BAB IV
PERANCANGAN

4.1 Analisis Sistem Terdahulu

Pencegahan akses terhadap URL berbahaya merupakan salah satu upaya dalam lingkup keamanan siber, untuk memastikan pengguna dalam jaringan tetap terjaga dari serangan seperti *phishing*, *malware*, ransomware dan sebagainya. Pendekatan yang umum dilakukan adalah memantau sistem SIEM secara manual, serta berpotensi melewatkan setiap aktivitas log pada jaringan yang terjadi dalam jumlah banyak dan dalam waktu yang cepat. Dampak yang terjadi pada pendekatan umum ini, selain dapat melewatkan informasi log dalam kurun waktu yang cepat, berpotensi tidak dapat memantau log dengan efektif, serta memerlukan waktu lebih untuk melihat dan menganalisa log yang sudah terlewat satu per satu.

Model LLM merupakan salah satu model dalam machine learning yang berfungsi untuk memproses, memahami serta mengidentifikasi pola dalam bentuk teks, yakni dalam penelitian ini adalah mengidentifikasi pola penulisan URL. Pendekatan dengan model LLM berbasis BERT pada penelitian ini, selain dapat memprediksi potensi adanya URL yang berbahaya namun belum diketahui, dapat membantu mencegah serta mendeteksi adanya URL berbahaya secara otomatis.

Pada penelitian yang dilakukan oleh Chang, et al (2022) dan Elsadig, et al (2022), model LLM berbasis BERT terbukti dengan memproses URL untuk mengelompokkan secara efektif dan melatih model BERT untuk mengklasifikasikan karakteristik string pendek, metode ini mencapai akurasi 98,30%, tingkat penarikan 95,21%, dan nilai F1 94,33%.

4.2 Spesifikasi Kebutuhan Sistem Baru

Untuk memastikan sistem berjalan dengan baik, diperlukan spesifikasi yang sesuai dengan rekomendasi dari sistem yang dijalankan. Pelatihan model Large Language Model dilakukan untuk mempelajari dan memprediksi potensi dari URL yang berbahaya, pelatihan model ini menggunakan dataset. Spesifikasi ini terdiri dari perangkat keras (*hardware*) dan perangkat lunak (*software*).

4.2.1 Spesifikasi Kebutuhan Perangkat Lunak

Berikut adalah komponen perangkat lunak yang digunakan dalam penelitian ini, beserta dengan rincian yang diperlukan.

Tabel 4.1 Tabel Kebutuhan Perangkat Lunak

No.	Perangkat Lunak	Deskripsi
1.	Visual Studio Code	Aplikasi untuk merancang kode pemrograman dan melatih model *Large Language Model*.
2.	Virtual Box	Aplikasi untuk menjalankan sistem SIEM dalam bentuk mesin virtual.
3.	ISO Wazuh	Sistem operasi SIEM yang digunakan untuk sistem deteksi, blokir dan logging.
4.	Web Browser	Aplikasi untuk menjelajahi situs web untuk mencari informasi.
5.	Python	Bahasa pemrograman yang digunakan untuk melatih model.

4.2.2 Spesifikasi Kebutuhan Perangkat Keras

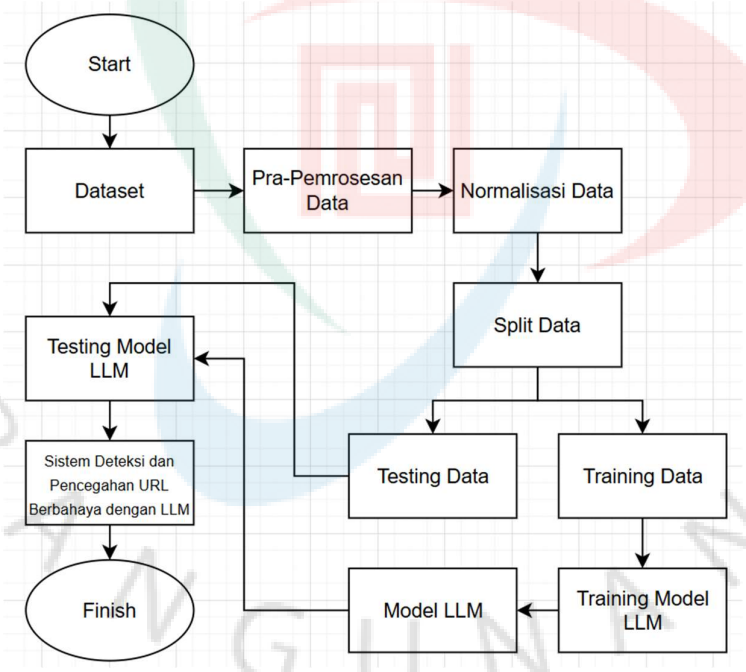
Berikut adalah komponen perangkat keras yang digunakan dalam penelitian ini, beserta dengan spesifikasi dari setiap perangkat.

Tabel 4.2 Tabel Kebutuhan Perangkat Keras

No.	Nama Perangkat Keras	Spesifikasi
1.	*Processor*	AMD Ryzen™ 5 2500U
2.	*RAM*	16 GB
3.	*Storage*	512 GB SSD + 1 TB HDD
4.	*VGA*	NVIDIA GTX 1050 Mobile 4 GB

4.3 Pembuatan Model

Perancangan pembuatan sistem untuk mencegah akses URL berbahaya menggunakan dan melatih model Large Language Model, dengan dataset yang berisi daftar situs-situs yang terindikasi berbahaya. Proses pelatihan model terdiri dari tahapan pada gambar berikut.



