BAB II
DASAR TEORI

2.1. Kajian Pustaka

Pada kajian pustaka ini menjadi dasar acuan penelitian dalam melakukan penelitian, sehingga dapat menjadi studi literatur dalam mengkaji sebuah penelitian. Penelitian sebelumnya memiliki perbedaan rancangan dalam melakukan penelitian, ini dilakukan tidak lain karena topik penelitian yang berkaitan dengan keamanan jaringaan seperti *Intrusion Prevention System* (IPS). Berikut beberapa kajian pustaka yang berisi penelitian yang sebelumnya terkait dengan penelitian ini.

Veeramreddy Jyothsna *and* Koneti Munivara Prasad dengan judul "*Anomaly-Based Intrusion Detection System*". Pada penelitian ini diketahui pengamanan jaringan dengan menggunakan *Intrusion Detection System* (IDS) sebagai sistem pengidentifikasian intrusi pada tingkat *false* alarm *high detection rate* dan *low detection rate*. Dari hal tersebut diketahui bahwa IDS dapat mengupayakan pendeteksian adanya intrusi dengan berbasis *Anomaly-based Detection* (AD) atau *Behavior-based Detection* (BD) untuk memainkan peran penting dalam melindungi jaringan dari aktivitas berbahaya yang belum terindentifikasi dan terprediksi. Serangan yang diujikan yaitu *Denial of Service* (DoS), *Remote To Local* (R2L) dan *User To Root* (U2R). Hasil penelitian menunjukkan bahwa akurasi FCAAIS dengan fitur optimal sebesar 91%, sedangkan akurasi FAIS dengan semua fitur adalah 88%. Ketepatan model FCAAIS dengan fitur optimal dan FAIS dengan semua fitur hampir mendekati 92%. Diketahui bahwa penerapan korelasi kanonik terhadap pemilihan atribut yang dioptimalkan memiliki peningkatan 3% dalam akurasi FAIS. Metrik kinerja lainnya seperti sensitivitas, spesifisitas, dan ukuran F dihitung pada FCAAIS melalui FAIS. Sensitivitas, spesifisitas, dan ukuran F adalah 96, 49 dan 95% untuk FCAAIS, sedangkan untuk FAIS adalah 95, 46, dan 91% [6].

Pada penelitian ini diketahui bahwa *Intrusion Detection System* (IDS) dengan menggunakan metode deteksi *Signature-based Detection* (SD) dapat mendeteksi serangan berdasarkan *rules*, sedangkan metode *Anomaly-based Detection* (AD) dapat mendeteksi serangan berdasarkan perbedaan yang terjadi secara signifikan dengan mengidentifikasi perubahan status sistem dari status normalnya. Serangan yang diujikan yaitu *Denial of Service* (DoS), *Remote to Local* (R2L) dan *User to Root* (U2R). Hasil penelitian ini menunjukan bahwa penggunaan pengamanan IDS dengan menggunakan metode *Signature-based Detection* (SD) memiliki tingkat akurasi deteksi yang baik untuk ancaman yang sudah terdefinisi tetapi tidak baik dalam menangani ancaman yang tidak terdefinisi, sedangkan metode *Anomaly-based Detection* (AD) sangat baik dalam akurasi deteksi ancaman yang belum terdefinisi akan tetapi dari hal tersebut diketahui bahwa kedua metode deteksi perlu digunakan secara bersamaan untuk memantau aktifitas trafik jaringan dengan status normal, *unnormal* dan *unknown* [7].

Pada penelitian ini meneliti dengan melakukan implementasi *Intrusion Detection* and *Prevention System* **(**IDPS) yang mampu mendeteksi dan memblokir adanya serangan dari penyusup, dengan metode *Network Intrusion Detection and Prevention System* **(**NIDPS) yang dapat dikolaborasikan dengan Iptables. Iptables ini digunakan untuk memfilter paket data yang masuk dan mendrop paket data yang terindikasi serangan. Untuk menentukan paket data layak atau tidak ke dalam jaringan dengan metode *Signature-based Intrusion Prevention System* dan *Anomaly-based Intrusion Prevention System*. Metode *Signature-based* ini digunakan untuk menilai paket yang dikirim berbahaya atau tidak yang dibandingkan dengan daftar yang telah tersedia. Sedangkan *Anomaly-based* digunakan untuk mengetahui pola paket apa saja yang akan ada pada sebuah jaringan yang tidak terdaftar di *rule database*. Selain itu menggunakan aplikasi Suricata untuk memeriksa *volume* lalu lintas paket data yang banyak tanpa menurangi jumlah *rules*. Adapun serangan yang dilakukan adalah *port scanning* pada protokol FTP dan Telnet. Hasil dari penelitian ini bahwa sistem keamanan

IDPS dapat di implementasikan dan dapat melindungi komputer *host* yang terhubung pada jaringan. *Administrator* Sistem keamanan jaringan IDPS dapat mengontrol lalu lintas data yang masuk pada jaringan komputer [8].

