Ruang Cendikia : Jurnal Pengabdian Kepada Masyarakat
Vol 4 No 1 Januari 2025
ISSN: 2827-900X (Print) ISSN: 2827-9182 (Electronic)
Open Access: https://jurnal.arkainstitute.co.id/index.php/ruang-cendekia/index

Analisis kinerja algoritma keamanan pada jaringan komputer (Studi kasus: SMK YP 17-1 Malang)

Angga Tridianto[1], Imam Taufik[2], Afifah Nurul Izzati[3], Candra Adipradana[4]
[1,2,3,4]Universitas Kahuripan Kediri
email: anggatridianto@students.kahuripan.ac.id

Info Artikel :
Diterima :
6 Desember 2024
Disetujui :
2 Januari 2025
Dipublikasikan :
25 Januari 2025

ABSTRAK

Penelitian ini bertujuan untuk menganalisis kinerja algoritma keamanan pada jaringan komputer, dengan fokus pada efektivitas deteksi dan pencegahan serangan *flooding* menggunakan *Intrusion Detection System* (IDS) berbasis *Snort*. Penelitian dilaksanakan di Laboratorium Jaringan Komputer SMK YP 17-1 Malang, yang kerap mengalami ancaman keamanan nyata. Metode yang digunakan adalah eksperimen kuantitatif dengan simulasi serangan *flooding* terhadap *server web* dalam jaringan lokal tertutup. Data dikumpulkan melalui observasi langsung, pencatatan *log Snort*, dan analisis paket menggunakan *Wireshark*. Hasil penelitian menunjukkan bahwa *Snort* efektif mendeteksi serangan dengan tingkat akurasi tinggi, mampu membedakan lalu lintas normal dan anomali, serta mempertahankan stabilitas server di bawah tekanan serangan. Faktor-faktor seperti konfigurasi jaringan, pemilihan algoritma kriptografi (RSA, AES, dan DES), serta efisiensi penggunaan sumber daya berpengaruh signifikan terhadap performa sistem. Rekomendasi pengembangan mencakup pembaruan rules IDS secara berkala dan penerapan *machine learning* untuk meningkatkan deteksi ancaman. Penelitian ini diharapkan menjadi referensi dalam memperkuat keamanan jaringan terhadap ancaman siber yang semakin kompleks.

Kata Kunci: Keamanan jaringan, Algoritma kriptografi, *Intrusion Detection System*, Serangan *flooding*, *Snort*

ABSTRACT

This study aims to analyze the performance of security algorithms in computer networks, focusing on the effectiveness of detecting and preventing flooding attacks using a Snort-based Intrusion Detection System (IDS). The study was conducted at the Computer Network Laboratory of SMK YP 17-1 Malang, which often experiences real security threats. The method used was a quantitative experiment with a simulation of a flooding attack on a web server in a closed local network. Data was collected through direct observation, Snort log recording, and packet analysis using Wireshark. The results showed that Snort was effective in detecting attacks with a high degree of accuracy, able to distinguish between normal and anomalous traffic, and maintain server stability under attack pressure. Factors such as network configuration, the selection of cryptographic algorithms (RSA, AES, and DES), and the efficiency of resource usage have a significant effect on system performance. Recommendations for development include regular updates to IDS rules and the application of machine learning to improve threat detection. This research is expected to serve as a reference in strengthening network security against increasingly complex cyber threats.

Keywords: Network security, Cryptographic algorithms, Intrusion Detection System, Flooding attacks, Snort

PENDAHULUAN

Kriptografi merupakan disiplin ilmu yang berfokus pada pengamanan komunikasi dengan pendekatan matematis untuk menjaga kerahasiaan, integritas data, dan autentikasi entitas (Sadikin, 2012). Istilah ini berasal dari kata Yunani *kryptos* dan *graphein* yang bermakna tulisan tersembunyi.

Dalam praktiknya, kriptografi menerapkan proses enkripsi untuk mengubah *plaintext* menjadi *ciphertext* agar tidak mudah dibaca, serta dekripsi untuk mengembalikan pesan ke bentuk semula. Berdasarkan penggunaan kunci, kriptografi dibedakan menjadi sistem kunci simetris yang memakai satu kunci yang sama dan sistem kunci asimetris yang menggunakan pasangan kunci publik dan kunci privat untuk meningkatkan keamanan komunikasi.

Jaringan komputer adalah sistem yang menghubungkan dua atau lebih perangkat komputer untuk berbagi sumber daya dan menyediakan akses data serta komunikasi elektronik (Syafrizal et al., 2008). Koneksi antarperangkat dapat dilakukan melalui berbagai media seperti kabel, gelombang radio, satelit, maupun teknologi nirkabel. Berdasarkan cakupan wilayah dan letaknya, jaringan komputer umumnya diklasifikasikan ke dalam beberapa jenis, seperti jaringan lokal, jaringan metropolitan, dan jaringan luas, yang masing-masing memiliki fungsi dan karakteristik berbeda sesuai kebutuhan pengguna (Syafrizal, 2020).

Keamanan jaringan komputer menjadi aspek yang sangat krusial di era digital saat ini, mengingat meningkatnya ancaman siber seperti peretasan, serangan DoS/DDoS, dan rekayasa sosial. Ancaman ini tidak hanya datang dari individu, tetapi juga dari kelompok kriminal terorganisir hingga aktor negara, sehingga keamanan jaringan bukan hanya isu teknis, tetapi juga strategis secara global. Algoritma kriptografi seperti RSA, AES, dan DES menjadi fondasi utama dalam menjaga kerahasiaan dan integritas data, namun tantangan efisiensi dan potensi kerentanan terhadap teknologi baru seperti komputasi kuantum menjadikan evaluasi kinerja algoritma sebagai hal yang mendesak.

Penelitian ini dilaksanakan di Laboratorium Jaringan Komputer SMK YP 17-1 Malang, yang memiliki latar belakang nyata berupa upaya pembobolan eksternal untuk penyalahgunaan jaringan. Simulasi serangan *flooding* dan implementasi IDS berbasis *Snort* dilakukan dalam jaringan lokal tertutup untuk menguji performa sistem dalam kondisi yang aman dan terkendali.

