BAB 2
TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1 Intrusion Detection System (IDS)

Konsep dalam keamanan jaringan komputer memiliki tantangan dibidang yang berfokus pada perlindungan integritas, kerahasiaan, dan ketersediaan data yang dikirimkan melalui jaringan. Ancaman yang sering terjadi pada jaringan komputer meliputi berbagai jenis serangan, seperti serangan *Denial of Service (DoS), Probe, User to Root (U2R), Remote to Local (R2L),* Serangan *Malware* ataupun intrusi lainnya. Sistem keamanan jaringan yang handal dan menggabungkan penggunaan berbagai teknik, cara dan alat untuk mengidentifikasi, mencegah, dan memitigasi ancaman ini secara efektif sangat dibutuhkan. Saat ini kompleksitas jaringan modern, yang melibatkan berbagai perangkat dan protokol, serta menyajikan hasil deteksi tersebut dalam bentuk yang mudah dipahami oleh pengguna sangat dibutuhkan (Joko Suwarno, Galuh Saputri, 2024).

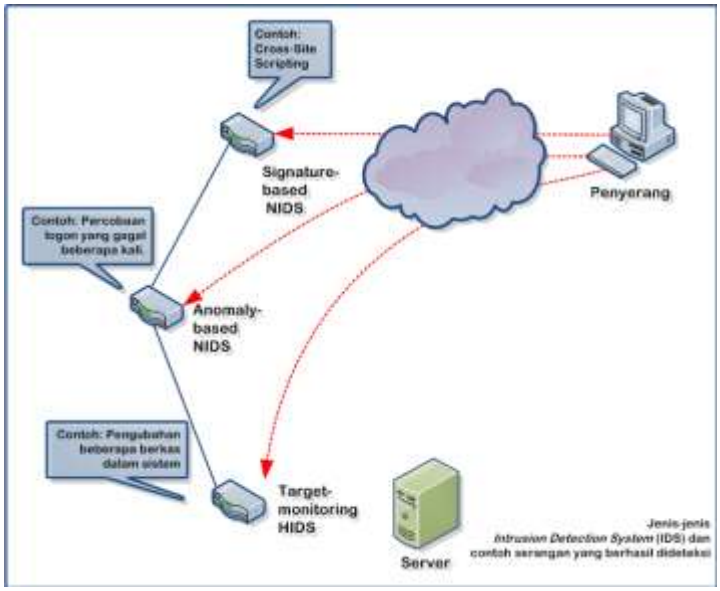
Intrusion Detection System (IDS) memiliki peran penting dalam menjaga keamanan jaringan komputer. *Intrusion Detection System* (IDS) merupakan sebuah sistem yang dirancang untuk mendeteksi adanya ancaman serangan siber (Antareza, 2024). IDS akan melakukan proses pengawasan kegiatan yang terjadi pada sistem komputer atau jaringan dan menganalisisnya untuk menentukan apakah kegiatan itu termasuk normal atau instrusi. IDS adalah *tools,* dengan mekanisme keamanan yang berfungsi memonitor aktivitas jaringan atau host untuk mengidentifikasi potensi serangan (Davies et al., 2025).

Kemampuan dari IDS adalah memberikan peringatan awal kepada administrator jaringan saat terjadinya sebuah aktifitas tertentu yang tidak diinginkan. IDS juga mampu melakukan pelacakan aktivitas yang merugikan suatu sistem. Suatu sistem IDS dapat melakukan *monitoring* terhadap paket-paket yang melewati jaringan dan berupaya menemukan apakah paket-paket tersebut berisi aktivitas-aktivitas yang mencurigakan atau tidak.

Ada dua jenis Klasifikasi IDS, yakni:

- *Network-based Intrusion Detection System (*NIDS) ➔ merupakan semua lalu lintas yang mengalir ke sebuah jaringan yang akan dianalisis untuk mencari apakah ada percobaan serangan atau penyusupan ke dalam sistem jaringan. Posisi NIDS berada dalam segmen jaringan penting di mana server berada atau pintu masuk sebuah jaringan..
- *Host-based Intrusion Detection System* (HIDS) ➔ merupakan aktivitas sebuah *host* jaringan individual yang dipantau apakah terjadi sebuah percobaan serangan atau penyusupan ke dalamnya atau tidak. HIDS seringnya berada pada server-server kritis di jaringan, seperti *firewall, web server*, atau server yang terkoneksi ke Internet.

Dalam mengenali pola serangan, ada beberapa metode yang dilakukan IDS dalam bekerja, yaitu *Signature Based* IDS dan *Anomali Based* IDS (Tambunan et al., 2020).



