BAB II
DASAR TEORI

2.1 KAJIAN PUSTAKA

Penelitian Putra, dkk. [5] yang berjudul "Implementasi dan Analisis Keamanan Jaringan Virtual HIPS Snort Pada Layanan *Web server* Dengan Penyerangan Dos dan DDOS" membahas tentang penerapan *Host Intrusion Prevention System* Snort pada *server* dengan Layanan *Web server*. Kemudian *server* yang telah dipasang sistem keamanan *Host Intrusion Prevention System* Snort tersebut dilakukan pengujian dengan serangan TCP SYN *Flood*. Hasil yang didapatkan bahwa sistem *Host Intrusion Prevention* Sytem yang diterapkan dapat mengenali dan menahan jenis serangan tersebut dengan menggunakan *rule* Snort yang telah dibuat.

Penelitian Firdaus dan Suartana [4] yang berjudul " Implementasi Keamanan Jaringan *Intrusion Detection*/*Prevention System* Menggunakan pfSense " membahas tentang penerapan *Intrusian Detection*/*Prevention System* (IDPS) pada *server*. Pada *server* tersebut diterapkan sistem keamanan yang menggunakan fitur IDPS Suricata pada *tools* pfSense. Sistem keamanan diuji dengan serangan DOS menggunakan *tools* hping3 pada kali linux. Tipe variasi serangan yang digunakan dalam penelitian ini yaitu UDP *Flooding* ,TCP SYN *Flooding, Smurf Attack, PUSH ACK Flood*, dan RST/Reset *Flood*. Hasil pengujian terhadap implementasi keamanan jaringan IDPS yaitu Suricata dapat mengirimkan alert ketika mendeteksi serangan dan melakukan action yaitu memblokir source IP penyerang

Penelitian Ralianto dan Cahyono [6] yang berjudul " Perbandingan Nilai Akurasi Snort dan Suricata dalam Mendeteksi Intrusi Lalu Lintas di Jaringan " membahas tentang analisa nilai akurasi *intrusion detection system* (IDS) menggunaksan aplikasi Snort dan Suricata. IDS Snort dan Suricata digunakan untuk mendeteksi intrusi di jaringan. Pada penelitian ini menggunakan 3 VM masing-masing dipasang Suricata, Snort, dan pytbull. Pytbull merupakan aplikasi untuk pengujian serangan terhadap VM yang sudah dipasang Snort dan Suricata. Berdasarkan pengujian yang telah dilakukan pada penelitian tersebut

menyimpulkan bahwa rata – rata nilai akurasi Suricata lebih tinggi (61%) daripada Snort (31%) karena *rule* Suricata lebih banyak dibandingkan Snort. Penggunaan memory juga Suricata lebih stabil daripada Snort karena menggunakan fitur *multithreading*.

Penelitian Fadhilah dan Marzuki [7] yang berjudul *" Performance Analysis of IDS Snort and IDS Suricata with Many-Core Processor in Virtual Machines Against Dos/DDoS Attacks "* membahas tentang analisa performa dari VM yang dipasang aplikasi Snort dan Suricata. Pada penelitian tersebut VM IDS Snort dan Suricata diuji dengan serang *Denial of Service*. Penelitian tersebut menyimpulkan performa CPU, *memory* dan *packet Capture* dari IDS Snort dan Suricata dalam mendeteksi serangan ketika menggunakan beberapa variasi jumlah *core* 1, 2, 4 *core*. Hasil dari pengujian pada penelitian tersebut menyimpulkan bahwa pada IDS Suricata peningkatan jumlah *core* tidak menunjukan peningkatan deteksi serangan. Sedangkan pada IDS Snort, peningkatan jumlah *core* menunjukkan peningkatan deteksi serangan Dos/DDos yang signifikan. Maka pada penelitian tersebut menyimpulkan bahwa performa multi-threading pada IDS Snort 3.0 lebih baik dibandingkan IDS Suricata.

Penelitian Santoso, dkk. [8] yang berjudul " Implementasi dan Analisa Snort dan Suricata Sebagai IDS dan IPS Untuk Mencegah Serangan DOS dan DDOS " membahas tentang analisa performa Snort dan Suricata dalam mendeteksi serangan (IDS) dan mencegah serangan (IPS). Pada penelitian tersebut dilakukan pengujian *ping flood, SYN flood,* dan *HTTP flood* dengan skema DoS dan DDoS. Hasil dari pengujian deteksi serangan (IDS) dengan skema DoS dan DDoS. Suricata unggul dalam mendeteksi serangan DoS *SYN flood,* DDoS *SYN flood,* DDoS *HTTP flood,* dan DDoS *ping flood*. Sedangkan Snort hanya unggul dalam mendeteksi serangan DoS *ping flood*. Hasil pencegahan serangan (IPS) dengan skema DoS dan DDoS. Suricata unggul dalam mencegah serangan DoS *SYN flood,* DDoS *SYN flood* dan DDoS *ping flood*. Sedangkan Snort unggul dalam mencegah serangan DoS *ping flood,* DoS *HTTP flood,* DDoS *ping flood,* dan DDoS *HTTP flood*. Berdasarkan hasil pengujian tersebut Suricata unggul dalam 7 skenario dan Snort unggul dalam 5 skenario pengujian.