Pada penelitian ini meneliti dengan melakukan implementasi IDPS pada LAN yang bertujuan untuk memproteksi jaringan dari seorang *intruder* yang berusaha melakukan penyusupan atau penyerangan. Metode yang digunakan yaitu IDS dan *firewall*. IDS yang digunakan adalah Snort yang bertugas untuk melakukan pencatatan dan pemberitahuan saat mendeteksi trafik yang mencurigakan, sedangkan *firewall* yang digunakan adalah Iptables untuk memblok paket data baik secara manual oleh *network administrator* maupun secara otomatis setelah diatur *security* policynya. Penelitian ini dilengkapi dengan barnyard dan BASE. Barnyard bertugas untuk menangani *file output* dari Snort, sehingga Snort dapat berfokus untuk melakuakan sensor terhadap trafik jaringan. Sedangkan *Basic Analysis Security Engine* (BASE) bertugas untuk merepresentasikan *log file* Snort kedalam format berbasis GUI. Adapun serangan yang diujikan adalah *Man In The Middle* (MiTM) dan DoS. Hasil implementasi dari penelitian ini fungsionalitas dari komponen dapat mengetahui adanya serangan pada LAN dan kemudian melakukan blok terhadap paket yang mencurigakan. [9]

Pada penelitian ini meneliti dengan melakukan implementasi *Intrusion Prevention System* (IPS) menggunakan Snort dengan mode *in-line* dan Iptables berbasis Linux yang bertujuan untuk mendeteksi adanya serangan dan melakukan *drop* paket serangan. Snort digunakan untuk memberikan *alert* dari adanya paket yang berbahaya, sedangkan Iptables digunakan untuk *drop* paket, *reject* paket dan melakukan pemblokiran yang lain melalui *rule*. Adapun serangan yang dilakukan yaitu *Denial of Service* (DoS). Hasil yang didapatkan pada penelitian ini konfigurasi Snort dapat dilakukan dengan dua cara yaitu IDS dan Iptables. Selain itu serangan dicegah dengan cara di *monitoring*, *drop* dan *filter* paket. Snort dan Barnyard menklasifikasikan paket serangan dari *port*, *source* IP dan *destination* IP [10].

2.2. Dasar Teori

Pada landasan teori ini akan dibahas tentang teori dasar yang melandasi permasalahan dan penyelesaian yang diangkat dalam penelitian ini adalah sebagai berikut.

2.2.1. Keamanan Jaringan

Sistem keamanan jaringan adalah proses untuk mengidentifikasi dan mencegah pengguna yang tidak sah dari suatu jaringan komputer. Tujuannya untuk mengantisipasi resiko ancaman berupa perusakan bagian fisik komputer maupun pencurian data [11].

Keamanan jaringan adalah aktivitas mencegah dan melindungi data dan informasi di dalam sistem jaringan. yang biasanya ada dalam sebuah korporasi, dari gangguan yang tidak sah atau bukan bagian dari pengakses info atau data yang sah [12]. Mencakup teknologi, piranti, dan proses untuk mengamankan semua yang terdapat di dalam sistem jaringan. Adapun prinsip keamanan jaringan sebagai berikut:

1. *Confidentiality* (Kerahasiaan)

   Kerahasian data informasi pada sistem atau *database*, adalah hal yang rahasia dan pengguna atau orang yang tidak berkepentingan tidak dapat melihat atau mengaksesnya [13]. Adapun ancaman yang muncul seperti lemahnya *password* yang digunakan maupun masuknya virus *backdoor* ke dalam sistem.

2. *Integrity* (Integritas)

   Integritas data tidak dirubah dari aslinya oleh orang yang tidak berhak, sehingga akurasi dan validitas data masih terjaga [13]. Adapun ancaman serangan pada prinsip *integrity* yaitu virus, *logic bomb*, *error coding* dan aplikasi yang menyebabkan data berubah.

3. *Availability* (Ketersediaan)

   Ketersediaan dalam menjamin setiap pihak yang berwenang dapat mengakses data, objek dan sumber daya [13]. Ancaman serangan yang dilakukan berupa kegagalan perangkat, *error* pada *software*, masalah pada

lingkungan seperti banjir, kebakaran, mati listrik, dan juga ancaman *hacker* seperti DoS *Attack*.

2.2.2. *Firewall*

Firewall digunakan untuk melindungi dengan cara menyaring, membatasi atau bahkan menolak hubungan segmen pada jaringan pribadi dengan jaringan luar yang bukan bagian ruang lingkupnya yang diterapkan baik terhadap *hardware*, *software* ataupun sistem itu sendiri. *Firewall* paket *filtering* termasuk salah satu jenis teknologi keamanan yang digunakan untuk mengatur paket-paket apa saja yang diizinkan masuk ke dalam sistem atau jaringan dan paket-paket apa saja yang diblokir [14]. Dan aplikasi *firewal* yang digunakan adalah Iptables.

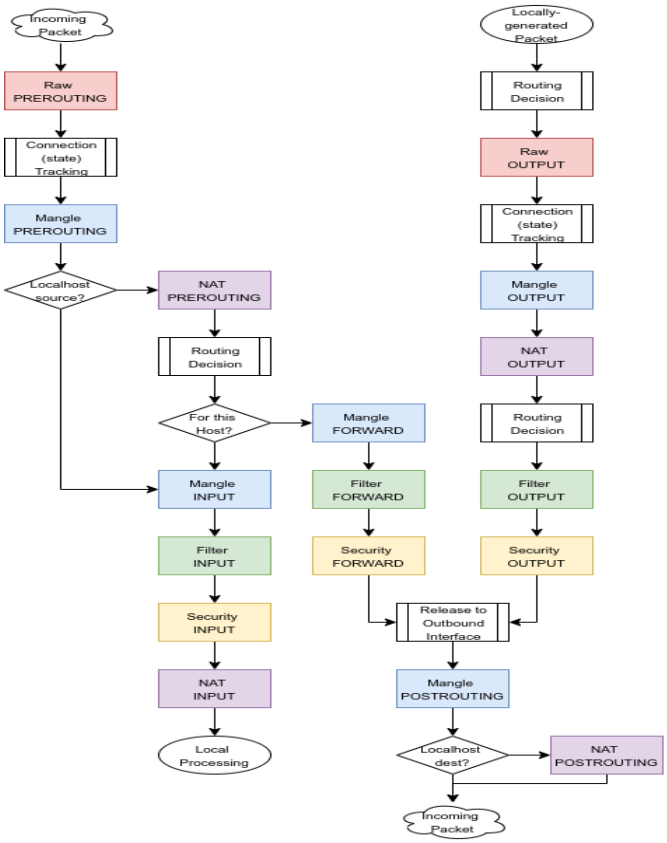
Dalam hal ini, Iptables berfungsi untuk menganalisis dan menyaring paket data yang masuk kedalam *firewall*, apakah paket data tersebut akan di *drop*, *accept* dan *reject*. Iptables ini juga sebagai alat untuk menyaring paket yang masuk, keluar dan sedang berlalu lintas didalam *firewall* melalui *server*.