Gambar 2. 1 Prinsip Kerja IDS (Wikipedia, n.d.)

a. *Signature Based* IDS

Cara kerja IDS dengan *Signature* ini seperti halnya yang dilakukan oleh beberapa antivirus, yang melibatkan pencocokan lalu lintas jaringan dengan basis data yang berisi cara-cara serangan dan penyusupan yang sering dilakukan oleh penyerang. Jenis ini membutuhkan pembaruan terhadap basis data *signature* IDS

yang bersangkutan karena IDS tidak mampu mendeteksi adanya sebuah percobaan serangan yang tidak terdapat dalam basis data *signature* system IDS, sehingga tidak dapat mendeteksi adanya jenis serangan baru.

b. *Anomaly Based* IDS

Pada metode *Anomaly Based* IDS ini melibatkan pola lalu lintas yang merupakan sebuah serangan yang sedang dilakukan oleh penyerang. Pada umumnya dilakukan dengan teknik statistik untuk membandingkan lalu lintas yang sedang dipantau dengan lalu lintas normal yang biasa terjadi. Metode ini memiliki kelebihan dibandingkan *signature-based IDS*, yaitu dapat mendeteksi bentuk serangan yang baru dan belum terdapat di dalam basis data *signature* IDS. Namun kelemahannya adalah metode ini sering mengeluarkan pesan *false positive*. Sehingga administrator harus memilah-milah mana yang merupakan serangan yang sebenarnya dari banyaknya laporan *false positive* yang muncul.

2.1.2 SNORT

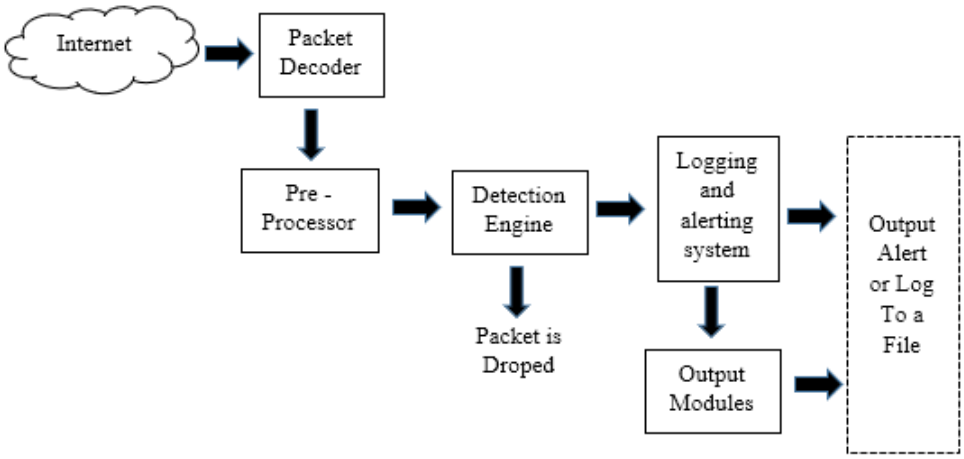
SNORT adalah sistem *Intrusion Detection System* (IDS) yang berfungsi untuk mendeteksi potensi ancaman atau serangan pada suatu jaringan dan bersifat *open source* (Antareza, 2024), yang terkenal karena kemampuannya untuk memeriksa lalu lintas IP secara waktu nyata (*real-time*), snort bekerja dengan menganalisis paket data untuk mendeteksi aktivitas mencurigakan seperti serangan siber dan melakukan pencatatan paket, lalu dapat menghasilkan peringatan atau bahkan memblokir lalu lintas berbahaya berdasarkan aturan (*rules*) yang ditentukan. SNORT mampu mengidentifikasi berbagai serangan, seperti upaya *port scanning*, serangan DDoS, dan lainnya.

Beberapa mode yang dilakukan oleh Snort dalam bekerja antara lain:

- *Packet Sniffer* ➔ Membaca paket dari *network* kemudian menampilkan dalam bentuk *continuous stream* di *console screen*
- *Packet Logger* ➔ Mencatat semua paket yang lewat di jaringan untuk dianalisa di kemudian hari
- *Intrsion Detection Mode* ➔ Pada mode ini snort akan berfungsi untuk mendeteksi serangan yang dilakukan melalui jaringan computer. Dalam

menggunakan mode IDS ini, harus dilakukan setup dari berbagai *rules*/aturan yang akan membedakan sebuah paket normal dengan paket yang membawa serangan.

Adapun Cara kerja SNORT dapat digambarkan seperti komponen-komponen pada gambar 2.2 berikut :



Gambar 2. 2 Cara Kerja SNORT
(Geologi, 2022)

SNORT bekerja dimulai dari *packet capture* dari jaringan luar, lalu *packet decoder* mengambil data hasil dari *packet capture* tersebut, selanjutnya akan dipisahkan data link seperti ethernet dan token ring kemudian protocol IP dan selanjutnya paket TCP dan UDP. Setelah pemisahan data selesai, snort telah memiliki informasi protokol yang dapat diproses selanjutnya, pada tahapan *preprocessing* dilakukan analisis atau manipulasi terhadap paket sebelum dikirim ke bagian *detection engine*. Pada proses *detect engine*, paket akan dibandingkan dengan rules yang telah dibuat sebelumnya oleh *rules database signature*. Selanjutnya Snort akan memberikan informasi terkait serangan seperti waktu kejadian, alamat IP sumber dan tujuan, protokol, serta jenis serangan yang akan dicatat dalam bentuk log dan memberikan alert/peringatan jika ditemukan aktivitas yang mencurigakan.