2.2 JARINGAN KOMPUTER

Sistem jaringan komputer terdiri dari komputer dan perangkat jaringan lainnya yang bekerja sama untuk mencapai tujuan yang sama. Tujuan jaringan komputer meliputi:

1. Membagi sumber daya misalnya dalam penggunaan perangkat keras dan koneksi internet.
2. Memudahkan komunikasi antar komputer, seperti pada penggunaan surat elektronik atau *e-mail* misalnya.
3. Akses informasi, misalnya menelusuri halaman *web*.
4. Sebagai bahan acuan dasar membangun jaringan warnet game online dan membangun jaringan di instansi dan perusahaan.

Berdasarkan jangkauan area atau lokasi, jaringan komputer dibedakan menjadi 3 tipe yaitu:

1. *Local Area Network* (LAN)

   Local Area Network (LAN) merupakan jenis jaringan yang hanya menghubungkan beberapa komputer di satu lokasi dalam jarak pendek dan terbatas, seperti di dalam ruangan atau gedung. Jaringan area lokal dapat menggunakan sarana koneksi seperti kabel dan nirkabel (tanpa kabel).

2. *Metropolitan Area Network* (MAN)

   Jaringan komputer jenis ini dapat menjangkau area yang lebih luas (sekitar 50 km) dibandingkan dengan jenis LAN. Selain itu, MAN memungkinkan transmisi data dengan kapasitas yang besar. MAN merupakan gabungan dari beberapa jaringan bertipe LAN, misalnya jaringan komputer antar kota dan provinsi.

3. *Wide Area Network* (WAN)

   Jenis jaringan ini memiliki cakupan yang lebih luas daripada jenis MAN. Cakupan jaringan WAN dapat tersebar di seluruh wilayah di berbagai negara. Jenis WAN ini bisa disebut internet atau jaringan global karena dapat mencakup seluruh jaringan komputer di dunia. [9].

2.3 *WEB SERVER*

Web server merupakan *software* yang menyediakan layanan berbasis data kepada klien melalui aplikasi *web browser* dengan protokol HTTP atau HTTPS. Data dikirim ke *server* dalam bentuk halaman *web*, biasanya dokumen HTML. Halaman *web* yang diminta dapat berisi teks, video, gambar, *file*, dan jenis konten lainnya.Terdapat beberapa *software* yang berfungsi sebagai *web server* salah satunya yaitu Apache. Apache merupakan sebuah *web server* yang berfungsi untuk melakukan *request-response* HTTP dan *logging* informasi. Selain itu, Apache juga adalah *web web server* paling umum digunakan dan mengikuti standar protokol HTTP. Berdasarkan survei yang dilakukan oleh Netcraft, bulan Januari 2005 jumlah pengguna Apache sekitar 68% dari pangsa *web server* yang berjalan di Internet [10].

2.4 SYARAT – SYARAT KEAMANAN JARINGAN

Keamanan Jaringan merupakan salah satu kajian *computer security* yang hanya fokus pada infrastruktur jaringan (*network*). Sehingga bisa dikatakan *network security* merupakan bagian dari *computer security*. *Network security* juga dapat dipandang sebagai bagian dari *cybersecurity*. Keamanan jaringan memastikan bahwa jaringan internal terhindar dari ancaman luar dengan melindungi infrastruktur dan menghambat akses ke sana. Perkembangan keamanan jaringan saat ini, tim keamanan menggunakan *machine learning* untuk membantu mengelola pemantauan keamanan jaringan dengan lebih baik. Hal ini memungkinkan mereka untuk mengidentifikasi lalu lintas yang tidak biasa dan waspada terhadap ancaman secara *real time*. [11].

Keamanan jaringan yang baik terdiri dari tiga syarat yang harus dipenuhi yaitu *prevention* (pencegahan), *observation* (observasi), dan *response* (respon). *Prevention* merupakan upaya pencegahan terhadap akses yang tidak sah dalam sistem jaringan yang dapat dicapai dengan teknologi IPS. Sedangkan *observation* adalah proses perawatan yang melacak aktivitas dalam log. Hal ini memungkinkan penggunaan sistem pengenalan intrusi sebagai bagian dari prosedur observasi.

Sedangkan *response* adalah tindakan yang diambil ketika ada penyusupan atau peretasan sistem jaringan.

Keamanan jaringan mempunyai 5 aspek sebagai berikut :

1. *Confidentiality*

   Confidetiality merupakan aspek yang mengharuskan suatu informasi yang hanya dapat diakses oleh pengguna yang sah dan memliki izin hak akses.

2. *Integrity*

   Integrity merupakan aspek keamanan jaringan yang mengharuskan suatu informasi hanya dapat diedit oleh *user* yang memiliki izin yang sah untuk mengubahnya.

3. *Availability*

   Availability merupakan aspek keamanan jaringan yang mengharuskan suatu informasi selalu tersedia bagi *user* yang memiliki izin hak akses yang sah.

4. *Authentication*

   Authentication merupakan aspek keamanan jaringan yang mengharuskan pengirim dan penerima suatu informasi dapat diidentifikasi berdasarkan identitas yang asli atau tidak palsu.

5. *Nonrepudiation*

   Nonrepudation merupakan aspek keamanan jaringan yang mengharuskan pengirim dan penerima suatu informasi tidak dapat menampik penerimaan dan pengiriman pesan[4].

