BAB III

METODO*LOG*I PENELITIAN

3.1 Analisa Kebutuhan

Penelitian ini membutuhkan beberapa alat untuk dapat membangun sistem pelaporan. Alat-alat tersebut diklasifikasikan dalam dua bentuk, yaitu perangkat keras dan perangkat lunak.

3.1.1 Perangkat Keras

Perangkat keras yang digunakan dalam penelitian ini adalah.

1. Sebuah komputer server (Pentium 8950, *memory* 2 GB, HDD 320 GB),
2. Sebuah komputer pengguna (Pentium 8950, *memory* 2 GB, HDD 320 GB),
3. Sebuah komputer penyerang, dan
4. koneksi ke internet

3.1.2 Perangkat Lunak

Perangkat lunak yang digunakan dalam penelitian ini adalah.

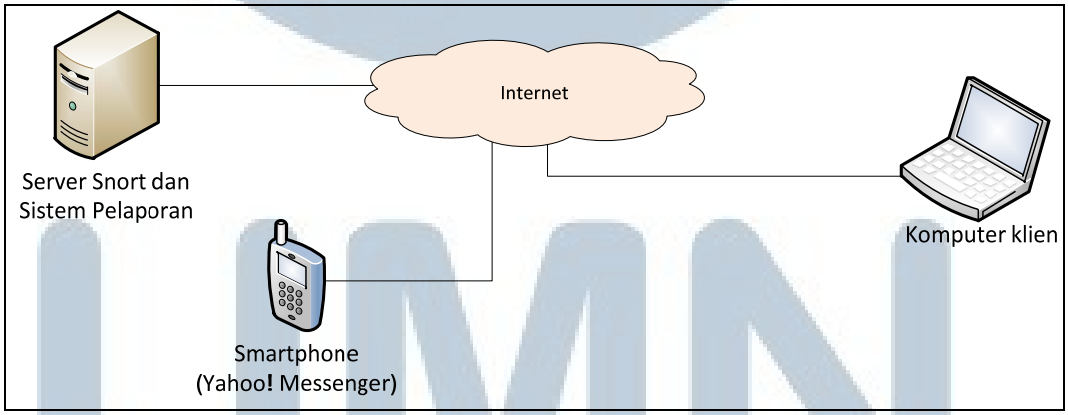
1. CentOS Linux 6.0
2. Snort IDS versi 2.1.9
3. BASE versi 1.4.4.
4. PHP 5.4
5. Yahoo! Messenger API
6. MySQL

3.2 Perancangan Sistem

Perancangan sistem dilakukan dengan membuat diagram-diagram yang merepresentasikan sistem. Diagram tersebut adalah diagram umum sistem untuk menggambarkan sistem secara keseluruhan; *flow chart* untuk menggambarkan proses yang terjadi pada sistem; dan *data flow diagram* untuk menggambarkan alur data yang terjadi pada sistem.

3.2.1 Diagram Umum Sistem

Secara keseluruhan, sistem digambarkan oleh gambar di bawah ini.



Gambar 3.1 Diagram Umum Sistem

Dalam Sistem ini, pengguna berinteraksi menggunakan komputer, telepon genggam, atau alat lain yang dapat mengakses internet dan kompatibel untuk Yahoo! Messenger. Pengguna akan mendapatkan notifikasi jika ada serangan melalui Yahoo! Messenger yang telah didaftarkan tersebut. Jadi pengguna dapat

menggunakan perangkat apapun untuk menerima peringatan tersebut asal terdapat aplikasi Yahoo! Messenger di dalamnya.

Pada *server* Snort, ditempatkan sebuah server sistem *reporting*. Sistem ini akan memonitor perubahan yang terjadi pada Snort menggunakan *trigger*. Jika terjadi perubahan pada server Snort, dengan kata lain terjadi insiden baru, maka server notifikasi akan mengirimkan peringatan kepada pengguna.