Adapun serangan yang dapat terdeteksi bisa berupa percobaan akses ilegal, pemindaian port, eksploitasi layanan ataupun penyisipan payload berbahaya. Namun Snort memiliki keterbatasan dalam mendeteksi *zero-day attack*, bergantung pada kualitas rules dan membutuhkan tuning untuk mengurangi *false positive*.

2.1.3 Jenis-jenis serangan pada IDS

Dalam sistem Itrusion Detection System (IDS), ada beberapa jenis serangan yang umum digunakan dalam simulasi intruder di jaringan, antara lain :

a. DDoS (*Distributed Denial of Service*), merupakan serangan yang dilakukan oleh banyak komputer secara bersamaan untuk membanjiri target dengan lalu lintas jaringan atau permintaan layanan. Adapun tujuan penyerang adalah untuk membuat server atau jaringan menjadi lambat, tidak dapat diakses atau berhenti beroperasi sehingga pengguna tidak dapat menggunakan layanan. Pada saat serangan terjadi, indikator yang menunjukkan bahwasanya serangan ini DDoS adalah ditandai dengan kata *"ddos", "flood" atau "dos"*.

   Ciri-ciri serangan ini antara lain :

   - Mengirimkan trafik dalam jumlah besar ke target
   - Menghabiskan sumber daya server sehingga layanan lambat dan tidak dapat diakses

   Karaketristik jaringan jika diserang DDoS adalah :

   - Jumlah paket meningkat sangat tinggi
   - Kapsaitas penggunaan CPU, RAM atau *bandwidth* yang meningkat drastis

   Contoh Alert : TCP SYN Flood, UDP *Flood Attack DoS Attemp*

b. Probe (Probing/Surveillance) : merupakan aktivitas pemindaian (scanning) dan pengumpulan informasi terhadap sistem target. Tujuan penyerang adalah menemukan port yang terbuka, layanan yang berjalan, sistem operasi atau celah keamanan yang bisa dimanfaatkan untuk melakukan serangan lanjutan. Pada saat serangan terjadi, indikator yang menunjukkan bahwasanya serangan ini Probe adalah ditandai dengan kata "portscan", "scan","probe","nmap", dan "ping".

   Ciri-ciri Probe antara lain :

   - Penyerang melakukan pengintaian terhadap target dengan tujuan mengumpulkan informasi sebelum melakukan serangan
   - Kegiatan ini tidak merusak sistem

   Karakteristik jaringan jika terjadi probe adalah :

- Banyaknya permintaan ke berbagai port dalam waktu singkat
- Pengiriman ICMP Echo Request secara berulang
- Upaya Identifikasi sistem operasi dan layanan yang aktif.

Contoh. Port Scanning, Nmap Scan, IP Sweep dan Vulnerability Scanning

c. R2L (Remote to Local) : merupakan serangan yang dilakukan dari komputer jarak jauh (remote) untuk mendapatkan akses ke sistem local tanpa akun yang sah. Tujuannya adalah menembus sistem dan mendapatkan hak akses sebagai pengguna biasa melalui pencurian password, brute force, eksploitasi layanan dan kelemahan autentikasi. Pada saat serangan terjadi, indikator yang menunjukkan bahwasanya serangan ini R2L adalah ditandai dengan kata "ssh brute force attack", "ftp brute force attack", "telnet login attack", "ssh", "ftp", "telnet", "login" dan "authentication".

   Karakteristik Jaringan jika ada serangan R2L adalah :

   - Banyak percobaan login yang gagal
   - Percobaan autentikasi berulang dari IP yang sama
   - Aktivitas terhadap layanan SSH, FTP atau telnet meningkat.

   Contoh. Brute Force Login, FTP Brute Force

d. U2R (User to Root) : merupakan serangan yang dilakukan oleh pengguna yang sudah memiliki akses ke sistem untuk meningkatkan hak akses menjadi administrator (root). Tujuannya adalah untuk mengambil alih sistem dengan memanfaatkan kerentanan priviledge escalation. Pada saat serangan terjadi, indikator yang menunjukkan bahwasanya serangan ini R2L adalah ditandai dengan kata "exploit","root","privilege","shellcode" dan "u2r".

   Karakteristik jaringan jika diserang U2R antara lain :

   - Eksekusi kode berbahaya
   - Percobaan privilege escalation
   - Manipulasi hak akses sistem
   - Aktivitas shell yang encurigakan

   Contoh. Privilledge Escalation Exploit, Local Exploit

2.1.4 Machine learning

Machine learning (Pembelajaran Mesin) merupakan sebuah sistem yang digunakan untuk belajar dan beradaptasi dengan informasi baru tanpa diprogram secara eksplisit untuk setiap situasi. Dalam konteks keamanan jaringan, ML berarti sistem yang dapat mempelajari pola lalu lintas normal dan abnormal dari waktu ke waktu, menyempurnakan dirinya sendiri untuk lebih membedakan ancaman yang sah dari alarm palsu (Aeraj et al., 2023). Sistem IDS yang berbasis pada aturan (rules) memiliki keterbatasan dalam mendeteksi serangan atau intruder yang belum pernah terjadi sebelumnya, begitu juga halnya dengan serangan yang menggunakan teknik baru. Hal inilah yang menjadikan machine learning sebagai solusi yang lebih adaptif dan canggih, dimana dengan kemampuan belajar dari data sebelumnya atau data training, maka ML mampu mengenali pola yang mencurigakan, sehingga machine learning dapat secara signifikan meningkatkan efektivitas deteksi intrusi. (Martha, 2024).