2.5 LAPISAN OSI REFERENCE MODEL

Protokol adalah sekumpulan aturan, konvensi, dan sinkronisasi yang digunakan oleh komputer untuk berbagi data di jaringan. Contohnya termasuk mengirimkan e-mail, mengirim *file*, mengakses halaman *web*, komunikasi di antara komputer, dan banyak lagi.

Protokol *Open System Interconnection* (OSI) merupakan gabungan dari berbagai protokol standar berbasis model OSI yang dikembangkan oleh lembaga internasional pengembang protokol jaringan. OSI berkembang karena kebutuhan

standar jaringan. Model referensi OSI membagi arsitektur jaringan menjadi 7 lapisan (*layer*). Setiap lapisan dipisahkan menurut fungsinya, dan istilah " *layer* " mengacu pada tingkat layanan dengan fungsi tertentu. Namun, jumlah *layer* tidak terlalu banyak untuk menghindari arsitektur yang rumit. Pembagian *layer* berdsarakan model OSI yaitu terdiri dari *layer Application*, *layer Presentation*, *layer Session*, *layer Transport*, *layer Network*, *layer* data *link* dan terakhir *layer Physical*.

1. Lapisan 7 – *Application*

   Pada *layer* ini berurusan dengan program komputer yang digunakan oleh user. Program komputer yang masuk dalam kategori ini hanya program khusus untuk akses ke jaringan Contoh: Aplikasi *Word Processing* hanya menangani pengolahan teks sehingga tidak berhubungan dengan OSI. Namun, jika terdapat fitur jaringan seperti pengiriman *e-mail* maka *layer* aplikasi baru dapat dianggap berhubungan. Ada beberapa *protocol* pada lapisan aplikasi ini yang biasanya diperlukan sebagai anatara lain yaitu WWW, HTTP, FTP, SMTP, TELNET.

2. Lapisan 6 – *Presentation*

   Layer ini bertanggung jawab untuk menjaga format data yang dapat diterima oleh berbagai jenis media. *Layer* ini juga memiliki kemampuan untuk mengkonversi format data, yang memungkinkan *layer* berikutnya untuk memahami format yang diperlukan untuk komunikasi. Jenis data yang didukung oleh *layer* presentasi yaitu teks, data, grafik, visual/gambar, dan suara.

3. Lapisan 5 – *Session*

   Layer Session/Sesi merupaka *layer* yang bertugas untuk memulai, mengontrol dan mengakhiri suatu komunikasi (biasa disebut *session*/sesi). Protokol pada *layer* session antara lain yaitu : SQL ,NFS, RPC, SCP, ASP.

4. Lapisan 4 – *Transport*

   Pada *layer* ini dapat memilih untuk menggunakan protokol yang mendukung pemulihan kesalahan atau tanpa pemulihan keasalahan. Apabila data yang datang tidak berurutan, *multi-plexing* mengurutkannya. Selain itu, komunikasi dari ujung ke ujung (*end-to-end*) diatur dengan cara yang

berbeda di lapis keempat ini. Akibatnya, lapis keempat ini sangat memengaruhi *transfer* data. Protokol pada *layer* ini antara lain yaitu TCP, UDP, SPX.

5. Lapisan 3 – *Network*

   Pada *layer* ini berfungsi untuk melakukan *addressing and routing*. *Addressing* pada *layer network* merupakan pengalamatan yang dilakukan secara *logical*. Sedangkan *Routing* merupakan perutean sebagi pengarah jalur paket data yang akan dikirim. Terdapat 2 jenis perutean antara lain yaitu *Routed* dan *Routing Protocol*. Protokol yang bekerja pada *layer* ini antara lain yaitu IP dan ICMP.

6. Lapisan 2 - *Data Link*

   Lapisan data link merupakan lapisan yang memiliki beberapa fungsi yaitu *Arbitration* (pemilihan media fisik), *Addressing* (pengalamatan fisik), *Error Detection* (menentukan apakah data telah berhasil terkirim), *Identify Data Encapsulation* (menentukan pola header pada suatu data).

7. Lapisan 1 – *Physical*

   Layer ini mengatur berbagai bentuk antarmuka media transmisi yang termasuk berbagai spesifikasi,seperti konektor, pin, penggunaan pin, arus listrik yang lewat, *encoding*, sumber cahaya, dan lain-lain. [9].

2.6 TCP

Transmission Control Protocol (TCP) merupakan salah satu protokol komunikasi pada jaringan komputer yang memiliki keunggulan yaitu mampu mengkoreksi kesalahan dalam pengiriman data. Pada protokol ini menjamin proses pengiriman data karena menggunakan metode *flow control*. Metode *Flow control* merupakan metode untuk menentukan kapan data harus dikirim kembali dan kapan menghentikan aliran data paket sebelumnya, sampai data tersebut berhasil ditransfer.[12].