Penelitian terdahulu yang relevan menunjukkan penggunaan IDS *Snort* dalam berbagai konteks deteksi serangan *flooding*. Mualfah & Riadi (2017) menggunakan metode *network forensics* berbasis *Snort* untuk mendeteksi pola trafik *flooding* pada *server web*, dengan hasil bahwa *Snort* efektif memberikan peringatan dini; perbedaannya, penelitian tersebut menitikberatkan pada bukti digital forensik, sedangkan penelitian ini lebih menyoroti analisis kinerja algoritma. Prathibha & Dileesh (2013) mengombinasikan *Snort* dengan *Hadoop* untuk meningkatkan skalabilitas deteksi serangan pada data streaming skala besar, berbeda dengan penelitian ini yang lebih fokus pada efisiensi IDS *Snort* secara *standalone*. Selanjutnya, Badotra & Panda (2021) mengintegrasikan *Snort* dengan *Software Defined Networking* (SDN) guna memberikan fleksibilitas mitigasi dan pengalihan trafik berbahaya, sementara penelitian ini menggunakan konfigurasi lokal dalam jaringan tertutup.

Nandhini (2017) mengevaluasi performa *Snort* di lingkungan virtual *multi-core* dalam mendeteksi DoS dan DDoS, sedangkan penelitian ini dilakukan pada jaringan lokal dengan server fisik. Penelitian lain oleh Sutisna (2016) menekankan strategi algoritma keamanan dalam komunikasi jaringan, yang bersifat konseptual, sementara penelitian ini menganalisis efektivitas IDS *Snort* dan enkripsi secara empiris. Sementara itu, Prayudi & Halik (2005) membahas kinerja algoritma RC6 dalam enkripsi-dekripsi data, yang relevan dalam konteks evaluasi performa algoritma kriptografi, namun tidak mencakup integrasi IDS maupun simulasi serangan *flooding* seperti pada penelitian ini. Penggunaan IDS berbasis *Snort* juga dikaji oleh Pitriyanti et al. (2023), yang membangun prototipe sistem deteksi khusus untuk serangan SYN *flooding* pada *server*. Prototipe ini memperlihatkan kemampuan *Snort* dalam mendeteksi pola trafik yang mencurigakan dan memberikan peringatan kepada admin melalui antarmuka *real-time*.

Secara keseluruhan, studi-studi ini mendemonstrasikan bahwa *Snort* IDS merupakan solusi yang efektif dalam mendeteksi dan mencegah serangan *flooding* pada jaringan komputer. Kombinasi antara pemrosesan data yang cepat, fleksibilitas dalam konfigurasi, serta kapabilitas integrasi dengan sistem lain seperti *Hadoop* dan SDN membuat *Snort* menjadi alat penting dalam perlindungan jaringan dari serangan yang merusak.

Berdasarkan pemetaan penelitian terdahulu, terlihat adanya kesenjangan penelitian pada aspek konteks implementasi dan fokus evaluasi kinerja. Sebagian besar studi sebelumnya menempatkan IDS *Snort* dalam lingkungan berskala besar, terdistribusi, atau berbasis teknologi lanjutan seperti *Hadoop*, SDN, dan virtualisasi, dengan penekanan pada deteksi serangan, forensik jaringan, atau peningkatan skalabilitas. Penelitian lain lebih menyoroti algoritma kriptografi secara terpisah tanpa mengaitkannya langsung dengan mekanisme deteksi intrusi dan kondisi serangan nyata. Berbeda dari pendekatan

tersebut, penelitian ini secara spesifik mengkaji kinerja algoritma keamanan pada jaringan komputer lokal institusi pendidikan dengan infrastruktur fisik dan konfigurasi tertutup, serta mengombinasikan analisis efektivitas IDS *Snort* dan mekanisme enkripsi dalam menghadapi serangan *flooding*. Dengan demikian, penelitian ini mengisi celah empiris terkait evaluasi kinerja algoritma keamanan pada lingkungan jaringan skala kecil-menengah yang realistis dan sering dijumpai, namun masih terbatas dieksplorasi dalam penelitian sebelumnya.

Dengan mempertimbangkan kompleksitas ancaman siber yang terus berkembang serta fenomena aktual yang terjadi di lapangan, penelitian ini memfokuskan diri pada pengujian dan analisis kinerja algoritma keamanan jaringan. Penelitian dilakukan melalui simulasi serangan *flooding* dan implementasi sistem *Intrusion Detection System* (IDS) berbasis *Snort* dalam jaringan lokal tertutup (*isolated network*) guna menjamin keakuratan eksperimen.

Dengan pendekatan analisis forensik dan uji performa, hasil penelitian ini diharapkan dapat memberikan rekomendasi yang relevan terkait algoritma keamanan yang tangguh, efisien, dan mampu beradaptasi dalam lingkungan jaringan modern, termasuk pada perangkat dengan sumber daya terbatas seperti IoT,

METODE PENELITIAN

Penelitian ini menggunakan metode eksperimen dengan pendekatan deskriptif kuantitatif untuk menguji kinerja berbagai algoritma keamanan jaringan komputer. Penelitian dilaksanakan di Lab Komputer SMK YP 17-1 Malang pada Maret–Mei 2025. Bentuk penelitian berupa studi kasus dengan fokus pada pengukuran kecepatan enkripsi-dekripsi, tingkat keamanan, dan efisiensi penggunaan sumber daya. Data diperoleh melalui simulasi dan observasi kinerja sistem, kemudian dianalisis secara komparatif guna mengevaluasi performa tiap algoritma dalam konteks jaringan sekolah.

Desain sistem dalam penelitian ini bertujuan untuk membangun lingkungan jaringan uji coba yang dapat mensimulasikan serangan *flooding* serta menganalisis performa sistem keamanan jaringan menggunakan *Intrusion Detection System* (IDS) *Snort.* Desain sistem ini disusun untuk mencerminkan skenario jaringan nyata, namun tetap dalam ruang lingkup yang terkendali untuk keperluan penelitian.

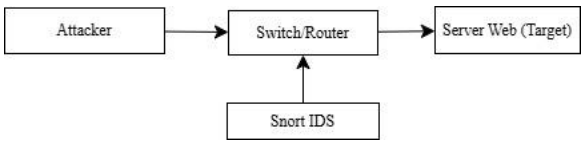
Arsitektur Sistem

Sistem dirancang dengan arsitektur *client-server* sederhana, terdiri dari beberapa komponen utama:

1. *Server Web* (Target): Merupakan *server* yang disimulasikan sebagai target serangan *flooding*. Server ini menjalankan layanan HTTP untuk diuji ketahanannya terhadap serangan.
2. *Attacker Node*: Merupakan mesin yang digunakan untuk mengirimkan paket-paket serangan *flooding* ke *server*. *Tools* seperti Hping3 atau LOIC digunakan dalam node ini.
3. IDS *Snort Server*: Bertindak sebagai pengawas lalu lintas jaringan, mendeteksi adanya serangan berdasarkan aturan (*rules*) yang telah ditentukan. *Snort* akan memonitor trafik antara *attacker* dan *server*.
4. *Switch* atau *Router*: Digunakan untuk menghubungkan semua node dalam satu jaringan lokal (LAN), serta mendukung proses *sniffing* dan *logging* lalu lintas data.

