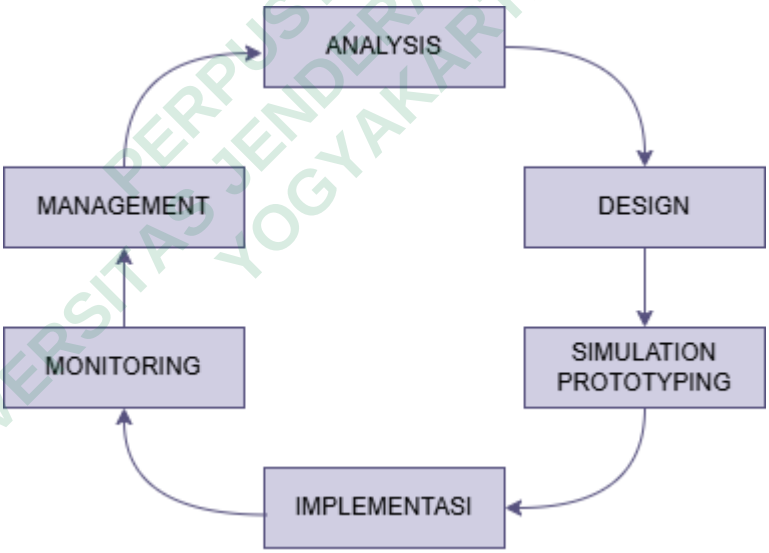
BAB 3
METODE PENELITIAN

Penelitian ini menggunakan menggunakan metode *Network Development Life Cycle* (NDLC) sebagai pendekatan sistematis dalam pengembangan dan implementasi sistem pemantauan keamanan jaringan. Metode NDLC digunakan untuk merancang dan membangun sistem honeypot yang terintegrasi dengan ELK (Elasticsearch, Logstash, Kibana). Dengan NDLC, penelitian ini memastikan bahwa setiap tahap pengembangan sistem, mulai dari analisis kebutuhan, perancangan, prototipe, implementasi, *monitoring* hingga manajemen dilakukan secara terstruktur dan efisien, guna memastikan keandalan serta efektifitas sistem yang dibangun. Penelitian ini bersifat eksperimental dan studi kasus, dimana sistem honeypot dan ELK *Stack* diterapkan langsung pada jaringan UNJAYA.



Gambar 3.1 Metode NDLC

Berdasarkan metode yang digunakan, dalam penelitian ini akan terdiri dari beberapa tahapan sebagai berikut:

1. Analisis Kebutuhan (*Analysis*)

   Tahap pertama dilakukan dengan mengidentifikasi kebutuhan dalam membangun sistem pemantauan keamanan jaringan. Identifikasi dilakukan melalui studi literatur terkait sistem honeypot, termasuk cara implementasi,

konfigurasi, dan integrasi dengan ELK stack yang diperlukan dalam membangun sistem pemantauan keamanan jaringan.

2. Perancangan (*Design*)

   Pada tahap ini dilakukan dengan cara desain arsitektur sistem yang mencakup pemilihan jenis honeypot yang akan digunakan, seperti OpenCanary dan cowrie, serta perancangan integrasi dengan ELK *Stack* untuk pengolahan dan visualisasi data *log*, serta skenario serangan yang mungkin akan terjadi.

3. Simulasi Prototipe (*Simulation Prototyping*)

   Sistem keamanan jaringan mulai dibangun dengan konfigurasi yang lengkap, namun belum diterapkan langsung pada lingkungan jaringan UNJAYA. Simulasi dilakukan dalam lingkungan terisolasi guna memastikan honeypot dan ELK *Stack* berfungsi sesuai yang diharapkan sebelum diterapkan di lingkungan sebenarnya.

4. Implementasi (*Implementation*)

   Setelah melalui tahap simulasi prototipe, sistem honeypot diterapkan langsung pada jaringan kampus UNJAYA. Pada tahap ini, honeypot mulai aktif menangkap lalu lintas jaringan, sementara ELK *stack* memproses, mengolah, dan memvisualisasikan data yang dikumpulkan dari honeypot.