1. *Filter, berfungsi* enentukan paket yang akan di *DROP*, *LOG*, *ACCEPT*, atau *REJECT*.
   a. *FORWARD*: Melakukan *filter* paket yang akan di *forward* dari NIC satu ke NIC yang lain seperti fungsi pada *router*.
   b. *INPUT*: Melakukan *filter* paket yang ditujukan untuk *firewall*.
   c. *OUTPUT*: Melakukan *filter* paket yang akan keluar dari *firewall*.
2. *Network Address Translation* (NAT), berfungsi untuk mentranslasikan atau merubah alamat asal atau tujuan dari sebuah paket.
   a. *PREROUTING*: Digunakan untuk mentranslasikan *address* sebelum proses routing terjadi, yaitu merubah IP tujuan dari paket data biasanya disebut dengan *Destination* NAT atau DNAT.
   b. POST*ROUTING*: Digunakan untuk mentraslasikan *address* setelah proses routing terjadi, yaitu merubah *source* IP dari paket data biasanya disebut dengan *source* NAT atau SNAT.
   c. OUTPUT: Digunakan untuk mentranslasikan *address* paket data yang

berasal dari *Firewall* itu sendiri.

3. *Mangle*, berfungsi untuk melakukan penghalusan (*mangle*) pada paket data seperti TTL, TOS, dan MARK. Untuk *mangle* sendiri mempunya 5 buah chain, yaitu: *PREROUTING*, *POSTROUTING*, *INPUT*, *OUTPUT*, *FORWARD* [15].

Salah satu aplikasi dari *firewall* adalah Iptables dan cara kejanya dapat dilihat pada Gambar 2.4.



Gambar 2.2. Alur kerja Iptables

2.2.3. *Intrusion Detection Prevention System* (IDPS)

IDPS merupakan IDS dan IPS yang dapat melakukan *monitoring traffic* jaringan, *detection* dan pencegahan terhadap paket data yang mencurigakan. Berikut merupakan penjelasan dari IDS dan IPS.

1. *Intrusion Detection System* (IDS)

Instrusion Detection System atau IDS adalah suatu sistem aplikasi yang dapat memonitor lalu lintas jaringan dari paket-paket data yang mencurigakan atau yang melanggar aturan keamanan jaringan dan kemudian membuat laporan dari aktivitas jaringan tersebut [16].

Jenis-Jenis IDS dilihat dari kemampuan mendeteksi serangan atau penyusupan di dalam jaringan, maka IDS dibagi menjadi 2 yaitu :

a. *Network-based Intrusion Detection System* (NIDS)

IDS ini di desain untuk menganilsa semua lalu lintas yang melewati ke sebuah jaringan yang akan mencari adanya percobaan serangan atau penyusupan ke dalam sistem jaringan [16]. Kelemahan dari NIDS adalah rumit untuk diimplementasikan dalam jaringan yang menggunakan *switch ethernet*, meskipun beberapa *vendor switch ethernet* sekarang telah menerapkan fungsi IDS di dalam *switch* buatannya untuk memonitor *port* atau koneksi.

b. *Host Intrution Detection System* (HIDS)

IDS jenis ini berjalan pada host yang berdiri sendiri atau perlengkapan dalam sebuah jaringan [16]. HIDS ini melakukan pengawasan terhadap paket yang berasal dari dalam maupun dari luar hanya pada satu aplikasi dan kemudian memberi peringatan kepada *user* atau *adminitrator* jaringan akan adanya kegiatan mencurigakan yang terdeteksi oleh HIDS.

2. *Intrusion Prevention System* (IPS)

Intrusion Prevention System merupakan gabungan antara IDS dan *firewall*. IPS digunakan untuk memecahkan masalah serangan yang terjadi pada jaringan komputer. IPS membuat akses kontrol dengan cara melihat konten aplikasi, dari pada melihat IP *address* atau *port*, yang biasanya dilakukan oleh *firewall*. Sistem setup IPS sama dengan sistem *setup* IDS. IPS mampu mencegah serangan yang datang dengan bantuan *administrator* [17] . IPS ini nantinya akan menghalangi suatu serangan yang belum terjadi dalam

melakukan eksekusi pada memori dan membandingkan *file* yang tidak semestinya mendapatkan izin untuk dieksekusi.