Dengan mengintegrasikan Pembelajaran Mesin (ML) dengan SNORT, sebuah sistem deteksi intrusi jaringan (NIDS) yang banyak digunakan, mekanisme keamanan yang lebih canggih dan adaptif dapat diciptakan. SNORT, dalam bentuk aslinya, terutama mengandalkan aturan yang ditentukan secara manual untuk mengidentifikasi ancaman jaringan. Aturan-aturan ini didasarkan pada *signatures* khusus untuk serangan yang sudah diketahui, yang dapat membatasi kemampuannya untuk mendeteksi ancaman baru atau varian dari serangan yang sudah ada. Namun, dengan mengintegrasikan dengan Machine Learning, kemampuan ini dapat ditingkatkan secara signifikan.

Machine Learning memiliki peran yang penting dalam deteksi intrusi dan menjadi salah satu pendekatan yang paling menjanjikan, hal ini dikarenakan kemampuannya untuk mengenali pola dari data historis dan mampu mendeteksi anomali secara otomatis (Polgan et al., 2024). Machine Learning diklasifikasikan menjadi tiga jenis utama yaitu supervised learning, unsupervised learning, dan reinforcement learning. Supervised learning merupakan klasifikasi yang memerlukan data yang telah diberi label untuk melatih model, sedangkan unsupervised learning dilakukan pada pengenalan pola dalam data yang tidak berlabel (Singh et al., 2021). Dalam penggunaan IDS, supervised learning lebih

umum digunakan dibandingkan unsupervised learning terutama algoritma seperti *Decision Tree, Random Forest, dan Support Vector Machine* (SVM), yang telah terbukti efektif dalam berbagai studi dan penelitian (Singh et al., 2021). *Classification* atau klasifikasi merupakan salah satu algoritma supervised learning. Pengukuran fitur statistik dalam klasifikasi menggunakan untuk mempelajari model dengan menggunakan algoritma. Metode klasifikasi dalam *machine learning* yang popular adalah *Support Vector Machine* (SVM). SVM merupakan sebuah model pembelajaran mesin berbasis kernel untuk tugas klasifikasi dan regresi (Cervantes et al., 2020).

2.1.5 Klasifikasi

Dalam penggunaan *machine learning* kategori *supervised learning*, salah satu tugas utamanya adalah klasifikasi. Metode ini adalah proses untuk menemukan model yang dapat membedakan kelas-kelas dari setiap data yang ada, dengan tujuan agar model tersebut dapat digunakan untuk memprediksi data-data yang belum diketahui kelasnya, klasifikasi merupakan salah satu tugas yang penting untuk beberapa aplikasi seperti *text categorization, tone recognition, dan data classification* (Maisat et al., 2023).

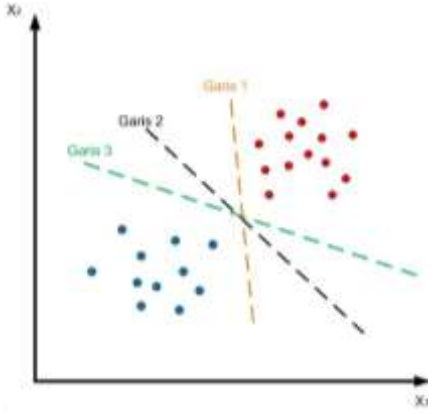
Dalam *machine learning*, proses klasifikasi diawali dengan tahap pengumpulan dan prapemrosesan data, selanjutnya dilakukan proses ekstraksi fitur yang relevan, pelatihan model, serta pengujian performa kerja dari algoritma *machine learning* tersebut dengan harapan model klasifikasi mampu melakukan generalisasi dengan memberikan prediksi yang tepat. Adapun beberapa algoritma yang sering digunakan dalam klasifikasi *machine learning* antara lain *Decision Tree, Random Forest, Naive Bayes, K-Nearest Neighbor (KNN), Support Vector Machines (SVM),* dan lainnya.

2.1.6 Algoritma *Support Vector Machine (SVM)*

Salah satu algoritma dalam *machine learning* adalah Algoritma *Support Vector Machine* (SVM) yaitu salah satu *algoritma supervised learning* yang digunakan untuk klasifikasi dan regresi. SVM bekerja dengan mencari *hyperplane* terbaik yang memisahkan data ke dalam kelas yang berbeda. Algoritma SVM terkenal

karena kemampuannya untuk menangani data berdimensi tinggi dan efektif dalam memisahkan data yang tidak linear melalui penggunaan kernel trick. Kernel trick memungkinkan SVM untuk memetakan data ke dalam ruang dimensi yang lebih tinggi, sehingga membuat data yang tidak dapat dipisahkan secara linear menjadi lebih mudah untuk diklasifikasikan (Muhammadiyah et al., 2024). SVM juga mampu mengatasi masalah *overfitting*, karena meminimalkan kesalahan klasifikasi sambil memaksimalkan margin antara kelas-kelas data, artinya, SVM tidak hanya mencoba untuk mengklasifikasikan data yang ada dengan benar, tetapi juga memastikan bahwa model yang dihasilkan tidak terlalu cocok dengan data pelatihan, sehingga meningkatkan kemampuan generalisasi terhadap data baru.. SVM memiliki beberapa keunggulan, yaitu memiliki akurasi yang tinggi, mampu mengklasifikasikan pada kondisi data yang bervariasi, efisien dalam penggunaan memori dan dapat menangani data yang tidak terdistribusi secara normal.