Gambar 4.1 Tahapan Penerapan Algoritma

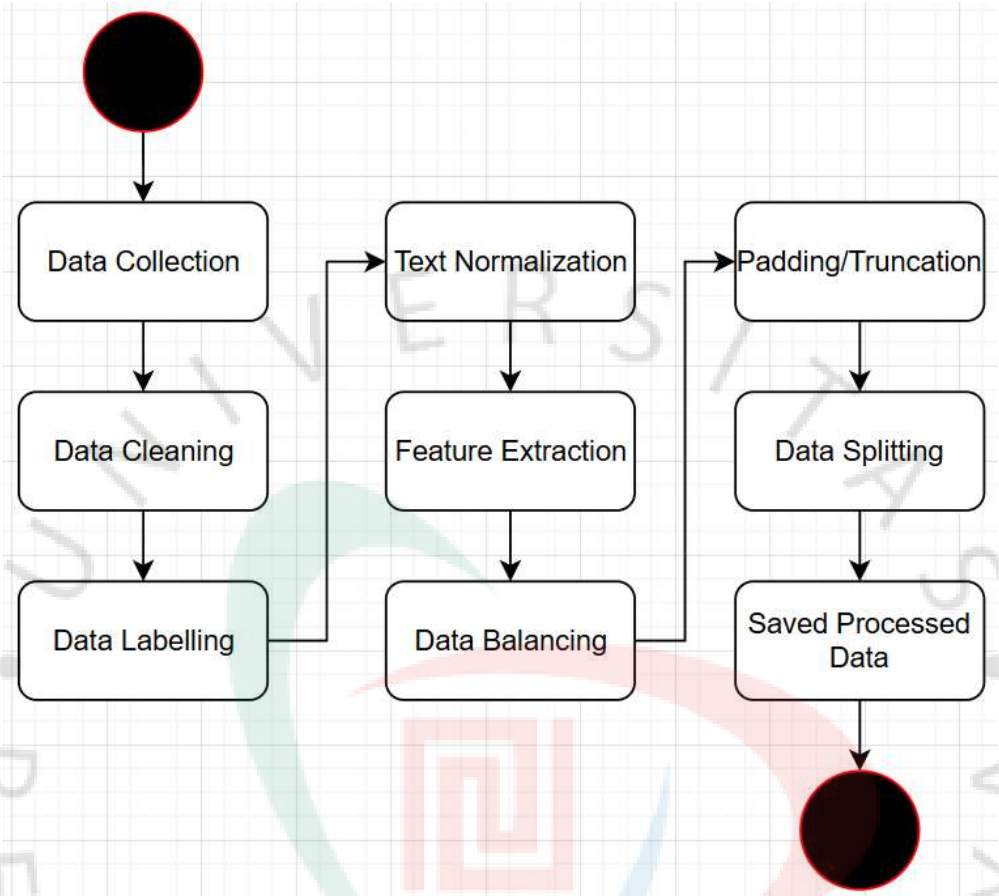
Melalui Gambar 4.1, tahap pembuatan model adalah prosedur dalam pengembangan sistem deteksi dan pencegahan akses pada URL berbahaya, menggunakan model Large Language Model dari dataset berisi daftar situs-situs yang terindikasi berbahaya. Prosedur tahap pembuatan model meliputi pengambilan dataset, pelatihan model dan pengujian model. Berikut penjelasan terkait langkah-langkah pada gambar diatas.

4.3.1 Dataset

Sumber dataset yang diperlukan berasal dari situs web kaggle.com, yang memiliki parameter data sesuai dengan kebutuhan dalam penelitian ini. Dataset ini bernama "*malicious*_phish.csv", parameter dataset terdiri dari; URL dan *Type* dengan total sebanyak 651.207 data. Parameter URL mengacu pada penulisan dari setiap situs web pada umumnya di internet. Selanjutnya, parameter *Type* mengacu pada sifat situs web yang terdaftar, yakni; *Benign* (Aman, dengan jumlah sebanyak 428.103 data), *Defacement* (Berbahaya, dengan jumlah sebanyak 96.457 data), *Phishing* (Berbahaya, dengan jumlah sebanyak 94.111 data) dan *Malware* (Berbahaya, dengan jumlah sebanyak 32.520 data). Kemudian, dataset ini digunakan untuk melatih model Large Language Model untuk mengidentifikasi pola penulisan URL, mengelompokkan URL yang bersifat aman dan berbahaya, serta melakukan prediksi adanya potensi URL yang berbahaya namun belum teridentifikasi.

4.3.2 *Pre-Processing Data* (Pra-Pemrosesan Data)

Perancangan model memerlukan kebutuhan data yang sesuai untuk memastikan sistem berjalan dengan baik. Langkah-langkah pre-processing data terdiri dari; pengumpulan data, pembersihan data, pemberian label pada data, normalisasi data dan penyimpanan data yang sudah diproses. *Pre-processing* dilakukan untuk menyiapkan dataset yang berisi situs web aman dan tidak aman, serta memastikan data yang diproses sesuai dengan kriteria yang dibutuhkan. Berikut ini adalah gambar alur dalam *pre-processing* data.



Gambar 4.2 Pre-processing

Merujuk pada gambar 4.2, tahap-tahap pre-processing dapat dijelaskan sebagai berikut;

a) *Data Collection*

Langkah pertama pada pre-processing adalah mengumpulkan data mentah dari berbagai sumber, yakni pada penelitian ini berasal dari situs web kaggle.com. Data diambil dengan format file CSV, yang kemudian menjadi berkas untuk melatih model.

b) *Data Cleaning*

Tahap data cleaning adalah proses pembersihan data mentah dari elemen-elemen yang tidak relevan, seperti data duplikasi, nilai kosong dan kesalahan format yang dapat mengganggu kinerja model.

c) *Data Labeling*

Dalam model *supervised learning*, yakni dalam penelitian ini berupa Large Language Model, setiap data perlu diberikan label menurut target klasifikasinya, untuk menentukan URL yang diklasifikasikan sebagai "aman" (0) atau berbahaya (1).

d) *Text Normalization*

Tujuan *text normalization* adalah membuat format data teks menjadi seragam, dalam hal ini; mengubah semua huruf menjadi huruf kecil, menghapus karakter khusus dan melakukan tokenisasi agar dapat diolah dengan model berbasis teks.

e) *Feature Extraction*

Data yang sudah ditokenisasi perlu dijadikan representasi numerik, guna melanjutkan proses oleh algoritma machine learning. Pada penelitian ini, tokenizer berbasis model LLM menggunakan BERT Tokenizer.

f) *Data Balancing*

Tahap ini adalah proses penyesuaian distribusi jumlah data antar kelas agar model yang dilatih tidak mengalami bias. Teknik yang digunakan pada tahap ini yaitu;

- *Oversampling*: Jika kelas data URL "aman" lebih sedikit dari kelas data URL "berbahaya", maka teknik ini digunakan untuk menambahkan data yang memiliki jumlah lebih sedikit (yakni URL "aman") untuk menyeimbangkan jumlah data yang lebih banyak (yakni URL "berbahaya").
- *Undersampling*: Teknik ini berfungsi untuk mengurangi jumlah data yang lebih banyak (yakni URL "berbahaya") dengan URL "aman", guna menyeimbangkan jumlah data yang selanjutnya akan dilatih oleh model.