Terdapat dua jenis *Intrusion Prevention System* (IPS) yaitu:

a. *Host-based Intrusion Prevention System* (HIPS)

HIPS ini akan melindungi sistem operasi dan aplikasi yang berjalan pada *host* dengan mengidentifikasi dan menghentikan *known attack* dan *unknown attack*. *Software* HIPS memiliki kemampuan dapat melakukan *deny* atau *permit request* ketika *request* tersebut diidentifikasi sebagai *request* yang berbahaya atau tidak [18]. Pada HIPS ini akan memproteksi *traffic* pada *interface network*, memproteksi *integrity file*, dan memproteksi *behavior* aplikasi.

b. *Network-based Intrusion Prevention System* (NIPS)

NIPS ini melindungi jaringan secara keseluruhan. IPS umunya diletakkan *in-line* dengan *firewall* dan *network*. Sederhananya, semua *traffic* yang melalui NIPS dari *internet* yang mengarah ke *firewall* dan keluar *firewall*, jika menghasilkan *alert* akan di *drop* oleh NIPS dan tidak akan diteruskan ke *network* [18]. NIPS harus memiliki kemampuan yang sama dengan NIDS, dimana secara berkelanjutan melakukan *monitoring* terhadap abnormal *traffic pattern*, men-*generate event log*, menghasilkan *alert* dan yang terpenting menghentikan intrusi yang masuk ke *network user.*

Dalam mengenali pola serangan, ada beberapa metode IDPS bekerja yaitu dengan teknik *Signature-based Detection* dan *Anomaly-based Detection*.

1. *Signature-based Dertection* atau *Misuse-based Detection*

IDS yang berbasis pada *signature* akan melakukan pengawasan terhadap paket-paket dalam jaringan dan melakukan pembandingan terhadap paket-paket tersebut dengan basis data *signature* yang dimiliki oleh sistem IDS [16]. Cara yang dilakukan hampir sama dengan cara kerja aplikasi antivirus dalam melakukan deteksi terhadap *malware*, dimana akan terjadi keterlambatan untuk mendeteksi sebuah serangan di *internet* dengan

signature yang digunakan. Selama waktu keterlambatan tersebut sistem IDS tidak dapat mendeteksi adanya jenis serangan baru.

2. *Anomaly-based Detection* atau *Behaviour-based Detection*

IDS jenis ini akan mengawasi lalu lintas dalam jaringan dan melakukan perbandingan lalu lintas yang terjadi dengan rata-rata lalu lintas yang ada (Stabil). Sistem akan melakukan identifikasi maksud dari jaringan yang normal, jumlah *bandwidth* yang digunakan, jenis protokol yang digunakan serta *port* maupun alat yang saling berhubungan satu sama lain didalam jaringan tersebut, dan memberi peringatan kepada *administrator* ketika dideteksi ada yang tidak normal [16]. Metode *anomaly-based Detection* dapat mendeteksi bentuk serangan yang baru dan belum terdapat di dalam *database signature Detection*. Namun kelemahannya adalah sering mengeluarkan pesan *false positive*. Sehingga tugas *Administrator* menjadi lebih rumit, dengan harus memilah-milah mana yang merupakan serangan yang sebenarnya dari banyaknya *False Positive* (FP) yang muncul.

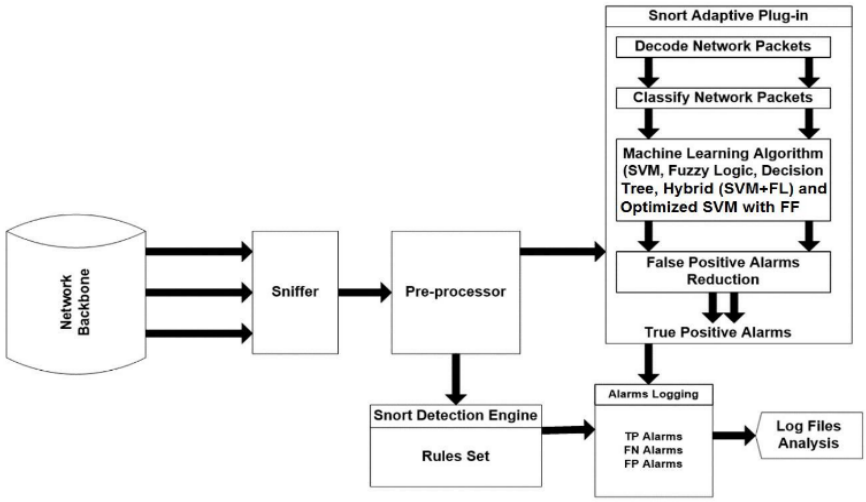
Diketahui ada perbandingan antara IDS dan IPS**.**

Tabel 2.1. Perbandingan IDS dan IPS [19]

	IDS	**IPS**
***Layer* OSI**	***Layer* 3**	***Layer* 2, 3 dan 7**
Manfaat	Identifikasi dan memeriksa semua paket yang melalui *traffic* jaringan, jika tidak terdapat anomali, maka IDS akan memberi *alert*.	Menggabungkan fungsi *firewall*, QoS dan IDS. Selain dapat mendeteksi anomali, IPS juga dapat menyediakan fungsi *allow*, *block* dan *log*.
Aktifitas	Mendeteksi serangan hanya ketika serangahn sudah masuk ke jaringan dan tidak dapat melakukan sesuatu untuk menghentikannya.	*Early detection*, teknik yang proaktif, dapat mencegah serangan masuk dan dapat menghentikan serangan dengan *block*.
Komponen	Tidak dapat mendeteks semua aktifitas *malicious code* setiap saat sehingga dapat mengakibatkan *False Negative* (FN) yang banyak.	Dapat mendeteksi *new signature* dan *behavior attack*, sehingga akan menurunkan tingkat *False Negative* (FN)
Compability	Tidak dapat dan mampu menggunakan ACL/ *script* dari komponen keamanan lain	Dapat diintegrasikan dengan ACL dan perimeter DMZ lainnya