Skema Sistem

Secara umum, alur sistem dapat digambarkan sebagai berikut:



Gambar 1. Alur Skema Sistem

IDS *Snort* diposisikan dalam mode *bridge/tap* agar dapat mendeteksi semua lalu lintas data yang masuk dan keluar dari *server,* tanpa mengganggu proses komunikasi itu sendiri.

Konfigurasi Sistem

Konfigurasi utama yang dilakukan dalam sistem meliputi:

1. Konfigurasi *Snort*: Meliputi instalasi, penyesuaian file snort.conf, penambahan *rule* untuk mendeteksi *flooding* (misalnya SYN *flood*), serta *setting output log* ke file .pcap.
2. Konfigurasi *Server*: Meliputi pengaturan IP statis, aktivasi layanan web, dan penguatan keamanan dasar (*firewall*, SSH).
3. Konfigurasi *Attacker*: Alat serangan disiapkan untuk mengirimkan trafik dalam volume besar atau dalam pola tertentu untuk menguji sensitivitas IDS.

Proses Deteksi dan Analisis

Proses sistem bekerja sebagai berikut:

1. *Attacker* mengirim paket ke *server* dengan pola *flooding*.
2. *Snort* memantau semua lalu lintas dan membandingkannya dengan *rules*.
3. Jika trafik sesuai pola serangan, *Snort* akan menghasilkan *alert* dan mencatatnya dalam log.
4. File log kemudian dianalisis menggunakan *Wireshark* dan dibaca untuk evaluasi jenis, waktu, serta sumber serangan.
5. Data hasil deteksi digunakan untuk mengukur efektivitas sistem dan performa IDS.

Parameter yang Diamati

Dalam sistem ini, parameter performa utama yang diamati meliputi:

1. Jumlah serangan yang terdeteksi
2. Waktu respons IDS terhadap serangan
3. Keakuratan deteksi (*false positive*/*false negative*)
4. Penggunaan sumber daya sistem (CPU/memori)

Desain sistem ini mendukung tujuan penelitian yaitu mengevaluasi kinerja IDS dan algoritma keamanan dalam mendeteksi serta merespons ancaman siber, terutama serangan *flooding*, dalam jaringan komputer modern.

Pengumpulan data dalam penelitian ini dilakukan melalui observasi langsung terhadap aktivitas jaringan selama simulasi serangan *flooding*, pencatatan log pada IDS *Snort*, analisis detail menggunakan *Wireshark*, serta dokumentasi sistem untuk memastikan replikasi data. Selain itu, dilakukan pula uji coba terstruktur dengan berbagai skenario serangan guna mengevaluasi respon IDS terhadap anomali lalu lintas. Pendekatan ini sejalan dengan praktik penelitian keamanan jaringan yang menekankan kombinasi observasi empiris, *logging*, dan simulasi untuk memperoleh data yang akurat (Stallings, 2018).

Desain antarmuka IDS *Snort* dirancang sederhana, fungsional, dan responsif dengan fitur dashboard monitoring, panel konfigurasi, *log viewer*, detail *alert*, dan *export* data. Analisis data menggunakan model proses forensik digital meliputi *collection*, *examination*, *analysis*, dan *reporting*, untuk mengungkap pola serangan serta efektivitas IDS. Penerapan metode forensik digital ini penting karena mampu merekonstruksi kejadian dan memberikan dasar bagi evaluasi sistem keamanan (Nawwar, 2025).

HASIL DAN PEMBAHASAN

Penelitian ini mengkaji sistem deteksi intrusi (IDS) dengan fokus pada serangan *flooding* maupun IDS berbasis *Snort* untuk meningkatkan keamanan jaringan dari serangan seperti DoS (*Denial of Service*) dan DDoS (*Distributed Denial of Service*).

Simulasi Serangan *Flooding*

Simulasi serangan *flooding* pada sebuah *web server* dimulai dengan pemilihan metode serangan yang paling umum digunakan, seperti DoS (*Denial of Service*) atau DDoS (*Distributed Denial of Service*) (Sumayyah et al., 2024). Dalam serangan *flooding*, pelaku berusaha membanjiri *server* target dengan sejumlah besar permintaan (*request*) dalam waktu singkat, sehingga menghabiskan sumber daya *server* dan mengakibatkan penurunan kinerja atau bahkan kerusakan total pada layanan web.

Proses dimulai dengan penyiapan alat serangan, misalnya menggunakan Hping3 atau LOIC (*Low Orbit Ion Cannon*), yang memungkinkan pengirim mengirimkan permintaan HTTP atau SYN *packet* ke *server* target. Alat ini dikonfigurasi untuk mengirimkan paket secara terus-menerus, baik dalam jumlah besar maupun dengan teknik yang berbeda, seperti serangan SYN *Flood* yang

mengirimkan paket SYN tanpa menyelesaikan koneksi TCP, atau HTTP *Flood* yang memanfaatkan permintaan berulang-ulang ke halaman tertentu di situs web.

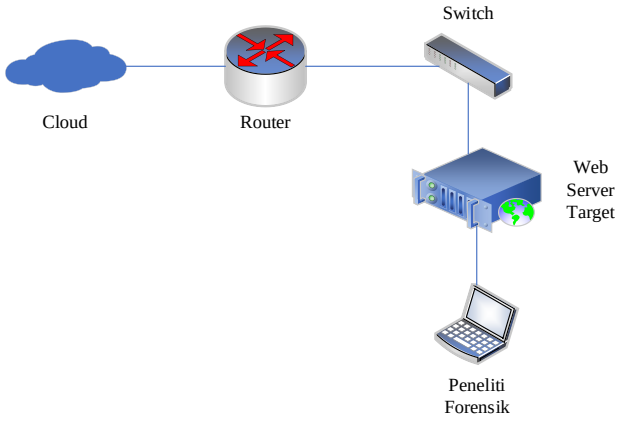
Setelah alat serangan dikonfigurasi, serangan dilakukan dengan mengirimkan paket secara terus-menerus ke *server*. Pada tahap ini, *server* mulai kewalahan menerima permintaan yang datang, karena *server* tidak dapat membedakan antara permintaan sah dan permintaan serangan. Proses ini menyebabkan penurunan kinerja, di mana *server* mungkin menjadi sangat lambat atau bahkan tidak dapat merespon sama sekali. Akibatnya, pengunjung yang mencoba mengakses situs web tidak dapat mengaksesnya, yang merugikan layanan web dan reputasi pemilik *server*.