3.2.2 *Flowchart* Sistem

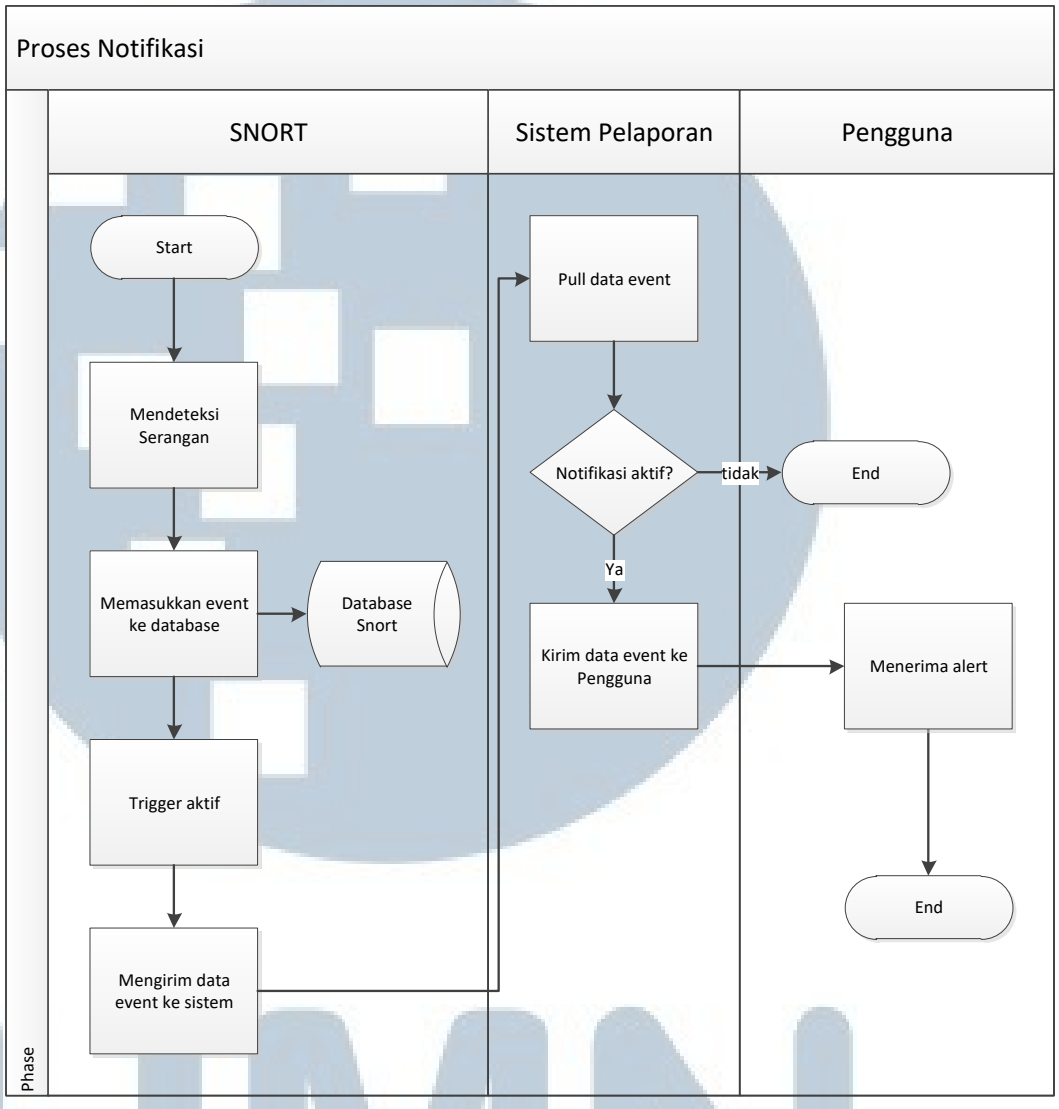
Secara garis besar, sistem ini terdiri dari dua proses, yaitu proses notifikasi dan proses komunikasi dengan pengguna. Proses notifikasi adalah proses yang menangani bagaimana sistem mengirimkan *alert* apabila terdapat temuan Snort akan insiden maupun serangan dalam jaringan. Sedangkan proses komunikasi dengan pengguna mengelola bagaimana pengguna dapat berinteraksi dengan sistem. Pengguna dapat berinteraksi dengan sistem untuk meminta detail tentang serangan yang tersimpan di *database* Snort.

Kedua proses tersebut didahului oleh sebuah proses yang mengautentikasi akun Yahoo! Messenger yang dipakai oleh *server*.

Berikut ini adalah *flowchart* yang digunakan untuk menggambarkan alur kedua proses tersebut.

3.2.2.1 *Flowchart* Proses Notifikasi

Proses notifikasi terjadi ketika Snort menemukan insiden atau serangan dalam jaringan. Gambar berikut menggambarkan alur proses notifikasi.