5. *Monitoring*

   Sistem yang telah diimplementasikan akan terus dipantau untuk merekam aktivitas jaringan, mendeteksi potensi serangan, serta mengumpulkan *log* aktivitas yang mencurigakan.

6. Manajemen (*Management*)

   Tahap terakhir bertujuan untuk menilai apakah sistem pemantauan keamanan jaringan yang sudah diimplementasikan berjalan secara optimal. Evaluasi dilakukan dengan meninjau *log* serangan yang terkumpul pada honeypot, serta efisiensi ELK *stack* dalam menyajikan informasi. Jika ditemukan kekurangan atau kebutuhan tambahan, maka akan dilakukan penyesuaian konfigurasi agar sistem berjalan secara optimal.

3.1 Alat dan Bahan Penelitian

3.1.1 Perangkat Keras

Dalam proses implementasi sistem pemantauan keamanan jaringan, kebutuhan perangkat keras menjadi salah satu aspek penting yang harus diperhatikan. Perangkat keras yang digunakan harus mampu mendukung seluruh proses instalasi, penyimpanan log, pemrosesan data, serta visualisasi dalam sistem pemantauan keamanan jaringan yang dirancang. Pada penelitian ini perangkat keras yang digunakan adalah satu unit server fisik yang berfungsi sebagai honeypot. Server ini menggunakan prosesor Intel i3 CPU 3.07 GHz, memori (RAM) sebesar 4 GB, serta penyimpanan internal berkapasitas 512 GB. Server ini terhubung langsung ke jaringan luar melalui IP publik, sehingga dapat meniru layanan jaringan tertentu dan mendeteksi aktivitas mencurigakan dari luar. Selain itu, server ini juga menjalankan *filebeat agent* untuk mengirimkan log serangan yang terdeteksi ke server ELK Stack guna diproses dan divisualisasikan.

3.1.2 Perangkat Lunak

Perangkat lunak merupakan komponen penting dalam sistem yang akan dibangun. Sistem ini menggunakan kombinasi perangkat lunak *open source* untuk memenuhi fungsi honeypot, pengumpulan log, pemprosesan data dan visualisasi.

1. VPS (*Virtual Private Server*)

   Virtual Private Server yang digunakan untuk ELK Stack memiliki spesifikasi sebagai berikut: prosesor dengan 4 VCPU, memori (RAM) sebesar 8192 MB, dan penyimpanan sebesar 20 GB. VPS ini dilengkapi dengan koneksi jaringan yang menggunakan dedicated public IP serta bandwidth yang stabil. Fungsinya adalah untuk menerima dan menyimpan log yang dikirimkan dari server honeypot, kemudian mengolah dan menampilkannya dalam bentuk visualisasi melalui dashboard.

2. *Proxmox VE* (*Virtual Environment*)

   Proxmox VE adalah platform virtualisasi berbasis Debian yang digunakan untuk mengelola server fisik honeypot. Proxmox memungkikan pengguna menjalankan sistem operasi seperti ubuntu server secara virtual,

sekaligus mengelola resource (CPU, RAM, *storage*) dan menyediakan fitur manajemen jarak jauh melalui antarmuka *web*.

3. Ubuntu Server 22.04 LTS

   Merupakan sistem operasi utama yang digunakan pada mesin virtual proxmox untuk menjalankan layanan OpenCanary. Ubuntu server dipilih karena ringan, stabil, dan memiliki dukungan komunitas luas dalam konteks keamanan dan jaringan.

4. Tailscale

   Tailscale adalah layanan jaringan privat yang menggunakan protokol *WireGuard* untuk menciptakan jaringan *overlay* antar perangkat. Dalam penelitian ini, tailscale digunakan untuk menghubungkan perangkat *client* (peneliti) dengan *server* ubuntu yang menjalankan OpenCanary, meskipun berada pada jaringan berbeda. Hal ini memungkinkan peneliti untuk melakukan konfigurasi, pemantauan dan pengujian tanpa harus berada pada jaringan fisik yang sama.

5. OpenCanary

   OpenCanary adalah honeypot ringan yang dirancang untuk meniru berbagai layanan umum seperti SSH, FTP, HTTP dan Lainnya. Ketika layanan palsu ini diakses oleh aktor yang tidak sah, OpenCanary menghasilkan log sebagai bentuk peringatan dini akan potensi serangan.