- *Class Weighting*: Berguna untuk memberikan bobot yang lebih besar pada jumlah data yang lebih sedikit, agar model lebih memperhatikan kesalahan pada jumlah data yang lebih sedikit.

g) *Padding/Truncation*

Padding/Truncation digunakan untuk menyamakan/menyeragamkan panjang input teks agar dapat diproses oleh BERT. Transformer memerlukan input dengan panjang yang seragam, maka URL yang memiliki panjang tulisan yang kurang dari batas maksimal akan diberikan padding, atau penambahan token khusus. Batas maksimal token adalah sebanyak 64, jika URL pada data hanya berisi "www.contohwebsite.com" (jumlah token tidak mencapai 64), maka URL tersebut akan dibuat menjadi "[CLS] www . contohwebsite . com [SEP] [PAD] [PAD] [PAD]….." hingga mencapai 64 token. Sedangkan jika URL pada data berisi www.contohwebsite.com/login/1/2/3/4/5/6/7/8/9/0..... (jumlah token melebihi 64), maka URL tersebut akan dipotong hingga panjang maksimal yang ditentukan menggunakan Truncation (pemotongan).

h) *Data Splitting*

Data splitting berperan untuk membagi dataset menjadi tiga bagian yakni; pelatihan data (training). validasi data (validation, dan pengujian data (testing). Langkah ini diperlukan untuk memastikan model dilatih secara bertahap dan dapat dievaluasi.

i) *Saved Pre-processed Data*

Setelah pre-processing selesai, data disimpan dalam bentuk format tertentu agar dapat digunakan kembali untuk pelatihan model tanpa harus pre-processing dari awal kembali.

4.3.3 Normalisasi Data

Normalisasi data digunakan untuk melakukan normalisasi atau diseragamkan, pemerataan panjang URL dengan *padding* dan *truncation* untuk

membatasi panjang URL agar dapat disesuaikan dengan input yang diperlukan oleh BERT.

4.3.4 Split Data

Langkah split data digunakan untuk memisahkan data untuk dibagi ke dalam bentuk subset, guna untuk melatih model. Subset data ini terbagi menjadi 2 bagian yakni, data latih (*training set*) dan data uji (*testing set*) dalam bentuk proporsi 80:20. 80% data digunakan model untuk mempelajari pola, hubungan dan karakteristik pada data, guna memahami konteks data dengan baik. 20% data disisihkan selama pelatihan model, guna untuk menguji kinerja model secara objektif. Hasil dari pengujian kinerja model ini akan membantu dalam generalisasi terhadap data baru yang akan diberikan selanjutnya untuk pelatihan.

4.3.5 Training Data

Training data adalah langkah dalam pelatihan untuk model mengidentifikasi label "aman" dan "berbahaya" dalam dataset URL. Disini model mempelajari pola, struktur dan ciri-ciri pada URL yang teridentifikasi aman dan berbahaya. Semakin baik data yang diolah/diberikan, efektifitas model dalam mengenali dan mengklasifikasikan URL semakin baik.

4.3.6 Training Model LLM

Model Large Language Model (LLM) berupa BERT dilatih dalam metode training data. Proses pelatihan ini dimulai dari perubahan URL menjadi bentuk representasi numerik menggunakan tokenizer, dan diproses oleh modl untuk mempelajari konteks kata atau karakter dalam URL. Proses ini bertujuan agar model dapat mengenal dan membedakan karakteristik URL berbahaya dan URL aman secara otomatis, dengan berdasarkan pola yang dipelajari dalam proses pelatihan.

4.3.7 Model LLM

Setelah selesai dilatih, model LLM yang sudah belajar dari data pelatihan disimpan sebagai model yang siap dipakai. Model yang telah dilatih sudah dapat melakukan prediksi dan menentukan URL yang aman atau berbahaya, serta akan diintegrasikan ke dalam SIEM sebagai sistem deteksi.

4.3.8 Testing Data

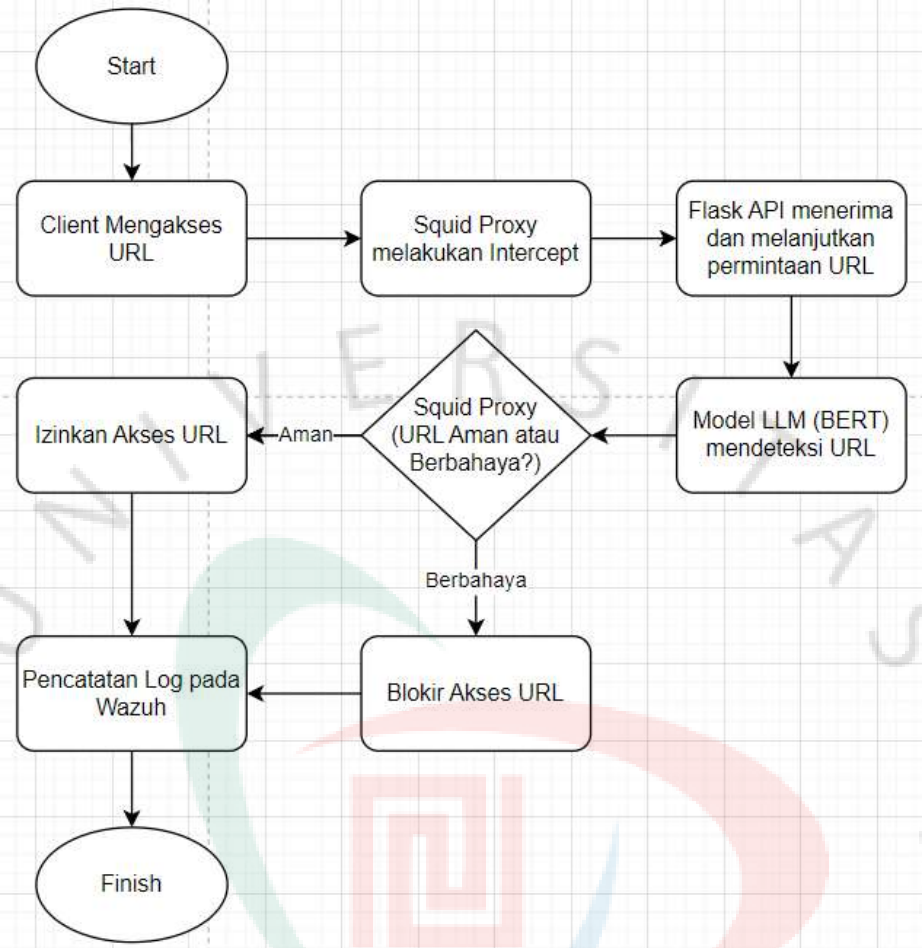
Testing data adalah proses untuk melakukan evaluasi performa model, dengan cara mengambil bagian dari dataset yang tidak digunakan selama proses pelatihan. Hasil dari evaluasi dengan testing data ini dapat memberikan gambaran objektif terhadap seberapa baik model dalam menggeneralisasi dan menerapkan prediksi terhadap data yang baru. Testing data juga diberikan label sehingga hasil prediksi dapat dibandingkan dengan hasil sebenarnya.

4.3.9 Testing Model LLM

Model LLM telah dilatih dan diuji menggunakan data pengujian (testing data). Tahap ini digunakan untuk menguji kinerja model dalam mendeteksi URL berbahaya berdasarkan data yang belum diketahui. Metrik evaluasi yang digunakan adalah akurasi, *precision*, *recall* dan *F1-score* untuk memastikan model dapat mengklasifikan URL dengan baik dan efektif.