Gambar 3.2 *Flowchart* Proses Notifikasi

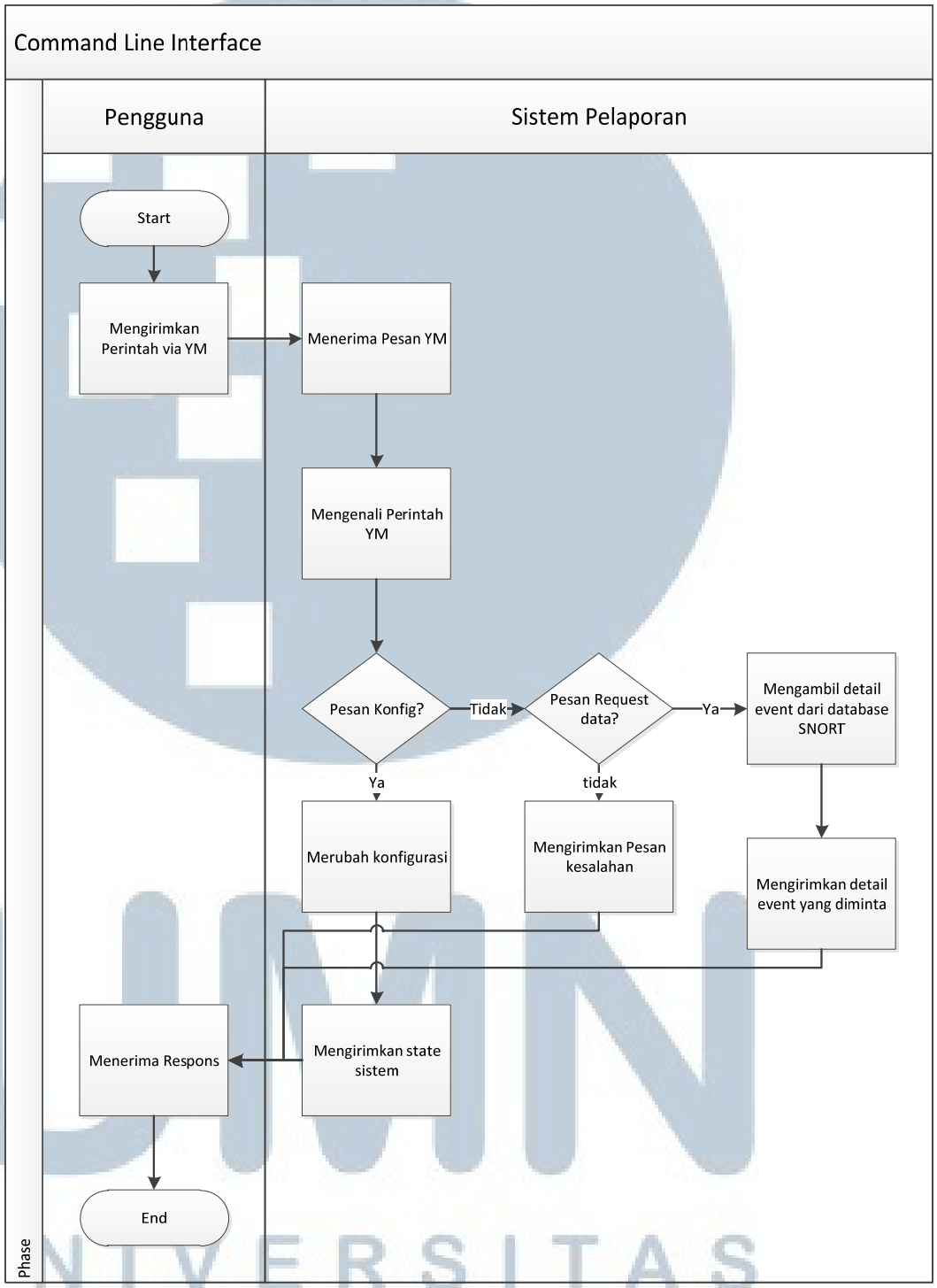
Proses ini dimulai ketika Snort mendeteksi adanya insiden yang terjadi pada jaringan. Secara otomatis, Snort akan memasukkan temuannya tersebut ke dalam *database*-nya. Ketika Snort menyisipkan data tersebut, *trigger* yang telah dipasang akan aktif. *Trigger* tersebut membuat Snort mengirimkan data serangan ke sistem. Sistem selanjutnya akan mengirimkan notifikasi ke pengguna melalui Yahoo! Messenger.