6. Filebeat

   Filebeat digunakan sebagai agen ringan untuk mengirimkan log yang dihasilkan oleh OpenCanary ke logstash.

7. Logstash

   Logstash berfungsi untuk menerima log dari OpenCanary, melalui parsing atau pemprosesan data, lalu meneruskannya ke Elasticsearch untuk disimpan dan dianalisis.

8. Elasticsearch

   Elasticsearch adalah *database* pencarian berbasis RESTful API yang digunakan untuk menyimpan dan mengindeks log hasil pemrosesan

dari Logstash. Elasticsearch mampu menangani pencarian dan analisis data skala besar dengan cepat.

9. Kibana

   Kibana merupakan *dashboard* visualisasi yang digunakan untuk menampilkan data *log* secara *real-time*. Penggunaan kibana dapat memantau tren serangan, aktivitas mencurigakan, dan pola akses terhadap honeypot melalui berbagai bentuk grafik, *chart* dan berbagai bentuk lain sesuai kebutuhan.

3.1.3 Kebutuhan Jaringan

Dalam perancangan dan implementasi sistem, kebutuhan jaringan merupakan elemen krusial yang menunjang konektivitas antar komponen sistem, serta memastikan proses pengumpulan dan pengiriman log dapat berjalan secara optimal dan aman. Kualitas dan struktur jaringan akan sangat mempengaruhi efektivitas sistem dalam mendeteksi serta merekam aktivitas mencurigakan.

Adapun kebutuhan jaringan dalam penelitian ini adalah sebagai berikut:

1. Koneksi Internet Stabil

   Diperlukan koneksi internet yang stabil untuk mendukung proses pengunduhan paket perangkat lunak, pembaruan sistem, serta komunikasi antar server, khususnya server OpenCanary dan VPS ELK stack. Stabilitas ini penting agar log dapat dikirimkan dan diterima secara real-time tanpa gangguan.

2. Jaringan *Private* (*tailscale*)

   Untuk menjaga keamanan komunikasi internal, digunakan tailscale sebagai solusi jaringan privat berbasis *WireGuard*. Tailscale memungkikan koneksi terenkripsi antar perangkat meskipun berada pada jaringan yang berbeda secara fisik. Hal ini sangat membantu dalam pengelolaan server ubuntu dan proses pemantauan jarak jauh secara aman

3. Alamat IP Statis dan IP Publik.

   Penggunaan alamat IP statis diterapkan pada tahap implementasi sistem di lingkungan jaringan laboratorium. Alamat IP statis memberikan

konsistensi dalam komunikasi antar komponen sistem, sehingga memudahkan proses pengujian, pemantauan, serta *troubleshooting* terhadap alur integrasi antara honeypot (OpenCanary), server log (Logstash), dan visualisasi (Kibana). Stabilitas alamat ini sangat penting agar setiap perangkat dapat saling berkomunikasi tanpa gangguan yang disebabkan oleh perubahan alamat secara dinamis.

Tujuan dari penggunaan IP publik adalah agar server dapat diakses dari jaringan luar, sehingga memungkinkan pengumpulan log dari berbagai aktivitas jaringan eksternal secara langsung dan aktual. Secara khusus, server VPS yang digunakan untuk menjalankan komponen ELK Stack telah menggunakan alamat IP publik sejak awal

4. Lingkungan Jaringan Laboratorium

   Penggunaan sebelum diterapkan pada jaringan publik, sistem terlebih dahulu diuji di lingkungan jaringan laboratorium. Jaringan lab ini bersifat lokal dan terisolasi, digunakan sebagai media simulasi untuk menguji integrasi antar komponen seperti OpenCanary, Logstash, dan visualisasi Kibana. Pengujian dalam lingkungan ini meminimalkan risiko dan memungkinkan perbaikan sistem sebelum diterapkan secara luas.