4.3.10 Sistem Deteksi dan Pencegahan URL berbahaya dengan LLM

Sesudah melakukan uji kinerja dan evaluasi pada model, langkah selanjutnya adalah mengintegrasikan model LLM yang sudah dilatih ke dalam sistem SIEM melalui API Flask, bersamaan dengan Squid Proxy sebagai sistem pemblokiran jika terdeteksi adanya akses pada URL berbahaya. Berikut ini adalah alur pemblokiran jika terjadi adanya akses pada URL berbahaya.



Gambar 4.3 Proses Pemblokiran URL

Merujuk pada Gambar 4.3, berikut adalah penjelasan proses blokir URL yang dilakukan. Pertama, client mengakses URL menggunakan aplikasi web browser, kemudian URL yang diakses oleh client akan dicegat/intercept oleh Squid Proxy. Squid proxy melanjutkan permintaan URL oleh client kepada Flask API, yang bertugas memanggil model LLM (BERT) untuk mengklasifikasikan URL yang diakses oleh client. Jika hasil klasifikasi dari model LLM (BERT) dinyatakan aman, maka URL yang diminta mendapatkan izin akses. Jika hasil klasifikasi dari model LLM (BERT) dinyatakan berbahaya/*malicious*, maka izin akses URL tersebut ditolak. Kemudian, semua aktivitas tersebut dicatat pada log dalam Wazuh untuk dianalisa. Berikut ini adalah gambar pemicu payload untuk memeriksa sifat URL yang diakses.

Gambar 4.4 Payload Pemicu

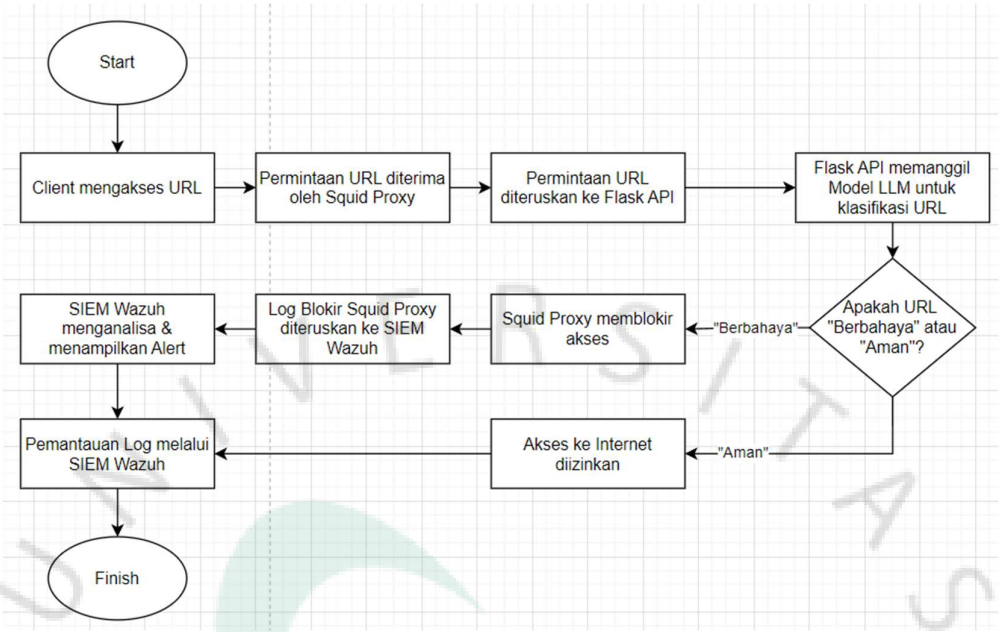
Berikut adalah penjelasan pada Gambar 4.4;

- curl -X POST: Mengirim HTTP request dengan metode POST.
- http://192.168.1.100:5000/predict: Alamat IP dan port Flask API + endpoint /predict untuk klasifikasi URL.
- -H "Content-Type: application/json": Header HTTP yang menunjukkan bahwa data yang dikirim berbentuk JSON.
- -d '{"URL":"br-icloud.com.br"}': Data JSON yang dikirim ke model, yaitu URL br-icloud.com.br (dicurigai sebagai URL *phishing*/*malicious*).
- prediction: "*Malicious*" adalah hasil dari Model BERT mengklasifikasikan URL tersebut sebagai URL berbahaya.
- URL: "br-icloud.com.br" merupakan URL yang diuji/diberikan sebagai input.

4.4 Perancangan Sistem

Pengembangan sistem pada penelitian ini memerlukan rancangan yang terstruktur untuk mengembangkan sistem deteksi dan pencegahan akses pada URL berbahaya. Rancangan ini akan direpresentasikan dalam bentuk; *flowchart*, *activity gambar* dan *use case gambar*.

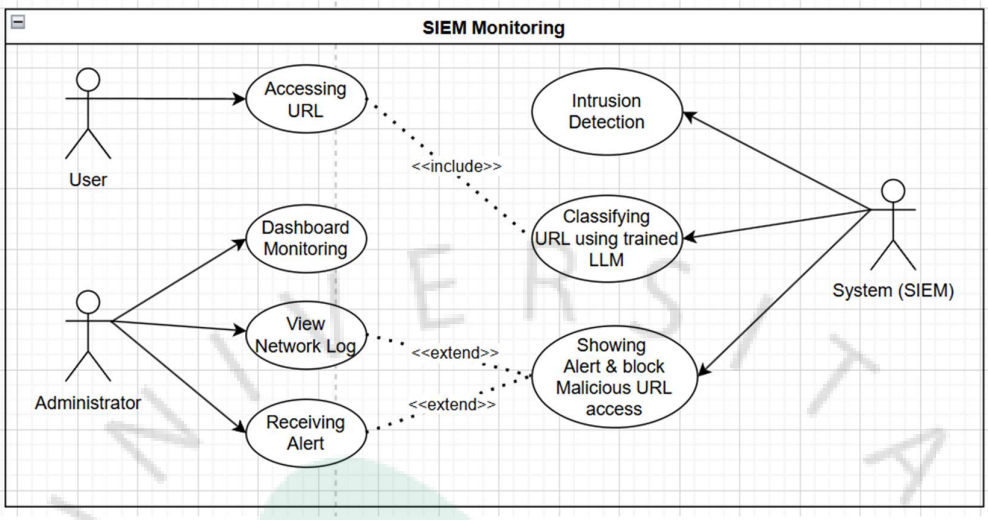
4.4.1 Flowchart Proses Sistem Deteksi dan Blokir



Gambar 4.5 Flowchart Sistem Deteksi dan Blokir

Merujuk pada Gambar 4.5 Flowchart Sistem Deteksi dan Blokir, rangkaian proses menunjukkan cara kerja sistem dalam melakukan deteksi dan blokir pada URL berbahaya. Langkah awal yang dilakukan adalah ketika pengguna mengakses suatu URL, maka permintaan akses URL diterima oleh Squid Proxy, dilanjutkan kepada Flask API yang memanggil model LLM untuk melakukan klasifikasi terhadap URL yang diakses. Jika LLM tidak mendeteksi adanya URL berbahaya, maka akses ke internet diizinkan. Jika model LLM mendeteksi adanya URL berbahaya yang diakses, maka Squid Proxy memblokir akses pada URL berbahaya, log blokir diteruskan ke SIEM Wazuh dan menampilkan peringatan/*alert*. Kemudian, semua log akses terhadap URL aman dan berbahaya, tetap dapat dipantau pada SIEM Wazuh.