3.2.2.2 *Flowchart* Proses Komunikasi

Proses komunikasi mengelola bagaimana cara pengguna berhubungan dengan sistem. Secara umum, ada dua alasan pengguna berkomunikasi dengan sistem, yaitu untuk meminta data tentang detail insiden yang terdeteksi oleh Snort dan untuk merubah konfigurasi sistem pelaporan. Pengguna melakukan komunikasi dengan sistem dengan cara mengirimkan pesan ke akun Yahoo! Messenger sistem.

Proses ini dimulai ketika pengguna mengirimkan pesan kepada sistem melalui Yahoo! Messenger. Sistem yang menerima pesan tersebut lalu mengenali pesan tersebut. Sesuai deskripsi di atas, terdapat dua jenis perintah yang mungkin dikirimkan oleh pengguna, yaitu pesan permintaan data dan pesan perubahan konfigurasi sistem. Jika pesan yang didapat merupakan pesan permntaan data, maka sistem akan melakukan *query* atas data yang diminta pengguna ke database Snort. Setelah mendapatkan data tersebut, sistem lalu mengirimkan data tersebut ke pengguna melalui Yahoo! Messenger.

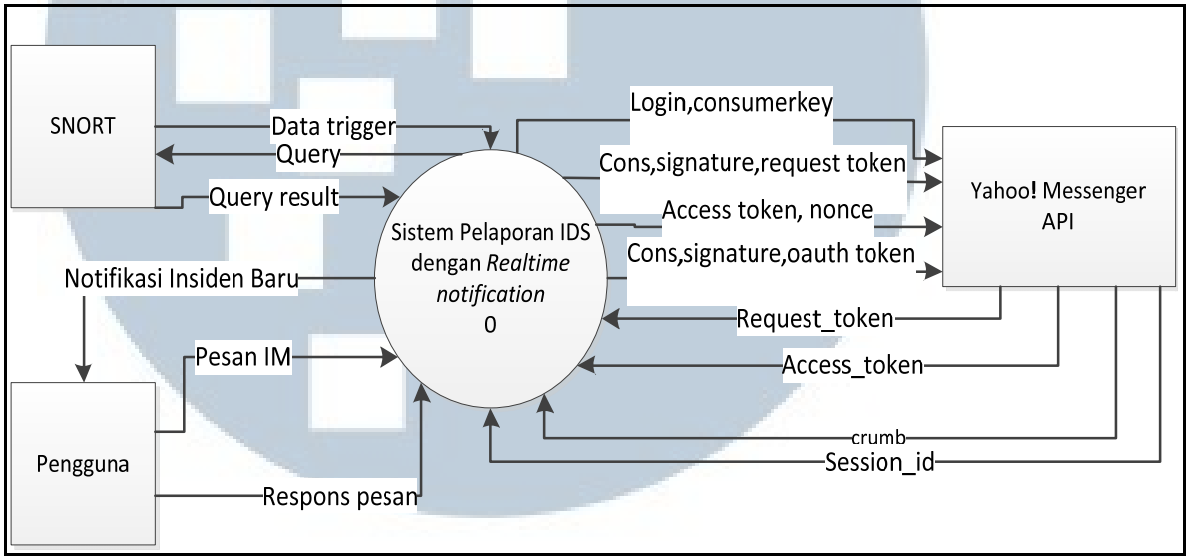
Ketika pesan yang didapat adalah pesan perubahan konfigurasi, sistem akan merubah konfigurasi sesuai permintaan pengguna. Setelah itu, sistem akan mengirimkan status sistem setelah perubahan dilakukan.



Gambar 3.3 *Flowchart* Proses Komunikasi

3.2.3 DFD Sistem

Sesuai gambaran umum yang dipaparkan pada Diagram Umum dan proses pada kedua diagram alur, dibuatlah sebuah *context diagram* dari *data flow diagram* sistem seperti berikut.