2.7 *DENIAL OF SERVICE (DOS)* DAN *DISTRIBUTED DENIAL OF SERVICE (DDOS)*

DOS merupakan suatu jenis serangan pada jaringan internet yang bertujuan untuk menghabiskan sumber daya target. Salah satu target serangan DOS yaitu *server*. Ketika suatu *server* terkena serangan DOS, maka layanan pada *server* tersebut akan terhambat sehingga pengguna lain tidak dapat mengakses layanan dari *server*. Sedangkan DDOS adalah serangan DOS yang menggunakan banyak *host* penyerang (*botnet*) untuk menyerang satu target dalam suatu jaringan[5]. Penggunaan banyak *botnet* pada serangan DDOS memungkinkan pengiriman paket serangan dalam jumlah yang lebih besar dibandingkan serangan DOS sehingga akses terhadap suatu layanan akan terhenti dengan cepat.[13].

2.8 *DATA FLOODING*

Data flooding adalah pengiriman data yang berlebihan ke jaringan, yang biasanya merupakan data yang tidak berguna. Data tersebut dikirim oleh penyerang yang mempunyai tujuan untuk memenuhi jalur *traffic* yang ada dalam jaringan. Serangan *data flooding* pada jam sibuk mengakibatkan *traffic* suatu data akan sangat padat. Baik data yang akan dikirim maupun data yang akan datang akan mengalami antrian data yang mengakibatkan keterlambatan dalam pengiriman dan penerimaan data. Selain itu, *data flooding* juga bisa mengakibatkan kerugian lain seperti kerusakan program. *Data flooding* memiliki jenis tertentu sesuai dengan fungsi dari data yang dikirim. Salah satu jenis *data flooding* yaitu *TCP SYN flooding.*

TCP SYN flooding merupakan salah satu jenis *data flooding* dimana penyerang (*attacker*) akan mengirimkan permintaan SYN kepada *server* yang menjadi target[4] . Hal ini akan menyebabkan *host* tujuan akan terus menunggu paket tersebut dengan menyimpannya kedalam *backlog*. Paket yang dikirimkan biasanya beukuran kecil namun jika pengiriman SYN dilakukan secara terus menerus akan memenuhi *backlog*. Apabila *backlog* sudah besar maka akan mengakibatkan *host* tujuan akan menolak paket SYN yang datang, sehingga *host* tersebut tidak bisa dikoneksi oleh *host* yang lain [12].

2.9 *INTRUSION DETECTION SYSTEM* (IDS)

IDS merupakan sistem yang bekerja untuk melakukan pengawasan terhadap kegiatan/perilaku pada lalu lintas data dalam suatu jaringan. Ketika IDS menemukan kegiatan yang mencurigakan pada lalu lintas jaringan maka IDS akan memberikan peringatan kepada sistem. Pada dasarnya IDS memiliki 2 metode dalam mendeteksi serangan yaitu *rule based system* dan *adaptive system* . *Rule based system* yaitu metode mendeteksi serangan berdasarkan *signature* dan *rule* yang tersimpan di database. Ketika IDS menemukan kecocokan antara kondisi lalu lintas jaringan dengan *rule/signature,* maka IDS akan menganggap hal tersebut merupakan suatu serangan. Metode selanjutnya yaitu *adaptive system* dimana pada metode ini IDS mendeteksi serangan tidak hanya berdasarkan *rule/signature* yang tersimpan di database, tetapi juga dapat mendeteksi jenis serangan yang baru [13].

2.10 *INTRUSION PREVENTION SYSTEM* (IPS)

IPS adalah suatu sistem yang berfungsi untuk melakukan identifikasi terhadap aktivitas pada lalu lintas jaringan yang dianggap sebagai suatu ancaman sekaligus melakukan tidakan berupa pemblokiran[4]. Cara kerja *system IPS* yaitu mengkombinasikan teknik *firewall* dan *intrusion detection system* (IDS). *System* ini akan mencegah serangan ataupun ancaman yang berusaha memasuki jaringan lokal. IPS memeriksa dan mencatat paket – paket data yang melintas pada *interface* kemudian ketika menemukan data yang dianggap suatu ancaman maka IPS akan memblokir alamat IP pengirim dan mencatat setiap paket data yang menjadi ancaman[14].

Dalam mengidentifikasi apakah suatu paket data dinyatakan sebagai ancaman atau tidak maka IPS menggunakan beberapa metode yaitu :

1. *Signature-based Intrusion Prevention System*

   Metode *signature* merupakan metode yang menyediakan daftar signature yang berfungsi untuk mengidentifikasi paket yang dikirim sebagai ancaman atau bukan. Pada metode ini paket akan dicocokan dengan daftar *signature* yang tersedia. Metode ini akan melindungi sistem berdasarkan serangan yang sudah diketahui sebelumnya sehingga daftar *signature* harus terus diperbarui agar dapat mengodentifikasi jenis serangan terbaru.

2. *Anomaly-based Intrusion Prevention System*

Metode *anomaly* merupakan metode yang dijalankan berdasarkan konfigurasi IDS dan IPS yang disesuaikan dengan kondisi transfer paket data yang dianggap normal pada lalu lintas jaringan. Berdasarkan konfigurasi tersebut maka IPS akan mendeteksi kondisi transfer data yang tidak normal yang disebut paket *anomaly*. Ketika IDS medeteksi anomaly pada lalu lintas jaringan maka IDS akan memberikan peringatan yang selanjutnya ditanggapi oleh IPS untuk menolak paket dan memblokir pengirim [13].