4.4.2 Use Case Gambar



Gambar 4.6 Use Case

Gambar 4.6 memvisualisasikan cara kerja sistem dengan use case gambar dalam sistem deteksi dan blokir akses pada URL berbahaya. Terdapat 3 actor dalam sistem ini yakni; User, Administrator dan System. User adalah actor/pengguna yang mengakses URL pada aplikasi penjelajahan situs web. Administrator adalah actor yang bertugas memantau dashboard SIEM, melihat log pada jaringan melalui SIEM dan menerima peringatan/alert jika System (SIEM) mendeteksi adanya akses pada URL berbahaya. Berikut adalah tabel skenario penggunaan use case.

Tabel 4.3 Accessing URL

1.	Use Case:	Accessing URL
2.	Deskripsi:	User melakukan akses internet
3.	Aktor:	User
4.	Action:	1. User membuka Web Browser. 2. User mengakses URL secara acak/sesuai keinginan.

Tabel 4.4 Dashboard Monitoring

1.	Use Case:	Dashboard Monitoring
2.	Deskripsi:	Administrator melakukan monitoring pada SIEM
3.	Aktor:	Administrator
4.	Action:	1. Administrator membuka SIEM melalui Web Browser. 2. Administrator memantau Dashboard Monitoring pada SIEM.

Tabel 4.5 View Network Log

1.	Use Case:	View Network Log
2.	Deskripsi:	Administrator melihat Network Log pada SIEM
3.	Aktor:	Administrator
4.	Action:	1. Administrator membuka SIEM melalui Web Browser. 2. Administrator melihat Network Log pada SIEM.

Tabel 4.6 Receiving Alert

1.	Use Case:	Receiving Alert
2.	Deskripsi:	Administrator menerima peringatan/alert pada SIEM
3.	Aktor:	Administrator
4.	Action:	1. Administrator membuka SIEM melalui Web Browser. 2. Administrator menerima peringatan/alert yang dikirim oleh SIEM jika terdeteksi adanya akses pada URL berbahaya.

Tabel 4.7 Intrusion Detection

1.	Use Case:	Intrusion Detection
2.	Deskripsi:	System (SIEM) memantau aktivitas pada lalu lintas jaringan
3.	Aktor:	System (SIEM)
4.	Action:	System (SIEM) melakukan deteksi intrusi dan menjalankan model LLM berdasarkan file ossec.conf dan local_rules.xml yang telah dikonfigurasi

Tabel 4.8 Classifying URL using trained LLM

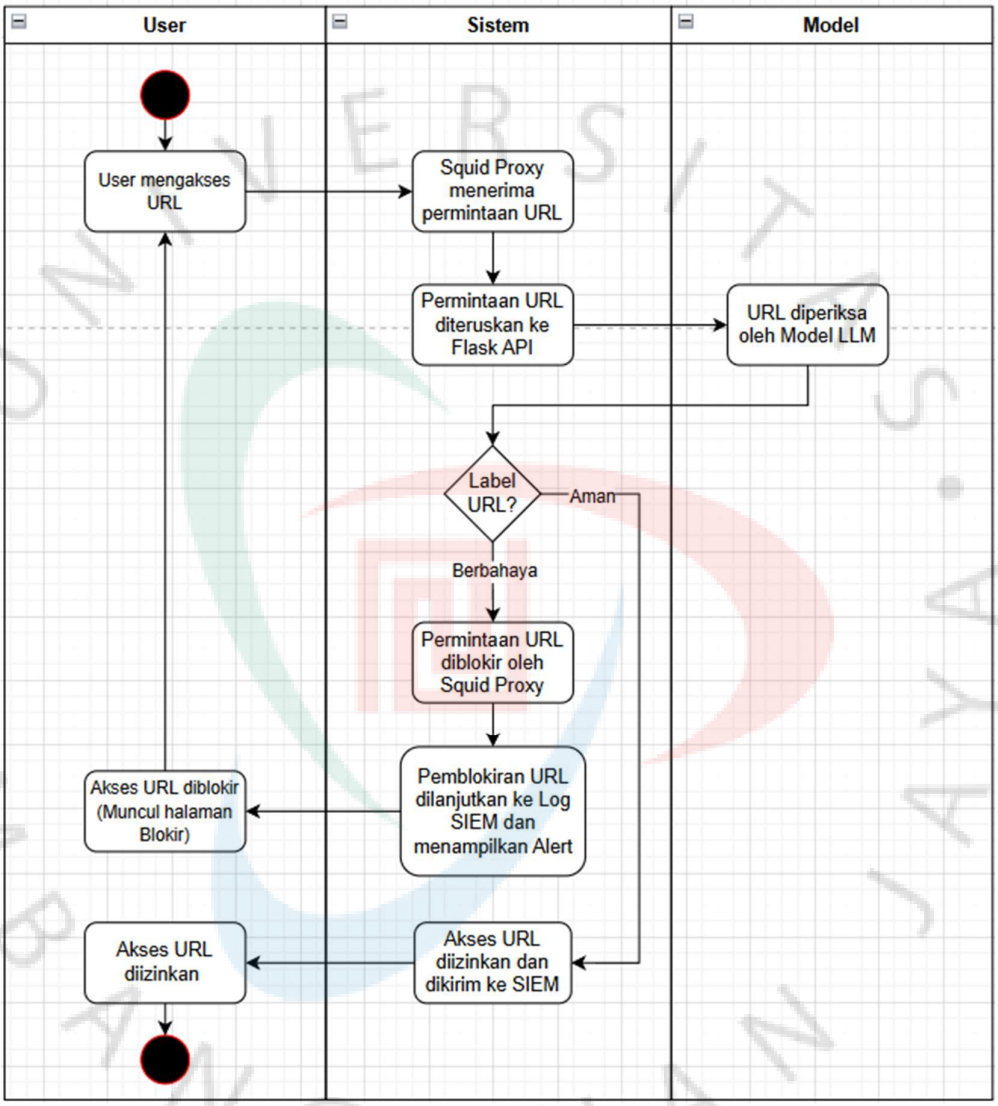
1.	Use Case:	Classifying URL using trained LLM
2.	Deskripsi:	System (SIEM) mengklasifikasikan URL yang diakses oleh User
3.	Aktor:	System (SIEM)
4.	Action:	1. System (SIEM) menjalankan service "llm_*malicious*_block" dengan Flask API yang sudah dikonfigurasi pada file ossec.conf. 2. Model LLM mengklasifikasikan URL yang diteruskan oleh F;ask. Jika URL terdeteksi aman, maka hasil deteksi "*malicious*", jika URL terdeteksi berbahaya, maka hasil deteksi "benign".