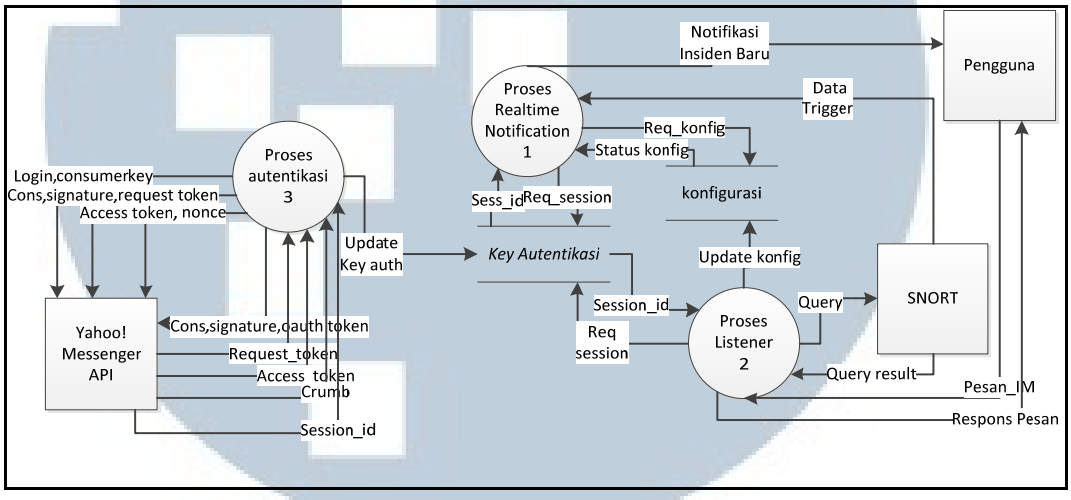


Gambar 3.4 *Context Diagram* Sistem

Diagram tersebut menggambarkan bahwa terdapat tiga entitas yang terhubung dengan sistem. Entitas tersebut adalah server Snort, pengguna, dan server Yahoo! Messenger API. Rencananya apliaksi ini akan digunakan oleh seorang *system* administrator. Yang dimaksudkan dengan admin di sini adalah admin yang mengurus jaringan yang diawasi oleh Snort. Admin inilah yang akan mendapatkan peringatan jika ada insiden yang terjadi pada jaringan tersebut.

3.2.3.1 DFD Level 0

Diagram konteks tersebut dapat dipecah menjadi DFD level 0 dengan tiga subsistem, yaitu subsistem proses *realtime notification,* proses *listener*, dan proses autentikasi.



Gambar 3.5 DFD Level 0

Snort memberikan beberapa *input* ke sistem: data *trigger*, dan *query result*. Data *trigger* adalah data yang berupa detail serangan yang baru terjadi. Snort mengirimkan data tersebut setelah terjadi serangan dengan menggunakan *trigger* dalam *database*. Sedangkan *query result* adalah hasil dari *query* yang dibutuhkan sistem untuk memberikan tanggapan terhadap permintaan data dari pengguna.

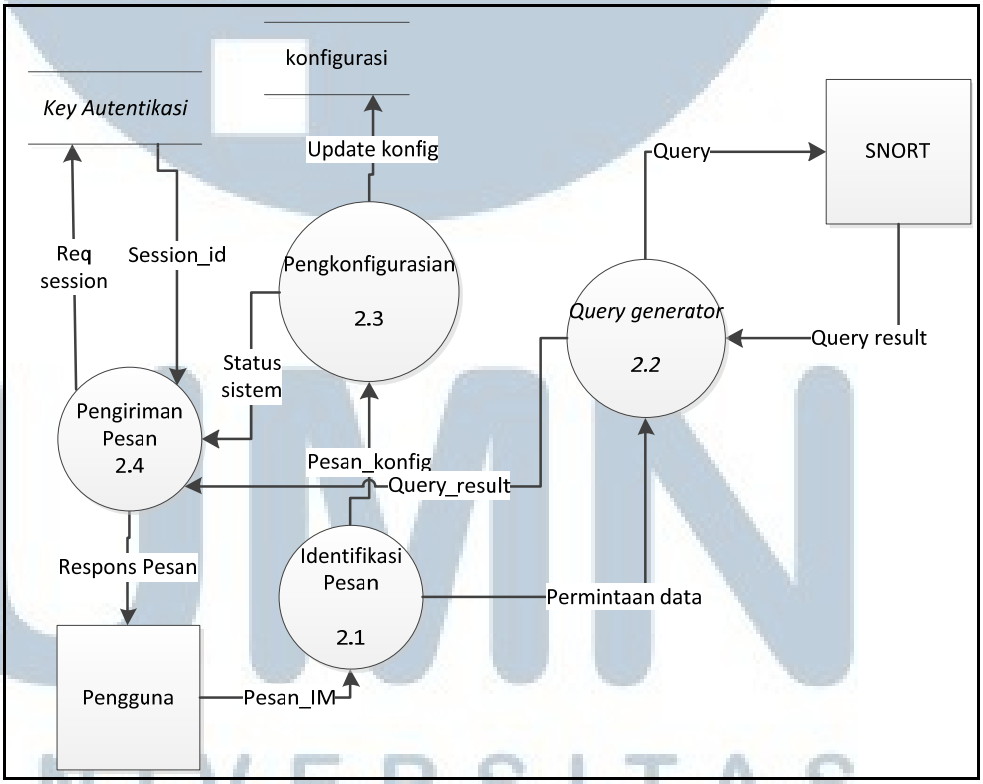
Server Yahoo! Messenger API bertukar data dengan sistem untuk melakukan autentikasi terhadap akun Yahoo! Messenger yang telah dikonfigurasikan. Akun inilah yang akan digunakan dalam berkomunikasi dengan pengguna. Detail autentikasi dan *session* akan disimpan dalam sebuah media