3.2 Jalan Penelitian

Jalannya penelitian yang akan dilaksanakan dalam implementasi sistem pemantauan keamanan jaringan, mengadopsi langkah dari metode *Network Development Life Cycle* (NDLC) yang terdiri dari enam tahapan. Setiap tahap dilakukan secara sistematis untuk memastikan keberhasilan sistem pemantauan keamanan jaringan di UNJAYA.

3.2.1 Tahap Analisis Kebutuhan

Informasi tentang kebutuhan yang menjadi bagian untuk melakukan implementasi sistem pemantauan keamanan jaringan berada pada tahap ini. Dengan menganalisis permasalahan utama dari objek penelitian ini yaitu UNJAYA dimana belum adanya sistem pemantauan terkait keamanan jaringan, peneliti mencoba

mendalami terkait kebutuhan sistem berdasarkan kondisi dan lingkungan dimana sistem akan diimplementasikan, seperti infrastruktur jaringan yang tersedia, spesifikasi server yang dibutuhkan, sumber daya perangkat lunak yang akan digunakan.

3.2.2 Tahap Perancangan

Perancangan dilakukan dengan melakukan desain sistem pemantauan keamanan jaringan yang mencakup pemilihan jenis honeypot yang akan digunakan, perancangan integrasi honeypot dengan ELK stack, serta pemilihan desain *dashboard* keamanan jaringan yang ada pada Kibana untuk memvisualisasikan data *log* serangan. Pada tahap ini, peneliti akan mencoba membuat gambaran secara terperinci tentang implementasi sistem pemantauan keamanan jaringan dengan berdasarkan topologi jaringan atau infrastruktur jaringan UNJAYA, sehingga mempermudah untuk memasuki tahapan selanjutnya.

3.2.3 Tahap Simulasi Prototipe

Perancangan yang telah dilakukan memberikan gambaran terkait sistem yang akan diimplementasikan yang selanjutnya di tahap ini akan dilakukan pembuatan sistem tersebut. Sistem akan dibuat dalam jaringan terkontrol agar menghindari kemungkinan kegagalan yang menyebabkan hal yang tidak diinginkan. Tujuan dari tahap ini juga untuk memastikan honeypot dapat menangkap aktivitas mencurigakan dan mengirimkan *log* ke ELK stack. Pada tahap ini juga dilakukan pengujian terhadap konfigurasi honeypot untuk memastikan layanan dapat berjalan dengan baik, serta menguji ELK Stack agar data serangan dapat divisualisasikan dengan baik.

3.2.4 Tahap Implementasi

Setelah simulasi prototipe selesai, langkah selanjutnya adalah mengimplementasikan sistem kedalam jaringan UNJAYA untuk mulai menangkap lalu lintas jaringan. Pada tahap ini, dilakukan kebijakan konfigurasi jaringan dan

konfigurasi agar sistem dapat berjalan dengan optimal tanpa mengganggu jaringan infrastruktur jaringan utama.