2.2.4. Snort

Snort merupakan suatu perangkat lunak untuk mendeteksi penyusup dan mampu menganalisa paket yang melintasi jaringan secara langsung dan melakukan pencatatan ke dalam penyimpanan data serta mampu mendeteksi berbagai serangan yang berasal dari luar jaringan. Snort memanfaatkan perangkat tcpdump untuk mengambil dan menganalisis paket data terhadap sekumpulan jenis serangan yang sudah terdefinisi. Snort dapat berjalan dalam tiga mode antara lain :



Gambar 2.3.Arsitektur Snort dengan *Adaptive Plug-in*

1. Paket *sniffer*, melihat paket yang lewat di jaringan.
2. Paket *logger*, mencatat semua paket yang lewat di jaringan untuk di analisis.
3. NIDS, mendeteksi serangan yang dilakukan melalui jaringan komputer dengan konfigurasi dari berbagai aturan yang akan membedakan sebuah paket normal dengan paket serangan [20].

Pada gambar 2.5 Snort arsitektur *plug-in* adaptif Snort diatas bertujuan untuk mengurangi *alarm false positive* menggunakan algoritma pembelajaran mesin. Arsitektur baru yang diusulkan dari Snort IPS beroperasi secara paralel dengan *set* aturan Snort.

Alasan untuk mengintegrasikan *plug-in* adaptif Snort paralel dengan *set* lain adalah karena aturan Snort hanya mendeteksi lalu lintas berbahaya yang

diketahui. Lalu lintas berbahaya yang tidak diketahui atau variannya dapat dideteksi oleh *plug-in* yang pada gilirannya dapat mengurangi *alarm* positif palsu. Pengurangan alarm *false positive* ini jelas akan meningkatkan akurasi pendeteksian. Pra-prosesor akan memasukkan lalu lintas jaringan ke dalam *plug-in* dan kumpulan aturan Snort dan keduanya akan beroperasi secara paralel untuk mendeteksi lalu lintas berbahaya dengan lebih akurat. *Plug-in* adaptif memiliki empat komponen sebagai berikut:

1. *Decode Network Packets*

   Komponen ini menerjemahkan data paket untuk mendapatkan informasi rinci: *address* IP, *port* IP *source* dan *destination*, MAC alamat, bingkai *Ethernet* dan ukuran paket.
2. Klasifikasikan Paket Jaringan

   Membedakan antara lalu lintas yang sah dan berbahaya.
3. Algoritma *machine learning*

   Adaptive plug-in bisa menggunakan algoritma *Decision Tree*, *Hybrid* SVM dan *Fuzzy Logic*, dan SVM yang dioptimalkan dengan *firefly algorithm* untuk memproses lalu lintas yang sah dan berbahaya.
4. Pengurangan *alarm False Positive* (FP).

   Komponen ini selanjutnya mengurangi alarm positif palsu untuk mengirim *alarm* positif benar ke *file log* Snort [20].

Karena Snort berkemampuan membuat NIDS dan NIPS berbasis *signature-based* yaitu *rules* telah ditentukan sebelumnya yang terdiri dari dua bagian diantaranya, *header* menentukan tindakan yang akan dilakukan Snort. Ini termasuk faktor tindakan yang terkait *rules*, seperti *log* atau *alert*. Bagian kedua dari rules Snort adalah *option*. *Option* merupakan tempat *administrator* keamanan menentukan apa yang terlibat dalam paket dan pesan yang akan ditampilkan ketika Snort berhasil mencocokan *rules* dengan paket [21].

Snort mengikuti sintaks tertentu. Secara terpisah, mirip dengan aturan *firewall*, tetapi bagian opsi disini lebih banyak dari pada *firewall*. Sintaksnya ditampilkan di bawah ini.

Tabel 2.2. Format *rules* Snort

Rules							
Header							***Option***
action	*protocol*	*source* IP	*source port*	*Direction*	*destination* IP	*destinationport*	*Messages*, etc

Pada penelitian ini menggunakan aplikasi Snort3, karena dengan Snort3 *rules* lebih cepat dan lebih efisien, pengguna memiliki kontrol Lebih besar yang dapat dijalankan pada berbagai sistem operasi. Snort3 memiliki beberapa keunggulan yakni :

1. Mendukung untuk menangani banyak pemrosesan paket *threads*.
2. Dapat mengkonfigurasi *sharing* data berdasarkan tabel atribut.
3. Konfigurasi yang mudah digunakan karena berbasis *script*.
4. Mendukung *key* sebagai komponen utama.
5. Melayani deteksi otomatis untuk konfigurasi tanpa *port*.
6. Mendukung *sticky buffer* dalam *rules*.
7. Dapat mengenerasi dokumentasi referensi secara otomatis.
8. Mendukung lintas platform atau *third party* yang lebih baik.

2.2.5. *Server*

Server adalah sebuah sistem komputer yang menyediakan jenis layanan (*Service*) tertentu dalam sebuah jaringan komputer. *Server* didukung dengan prosesor yang bersifat *scalable* dan RAM yang besar, juga dilengkapi dengan sistem operasi khusus, yang disebut sebagai sistem operasi jaringan (*Network operating system*). *Server* juga menjalankan perangkat lunak administratif yang mengontrol akses terhadap jaringan dan sumber daya yang terdapat di dalamnya, seperti halnya berkas atau alat pencetak (*Printer*), dan memberikan akses kepada *workstation* anggota jaringan. Adapun jenis-jenis dari *server* yaitu:

1. *Proxy server*

   Proxy server berfungsi sebagai gerbang antara jaringan *local* dengan jaringan *public* (*Internet*). *Proxy server* bekerja membatasi permintaan pada data dengan berbagi *file server* dengan *client* di luar jaringan tersebut [22].