penyimpanan dan akan digunakan nantinya pada proses-proses yang membutuhkan.

Pengguna dapat memberikan masukan ke sistem dalam hal untuk mendapatkan informasi tentang insiden maupun merubah konfigurasi sistem. Admin dapat mengirimkan pesan ke sistem dan akan mendapatkan jawaban sesuai pesan yang dikirimkan.

3.2.3.2 DFD Level 1 – Proses *Listener*

Sub Sistem Proses *Listener* digambar dengan diagram di bawah ini.



Gambar 3.6 DFD Level 1 *Command Line Interface*

Proses ini dilakukan ketika pengguna berinteraksi dengan sistem menggunakan Yahoo! Messenger. Oleh karena itu, ada dua proses yang

melibatkan Yahoo! Messenger, yaitu *listener* dan pengiriman *message*. Proses-proses tersebut digunakan untuk menerima dan mengirim pesan kepada pengguna melalui YM. Selain kedua proses tersebut, terdapat pula proses *interpreter* dan pengaturan opsi sistem.

Alur data pada subsistem ini adalah sebagai berikut. Pengguna mengirimkan perintah ke akun YM sistem. Pesan tersebut akan diterima oleh *listener* Yahoo! Messenger. Pesan tersebut selanjutnya diteruskan ke *command line interpreter* yang akan menerjemahkan pesan yang terkirim. Pesan tersebut diklasifikasikan menjadi dua jenis, yaitu pesan permintaan data dan pesan perubahan konfigurasi. Pesan permintaan data selanjutnya akan diteruskan ke proses pengolahan *query* untuk diproses sehingga akan mendapatkan data yang diminta. Data yang diminta selanjutnya akan dikirimkan ke proses pengiriman pesan. Data tersebut selanjutnya akan dikirimkan ke pengguna sebagai sebuah pesan.

3.2.4 Perancangan Antarmuka

Antarmuka yang digunakan oleh sistem ini adalah antarmuka berbasis *command line*. Antarmuka jenis ini hanya menggunakan teks sebagai alat komunikasi. Dalam perancangan antarmuka ini, hal-hal yang diperhatikan adalah pemilihan perintah-perintah yang dikenal oleh sistem.

Secara umum, sistem ini hanya bisa menjalankan dua operasi, yaitu *real-time notification* dan permintaan data insiden yang terjadi. Oleh karena itu, perintah yang dikenali sistem menyangkut kedua proses di atas.

Untuk proses *real-time notification*, pengguna hanya dapat merubah konfigurasi proses tersebut. Hal itu dikarenakan pada proses ini, pengguna hanya tinggal mendapatkan notifikasi dari sistem jika terjadi serangan. Konfigurasi yang bisa dirubah berupa aktivasi proses tersebut.