Pada Algoritma SVM, proses yang dilakukan guna mencari *hyperplane* atau fungsi pemisah (*decision boundary*) terbaik yang mampu memisahkan dua buah kelas atau lebih pada ruang input, dimana diperlukan optimasi pada SVM untuk menemukan jarak maksimum *hyperplane* dengan kedua kelas tersebut. *Hyperplane* dapat berupa line atau garis pada 2 dimensi dan dapat berupa flat plane pada multiple plane seperti pada gambar 2.3.



Gambar 2. 3 Ilustrasi Cara Kerja Support Vector Machine
(Mega Bagus ID, 2019)

Gambar 2.3 menunjukkan bahwa ada 2 kelompok data yang akan diklasifikasi. Dimana SVM akan membagi 2 kelompok ini sebaik mungkin, garis batasnya bisa memisahkan 2 kelompok dengan jarak terjauh antara titik terluar di

masing-masing kelompok dengan garis pembatas itu sendiri. Keunggulan utama SVM adalah kemampuannya untuk melakukan generalisasi yang baik dan ketahanannya terhadap *overfitting* pada dataset dengan jumlah fitur yang besar. Adapun performa SVM sangat dipengaruhi oleh pemilihan hyperparameter yang tepat dengan melakukan proses optimasi untuk mencapai hasil yang optimal.

2.1.7 Hyperparameter pada SVM

Hyperparameter Tuning adalah proses mencari kombinasi nilai hyperparameter terbaik untuk sebuah model *machine learning* seperti SVM guna mendapatkan performa yang optimal. Proses ini digunakan untuk memaksimalkan akurasi model dan meminimalkan kesalahan prediksi, serta mencegah masalah seperti *underfitting* (model terlalu sederhana) dan *overfitting* (model terlalu kompleks*). Hyperparameter* Utama pada SVM :

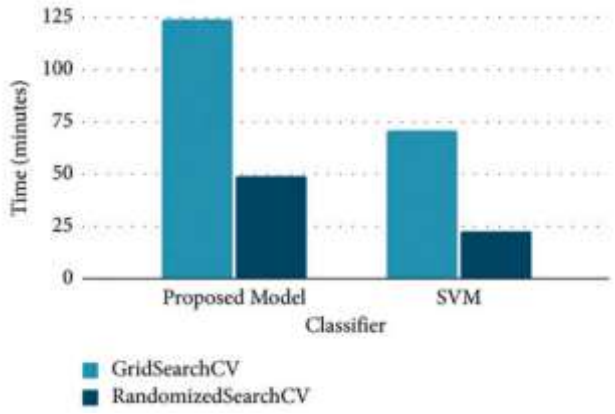
a. C (Parameter Regularisasi/Penalti) : C mengontrol *trade-off* antara memiliki margin pemisahan yang lebar dengan akurasi klasifikasi titik data pelatihan. Jika nilai C kecil, maka margin lebih lebar, tetapi berpotensi terjadi *underfitting,* sedangkan jika C besar, maka margin lebih sempit, menekankan klasifikasi yang benar dari setiap titik latih, dan berpotensi menyebabkan *overfitting.*
b. Gamma (untuk kernel rbf, poly dan sigmoid)
   Gamma akan menentukan seberapa jauh pengaruh satu titik data pelatihan tunggal, Jika nilai gamma kecil, maka pengaruhnya jauh (batas keputusan sangat mulus/halus), sebaliknya jika gamma besar, maka pengaruhnya dekat batas keputusan dan berpotensi *overfitting.*
c. Kernel
   Hyperparameter ini menentukan fungsi kernel yang digunakan untuk mentransformasi data ke ruang fitur berdimensi lebih tinggi, sehingga pengamatan lebih mudah dipisahkan secara linear

Metode Tuning Hyperparameter :

a. *Grid SearchCV* : *Grid Search Cross Validation* merupakan metode yang dapat digunakan untuk menemukan parameter yang optimal pada suatu

model. Sedangkan *Cross Validation* merupakan teknik pada *machine learning* yang membagi dataset menjadi subset lalu melatih dan menguji model pada subset tersebut secara berulang (Rusman et al., 2023). Algoritma ini bekerja dengan mencoba semua kombinasi yang mungkin berdasarkan pada nilai parameter yang telah disediakan oleh pengguna. metode ini menyimpan hasil parameter-parameter yang telah dikombinasikan ke dalam grid kemudian memilih parameter terbaik dan mencari parameter dengan nilai kesalahan terkecil (Fajri & Primajaya, 2023).

b. *RandomizedSearchCV* : Metode ini mengambil sampel acak dari kombinasi hyperparameter yang ditentukan dalam ruang pencarian lalu mengombinasikannya. *Randomsearch* berfokus pada eksplorasi nilai parameter yang memiliki dampak yang signifikan pada performa model saja. Berikut pada gambar 2.4 menunjukkan perbandingan lama waktu komputasi (dalam menit) antara dua teknik pencarian hyperparameter menggunakan Grid SearchCV dan RandomizedSearchCV.