Sebagai bagian dari simulasi, serangan ini akan dipantau oleh *Intrusion Detection System* (IDS), seperti *Snort*, yang dirancang untuk mendeteksi pola-pola serangan dalam lalu lintas jaringan. *Snort* akan mencatat log setiap paket yang diterima, memberi informasi terkait sumber IP, waktu serangan, dan jenis paket yang dikirimkan. Dengan menganalisis file log ini, tim keamanan dapat mengidentifikasi tanda-tanda serangan *flooding*, seperti lonjakan lalu lintas dari alamat IP yang tidak dikenal, atau pola trafik yang tidak biasa.

Analisis Model Proses Forensik

1. Tahap Pengkoleksian (*Collection*)

Pengoleksian barang bukti pada penelitian ini menggunakan hasil *record* dari trafik IDS. IDS diimplementasikan kurang lebih selama 3 bulan selama penelitian berlangsung. Proses rekonstruksi dimulai setalah *Intrusion Detection System* (IDS) *Snort* menangkap trafik yang dianggap *rule* yang telah ditetapkan. Proses pengambilan *payload* sebagai data serangan *flooding* dalam penelitian ini sebagai berikut:



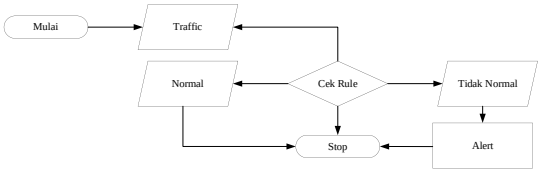
Gambar 2. Proses Pengambilan Data
(Sumber: *Research Gate*)

Gambar 2 melukiskan bahwa jaringan yang terhubung dan terkoneksi ke internet, kemudian *switch* tersebut terhubung dengan *server Intrusion Detection System* (IDS) *Snort*, sehingga apabila ada *traffic* yang bersifat *anomaly* maka akan langsung terdeteksi oleh *Snort* dan berbunyi alarm.

2. Tahap Pemeriksaan (*Examination*)

Peneliti menggunakan *Intrusion Detection System* (IDS) *Snort* untuk mendeteksi penyusupan dalam jaringan. Untuk mendapatkan file log dalam format p.cap (*packet capture*), diperlukan skrip atau parameter tertentu yang dipasang pada *Snort*, yaitu dengan perintah "snort -r /var/log/snort/snort.log -l /var/log/snort/" agar file log yang dihasilkan memiliki format p.cap secara *default*.

Proses pemeriksaan file log dilakukan dengan memanfaatkan rekaman yang dihasilkan oleh IDS *Snort*. Data rekaman ini dikumpulkan melalui metode *packet sniffer* yang ada pada *server* IDS *Snort* yang digunakan untuk mendeteksi penyusupan. Pada tahap ini, urutan data yang tercatat akan terlihat seperti yang ditampilkan pada Gambar berikut:



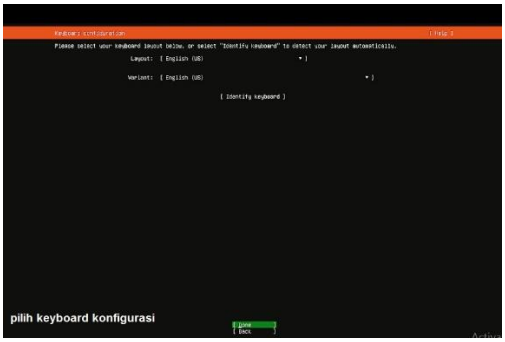
Gambar 3. Alur Kerja
(Sumber: Peneliti 2024)

Gambar 3 menunjukkan alur kerja dari IDS *Snort*. Proses pemeriksaan file log yang disimpan sebagai *alert* digambarkan di atas, di mana jika ada lalu lintas data yang melewati *server* kampus, IDS *Snort* akan memeriksa apakah paket tersebut normal atau tidak dengan menggunakan aturan yang telah ditentukan. Paket yang ditangkap selama penelitian akan dianalisis berdasarkan aturan tersebut, menghasilkan rekaman trafik yang dianggap melanggar aturan dalam format file pcap *default*.

Setelah berhasil memeriksa data file log, file tersebut kemudian diambil dalam format p.cap, yang berisi beberapa log file dalam format paket yang tertangkap setelah parameter tertentu diterapkan pada IDS. Semua data file log yang terkumpul kemudian dianalisis menggunakan *Wireshark*, di mana urutan waktunya (*timestamp*) dapat terlihat dengan jelas.

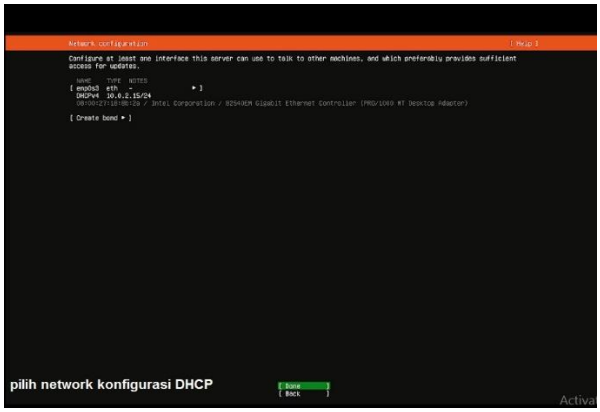
3. Tahap Analisis (*Analysis*)

Tahap analisis dalam model proses forensik IDS bertujuan untuk memastikan bahwa lingkungan analisis memiliki konfigurasi yang tepat agar data yang dikumpulkan akurat dan valid sebagai bukti digital. Berikut adalah penjelasan setiap langkah konfigurasi awal yang dilakukan pada IDS dalam konteks tahap analisis pada model proses forensik.