2. *Mail server*

   Mail server merupakan *server* yang bertugas untuk menangani e-*mail* dan menangani permintaan *client* [22].

3. Telnet *server*

   Telnet *server* adalah layanan yang mengatur suatu komputer dengan melakukan *login* dan *logout* pada komputer *host* [22].

4. *File Transfer Protocol* (FTP) *server*

   FTP *server* berfungsi untuk mengatur *transfer* data pada suatu jaringan pada setiap perangkat [22].

5. *Web server*

   Web server berfungsi untuk menyimpan konten sebuah *website* dan melakukan komunikasi melalui *protocol* HTTP [22].

6. *Secure Shell* (SSH) *server*

   SSH *server* berfungsi untuk menerima permintaan dari *client* dan bertugas untuk mendeskripsi atau enkripsi *client* dan menjalankan perintah dari *client*. SSH *server* berjalan pada *port* 22 dengan koneksi TCP. SSH *Server* bisa digunakan untuk melakukan pertukaran data seperti *File Transfer Protocol* (FTP) [22].

7. *Domain Name System* (DNS) *server*

   DNS *server* adalah sebuah sistem *server* yang menerima permintaan dari *client* untuk mengetahui alamat IP yang digunakan oleh sebuah *domain*. DNS (*Domain Name System*) adalah *server* yang digunakan untuk mengetahui IP *Address* suatu *host* lewat *host name*. Fungsi utama dari sebuah DNS *server* adalah menerjemahkan nama *host* menjadi alamat IP atau sebaliknya sehingga nama sebuah *host* akan lebih mudah diingat oleh pengguna. Fungsi lain dari DNS adalah memberikan informasi tentang suatu *host* ke seluruh *internet* [22].

2.2.6. *Port scanning*

Port scanning adalah kegiatan *probe* dalam jumlah yang besar dengan

menggunakan aplikasi secara otomatis, dalam hal ini adalah Nmap. Sebuah *scanner* sebenarnya adalah *scanner* untuk *port* TCP/ IP, yaitu sebuah program yang menyerang *port* TCP/ IP dan melayani SSH, Telnet, FTP, HTTP, HTTPS serta mencatat respon dari komputer target. Dengan cara seperti ini, *user* program *scanner* dapat memperoleh informasi yang berharga dari *host* yang mejadi target [23].

2.2.7. *Brute force*

Serangan *brute force* adalah sebuah teknik serangan terhadap sebuah sistem keamanan komputer yang menggunakan percobaan terhadap semua kunci yang mungkin. Pendekatan ini pada awalnya merujuk pada sebuah program komputer yang mengandalkan kekuatan pemrosesan komputer dibandingkan kecerdasan manusia. Sebagai contoh, untuk menyelesaikan sebuah persamaan kuadrat seperti $x^2+7x-44=0$, di mana x adalah sebuah *integer*, dengan menggunakan teknik serangan *brute force*, penggunanya hanya dituntut untuk membuat program yang mencoba semua nilai *integer* yang mungkin untuk persamaan tersebut hingga nilai x sebagai jawabannya muncul. Istilah *brute force* sendiri dipopulerkan oleh Kenneth Thompson, dengan mottonya: "*When in doubt, use brute force*" (Jika ragu, gunakan *brute force*). Secara sederhana, menebak kode dengan mencoba semua kombinasi karakter yang mungkin. *Brute force attack* digunakan untuk menjebol akses ke suatu *host* (*Server*/ *workstation*/ *network*) atau kepada data yang terenkripsi [24].

Sebuah *password* dapat dibongkar dengan menggunakan program yang disebut sebagai *password cracker*. Program *password cracker* adalah program yang mencoba membuka sebuah *password* yang telah terenkripsi dengan menggunakan sebuah algoritma tertentu dengan cara mencoba semua kemungkinan. Teknik ini sangatlah sederhana, tapi efektivitasnya luar biasa, dan tidak ada satu pun sistem yang aman dari serangan ini, meski teknik ini memakan waktu yang sangat lama, khususnya untuk *password* yang rumit.

Namun ini tidak berarti bahwa *password cracker* membutuhkan *decrypt*. Pada prakteknya, mereka kebayakan tidak melakukan itu. Umumnya, kita tidak dapat melakukan *decrypt passwords* yang sudah terenkripsi dengan algoritma yang kuat. Proses enkripsi modern kebanyakan hanya memberikan satu jalan, di mana tidak ada proses pengembalian enkripsi. Namun, anda menggunakan aplikasi simulasi yang mempekerjakan algoritma yang sama yang digunakan untuk mengenkripsi *password* orisinal. Aplikasi tersebut membentuk analisis komparatif dengan mencoba kata demi kata dalam kecepatan tinggi. Mereka menganut "Azaz Keberuntungan", dengan harapan bahwa pada kesempatan tertentu mereka akan menemukan kata atau kalimat yang cocok [24]. Teori ini mungkin tepat mengena untuk yang terbiasa membuat *random password*. Pada kenyataannya, *password* yang baik sulit untuk di tebak oleh program *password cracker*.