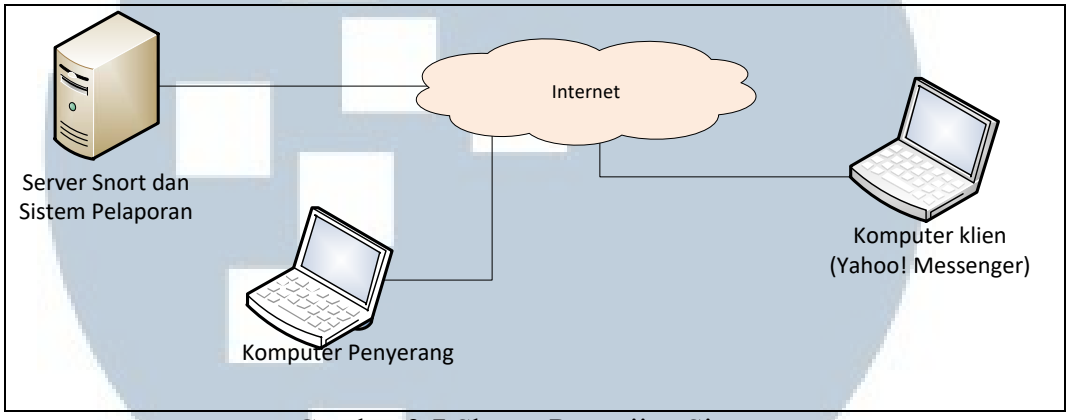
Pada proses permintaan data, pengguna dapat meminta data tentang insiden yang terjadi. Terdapat beberapa parameter yang bisa digunakan oleh pengguna untuk mendapatkan data dengan detail yang diinginkan. Beberapa detail yang dapat diminta oleh *user* adalah waktu dan jumlah data. Dimana pengguna dapat meminta data insiden yang terjadi pada kurun waktu tertentu. Selain itu, pengguna dapat membatasi berapa data yang akan ditampilkan.

Tabel 1. Daftar Perintah

Perintah	Parameter	Keterangan
show*Alert*		Menampilkan Insiden yang terdeteksi Snort.
	/d /m /y	Menampilkan insiden dalam waktu (hari/bulan/tahun) terakhir.
	/l	Membatasi jumlah insiden yang ditampilkan. /l all untuk menampilkan semua insiden.
set*Alert*		Menghidupkan / mematikan *service* notifikasi
	ON	Menghidupkan *service*
	OFF	Mematikan *service*
status*Alert*		Untuk mengetahui status dari fitur *real-time notification*

3.3 Pengujian

Pengujian dilakukan pada sistem yang telah diaplikasikan pada *server* Universitas Multimedia Nusantara. Berikut adalah skema pengujian sistem.



Gambar 3.7 Skema Pengujian Sistem

3.3.1 Pengujian Fungsionalitas

Pengujian ini dilakukan untuk mengetahui apakah fitur-fitur yang tersedia dalam sistem sudah berjalan dengan baik dan benar. Oleh karena itu, skenario-skenario yang dilakukan dalam pengujian ini adalah dengan cara mencoba semua fitur dengan kemungkinan masukan-masukan yang ada.

Fitur yang menjadi perhatian utama pada pengujian ini adalah fitur *command line interface*. Hal ini dikarenakan *syntax* dan *log*ika dari pesan yang dikirimkan oleh pengguna tidak dapat dibatasi. Dengan kata lain, pengguna dapat mengirimkan pesan berisi apapun ke server.

3.3.2 Pengujian *Atomicity*

Pengujian *atomicity* bertujuan untuk mengetahui seberapa besar tingkat kesuksesan pengiriman *alert* yang ditujukan kepada pengguna. Pengujian ini

dikatakan berhasil apabila tingkat kesuksesan mencapai 100%, dengan kata lain seluruh *alert* berhasil dikirimkan ke pengguna.

Untuk menguji aplikasi, dikirimkan paket yang berisi serangan yang akan dideteksi oleh Snort. Sistem kemudian akan mengirimkan *alert* ke pengguna. Dari sisi pengguna dilakukan pencatatan *alert* yang sudah diterima. Setelah itu, dilakukan perbandingan dari kedua catatan yang diperoleh dari server dan pengguna.

3.3.3 Pengujian *Delay* Notifikasi

Pengujian ini bertujuan untuk mengetahui berapa waktu *delay* yang terjadi ketika sistem mengirimkan notifikasi kepada pengguna. Hal ini berkaitan erat dengan sifat sistem yang *real-time*.

Untuk menghitung *delay* notifikasi, perlu dilakukan sinkronisasi jam atau *clock* pada sisi pengguna dan server. Setelah dilakukan sinkronisasi, *alert* akan dibangkitkan dan akan dikirimkan ke pengguna. Pesan notifikasi yang dikirimkan untuk pengujian ini telah dimodifikasi untuk menampilkan waktu terjadinya serangan. Dari waktu terjadinya serangan dan waktu terima pengguna, maka akan didapatkan selisih waktu yang dibutuhkan notifikasi untuk mencapai pengguna.