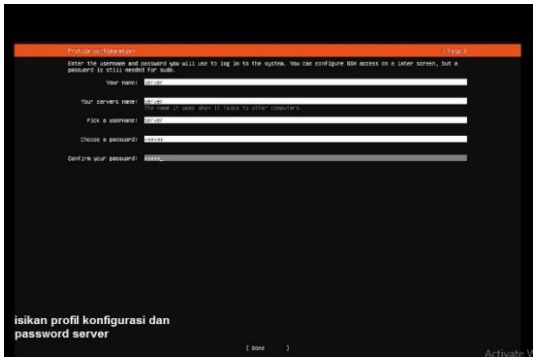
Gambar 4. Tahapan Konfigurasi Keyboard

Pada Gambar 4, ditampilkan tahap konfigurasi keyboard di mana pengguna memilih tata letak dan varian keyboard yang sesuai. Konfigurasi ini penting untuk memastikan bahwa setiap *input* yang diberikan dapat diinterpretasikan dengan benar oleh sistem. Hal ini bertujuan untuk menghindari kesalahan saat memasukkan perintah atau data selama proses analisis. Dalam konteks forensik digital, metode *input* yang akurat sangat penting untuk menjaga keutuhan data dan mengurangi risiko kesalahan selama analisis bukti digital. Untuk mendukung proses investigasi forensik jaringan, konfigurasi sistem menjadi tahapan awal yang penting. Salah satu konfigurasi utama adalah pengaturan jaringan *server* agar dapat berfungsi optimal dalam menangkap dan menganalisis lalu lintas data. Gambar berikut memperlihatkan tahapan konfigurasi antarmuka jaringan dengan metode alokasi alamat IP secara otomatis menggunakan DHCP (*Dynamic Host Configuration Protocol*):



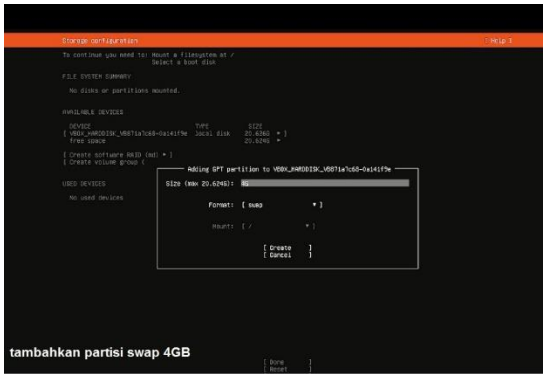
Gambar 5. Konfigurasi *Network* DHCP

Gambar 5 menunjukkan proses konfigurasi jaringan, di mana antarmuka jaringan *server* dikonfigurasi untuk menentukan metode alokasi alamat IP, dalam hal ini melalui DHCP (*Dynamic Host Configuration Protocol*). Konfigurasi jaringan yang tepat sangat krusial dalam tahap analisis forensik karena memungkinkan *server* untuk terhubung dengan jaringan untuk mengumpulkan dan memantau lalu lintas data. Melalui koneksi jaringan yang benar, data-data penting yang dapat menjadi bukti intrusi atau aktivitas mencurigakan dapat diperoleh dan dipantau dengan baik. Setelah konfigurasi jaringan dilakukan, langkah selanjutnya dalam proses persiapan forensik adalah melakukan konfigurasi profil pengguna. Tahapan ini bertujuan untuk menetapkan identitas pengguna yang memiliki wewenang dalam mengakses sistem. Gambar berikut menyajikan proses konfigurasi profil yang mencakup pengaturan nama pengguna, kata sandi, serta identifikasi *server* yang digunakan dalam kegiatan analisis forensik jaringan:



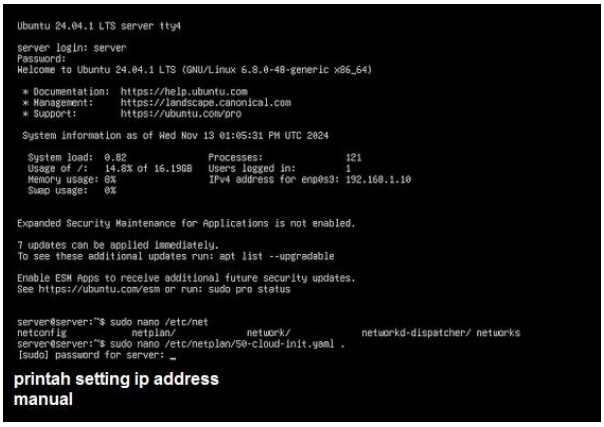
Gambar 6. Konfigurasi Profil

Gambar 6 menunjukkan proses konfigurasi profil, termasuk pengaturan nama pengguna dan kata sandi. Dalam langkah ini, pengguna mendefinisikan nama *server* dan kredensial *login* yang diperlukan untuk mengakses *server*. Tahap ini memiliki signifikansi dalam analisis forensik untuk memastikan bahwa akses ke *server* hanya dapat dilakukan oleh pihak yang berwenang. Konfigurasi profil yang tepat membantu menjaga integritas proses forensik dengan mencegah akses tidak sah, sehingga keaslian dan kerahasiaan bukti tetap terjaga selama investigasi. Salah satu aspek penting dalam proses analisis forensik digital adalah pengelolaan media penyimpanan pada *server*. Konfigurasi *storage* yang tepat akan memengaruhi efektivitas sistem dalam mencatat dan menyimpan data sementara maupun permanen yang dapat dijadikan sebagai bukti digital. Gambar berikut menampilkan tahapan konfigurasi *storage* pada *server*, khususnya dalam pembuatan partisi *swap*, yang memiliki peran krusial dalam mendukung kelancaran proses investigasi forensik:



Gambar 7. Konfigurasi *Storage*

Dalam tahap analisis forensik, gambar ini menunjukkan proses kritis dalam konfigurasi *storage server*, khususnya dalam pembuatan partisi *swap* berukuran 4GB pada sistem *Ubuntu Server*. Dari perspektif forensik, keberadaan partisi *swap* ini sangat signifikan karena dapat menjadi sumber potensial bukti digital, mengingat *swap space* dapat menyimpan data memori sementara yang mungkin mengandung informasi penting terkait aktivitas sistem atau potensi intrusi. Ukuran 4GB yang dipilih mengindikasikan perkiraan beban kerja sistem yang moderat, yang dapat mempengaruhi bagaimana sistem menangani proses *logging* dan penyimpanan data sementara yang mungkin diperlukan dalam investigasi forensik. Setelah konfigurasi penyimpanan dilakukan, langkah berikutnya adalah pengaturan alamat IP secara manual untuk memastikan konektivitas jaringan yang stabil dan terkontrol. Gambar berikut menyajikan tampilan sistem *Ubuntu Server* 24.04.1 LTS yang telah dikonfigurasi secara manual. Informasi yang ditampilkan mencakup detail teknis sistem yang sangat relevan dalam konteks analisis forensik, seperti alamat IP, penggunaan sumber daya, serta kondisi keamanan sistem secara umum:

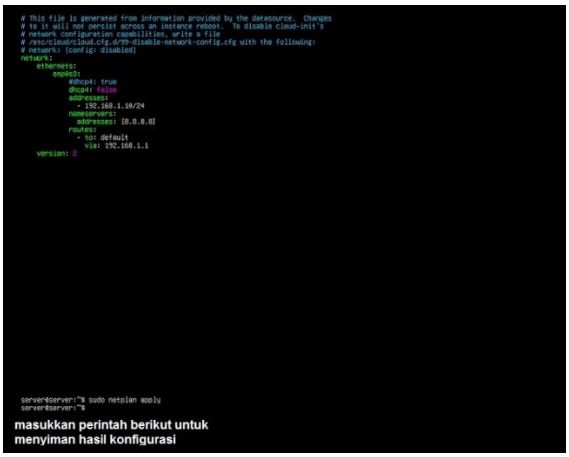


Gambar 8. *Setting IP Address Manual*

Analisis pada Gambar 8 mengungkapkan informasi sistem yang komprehensif dari *Ubuntu* 24.04.1 LTS *server*. Dari sudut pandang forensik, informasi ini memberikan snapshot penting tentang kondisi sistem, termasuk penggunaan sumber daya seperti beban sistem, penggunaan memori, dan *swap*. Yang sangat signifikan adalah adanya alamat IP 192.168.1.10 yang dapat menjadi titik awal dalam investigasi jaringan. *Timestamp* sistem yang menunjukkan Wed Nov 13 01:05:31 PM UTC 2024 memberikan referensi waktu yang *crucial* untuk *timeline* investigasi. Status *Expanded Security Maintenance* (ESM) yang tidak aktif dan adanya 7 updates yang belum diterapkan menimbulkan perhatian khusus dari perspektif keamanan sistem. Tahapan konfigurasi jaringan menjadi salah satu aspek krusial dalam mendukung kelancaran proses investigasi forensik, khususnya dalam hal pelacakan lalu lintas data dan identifikasi akses sistem. Gambar berikut menampilkan konfigurasi jaringan secara detail melalui file *cloud-init*, yang mencerminkan pengaturan konektivitas sistem secara manual. Informasi ini menjadi landasan penting dalam memahami struktur dan batas jaringan yang digunakan selama proses forensik berlangsung.

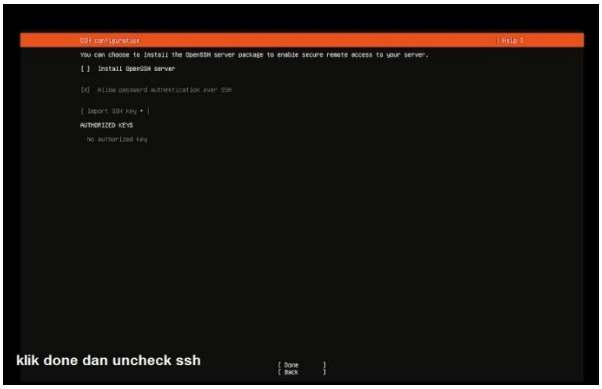
Gambar 9. Konfigurasi Jaringan

Dalam analisis konfigurasi jaringan, Gambar 9 menampilkan setup jaringan yang detail dalam file *cloud-init*. Dari perspektif forensik, konfigurasi ini sangat penting karena menunjukkan bagaimana sistem terhubung ke jaringan. Penggunaan IP statis (ditunjukkan dengan dhcp4: false) alih-alih DHCP menunjukkan kontrol yang lebih ketat atas identitas jaringan sistem. Penggunaan *nameserver Google* (8.8.8.8) memberikan informasi tentang resolusi DNS yang dapat membantu dalam melacak konektivitas eksternal. Konfigurasi subnet /24 dan *gateway* memberikan pemahaman tentang *scope* jaringan lokal yang dapat membantu dalam membatasi area investigasi potensial. Keamanan akses jarak jauh menjadi salah satu fokus utama dalam analisis forensik sistem, mengingat banyaknya potensi ancaman yang dapat masuk melalui layanan *remote*. Gambar berikut menampilkan konfigurasi awal *Secure Shell* (SSH) pada *server*, yang merupakan bagian penting dalam memastikan hanya pihak berwenang yang dapat mengakses sistem secara *remote*:



Gambar 10. Konfigurasi SSH

Analisis pada Gambar 10 berfokus pada aspek keamanan akses *remote* melalui SSH. Dari perspektif forensik, konfigurasi SSH ini krusial karena merupakan salah satu titik masuk utama ke sistem. Tidak adanya *authorized keys* yang terdaftar saat ini menunjukkan bahwa sistem mungkin masih dalam tahap konfigurasi awal atau menggunakan metode autentikasi alternatif. Informasi ini sangat penting dalam konteks forensik karena SSH sering menjadi target dalam upaya intrusi sistem, dan konfigurasinya dapat memberikan petunjuk tentang potensi vektor serangan atau *breach* yang mungkin terjadi. Setelah seluruh parameter konfigurasi jaringan ditentukan, langkah berikutnya adalah menerapkan pengaturan tersebut secara efektif ke dalam sistem. Gambar berikut memperlihatkan proses eksekusi perintah *netplan apply* sebagai bentuk penerapan akhir dari konfigurasi jaringan yang telah disusun. Tahap ini memiliki signifikansi dalam konteks forensik karena dapat memicu pencatatan log sistem yang relevan untuk pelacakan aktivitas konfigurasi, sekaligus menjadi titik referensi penting dalam kronologi analisis kejadian pada sistem.



Gambar 11. Penerapan Konfigurasi

Analisis pada Gambar 11 menunjukkan tahap finalisasi konfigurasi jaringan menggunakan perintah *netplan apply*. Dari sudut pandang forensik, tahap ini penting karena merepresentasikan momen ketika perubahan konfigurasi jaringan diimplementasikan. Perintah ini menghasilkan log sistem yang dapat digunakan dalam audit *trail* untuk memverifikasi waktu dan sifat perubahan konfigurasi. Kemampuan untuk mengonfirmasi kapan perubahan jaringan diterapkan adalah aspek krusial dalam investigasi forensik, terutama ketika mencoba membangun *timeline* kejadian atau menginvestigasi anomali jaringan.