2.2.8. *Distributed Denial of Service* (DDoS)

Distributed Denial of Service (DDoS) merupakan jenis serangan yang bertujuan menggangu hak akses pengguna jaringan yang dilakukan secara massif [25]. Secara umum serangan DDoS terdiri dari beberapa jenis serangan antara lain sebagai berikut.

1. *User Datagram Protocol* (UDP) *flood*

   Serangan ini membanjiri target dengan paket UDP. Tujuan serangan ini adalah untuk membanjiri *port* acak pada *host* secara jarak jauh. Hal ini menyebabkan *host* berulang kali memeriksa aplikasi yang mendengarkan di *port* tersebut, dan jika tidak ada aplikasi yang ditemukan maka akan membalas dengan paket ICMP *Destination Unreachable* [26]. Proses ini dapat menghabiskan sumber daya *host*, sehingga tidak dapat diakses.

2. *Internet Control Message Protocol* (ICMP) *flood*

   ICMP membanjiri sumber daya target dengan paket *Echo Request* (*Ping*) ICMP yang umumnya mengirim paket secepat mungkin tanpa menunggu balasan [26]. Jenis serangan ini dapat menghabiskan *bandwidth*

inbound dan *outbound*, karena *server* target akan sering merespons dengan paket ICMP *Echo Reply*, sehingga mengakibatkan lambatnya sistem secara keseluruhan.

3. *Ping of Death (PoD)*

Serangan *Ping of Death* melibatkan penyerang mengirim beberapa *ping* yang salah atau berbahaya ke komputer. Panjang paket maksimum dari sebuah paket IP (termasuk *header*) adalah 65.535 *byte*. Namun, Data *Link Layer*/Lapisan Data *Link* biasanya membatasi ukuran bingkai maksimum, misalnya 1500 *byte* melalui jaringan *ethernet*.

Dalam hal ini, paket IP yang besar dibagi menjadi beberapa paket IP (dikenal sebagai fragmen), dan host penerima memasang kembali fragmen IP tersebut ke dalam paket lengkap. Dalam skenario *Ping of Death*, setelah manipulasi konten fragmen yang berbahaya, penerima akan mendapatkan paket IP yang lebih besar dari 65.535 *byte* saat dipasang kembali. Ini dapat meluap *buffer* memori yang dialokasikan untuk paket, menyebabkan penolakan layanan untuk paket yang sah [26].

4. *Hypertext Transfer Protocol* (HTTP) *Flood*

Dalam serangan HTTP *flood* DDoS, penyerang mengeksploitasi permintaan HTTP *GET* atau *POST* yang tampaknya sah untuk menyerang *server web* atau aplikasi. Banjir HTTP tidak menggunakan paket yang salah format, *spoofing* atau teknik refleksi, dan membutuhkan lebih sedikit *bandwidth* daripada serangan lain untuk menjatuhkan situs atau *server* yang ditargetkan. Serangan tersebut paling efektif jika memaksa *server* atau aplikasi untuk mengalokasikan sumber daya semaksimal mungkin sebagai respons untuk setiap permintaan [26].

5. *Synchronize* (SYN) *flood*

Serangan ini mengeksploitasi kelemahan yang diketahui dalam urutan koneksi TCP ("*three way handshake*") [26], di mana membutuhkan tiga pertukaran paket agar *client* dapat berkomunikasi sepenuhnya dengan *server*. *Client* akan mengirimkan *request* kepada *server* berupa paket SYN. setelah

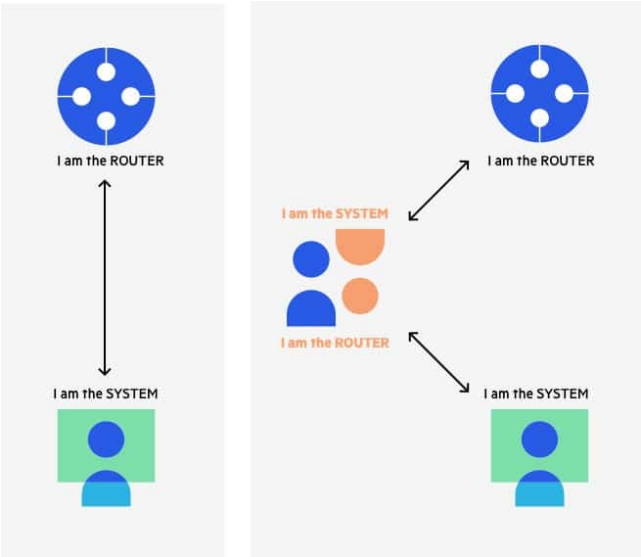
mendapatkan paket SYN dari *client*, *server* akan mambalas *request* dari *client* dengan mengirimkan paket SYN-ACK dan menunggu *client* untuk mengirimkan ACK kepada *server*. Namun, sangat mungkin untuk mengirimkan paket SYN tanpa mengirimkan kembali paket ACK. Hal tersebut membuat server akan terus menunggu balasan dari *client* yang sudah memang tidak mengirimkan ACK [27]. Di sisi lain, *client* terus menerus mengirimkan paket SYN dan membuat *server* kewalahan dan dapat mengakibatkan penurunan performa dari *server* itu sendiri

2.2.9. *Man-in-The-Middle* (MiTM)