2.10.1 *NETWORK-BASED INTRUSION PREVENTION SYSTEM* (NIPS)

NIPS merupakan sistem pemantauan dan proteksi yang mencakup satu jaringan secara menyeluruh sehingga mampu melindungi beberapa *host* sekaligus. NIPS dipasang pada *interface router* untuk melakukan proteksi terhadap lalu lintas jaringan yang melewati *interface router* tersebut. NIPS dapat menerapkan metode pendeteksian *signature, anlomaly,* dan *monitoring* berkas – berkas pada sistem opersi *host*[14].

2.10.2 *HOST-BASED INTRUSION PREVENTION SYSTEM* (HIPS)

HIPS merupakan sistem pemantauan dan proteksi yang melindungi hanya satu *host* saja. HIPS dipasang secara langsung pada *host* yang menerapkan IPS untuk memonitor aktifitas sistem internalnya. Fungsi utama dari HIPS yaitu untuk memantau dan mencegah *system call* yang dicurigai sebagai ancaman terhadap *host*. Selain itu, HIPS juga dapat memantau aliran data dan aktivitas pada aplikasi tertentu yag terpasang pada HIPS. Namun jika diamati dari sudut pandang *performnace*, HIPS memberikan dampak negatif terhadap performa *host*. Karena pemasangan HIPS yang secara langsung pada sistem operasi mengakibatkan penggunaan sumber daya *host* semakin besar [14].

2.11 SNORT

Snort merupakan perangkat lunak *intrusian detection system* (IDS) dan *intrusion perevention system* (IPS) yang bersifat *free* dan *open source*. *Software* ini

dibuat oleh Martin Roesch (*Sourcefire*) di tahun 1998 yang kemudian dikembangkan oleh perusahaan Cisco dengan berbasis *single -threaded* [6]. *Software* ini menggunakan *signature*, protokol, dan *anomaly* sebagai basis pada metode inspeksi IDS dan IPS[5].Program Snort dapat dioperasikan dengan tiga mode yaitu :

1. Paket *sniffer*

   Pada mode ini, Snort akan membaca semua paket pada lalu lintas jaringan kemudian menampilkannya dalam bentuk aliran yang tak terputus (layar).

2. Paket *logger*

   Pada mode ini, Snort akan mencatat log dari semua paket ke dalam *disk* kemudian dapat disimpan dengan cara mencantumkan direktori *logging*.

3. NIDS (*Network Intrusion Detection System*)

   Snort pada mode ini berperan sebagai IDS yang berfungsi untuk melakukan deteksi ancaman pada lalu lintas jaringan komputer.

Untuk melakukan fungsinya, Snort tersusun atas beberapa komponen antara lain yaitu :

1. *Decoder*

   Komponen ini bertugas untuk mengidentifikasi protokol, *decode* IP, TCP dan UDP dari paket yang ter*capture* dalam bentuk struktur data untuk mendapatkan informasi yang dibutuhkan seperti *port number* dan *IP address*. Jika Snort menemukan paket yang cacat maka Snort akan memberikan suatu peringatan.

2. *Preprocessors*

   Komponen *Preprocessors* memilki fungsi untuk mengambil paket data yang dinilai memilki potensi sebagai ancaman. Selanjutnya komponen ini mengirirm paket tersebut ke komponen *detection* engine untuk dikenali polanya.

3. *Rule*s *File*

   Rules file adalah suatu komponen dari Snort yang berupa *file* teks yang memuat aturan – aturan dalam bentuk sintaks. Sintaks berisi tentang pendefinisian aturan seperti protokol, *address*, *output*, *plug-ins* dan aturan

spesifik lainnya dengan parameter tertentu berdasarkan kebijakan keamanan jaringan yang diterapkan.

4. *Detection Engine*

   Detection engine merupakan komponen yang berfungsi untuk mencocokan paket yang dianggap berbahaya dengan rule yang sudah diterapkan. Jika ditemukan paket yang cocok maka komponen ini akan menginisialisasikan paket tersebut sebagai sebuah serangan.

5. *Output Plug-ins*

   Komponen ini meuapakan modul yang berfungsi untuk mengatur format keluaran *alert* dan *file logs* yang dapat diakses dengan berbagai cara seperti *extern files, database,* dan *console* [15].

Beberapa karakteristik yang dimiliki Snort antara lain yaitu :

1. Snort berukuran kecil yaitu hanya 2256 KB, sehingga tidak memerlukan penyimpanan yang besar.
2. Snort dapat dipasang diberbagai OS seperti Linux, Windows, Solaris, BSD, dll.
3. Snort cukup cepat sehingga mampu mendeteksi serangan pada *network* 100 Mbps.
4. Snort cukup mudah dikonfigurasi sesuai topologi jaringan dan *rule* yang akan digunakan
5. Snort bersifat *free* dan *open source* [8].

2.12 SURICATA

Suricata adalah *software* IDS dan IPS *open source* yang dikembangkan oleh komunitas *non-profit Open* Information *Security* Foundation (OISF) [4]. Suricata dapat melakukan pemantauan lalu lintas jaringan, mendeteksi ancaman secara *real-time*, pencegahan intrusi *inline*, dan pemrosesan *pcap offline*. Dalam memeriksa lalu lintas jaringan, Suricata menggunakan dengan *rules* yang sudah ditentukan. [6].