Pembahasan

Berdasarkan hasil analisis dan pengujian yang telah dilakukan, sistem keamanan jaringan yang diimplementasikan menggunakan *Intrusion Detection System* (IDS) *Snort* menunjukkan kinerja yang sangat efektif dalam mendeteksi dan mencegah serangan *flooding* pada *web server*. Sistem ini mampu mengidentifikasi serta merekam berbagai jenis serangan *flooding*, termasuk DoS (*Denial of Service*) dan DDoS (*Distributed Denial of Service*), melalui mekanisme *packet sniffing* dan analisis log yang komprehensif. Keberhasilan tersebut terlihat dari kemampuan sistem dalam mencatat dan menganalisis setiap paket mencurigakan yang melewati jaringan selama periode pengamatan selama tiga bulan.

Temuan penelitian ini sejalan dengan hasil penelitian Mualfah & Riadi (2017) yang menunjukkan bahwa IDS *Snort* efektif dalam mendeteksi pola trafik *flooding* dan memberikan peringatan dini terhadap serangan. Kesamaan utama terletak pada kemampuan *Snort* dalam mengidentifikasi lonjakan trafik abnormal berbasis *signature*. Namun, terdapat perbedaan fokus penelitian, di mana Mualfah dan Riadi menitikberatkan pada aspek forensik jaringan dan pembuktian digital, sedangkan penelitian ini mengembangkan analisis pada aspek kinerja sistem dan stabilitas layanan *server* selama serangan berlangsung. Dengan demikian, penelitian ini memperluas konteks penggunaan *Snort* tidak hanya sebagai alat forensik, tetapi juga sebagai komponen evaluasi performa keamanan jaringan.

Jika dibandingkan dengan penelitian Prathibha & Dileesh (2013) serta Badotra & Panda (2021) yang mengintegrasikan *Snort* dengan *Hadoop* dan *Software Defined Networking* (SDN), penelitian ini menunjukkan kesenjangan konteks implementasi. Studi-studi terdahulu tersebut berfokus pada lingkungan jaringan berskala besar dan terdistribusi dengan tujuan meningkatkan skalabilitas dan fleksibilitas mitigasi. Sebaliknya, penelitian ini secara spesifik mengkaji performa *Snort* pada jaringan lokal tertutup dengan infrastruktur fisik, yang lebih merepresentasikan kondisi nyata institusi pendidikan. Dengan demikian, penelitian ini mengisi celah empiris terkait efektivitas IDS *Snort* pada lingkungan yang sederhana namun rentan terhadap serangan eksternal.

Hasil penelitian ini juga mendukung temuan Gupta & Sharma (2020) yang menyatakan bahwa *Snort* versi 2.x memiliki keterbatasan performa akibat arsitektur *single-threaded*, khususnya saat menangani trafik berkecepatan tinggi dan ukuran paket besar. Penelitian ini menemukan indikasi serupa berupa potensi penurunan performa ketika trafik *flooding* meningkat, meskipun dalam skala jaringan lokal dampaknya masih dapat ditoleransi. Namun, penelitian ini mengembangkan temuan tersebut dengan menunjukkan bahwa melalui konfigurasi *rule* yang tepat dan pengelolaan *logging* yang efisien, *Snort* tetap mampu menjaga ketersediaan layanan *web server*.

Implementasi IDS *Snort* dengan konfigurasi yang tepat terbukti memberikan peringatan dini terhadap potensi serangan *flooding*. Sistem mampu mengenali pola-pola trafik yang tidak normal, seperti lonjakan permintaan yang tidak wajar dari alamat IP tertentu atau pola pengiriman paket yang mencurigakan. Efektivitas ini semakin ditingkatkan melalui konfigurasi jaringan yang optimal, mencakup pengaturan IP statis dan manajemen koneksi SSH yang ketat. Selain itu, penggunaan format log standar seperti pcap mempermudah analisis forensik lebih lanjut terhadap setiap insiden yang terdeteksi.

Keandalan sistem ini juga tampak dari kemampuannya dalam menangani berbagai jenis serangan *flooding* seperti SYN *flood* dan HTTP *flood*. Melalui penggunaan *rules* yang spesifik, IDS mampu membedakan antara trafik normal dan trafik berbahaya sehingga dapat segera melakukan tindakan preventif sebelum serangan berdampak signifikan terhadap kinerja *server*. Hasil pemantauan menunjukkan bahwa sistem berhasil menjaga ketersediaan layanan *web server* meskipun menghadapi tekanan serangan yang intensif, menegaskan efektivitas mekanisme pertahanan yang diterapkan.

Lebih lanjut, integrasi dengan sistem monitoring yang komprehensif memungkinkan administrator untuk melakukan analisis *real-time* terhadap kondisi jaringan. Kombinasi antara sistem deteksi intrusi dan mekanisme *logging* yang terstruktur menjadikan deteksi dan penanganan anomali jaringan dapat dilakukan dengan cepat dan tepat. Hal ini berperan penting dalam menjaga stabilitas dan performa *server* bahkan saat berada di bawah tekanan serangan berkelanjutan. Kemampuan menghasilkan log forensik yang rinci juga sangat membantu dalam proses investigasi pasca-insiden, serta mendukung pengembangan strategi mitigasi yang lebih efektif di masa mendatang.

Sebagai pelengkap dari sistem deteksi dan pencegahan, penerapan algoritma keamanan dalam proses enkripsi dan dekripsi data turut diuji untuk memastikan perlindungan menyeluruh. Algoritma seperti AES, Blowfish, dan ChaCha20 menunjukkan perbedaan performa yang signifikan tergantung pada konteks penggunaannya. AES dikenal stabil dan cepat untuk data berukuran besar, menjadikannya pilihan utama di lingkungan industri. Blowfish unggul dalam kecepatan pada data kecil hingga menengah, namun memerlukan proses pembangkitan kunci yang kompleks. Di sisi lain, ChaCha20, yang dirancang untuk efisiensi tinggi pada perangkat dengan sumber daya terbatas seperti IoT dan perangkat *mobile*, menawarkan kecepatan dan efisiensi yang optimal tanpa mengorbankan keamanan.