3.2.5 Tahap *Monitoring*

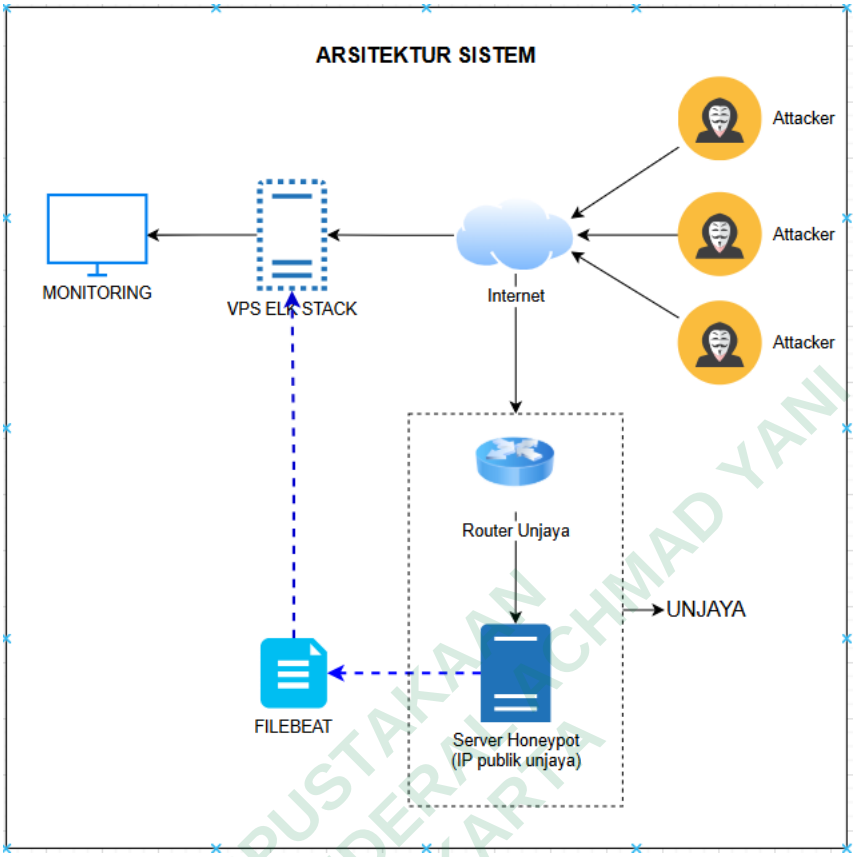
Dilakukan *monitoring* untuk mengawasi sistem yang keamanan pemantauan keamanan jaringan yang telah diimplementasikan, melihat kondisi aktivitas jaringan yang telah ditangkap, mendeteksi serangan yang ditampilkan dalam *dashboard*, serta mencatat informasi apa yang didapatkan. Dalam aktivitas monitoring ini, ditentukan seberapa lama waktu yang dibutuhkan untuk melakukan pemantauan jaringan. Penentuan lama pemantauan jaringan dilakukan dengan berdasar pada beberapa aspek seperti besarnya penyimpanan yang digunakan, data serangan per-hari, dan kemungkinan lain sesuai dengan kebutuhan yang terjadi.

3.2.6 Tahap Manajemen

Manajemen adalah tahap terakhir yang dilakukan pada penelitian ini, berfungsi untuk menilai apakah sistem pemantauan keamanan jaringan yang sudah diimplementasikan berjalan secara optimal. Evaluasi dilakukan dengan meninjau *log* serangan yang terkumpul pada honeypot, serta keinformatifan ELK *stack* dalam menampilkan informasi. Jika ditemukan kekurangan atau kebutuhan tambahan, maka akan dilakukan penyesuaian konfigurasi agar sistem berjalan secara optimal.

3.3 Perancangan Arsitektur Sistem

Perancangan arsitektur sistem yang dirancang untuk menggambarkan sistem yang dikembangkan dalam penelitian ini serta menjelaskan keterkaitan antar komponen dalam implementasi sistem pemantauan keamanan jaringan. Dengan rancangan arsitektur sistem ini dapat dijelaskan bagaimana sistem akan beroperasi dalam mendeteksi, menganalisis, serta memvisualisasikan serangan siber secara efektif guna meningkatkan keamanan siber di UNJAYA.



Gambar 3.2 Rancangan Arsitektur Sistem

Penjelasan berikut memberikan deskripsi mengenai arsitektur sistem untuk memahami alur kerja dan keterkaitan antar komponen dalam sistem pemantauan keamanan jaringan di UNJAYA.

1. *Attacker* (penyerang) dalam arsitektur ini merepresentasikan sistem bot yang berusaha menyerang jaringan UNJAYA. Pihak penyerang dapat menggunakan berbagai metode serangan seperti scanning port, brute force atau metode serangan Lainnya.
2. Internet bertindak sebagai media penghubung antara penyerang dengan infrastruktur UNJAYA.
3. *Router* UNJAYA sebagai penghubung antara infrastruktur jaringan UNJAYA dengan jaringan internet publik. *Router* mempunyai IP publik yang digunakan oleh *server* OpenCanary.
4. Server OpenCanary sebagai jebakan bagi penyerang dengan mensimulasikan berbagai layanan. Karena menggunakan IP publik

UNJAYA, server ini menjadi target serangan yang dirancang menyerupai sever dengan layanan aktif. Setiap interaksi penyerangan dengan server OpenCanary akan dicatat dan diteruskan ke VPS ELK Stack melalui mekanisme *log forwarding*.