Suricata dirancang sebagai sistem *multi-threaded*, sehingga Suricata menunjukkan kinerja yang optimal pada mesin *multi-core*. Penggunaan jumlah core yang lebih banyak mempermudah Suricata dalam memeriksa lalu lintas jaringan

pada volume yang padat. Suricata juga memiliki kemampuan untuk memberikan visibilitas ke lapisan Aplikasi dan penguraian HTTP yang cepat [13].

Arsitektur IDS Suricata memiliki kemiripan dengan IDS Snort namun memiliki perbedaan dimana IDS Suricata tidak menggunakan komponen *preprocessor* dalam arsitekturnya tetapi menggunakan paket *decoder* dan *detection engine*. IDS Suricata menggunakan *library af_packet* untuk meningkatkan kinerja dalam menangkap paket dalam jaringan dan menganalisisnya dari NIC. Kemudian paket – paket tersebut diteruskan ke paket *decoder*. Dalam modul paket *decoder*, setiap paket dari protokol data *link* ke protokol *transport* akan diproses dan diubah menjadi sebuah struktur data. Kemudian paket – peket tersebut diteruskan ke *detection engine*. Dalam proses ini, *detection engine* diatur oleh *rules*. *Rules* IDS Suricata mendukung *layer* 3, *layer* 4, dan *layer* 7. *Rules* berisi *signature* yang cocok dengan representasi paket internal. Jika ada paket berbahaya yang sesuai dengan aturan, maka IDS Suricata akan memberikan peringatan berupa *alert* [7].

2.13 *FIREWALL*

Firewall adalah sistem pada keamanan jaringan yang bertugas untuk memeriksa masuk keluarnya paket data pada lalu lintas jaringan. Biasanya IPS diterapkan secara *inline* setelah *firewall* [8].

Contoh cara kerja *firewall* yaitu ketika pengguna mencoba membuka google di dalam sistem yang menggunakan *firewall*, maka permintaan pengguna akan masuk ke *firewall* kemudian *firewall* akan memeriksa kebijakannya dan mencari tahu apakah pengguna diizinkan atau tidak. Jika diizinkan, maka *firewall* akan meneruskan permintaan pengguna ke *router*, lalu *router* akan mencari jalur terbaik untuk mencapai google. Dengan cara yang sama, jika situs *web* tidak diizinkan, *firewall* akan memblokir permintaan tersebut. Jika *router* tidak dapat menemukan jalur untuk permintaan tersebut, maka *router* akan mengirimkan pesan ke *firewall* bahwa situs *web* tidak tersedia. Artinya permintaan situs *web* tidak dapat dijangkau[16].

2.14 PFSENSE

Perangkat pfSense adalah *software* komputer yang dapat berfungsi sebagai *firewall* / *router* yang bersifat *open source* dengan sistem operasi *Freebsd*. Perangkat ini dapat dipasang pada komputer secara fisik atau secara virtual untuk membuat simulasi *firewall* / *router* pada jaringan komputer. Perangkat ini dapat dioperasikan dengan tampilan *web* GUI yang mempermudah *administrator* jaringan. Selain dapat digunakan sebagai *firewall* dan *router*, pfSense juga memiliki beberapa *service* dan paket aplikasi yang dapat meningkatkan sistem keamanan [4]. Beberapa *service* yang disediakan perangkat pfSense selain firewall dan routing antara lain yaitu *server* DHCP, *server* DNS, sebagai VPN dan pemasangan paket pihak ketiga melalui *package manager* seperti Snort, Suricata, dan sebagainya [15].

2.15 NPING

Nping merupakan *open source tool* yang digunakan untuk menghasilkan paket jaringan, menganalisis respons, dan mengukur waktu respons. Nping memungkinkan pengguna untuk menghasilkan paket jaringan dari berbagai macam protokol seperti TCP, UDP, ICMP, dan ARP. Nping dapat digunakan sebagai perangkat ping sederhana untuk mendeteksi *node* aktif dalam jaringan dan dapat digunakan sebagai paket generator untuk *stress test* tumpukan jaringan, mengirimkan permintaan ARP, *denial of service*, pelacakan rute, dan tujuan lainnya [17].

Pada *tools* ini terdapat *TCP connect mode* yang merupakan mode *default* yang tidak memerlukan hak akses is*time*wa untuk menggunakan perintah tersebut. Pada mode ini Nping akan membuat koneksi ke *port* target dengan melakukan panggilan sistem koneksi. Panggilan ini sama seperti permintaan koneksi yang digunakan *browser web* dan *client point to point*. Mode ini akan menampilkan jumlah koneksi TCP yang berhasil dilakukan [18].

2.16 QOS

Quality of Service (QoS) adalah suatu metode untuk mengukur kualitas dan mengidentifikasi karakteristik jaringan. Pada implementasinya, pengguna menentukan persyaratan kinerjanya berdasarkan parameter QoS . Beberapa

parameter QoS yang digunakan untuk mengukur kualitas dan mengidentifikasi karakteristik jaringan antara lain yaitu *delay, packetloss, throughput, bandwidth, jitter,* dan *Round Trip Time* [19].