Tabel 4.9 Showing Alert & block *Malicious* URL access

1.	Use Case:	Showing Alert & block *Malicious* URL access
2.	Deskripsi:	System (SIEM) menunjukkan peringatan akses dan blokir pada URL berbahaya
3.	Aktor:	System (SIEM)
4.	Action:	1. Squid Proxy mendeteksi adanya akses terhadap URL Berbahaya melalui verifikasi dari Flask API yang menjalankan "llm_*malicious*_block" dengan model LLM 2. Akses pada URL berbahaya yang diakses oleh User diblokir dan menunjukkan halaman tidak tersedia/terblokir oleh Flask API

4.4.3 Activity Gambar

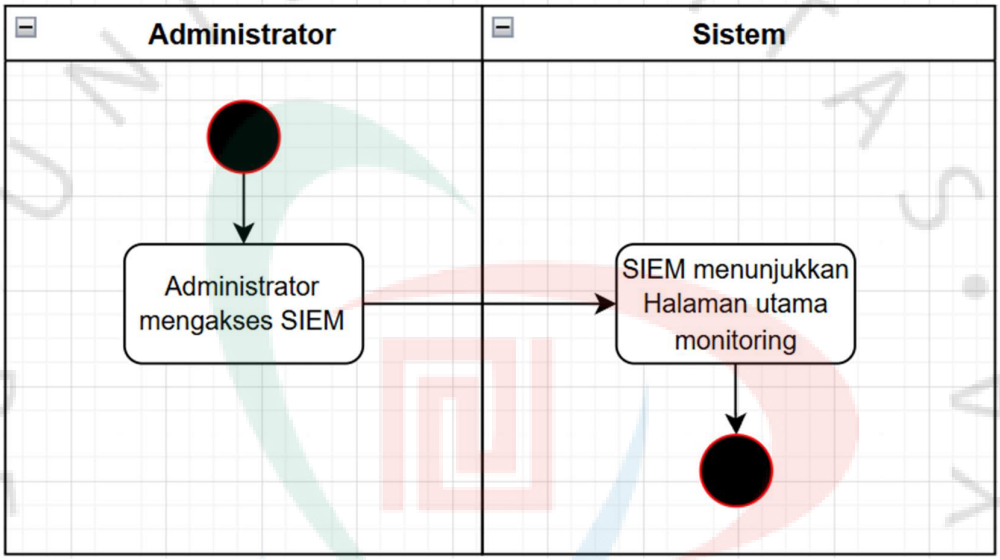
Activity gambar pada penelitian ini berfungsi untuk menjabarkan urutan aktivitas dalam use case yang sebelumnya telah dibuat. Berikut adalah semua activity gambar beserta penjelasannya.



Gambar 4.7 Activity Diagram: User

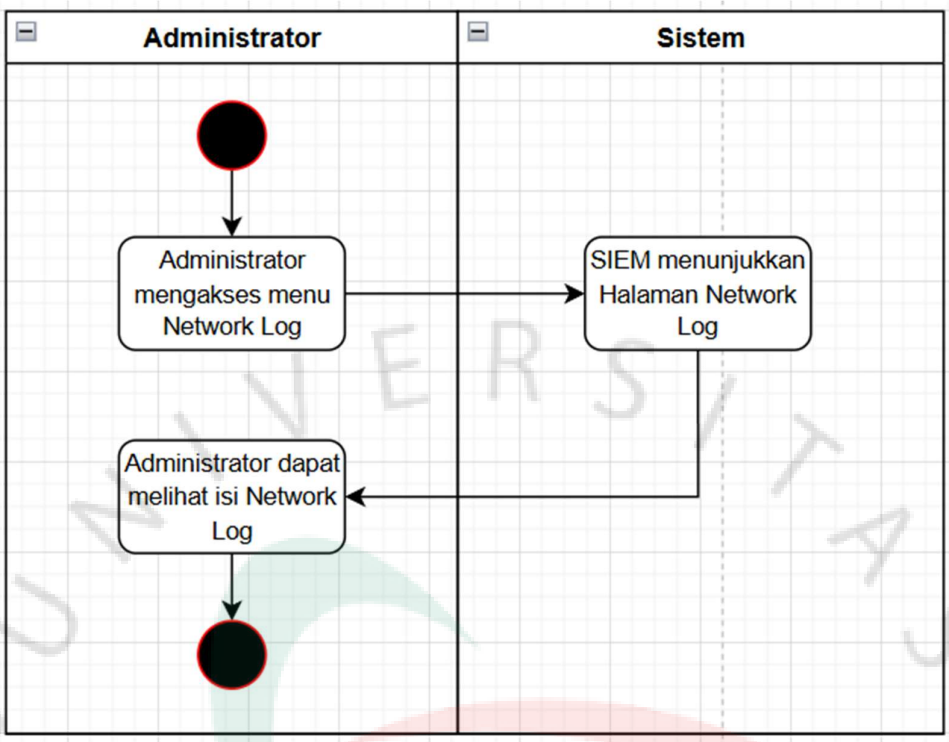
Merujuk pada Gambar 4.7, ada tiga peran dalam sistem deteksi dan pencegahan terhadap akses URL berbahaya yaitu; User, Sistem dan Model. Langkah pertama dimulai dari ketika User mengakses URL apapun dalam aplikasi web browser, lalu permintaan akses URL oleh User diteruskan ke Squid Proxy sebagai pencegat/*interceptor*. Kemudian, Squid Proxy melanjutkan permintaan

URL kepada Flask API untuk memanggil Model LLM yang berperan dalam mengklasifikasikan sifat URL. Ketika Model LLM sudah menentukan sifat dari URL yang diminta, maka jika terdeteksi berbahaya Squid Proxy memblokir akses, mengirimkan pesan blokir kepada SIEM, memberikan peringatan/*alert* terkait URL yang diminta dan menunjukkan halaman Error pada User. Jika URL terdeteksi aman, maka Squid Proxy memperbolehkan akses, mengirimkan pesan perizinan akses kepada SIEM dan mengizinkan User untuk mengakses URL yang terdeteksi aman.