Gambar 2. 4 Grafik perbandingan lama komputasi antara Grid SearchCV dan RandomizedSearch CV
(Samara et al., 2023)

2.1.8 Metode Grid Search CV untuk SVM

Grid Search adalah metode optimasi *hyperparameters* yang banyak digunakan pada algoritma *Support Vector Machine* (SVM). Metode ini melakukan pencarian menyeluruh melalui subset tertentu dari ruang *hyperparameter* sebuah algoritma pembelajaran. Pada *Grid SearchCV*, setiap kombinasi hyperparameter yang telah ditentukan akan diuji secara menyeluruh menggunakan skema validasi

silang, sehingga memungkinkan pemilihan konfigurasi parameter yang benar-benar memberikan kinerja terbaik berdasarkan metrik evaluasi yang digunakan. Algoritma *Grid Search* diuji menggunakan beberapa metrik kinerja, biasanya dengan metode *cross-validation* pada data pelatihan (Arfian et al., 2025), misalnya menggunakan k-fold CV untuk hasil yang lebih stabil/tidak terlalu bergantung pada satu pembagian data train/test.

K-Fold Cross Validation merupakan metode validasi model yang membagi data pelatihan menjadi K bagian (fold) dengan ukuran yang relatif sama, misalnya nilai K=10, maka data pelatihan akan dibagi menjadi 10 subset. Pada setiap iterasi 9 subset digunakan sebagai data pelatihan dan 1 subset sebagai data validation.

Penggunaan metode Grid SearchCV pada SVM memiliki jumlah hyperparameter SVM yang relatif terbatas dan terdefinisi dengan jelas, seperti parameter c, gamma dan kernel, sehingga ruang pencarian parameter tidak terlalu besar dan dapat mengevaluasi seluruh kombinasi *hyperparameter* secara efisien tanpa memerlukan sumber daya komputasi yang berlebihan.

Manfaat SVM dengan optimasi parameter via Grid Search adalah sebagai berikut :

- Memastikan performa maksimal model klasifikasi
- Mendapatkan parameter yang cocok, spesifik ke karakter data yang digunakan
- Memberikan *baseline* yang kuat sebelum bereksperimen dengan model yang lebih kompleks (Neural Network, ensemble dll).

2.1.9 Evaluasi Kinerja Model

Evaluasi terhadap kinerja model klasifikasi sangat penting karena mencerminkan sejauh mana model mampu mengklasifikasikan data. Evaluasi umumnya dilakukan dengan menggunakan *confusion matrix.* Terdapat empat macam istilah yang digunakan untuk pengukuran kinerja pada *confusion matrix* yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) *dan False Negative* (FN).

Tabel 2. 1 *Confusion Matrix*

Kelas Asli	Nilai Prediksi	
	Positif	Negatif
Positif	TP	FN
Negatif	FP	TN

Penjelasan :

TP = Jumlah data positif yang diprediksi positif

TN = Jumlah data negatif yang diprediksi negatif

FP = Jumlah data negatif yang salah prediksi sebagai positif

TN = Jumlah data positif yang salah prediksi sebagai negatif

Evaluasi dengan *confusion matrix* menghasilkan nilai *accuracy, precision, recall dan f1-score* sebagai berikut :

a. *Accuracy* merupakan nilai perbandingan antara data yang terklasifikasikan benar dengan keseluruhan data. *Accuracy* tinggi menunjukkan bahwa model memiliki kinerja yang baik dalam mengklasifikasikan data. *Accuracy* secara sistematis dinyatakan dalam persamaan berikut:

$$Accuracy = \frac{(TP + TN)}{(TP+TN+FP+FN)}$$

b. *Precision* merupakan rasio dari data positif yang diklasifikasikan bernilai benar terhadap jumlah total hasil prediksi positif. Dalam konsteks IDS, precision mengukur seberapa banyak prediksi serangan yang benar-benar merupakan serangan. *Precision* secara matematis dinyatakan dalam persamaan berikut :

$$Precision = \frac{(TP)}{(TP + FP)}$$

c. *Recall* merupakan rasio dari jumlah data positif yang diprediksi dengan benar (TP) terhadap jumlah data yang secara aktual bernilai positif. Dalam konteks IDS, recall akan mengukur kemampuan sistem dalam menemukan seluruh serangan yang ada. *Recall* secara matematis dinyatakan dengan persamaan :

$$Recall = \frac{(TP)}{(TP + FN)}$$

d. F-1 Score merupakan rasio keseimbangan antara precision dan recall. F1-score secara matematis dinyatakan dalam persamaan :

$$F1\text{-}Score = \frac{2(Precison \times Recall)}{(Precison + Recall)}$$

2.1.10 Dataset NSL-KDD

NSL-KDD merupakan dataset yang banyak digunakan dalam penelitian keamanan jaringan, khususnya pada pengembangan dan evaluasi *Intrusin Detection System.* Dataset NSL-KDD berisi data lalu lintas jaringan yang direpresentasikan dalam bentuk koneksi jaringan. Setiap baris data menggambarkan satu koneksi jaringan yang terjadi antara sumber dan tujuan tertentu dalam periode waktu tertentu. Setiap koneksi tersebut dilengkapi dengan sejumlah atribut yang merepresentasikan karakteristik koneksi, baik dari sisi protokol, isi koneksi, maupun pola trafik jaringan.