2.16.1 *DELAY*

Delay merupakan waktu yang dibutuhkan untuk mengirim paket data sampai pengirim mendapatkan respon paket data dari penerima [19]. Beberapa hal yang mempengaruhi nilai *Delay* antara lain yaitu media transmisi, jarak, dan lamanya proses. Jika pada suatu kondisi jaringan dengan nilai *Delay* yang tinggi maka mengindikasikan adanya hambatan dalam lalu lintas jaringan atau peningkatan penggunaan yang signifikan. Kategori untuk nilai *delay* ditunjukkan pada Tabel 2.1. Hal ini perlu menjadi perhatian dan segera diambil tindakan untuk mengindari kondisi jaringan yang *overload* [20].

Tabel 2.1 Kategori *Delay* (sumber TIPTHON) [21]

Kategori *Delay*	*Delay* (ms)	Indeks
Sangat Bagus	< 150	4
Bagus	150 – 300	3
Sedang	300 – 450	2
Buruk	>450	1

Perhitungan *Delay* dapat ditentukan dengan rumus sebagai berikut :
Delay= Waktu Paket Diterima –Waktu Paket Dikirim[19].

2.16.2 *PACKET LOSS*

Packet Loss merupakan parameter yang menunjukan jumlah paket data yang hilang atau jumlah paket data dikirim yang gagal mencapai penerima. *Packet loss* dapat terjadi karena beberapa faktor seperti *collision* dan *congestion* dalam lalu lintas jaringan. Kategori untuk nilai *packet loss* ditunjukkan pada Tabel 2.2 [20].

Tabel 2. 2 Kategori *Packet Loss* (sumber TIPHON) [21]

Kategori *Packet loss*	*Packet Loss* (%)	Indeks
Sangat Bagus	0 – 2	4
Bagus	3 – 14	3
Sedang	15 – 24	2
Buruk	>25	1

Perhitungan untuk menentukan persentase nilai *Packet Loss* yaitu:

$$Packet\ Loss = \frac{paket\ data\ dikirim - paket\ data\ diterima}{Paket\ data\ dikirim} \times 100$$ [20]

2.16.3 *THROUGHPUT*

Throughput adalah parameter yang menunjukkan kecepatan (*rate*) transfer data efektif. Kategori untuk nilai *throughput* ditunjukkan pada Tabel 2.3. Parameter ini memperlihatkan jumlah data yang dapat dikirimkan dan sampai ke penerima dalam interval waktu tertentu[20].

Tabel 2.3 Kategori *Throughput* (sumber TIPHON) [21]

Kategori *Troughput*	*Throughput*	Indeks
Bad	0 – 338 kbps	0
Poor	338 – 700 kbps	1
Fair	700 – 1200 kbps	2
Good	1200 kbps – 2,1 Mbps	3
Excelent	> 2,1 Mbps	4

Nilai *Throughput* dapat diketahui dengan persamaan berikut:

$$Throughput = \frac{Jumlah\ Data\ yang\ dikirim}{Waktu\ pengiriman\ data}$$ [20]

2.17 *PARAMETERS FOR BOTTLENECK EVALUATION OF THE WEB-BROWSING SERVICE*

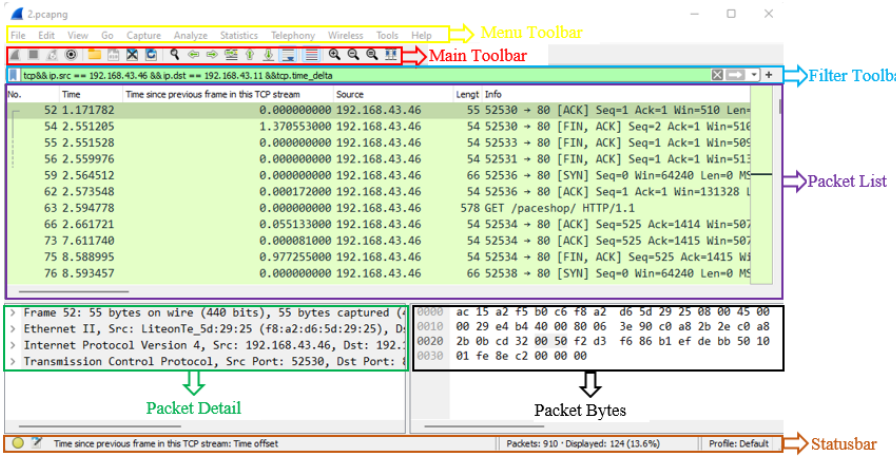
Bottleneck merupakan beberapa parameter yang berfungsi untuk mendiagnosis alasan dari penurunan QoS layanan *web-browsing*. Parameter yang termasuk *bottleneck* yaitu parameter a*plication layer, transport layer, network layer,* dan parameter karakteristik halaman web.

Round trip time (RTT) merupakan bagian dari parameter *network layer* yang dapat berfungsi untuk mendiagnosis penurunan QoS layanan *web-browsing*. mengacu pada waktu *round-trip* antara terminal dan *server web*. RTT dapat dihitung dengan *request Ping* dan *Ping reply*, yang biasanya digunakan untuk mengevaluasi QoS jaringan. RTT juga dapat dihitung dengan *three*-way *handshake* dari protokol TCP [22].