5. Filebeat digunakan untuk mengirim dan mentransfer log dari berbagai sumber ke Logstash atau Elasticsearch untuk dianalisis lebih lanjut.
6. VPS ELK *Stack*, sebagai tempat instalasi dan berjalannya ELK *stack* yang akan mendapatkan *log file* dari server OpenCanary. Logstash untuk mengumpulkan dan menerima *log* dari OpenCanary, Elasticsearch menyimpan *log* untuk dianalisis lebih lanjut, Kibana menyediakan visualisasi data dari data *log* yang di kumpulkan.
7. *Dashboard monitoring* untuk menampilkan *dashboard* yang sudah dibuat oleh Kibana dan ditampilkan dalam monitor.

Skema perancangan ini bertujuan untuk memastikan bahwa seluruh log aktivitas yang ditangkap oleh honeypot dapat diteruskan, diproses, disimpan, dan divisualisasikan secara efisien untuk mendukung proses deteksi dan analisis ancaman siber di UNJAYA.

3.4 Pemilihan Jenis Honeypot

Honeypot merupakan elemen utama dalam sistem deteksi dini terhadap ancaman siber. Pemilihannya tidak hanya didasarkan pada banyaknya layanan yang dapat disimulasikan, tetapi juga pada kompatibilitasnya dengan sistem monitoring log, kemudahan implementasi, efisiensi sumber daya, serta keberlanjutan pengembangan perangkat lunak.

Analisis perbandingan dilakukan terhadap beberapa honeypot yang umum digunakan, yakni OpenCanary, *Dionaea*, *Cowrie*. Tabel 3.1 menyajikan perbandingan aspek-aspek penting dari masing-masing honeypot tersebut.

Tabel 3.1 Perbandingan Honeypot

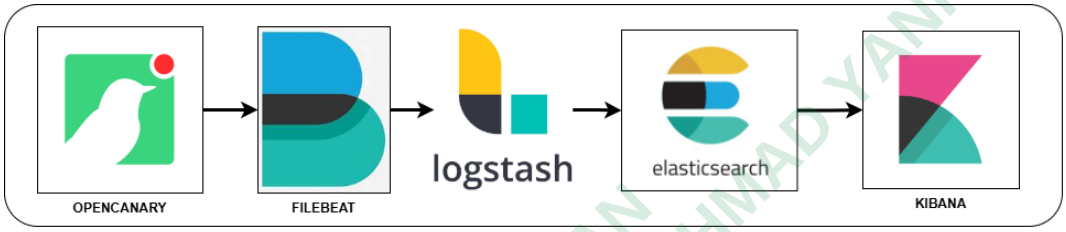
Kriteria	OpenCanary	Dionaea	Cowrie
Kemudahan Konfigurasi	Sangat mudah, cukup melalui file YAML	Kompleks, membutuhkan banyak dependensi	Sedang, berbasis shell dan log
Layanan yang disimulasikan	SSH, FTP, HTTP, MySQL, Telnet, dll.	SMB, HTTP, FTP, SIP, MSSQL	SSH, Telnet dengan shell emulasi
Logging & Output	JSON, Kompatibel dengan filebeat	SQLite, perlu parsing tambahan	File log, format perlu disesuaikan
Kompitabilitas dengan ELK	Sangat Kompatibel	Terbatas, butuh plugin tambahan	Kompatibel, lebih kompleks konfigurasi
Penggunaan Sumber Daya	Ringan	Menengah hingga berat	Menengah
Aktivitas Pengembangan	Aktif dan terdokumentasi baik	Kurang aktif	Aktif

Dari tabel 3.1 Perbandingan Honeypot tersebut dapat disimpulkan bahwa OpenCanary adalah pilihan yang paling tepat untuk implementasi sistem dalam penelitian ini. OpenCanary tidak hanya mudah diinstal dan dikonfigurasi, tetapi juga mendukung berbagai layanan umum yang rentan terhadap eksploitasi, serta sangat kompatibel dengan sistem ELK Stack yang digunakan untuk monitoring. Kemampuannya menghasilkan *output log* dalam format JSON memungkinkan integrasi langsung dengan *Filebeat* tanpa perlu konversi tambahan, sehingga mempercepat alur pemrosesan data. Dengan keunggulan tersebut, OpenCanary dipilih sebagai honeypot utama yang akan digunakan dalam tahap implementasi dan simulasi prototipe sistem pemantauan keamanan jaringan.

