeteksi dan IPtables dalam memitigasi serangan injeksi kode dari sisi klien pada lalu lintas HTTP.

8. Pengujian pada Serangan SQLi (SQL *Injection*):

   Penyerang memasukkan perintah SQL melalui input aplikasi *web* dengan tujuan untuk manipulasi pada *database* yang dilayani oleh server target.

9. Pengujian pada Serangan *Slow* HTTP:

   Penyerang memasukkan mengirimkan permintaan HTTP yang sangat lambat ke server target melalui *tool slow*http, bertujuan untuk menghabiskan sumber daya server dengan koneksi yang tidak lengkap.

3.5. Evaluasi Kerja Sistem

Untuk memastikan bahwa proses *monitoring* berjalan sesuai harapan dan sistem keamanan berfungsi efektif, beberapa metrik kinerja dievaluasi:

1. Kecepatan Pendeteksian dan Pencegahan

   Waktu yang diperlukan oleh Snort untuk mendeteksi ancaman dan melakukan tindakan pencegahan atau pemblokiran oleh IPtables setelah lalu lintas berbahaya masuk ke jaringan diukur dan dianalisis. Ini juga mencakup waktu IP sumber/tujuan, waktu kejadian, protokol, dan *action* yang diambil oleh Snort dari deteksi hingga data log tiba di Elasticsearch dan dapat divisualisasikan di Grafana.

2. Kinerja Notifikasi Telegram

   Integrasi dengan Telegram dievaluasi dengan mengukur respons *time* dari administrator setelah menerima notifikasi. Hal ini penting untuk menilai

apakah notifikasi Telegram benar-benar mempercepat tindakan pencegahan dan mitigasi, serta memastikan informasi yang disampaikan jelas dan relevan.

3. Akurasi Visualisasi Grafana

Akurasi dan relevansi informasi yang ditampilkan di *dashboard* Grafana dievaluasi untuk memastikan bahwa visualisasi secara tepat merefleksikan kejadian keamanan dan memberikan *insight* yang berguna bagi administrator. Ini juga mencakup validasi data yang masuk dari Logstash ke Elasticsearch.

4. Stabilitas dan Performa Sistem Keseluruhan

Selama proses *monitoring*, stabilitas dan performa keseluruhan dari sistem (Snort, IPtables, Grafana, dan notifikasi Telegram) juga diawasi. Hal ini untuk memastikan bahwa penambahan fitur deteksi, notifikasi, *logging*, dan visualisasi tidak memperlambat server atau menyebabkan masalah operasional lainnya.

3.6. Pelaporan

Hasil penelitian yang telah diperoleh akan didokumentasikan dalam bentuk laporan tertulis berupa skripsi. Laporan hasil pengujian ini berisi hal-hal seperti perancangan sistem, implementasi, dan hasil pengujian. Laporan tersebut dapat menjadi kontribusi terhadap pengembangan sistem keamanan server yang lebih optimal, terutama dalam hal deteksi dan pemblokiran intrusi.