MiTM adalah salah satu teknik dalam keamanan jaringan dimana penyusup menempatkan dirinya berada di tengah-tengah dua perangkat atau lebih yang saling berkomunikasi. [28] *Man-in-The-Middle attack* pada dasarnya penyerang memasukkan dirinya di antara dua pihak atau perangkat dalam mode sembunyi-sembunyi sehingga semua paket yang berlintas antara kedua pihak yang sah itu dialihkan melalui penyerang tersebut. Serangan ini cukup berbahaya karena penyerang kemudian dapat mengubah informasi dari paket yang dikirimkan, dan berpotensi mengirim data yang dipalsukan ke salah satu pihak. [29]

2.2.10. ARP *spoofing* (ARP *poisoning*)

ARP atau *Address Resolution Protocol* merupakan Protokol yang digunakan untuk memetakan alamat jaringan atau yang lebih dikenal dengan nama IP *address* ke alamat fisik atau MAC *address*. IP *address* atau alamat IP adalah alamat *virtual* yang ditetapkan pada setiap perangkat yang terhubung ke jaringan yang menggunakan IP. Alamat IP ini dapat berubah dan biasanya diatur oleh *Dynamic Host Control Protocol* (DHCP). Sedangkan MAC *address* atau alamat MAC ini pada dasarnya adalah alamat unik untuk setiap perangkat, bersifat tetap dan terdiri dari bilangan 48 bit yang direpresentasikan dengan bilangan heksadesimal yang terdapat pada Gambar 2.2 [30].



Gambar 2.4.Cara kerja serangan ARP *poisoning* [34]

2.2.11. DNS *spoofing*

Dengan semakin banyaknya orang yang terus menggunakan *web* untuk mendapatkan atau tukar-menukar informasi, DNS *Spoofing* telah menjadi metode penyerangan yang menarik bagi *hacker*s untuk mengumpulkan informasi yang berharga. *Web spoofing* memungkinkan penyerang untuk menciptakan "*shadow copy*" dari seluruh *World Wide Web* (WWW). Akses ke *web* bayangan diarahkan ke mesin penyerang yang memungkinkan penyerang untuk memonitor seluruh aktivitas pengguna termasuk mendapatkan dan mengubah informasi yang ditransfer melalui *web* tersebut. Serangan yang demikian memberikan metode bagi penyerang untuk mendapatkan informasi yang bersifat pribadi *password*, nomor rekening, alamat, nomor telepon dan lain-lain [31]. Sebagai tambahan, serangan ini dapat digunakan untuk memberikan informasi palsu yang menyesatkan pengguna sehingga menyebabkan salah satu tipe dari "*Denial of Service*" *attack* dengan meniadakan akses pengguna ke informasi *website* yang diinginkan.

2.2.12. *Social engineering*

Social engineering atau rekayasa sosial, adalah sebuah teknik manipulasi

yang memanfaatkan kesalahan manusia untuk mendapatkan akses pada informasi pribadi atau data-data berharga. Dalam dunia *cybercrime*, jenis penipuan *human hacking* ini dapat memikat pengguna untuk tidak menaruh curiga. Serangan seperti ini dapat terjadi secara *online*, langsung, dan melalui interaksi lainnya yang sulit untuk diduga. Umumnya, rekayasa sosial memiliki dua tujuan spesifik, yakni untuk sabotase dan mencuri [32].

2.2.13. *Confusion matrix*

Confusion matrix merupakan suatu metode yang berfungsi untuk melakukan analisis untuk menentukan tingkat *accuracy* sistem dengan baik. Nilai *True Positive* (TP) dan *True Negative* (TN) memberikan informasi ketika tingkat *accuracy* data bernilai benar. Sedangkan *False Positive* (FP) dan *False Negative* (FN) memberikan informasi tingkat *accuracy* dalam suatu sistem salah dalam melakukan pendeteksian. [33]

		Nilai Sebenarnya	
		TRUE	FALSE
Nilai Prediksi	TRUE	TP (True Positive)	FP (False Positive)
	FALSE	TN (True Negative)	FN (False Negative)

Gambar 2.5.Prediksi pada *confusion matrix*

Keterangan:

TP (*True Positive*) : Jumlah data dengan nilai sebenarnya positif dan nilai prediksi positif

FP (*False Positive*) : Jumlah data dengan nilai sebenarnya negatif dan nilai prediksi positif

FN (*False Negative*) : Jumlah data dengan nilai sebenarnya positif dan nilai prediksi negatif

TN (*True Negative*) : Jumlah data dengan nilai sebenarnya negatif dan nilai prediksi negatif

Banyak metrik digunakan untuk mengevaluasi metode pembelajaran mesin. Model optimal dipilih menggunakan metrik ini. Untuk mengukur efek deteksi secara komprehensif, beberapa metrik sering digunakan secara bersamaan dalam penelitian

1. *Accuration*

   Didefinisikan sebagai rasio sampel yang diklasifikasikan dengan benar terhadap sampel total. Akurasi adalah metrik yang cocok jika kumpulan data seimbang.

   Rumus:

$$Akurasi = \frac{TP + TN}{TP + FP + FN + TN} x\,100\,\% \tag{2.1}$$

2. *Precision* (P)

   Didefinisikan sebagai rasio sampel positif benar dengan sampel positif yang diprediksi itu mewakili kepercayaan deteksi serangan.

   Rumus:

$$P = \frac{TP}{TP + FP} x\,100\,\% \tag{2.2}$$

3. *Recall* (R)

   Didefinisikan sebagai rasio sampel positif benar terhadap sampel positif total dan juga disebut laju deteksi. Tingkat deteksi mencerminkan kemampuan model untuk mengenali serangan, yang merupakan metrik penting dalam IDS.

   Rumus:

$$R = \frac{TP}{TP + FN} x\,100\,\% \tag{2.3}$$