Dataset NSL-KDD memiliki 41 atribut fitur dan 1 atribur label kelas. Atribut-atribut ini terdiri dari atribut dasar koneksi, atribut berbasis konten, atribut berbasis trafik, dan atribut berbasis host. Kombinasi atribut ini memungkinkan model *machine learning* untuk mempelajari pola serangan jaringan secara menyeluruh, baik dari perilaku koneksi individual maupun dari pola trafik yang berulang, namun tidak semua fitur pada dataset ini yang bisa digunakan tergantung kebutuhan dan kesesuaian dengan fitur pada data realtime.

2.2 Penelitian Terdahulu

Bagian ini merupakan studi literatur yang merujuk pada beberapa penelitian terdahulu di antaranya adalah sebagai berikut :

Berdasarkan jurnal sebelumnya, Penelitian yang dilakukan untuk mengimplementasikan IDS menggunakan Snort yang dipadukan dengan pendekatan TPACK untuk mendeteksi dan memitigasi serangan *Denial of Service (DoS)* pada jaringan. Sistem berhasil mengidentifikasi anomali lalu lintas secara tepat dengan aturan Snort yang dirancang khusus, menunjukkan bahwa IDS berbasis *signature* mampu menangkap pola serangan yang telah dikenali sebelumnya. Kelemahannya, penelitian ini hanya berfokus pada deteksi serangan DoS dan intrusi lain, tapi tidak melibatkan algoritma *machine learning* sebagai penguat analisis, sehingga kemampuan deteksinya terbatas pada serangan yang sudah memiliki pola *signature* (S et al., 2025).

Penelitian selanjutnya adalah dengan melakukan monitoring keamanan jaringan dari serangan DDos seperti ICMP flooding, TCP flooding dan UDP

flooding dengan melibatkan telegram sebagai notifikasi, sama dengan penelitian sebelumnya, ini hanya berfokus pada deteksi serangan DoS dengan notifikasi telegram, dimana data log digunakan sebagai dasar analisis untuk mengevaluasi kemampuan sistem. Penelitian ini tidak menggunakan algoritma *machine learning* sebagai penguat analisis sehingga kemampuan deteksinya kurang adaptif dan juga terbatas dalam menghadapi serangan DDos yang berkembang (Nainggolan et al., 2022).

Kemudian, Penelitian selanjutnya mengintegrasikan Snort dengan algoritma SVM untuk meningkatkan akurasi deteksi dan menurunkan tingkat *false positive*, memanfaatkan dataset lalu lintas jaringan sebagai input model klasifikasi. Hasil eksperimen menunjukkan performa yang sangat tinggi dengan precision mencapai 99%, *false positive rate* hanya 1, dan tidak terdapat *false negative*, sehingga menegaskan potensi SVM sebagai *classifier* utama dalam penguatan kemampuan IDS berbasis Snort. Kelemahannya**,** penelitian ini belum membahas *hyperparameter tuning* yang optimal, tidak mengevaluasi pengaruh teknik optimasi seperti Grid Search atau metaheuristik terhadap peningkatan kinerja SVM, serta belum diuji pada lingkungan real-time atau dataset skala besar sehingga kurang merepresentasikan kondisi serangan modern yang dinamis (Aeraj et al., 2023)

Kemudian pada Penelitian jurnal sebelumnya juga mencoba menggabungkan Snort sebagai sensor deteksi jaringan dengan model machine learning untuk meningkatkan kemampuan Snort dalam mengidentifikasi anomali yang belum memiliki *signature*. Model pembelajaran yang digunakan mampu mengklasifikasikan aktivitas jaringan berdasarkan pola perilaku, sehingga membantu memperluas fungsi Snort dari sekadar *signature-based* menjadi *hybrid detection*. Kelemahannya, penelitian tidak melakukan optimasi *hyperparamete*r pada algoritma klasifikasi, sehingga performa model belum mencapai tingkat optimal, selain itu, penelitian tidak membahas mekanisme pembaruan model secara dinamis, sehingga IDS masih rentan terhadap jenis serangan baru yang berkembang cepat (Aeraj et al., 2023).