3.5 Alur Integrasi Sistem

Sistem pemantauan keamanan jaringan yang dikembangkan dalam penelitian ini dirancang untuk mengintegrasikan fungsi deteksi, pengumpulan,

pengolahan, hingga visualisasi data log serangan siber secara terpusat dan otomatis. Arsitektur sistem ini memadukan honeypot OpenCanary sebagai penghasil log, Filebeat sebagai agen pengirim data, serta ELK Stack (Logstash, Elasticsearch, Kibana) sebagai komponen pemroses dan visualisasi log. Tujuan utama dari integrasi ini adalah membentuk sebuah alur data yang berkesinambungan untuk mendeteksi, mencatat, dan menganalisis aktivitas mencurigakan yang terjadi pada jaringan UNJAYA.



Gambar 3.3 Alur Integrasi Sistem

Alur integrasi sistem ini terdiri atas lima tahapan utama, yaitu: pembuatan log**,** pengiriman log**,** pengumpulan dan pemrosesan data**,** pengindeksan data**,** dan visualisasi data. Berikut adalah penjelasan teknis dari masing-masing tahap:

1. Pembuatan Log (OpenCanary)

   Honeypot OpenCanary dikonfigurasi untuk mensimulasikan berbagai layanan jaringan seperti SSH, FTP, HTTP, *Telnet*, dan lain sebagainya. Ketika terdapat upaya akses terhadap layanan-layanan ini, OpenCanary secara otomatis menghasilkan *log* dalam format JSON. *Log* tersebut disimpan ke dalam file lokal, misalnya pada direktori */var/tamp/*OpenCanary.*log*. Informasi yang tercatat meliputi jenis layanan yang diakses, alamat IP sumber, port tujuan, serta waktu kejadian.

2. Pengiriman Log (Filebeat)

   Filebeat merupakan agen ringan yang dirancang untuk mengirimkan log dari sistem endpoint ke Logstash. Dalam konteks ini, Filebeat dikonfigurasi untuk membaca file log yang dihasilkan oleh OpenCanary secara *real-time*. Selain itu, Filebeat juga dapat menambahkan metadata tambahan seperti nama host, path file log, dan timestamp sebelum log

dikirimkan ke Logstash melalui protokol *Beats input*. Penggunaan Filebeat mempercepat pengiriman log dan mengurangi beban pada sistem honeypot.

3. Pengumpulan dan Pemprosesan Data (Logstash)

   Log yang dikirim oleh filebeat diterima oleh Logstash melalui input *beats*. Selanjutnya, Logstash memproses data log menggunakan *plugin JSON* untuk mengurai struktur data yang berbasis format *JSON*. Proses ini memungkinkan setiap entri log dipisah menjadi field-field terstruktur seperti *src_ip*, *dst_port*, *event_type*, dan *timestamp*, sehingga memudahkan proses pengindeksan di Elasticsearch serta mendukung analisis yang lebih detail dan akurat terhadap aktivitas serangan yang tercatat.

4. Pengindeksan Data (Elasticsearch)

   Log yang telah diproses oleh Logstash selanjutnya dikirim ke Elasticsearch. Di sini, data disimpan sebagai dokumen JSON dalam indeks log yang bersifat *searchable*. Elasticsearch mendukung pencarian cepat dan pengelompokan berdasarkan berbagai parameter, sehingga memudahkan analisis log untuk keperluan deteksi dini dan respons terhadap insiden keamanan.

5. Visualisasi Data (Kibana)

   Kibana digunakan untuk menampilkan data log yang telah diindeks oleh Elasticsearch dalam bentuk *dashboard* interaktif. Administrator jaringan dapat memantau serangan yang terjadi secara *real-time*, melihat tren berdasarkan jenis layanan atau alamat IP, serta menelusuri detail aktivitas mencurigakan dengan tampilan visual yang informatif. *Dashboard* ini menjadi antarmuka utama dalam kegiatan monitoring keamanan jaringan di lingkungan UNJAYA.