RTT menunjukan waktu yang digunakan oleh sebuah paket dari awal dikirim kemudian diterima oleh penerima sampai paket tersebut kembali ke pengirim. RTT juga dapat didefinisikan sebagai waktu percobaan koneksi internet untuk mengetahui kestabilan lalu lintas jaringan. Pada suatu kondisi koneksi internet ketika jumlah paketnya besar biasanya dikontrol oleh ukuran window TCP. Dalam fase penghindaran kemacetan (*congestion avoidance*), jendela kemacetan (*congestion window*) meningkat sebesar 1 paket secara umum pada setiap perhitungan RTT. Artinya dalam setiap detik, *Throughput* dari *node* meningkat 1/RTT paket/detik. Maka dapat disimpulkan bahwa ketika nilai waktu RTT lebih kecil maka terjadi peningkatan *Throughput* yang lebih tinggi [23].

2.18 *WIRESHARK*

Wireshark adalah aplikasi penganalisis protokol jaringan sumber terbuka yang tersedia secara gratis untuk tujuan pribadi, pendidikan, dan komersial yang dikelola oleh komunitas lebih dari 1.800 pengembang. Aplikasi ini biasanya digunakan oleh administrator jaringan atau insinyur jaringan untuk menganalisis pola lalu lintas jaringan, memecahkan masalah protokol jaringan masalah, dan melakukan analisis mendalam tentang celah keamanan jaringan. *Wireshark* juga menyediakan GUI yang mudah digunakan dan mudah dipahami dan kemampuan pemfilteran lanjutan untuk mencari melalui jutaan paket yang memudahkan administrator jaringan untuk menganalisis peristiwa di jaringan dengan cepat yang ditunjukkan pada Gambar 2.1.



Gambar 2.1 Struktur Tampilan GUI *Wireshark*

Berikut merupakan penjelasan dari Struktur pada tampilan GUI *wireshark*:

1. *Menu Toolbar* merupakan daftar *menu* yang disediakan *Wireshark* untuk akses beberapa fitur.
2. *Main Toolbar* menampilkan akses cepat terhadap item yang sering digunakan dari *menu toolbar*
3. *Filter Toolbar* berfungsi untuk mendefinisikan filter pada hasil *capture* paket data pada lalu lintas jaringan
4. *Packet List* berfungsi untuk menampilkan semua paket data dari hasil *capture* berdasarkan *filter*.
5. *Packet Details* berfungsi untuk menampilkan isi dari setiap paket data hasil *capture*.
6. *Packet Byte* berfungsi untuk menampilkan *Byte* data dari hasil *capture* jaringan
7. *Status Bar* berfungsi untuk menampilkan jenis paket data dan jumlah persentase paket data yang disaring[24].

2.19 APACHE JMETER

Apache Jmeter merupakan aplikasi *open source* yang menggunakan perangkat lunak java yang dirancang untuk mengukur kinerja dan pengujian beban fungsional. Apache jmeter mensimulasikan beban berat pada *server* untuk menguji dan menganalisis kinerjanya berdasarkan jenis beban tertentu. Jenis aplikasi/*server*/protokol yang dapat diuji kinerjanya menggunakan Apache jmeter antara lain yaitu *web* HTTP dan HTTPS (Java, *Node*JS, PHP, ASP.NET, …), SOAP / REST *Webservices*, FTP, TCP, dan Java *Objects* [25].

2.20 TSHARK

TShark merupakan versi CLI dari Wireshark, yang dirancang untuk menangkap dan menganalisis lalu lintas jaringan. Aplikasi ini mendukung opsi yang sama dengan Wireshark. Aplikasi *Wireshark* berbasis GUI atau aplikasi tshark berbasis CLI ini dapat digunakan untuk menangkap lalu lintas yang melintasi jaringan. Aplikasi direkomendasikan untuk perangkat yang hanya memiliki akses

CLI seperti Ubuntu. Pengguna dapat menginstal TShark menggunakan perintah *apt install Tshark*. Setelah diinstal, pengguna dapat meninjau opsi CLI untuk Tshark menggunakan perintah Tshark dengan opsi --*help* atau perintah *man tshark* [24].

2.21 *NETWORK ADDRESS TRANSLATION (NAT)*

Dalam penggunaannya yang paling umum, *Network Address Translation* (NAT) memungkinkan banyak komputer yang menggunakan IPv4 untuk terhubung ke Internet menggunakan satu alamat IPv4 publik. Perangkat lunak pfSense memungkinkan penerapan NAT, dan mengakomodasi konfigurasi NAT yang jauh lebih canggih dan kompleks yang diperlukan dalam jaringan dengan beberapa alamat IP publik. NAT dikonfigurasi dalam dua arah yaitu *outbound* dan *inbound*. *Outbound* NAT mendefinisikan lalu lintas yang meninggalkan jaringan lokal menuju jaringan jarak jauh, seperti akses ke internet. NAT *inbound* mengacu pada lalu lintas dari jaringan jarak jauh ke jaringan lokal. Jenis NAT *inbound* yang paling umum digunakan yaitu *port forwarding*. *Port Forward* memungkinkan akses ke *port* tertentu, rentang *port*, atau protokol pada perangkat jaringan internal yang dialamatkan secara pribadi. Aturan *port forward* dapat mengarahkan seluruh protokol seperti GRE, ESP, TCP dan UDP. *Port forward* sering digunakan untuk melakukan *hosting server*, atau menggunakan aplikasi yang memerlukan koneksi masuk dari Internet [26].