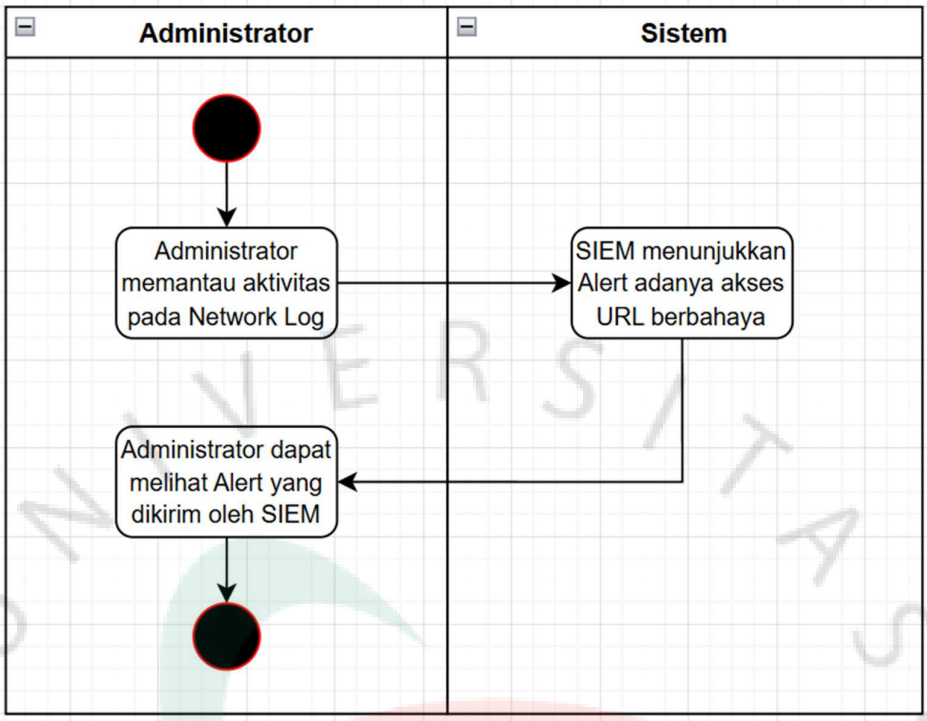
Gambar 4.8 Activity Diagram: Dashboard Monitoring

Gambar 4.8 menunjukkan aktivitas administrator dalam mengakses sistem SIEM. Pertama, administrator menggunakan web browser dengan memanggil alamat IP milik SIEM, kemudian masuk pada sistem dan SIEM menunjukkan dashboard monitoringnya.



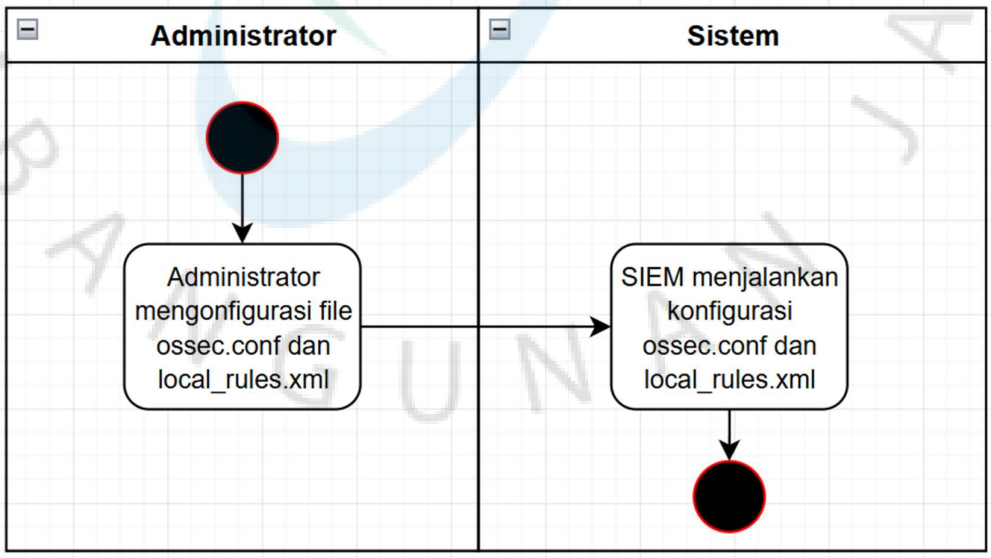
Gambar 4.9 Activity Diagram: View Network Log

Merujuk pada Gambar 4.9, proses diatas menunjukkan proses administrator mengakses menu “Network Log” pada SIEM untuk memantau aktivitas yang terjadi pada jaringan.



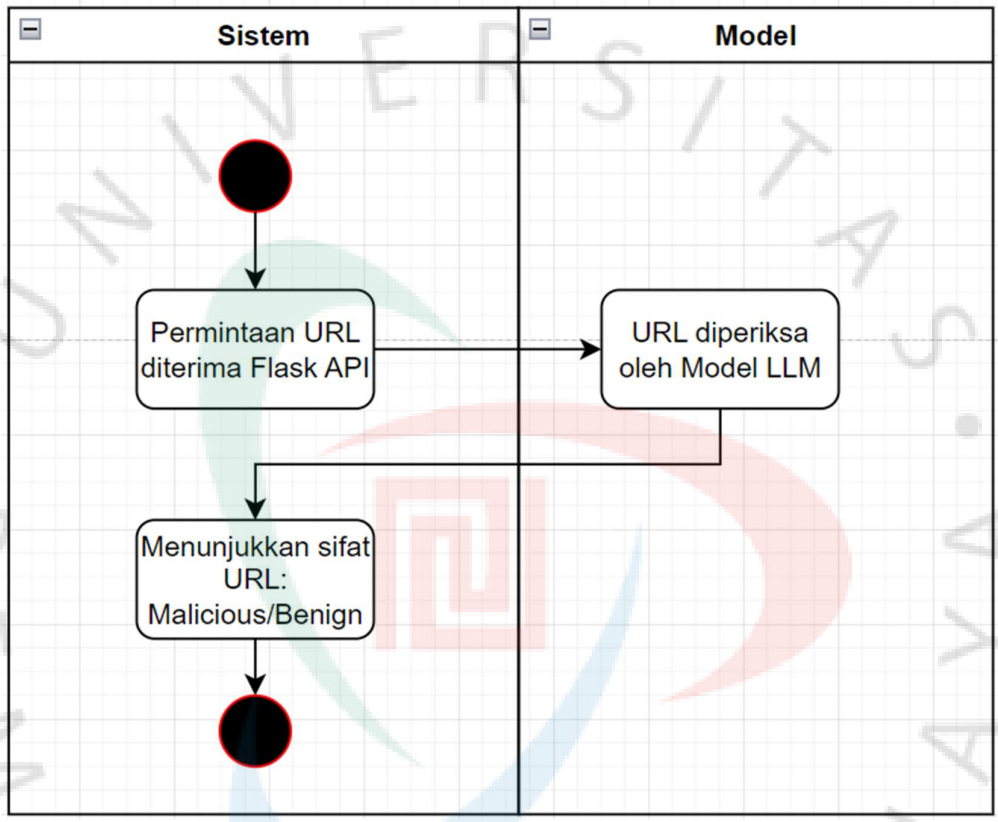
Gambar 4.10 Activity Diagram: Receiving Alert

Gambar 4.10 menjelaskan alur proses ketika administrator menerima alert/peringatan jika terdeteksi adanya akses pada URL berbahaya dengan cara melihat pada menu Network Log.