3.6 Desain Dashboard

Desain *dashboard* keamanan jaringan di kibana merupakan tahap penting dalam sistem pemantauan keamanan jaringan karena berfungsi sebagai antarmuka utama bagi administrator untuk melakukan analisis, pemantauan, dan deteksi aktivitas mencurigakan secara *real-time*. *Dashboard* ini dibangun dengan

memanfaatkan data log yang telah diproses dan diindeks oleh Elasticsearch dari honeypot OpenCanary.

Perancangan *dashboard* dilakukan dengan mempertimbangkan kebutuhan pengguna dalam melihat informasi yang relevan, mudah dipahami, serta mendukung pengambilan keputusan cepat dalam situasi insiden keamanan. Dalam implementasinya, *dashboard* ini menampilkan beberapa elemen visualisasi utama yang telah dipilih berdasarkan keperluan analisis data log. Berikut komponen visualisasi dashboard yang akan diimplementasikan:

1. Jumlah Serangan

   Komponen ini divisualisasikan dalam bentuk *line chart* yang menunjukkan tren jumlah serangan dalam rentang waktu tertentu. Visualisasi ini membantu dalam mengidentifikasi waktu terjadinya lonjakan serangan dan pola frekuensi aktivitas mencurigakan.

2. Top IP Penyarang (*src_host.keyword*)

   Menampilkan daftar IP address terbanyak yang melakukan interaksi atau upaya serangan terhadap honeypot. Informasi ini penting untuk mengetahui entitas eksternal yang aktif menyerang sistem, dan divisualisasikan dalam bentuk *horizontal bar chart.*

3. Port Tujuan yang sering diserang (*dst_port)*

   Komponen ini menunjukan port layanan yang paling sering ditargetkan oleh upaya serangan. Visualisasi ini membantu dalam mengidentifikasi layanan atau protokol yang menjadi sasaran utama

4. Kategori Log Aktivitas Serangan (*logtype*)

   Merupakan pengelompokan log berdasarkan tipe aktivitas serangan seperti *scanning, login attempts*, atau *malware interaction*. Visualisasi dilakukan dalam bentuk *donut chart* untuk menunjukkan proporsi masing-masing kategori secara intuitif.

5. *Username* yang sering dicoba (*logdata.username.keyword*)

   Komponen ini berupa tabel atau *data table* yang menampilkan daftar *username* yang paling sering digunakan oleh pelaku dalam mencoba

mengakses sistem. Data ini penting untuk mendeteksi serangan brute force dan kredensial umum yang sering disalahgunakan

6. Negara Penyerang (*geoip.country_name.keyword*)

   Visualisasi ini menunjukkan distribusi serangan berdasarkan negara asal penyerang yang teridentifikasi dari IP address menggunakan field *geoip.country_name*. Komponen ini divisualisasikan dalam bentuk *vertical bar chart* untuk mempermudah analisis geografis sumber ancaman.

7. Peta Negara Penyerang

   Komponen ini menampilkan representasi geografis dari asal serangan yang terdeteksi, menggunakan pemetaan lokasi berdasarkan alamat IP melalui *field GeoIP*. Visualisasi dilakukan dalam bentuk peta interaktif (*Geo Map*) yang menunjukkan sebaran titik-titik asal penyerang di berbagai negara. Peta ini memberikan gambaran visual yang intuitif mengenai distribusi global aktivitas serangan, serta memudahkan dalam mengidentifikasi wilayah dengan frekuensi serangan tertinggi secara spasial.

Desain *dashboard* ini merupakan komponen akhir dari pipeline integrasi honeypot dan ELK Stack. Dengan menggabungkan data yang telah dianalisis dan divisualisasikan secara *real-time*, *dashboard* ini memungkinkan pengawasan menyeluruh terhadap ancaman jaringan yang terjadi di lingkungan UNJAYA. Selain itu, *dashboard* juga mempermudah pelaporan serta perencanaan strategis untuk peningkatan sistem keamanan jaringan ke depannya.