Penelitian selanjutnya membangun *Collaborative* IDS (CIDS), dimana beberapa node Snort yang saling berbagi log ke central node dan SIEM LogScale untuk meningkatkan deteksi ancaman dengan pendekatan kolaboratif. Hasil

evaluasi menunjukkan bahwa agregasi data dari banyak sensor mampu menurunkan *false positive* dan menangani serangan seperti port scanning dan ICMP flood lebih efektif dibandingkan Snort tunggal. Adapun kelemahannya penelitian ini tidak menggunakan machine learning untuk klasifikasi serangan, sehingga tidak membahas bagaimana model ML seperti SVM dapat meningkatkan deteksi berbasis Snort; selain itu desain CIDS menambah kompleksitas arsitektur dan belum dievaluasi pada skenario serangan modern yang lebih bervariasi atau lingkungan real-time (Davies et al., 2025)

Tabel 2. 2 Penelitian Terkait

No	Penulis	Tujuan Penelitian / Fokus Pengembangan Model	Dataset yang digunakan
1	Danu Satin S., Wahyuddin, Ahmad Kautsar, Agus Setyawan, 2025	Mengembangkan IDS berbasis Snort dengan sistem notifikasi real-time menggunakan Bot Telegram untuk mendeteksi dan memberikan peringatan terhadap serangan jaringan seperti DoS, FTP login, dan port scanning.	Tidak menggunakan dataset khusus; pengujian dilakukan pada lalu lintas jaringan real-time yang dibangkitkan selama eksperimen
2	Lasma Feronika Nainggolan, Naikson F. Saragih, Fati G.N. Larosa, 2022	Melakukan monitoring keamanan jaringan dari serangan DDos seperti ICMP flooding, TCP flooding dan UDP flooding dengan melibatkan telegram sebagai notifikasi.	Penelitian ini tidak menggunakan dataset publik melainkan data hasil simulasi serangan.
3	Ouafae El Aeraj, Cherkaoui Leghris, 2023	Menggabungkan Snort sebagai sensor deteksi jaringan dengan model machine learning untuk memperluas kemampuan Snort dari *signature-based* menjadi hybrid detection yang mampu mendeteksi anomali yang belum memiliki pola *signature*	Tidak disebutkan dataset spesifik; fokus pada integrasi sistem Snort dan model klasifikasi

No	Penulis	Tujuan Penelitian / Fokus Pengembangan Model	Dataset yang digunakan
4	Ouafae El Aeraj, Cherkaoui Leghris, 2024	Membuat sistem IDS yang mengintegrasikan Snort dengan algoritma Support Vector Machine untuk meningkatkan tingkat akurasi deteksi, meminimalkan *false positive*, dan memperkuat kemampuan identifikasi intrusi.	Waveform Dataset digunakan sebagai sumber data untuk pelatihan dan pengujian model SVM
5	Tom Davies, Max Hashem Eiza, Nathan Shone, Rob Lyon, 2025	Mengembangkan sistem Collaborative IDS (CIDS) dengan menghubungkan beberapa node SNORT yang mengirimkan log ke central SIEM LogScale untuk meningkatkan deteksi serangan seperti port scanning dan ICMP flood melalui pendekatan kolaboratif antar sensor.	Tidak menggunakan dataset statis tetapi menggunakan log jaringan realtime dari banyak node
6	Penelitian yang dilakukan	Mengembangkan Hybrid IDS Snort yang diperkuat dengan algoritma SVM teroptimasi melalui teknik hyperparameter tuning (misalnya Grid SearchCV atau metode optimasi lainnya) untuk meningkatkan akurasi deteksi, efisiensi komputasi, dan kemampuan identifikasi serangan modern baik *signature-based* maupun *anomaly-based*	Menggunakan dataset hasil simulasi serangan dari kali linux (attacker) ke server ubuntu (IDS Snort) yang tersimpan pada direktori log snort

Penelitian ini berjudul Optimasi *Hybrid Intrusion Detection System* : Integrasi Snort Dengan *Support Vector Machine* (SVM) Teroptimasi Untuk

Klasifikasi Jenis Serangan Jaringan. Adapun perbedaan penelitian ini dengan penelitian sebelumnya terdapat pada mode pengembangan *Hybrid* IDS Snort yang diperkuat dengan algoritma SVM teroptimasi melalui teknik Hyperparameter tuning, mnggunakan metode *Grid Search Cross Validation*, yaitu dengan mencoba setiap perpaduan atau kombinasi nilai hyperparameter yang telah ditentukan dalam grid untuk meningkatkan akurasi deteksi efisiensi komputasi, dan kemampuan identifikasi serangan modern baik *signature-based* maupun *anomaly-based*.

Berdasarkan karakteristik Snort sebagai sistem deteksi intrusi berbasis *signature* yang efektif dalam mengenali serangan yang telah dikenal, serta SVM sebagai algoritma *machine learning* yang memiliki kemampuan klasifikasi adaptif terhadap pola serangan yang kompleks, integrasi kedua pendekatan ini berpotensi menghasilkan sistem IDS yang lebih cerdas dan komprehensif. Kombinasi tersebut memungkinkan pemanfaatan kekuatan Snort dalam deteksi awal serangan, sekaligus kemampuan SVM dalam melakukan analisis lanjutan untuk meningkatkan akurasi klasifikasi dan mengurangi kesalahan deteksi. Dengan demikian, model hybrid yang mengintegrasikan Snort dan SVM mampu meningkatkan efektivitas sistem deteksi intrusi dalam menghadapi dinamika dan kompleksitas serangan jaringan modern