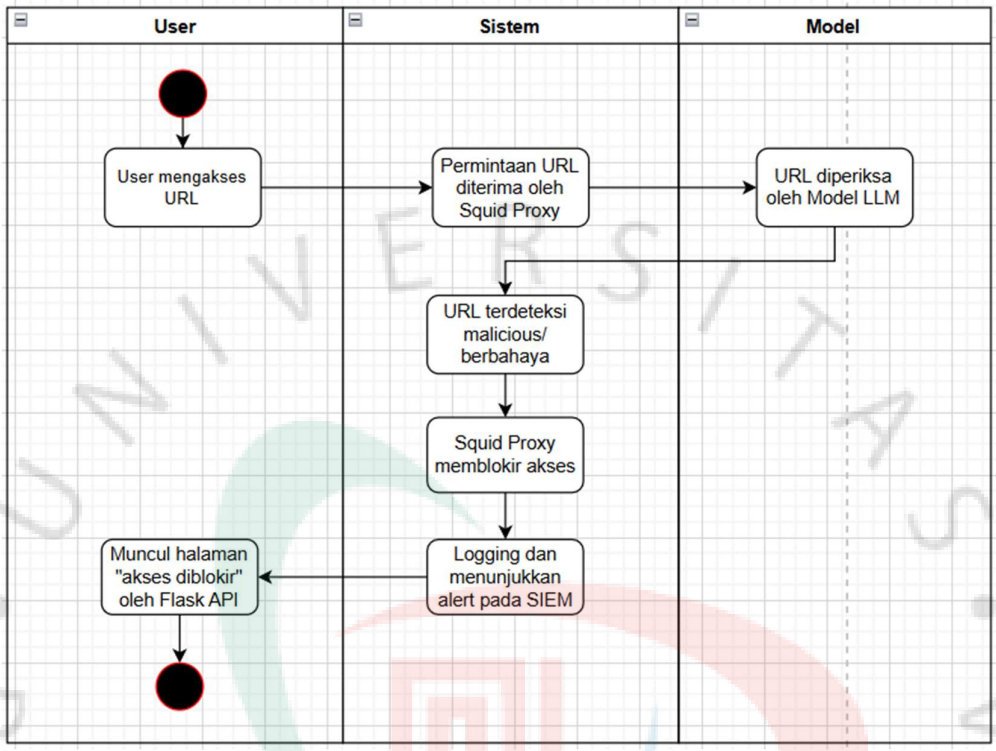
Gambar 4.11 Activity Diagram: Intrusion Detection

Pada Gambar 4.11, administrator mengonfigurasi file ossec.conf dan local_rules.xml pada sistem SIEM yang berfungsi dalam pemantauan deteksi intrusi. File ossec.conf dan local_rules.xml ini juga ditambahkan dengan konfigurasi untuk SIEM dapat mendeteksi dan memblokir jika terjadi akses pada URL berbahaya.



Gambar 4.12 Activity Diagram: Classifying URL using trained LLM

Merujuk pada Gambar 4.12, permintaan akses URL yang dilakukan oleh user akan diproses melalui Flask API yang selanjutnya diteruskan kepada model LLM. Kemudian, model LLM akan memproses URL dan menunjukkan hasil berupa "benign"(aman) atau "*malicious*"(berbahaya).



Gambar 4.13 Activity Gambar: Showing Alert & block *Malicious* URL access

Gambar 4.13 menunjukkan alur sistem menunjukkan peringatan/alert dan pemblokiran akses pada URL berbahaya. Squid Proxy mendeteksi adanya akses URL berbahaya, kemudian proses pemblokiran dimulai dari menunjukkan alert pada SIEM dan berakhir dengan menunjukkan halaman “akses diblokir” oleh Flask API.

4.5 Skenario Pengujian

Model LLM yan g sudah dilatih dan diintegrasikan ke dalam SIEM untuk mendeteksi dan mencegah akses pada URL berbahaya memerlukan verifikasi pengujian. Verifikasi dan pengujian ini merujuk pada bagian Bab III “Metode Pengujian”. Berikut adalah detail dari setiap metode pengujian yang dilakukan.

4.5.1 Pengujian White Box

Metode pengujian white box merujuk pada algoritma pada model LLM yang sudah dilatih untuk mengklasifikasikan antara URL yang berbahaya dan yang aman. Berikut ini adalah tabel penjelasan dalam metode white box yang dilakukan.

Tabel 4.10 Pengujian White Box

No	Modul yang Diuji	Output yang Diharapkan
1	Tokenisasi URL	Format input_ids dan attention_mask valid dan sesuai
2	Klasifikasi oleh Model BERT	Output label *malicious* atau benign + confidence score
3	Flask API dan Squid Proxy Filtering	Flask API menerima respon dan Squid Proxy memblokir akses URL berbahaya
4	Logging ke Wazuh	Alert muncul pada log alerts Wazuh CLI
5	Rule Matching dan Active Response	Penyesuaian Rules yang sudah ada dan penambahan rules untuk pemblokiran URL berbahaya dari Flask API dan Squid Proxy

4.5.2 Pengujian Black Box

Metode pengujian black box merujuk pada performa akan ditunjukkan ketika mengeksekusi sistem yang telah dikembangkan. Berikut ini adalah tabel penjelasan dalam metode black box yang dilakukan.

Tabel 4.11 Pengujian Black Box

No	Skenario Pengujian	Hasil yang Diharapkan
1	Client mengakses URL aman	Halaman ditampilkan dengan normal
2	Client mengakses URL berbahaya	Akses diblokir, halaman blokir Squid tampil

3	Flask API menerima dan merespons klasifikasi URL	JSON response dengan label benar dan waktu respon < 1 detik
4	Log dikirim ke Wazuh	Terdapat log baru di Kibana/Wazuh Alerts
5	Wazuh menampilkan alert	Alert muncul sesuai rule 100101 pada ossec.conf
6	Admin memantau aktivitas	Dapat melihat riwayat event akses URL, pemblokiran, dan alert