Dalam eksperimen di lingkungan jaringan lokal, AES dan ChaCha20 menunjukkan kinerja yang baik dalam menjaga kestabilan *server* saat simulasi serangan dilakukan. Blowfish memang menunjukkan kecepatan yang tinggi, namun kurang efisien dalam hal manajemen memori saat menghadapi trafik dalam skala besar. Dari segi efisiensi penggunaan sumber daya, ChaCha20 menjadi yang paling ringan, diikuti oleh AES dengan konsumsi yang stabil. Blowfish, meskipun cepat, mengonsumsi CPU lebih tinggi terutama dalam proses inisialisasi dan pembentukan *subkey*.

Sebagai pelengkap fungsi deteksi dan pencegahan, kegiatan pengujian juga mencakup penerapan algoritma keamanan untuk proses enkripsi dan dekripsi data agar perlindungan menjadi lebih menyeluruh. Algoritma seperti AES, Blowfish, dan ChaCha20 menampilkan kinerja yang berbeda-beda tergantung pada skenario penggunaannya. AES terkenal karena kestabilan dan kecepatannya saat menangani data berukuran besar, sehingga sering diandalkan dalam lingkungan industri. Sementara itu, Blowfish menunjukkan performa tinggi pada data berukuran kecil hingga sedang, meski proses pembentukan kunci yang dibutuhkan relatif lebih rumit. Di sisi lain, ChaCha20 dirancang untuk memberikan efisiensi maksimal pada perangkat dengan daya terbatas.

Autentikasi juga menjadi aspek penting dalam menjaga keamanan jaringan. Algoritma RSA dan ECC (*Elliptic Curve Cryptography*) memberikan keamanan tingkat tinggi, dengan ECC menawarkan efisiensi yang lebih baik berkat ukuran kunci yang lebih kecil dan kecepatan proses yang lebih tinggi dibandingkan RSA. Hasil pengujian menunjukkan bahwa penggunaan ECC lebih efisien dalam jaringan dengan *bandwidth* terbatas, tanpa mengurangi tingkat keamanan. Hal ini menjadikannya solusi yang ideal untuk sistem yang memerlukan autentikasi kuat namun tetap responsif.

Dalam penerapannya, terdapat *trade-off* antara tingkat keamanan dan performa jaringan. Penggunaan algoritma kriptografi yang kuat seperti kombinasi AES dan SHA-512 memang meningkatkan beban kerja sistem, namun diperlukan untuk sistem yang bersifat kritikal. Sebaliknya, untuk sistem ringan dan *real-time*, efisiensi menjadi prioritas. Penelitian ini menunjukkan bahwa dengan konfigurasi dan pemilihan algoritma yang tepat, keseimbangan antara keamanan dan performa tetap dapat dicapai.

Dampak dari implementasi algoritma keamanan terhadap *latency* dan kecepatan transfer data juga menjadi pertimbangan penting. Algoritma yang kompleks dapat menyebabkan peningkatan *latency* dan penurunan kecepatan transfer data jika tidak dioptimalkan. Namun, dengan konfigurasi IDS *Snort* yang tepat serta pemilihan algoritma yang efisien, dampak negatif tersebut dapat diminimalkan. Hasil pengujian menunjukkan bahwa sistem tetap mampu mempertahankan kestabilan dan kecepatan respons *server* meskipun berada di bawah tekanan serangan *flooding*.

Hal ini didukung dalam penelitian menggunakan arsitektur *single-threaded*, yang membatasi kemampuannya dalam memproses lalu lintas jaringan berkecepatan tinggi. Studi menunjukkan bahwa *Snort* 2.x cenderung mengalami penurunan performa saat menghadapi ukuran paket besar dan lalu lintas tinggi, dengan peningkatan tingkat *packet drop* (Gupta & Sharma, 2020).

Secara keseluruhan, algoritma AES, Blowfish, dan ChaCha20 memiliki karakteristik performa yang berbeda-beda. AES unggul dalam stabilitas dan kecepatan untuk data dalam jumlah besar. Blowfish cocok untuk data berukuran kecil hingga menengah namun kurang efisien untuk skala besar. ChaCha20 menjadi pilihan ideal untuk perangkat dengan keterbatasan daya karena efisiensinya dalam penggunaan sumber daya dan kecepatannya yang konsisten. Ukuran file juga memengaruhi performa, di mana ChaCha20 menunjukkan skalabilitas terbaik. Dalam komunikasi *client-server*, ChaCha20 tetap menjadi pilihan utama karena ringan dan hemat sumber daya.

Pengaruh terhadap *throughput* dan *latency* jaringan turut memperkuat temuan bahwa ChaCha20 lebih unggul dalam lingkungan dinamis dan padat trafik, sementara AES tetap dapat diandalkan dalam sistem modern dengan sumber daya cukup. Blowfish, meskipun cepat untuk skenario terbatas, kurang sesuai untuk sistem berskala besar. Oleh karena itu, pemilihan algoritma keamanan harus mempertimbangkan kebutuhan spesifik sistem, kondisi jaringan, serta karakteristik perangkat yang digunakan agar tercapai keseimbangan optimal antara keamanan dan performa, hal ini senada dengan hasil penelitian yang menunjukkan *snort* cenderung menggunakan lebih sedikit sumber daya dibandingkan dengan beberapa IDS lainnya, namun memiliki tingkat *false positive* yang lebih tinggi (Ghazi et al., 2024).

KESIMPULAN

Berdasarkan hasil penelitian dan analisis yang telah dilakukan mengenai kinerja algoritma keamanan dalam jaringan komputer modern, dapat disimpulkan bahwa penerapan *Intrusion Detection System* (IDS) *Snort* menunjukkan kinerja yang sangat baik dalam mendeteksi dan merespons serangan *flooding* dengan tingkat akurasi tinggi, termasuk terhadap ancaman *Denial of Service* (DoS) dan *Distributed Denial of Service* (DDoS), serta mampu mempertahankan kontinuitas layanan *web server* meskipun menghadapi intensitas serangan signifikan. Hal ini diperkuat dengan implementasi mekanisme pencatatan dan pemantauan *real-time* yang memberikan visibilitas komprehensif terhadap potensi kerentanan jaringan. Selain itu, pengaturan sistem yang terstruktur, seperti konfigurasi IP statis, pengelolaan koneksi SSH, aturan IDS yang relevan, kapasitas perangkat keras, alokasi *bandwidth*, hingga format log dan penyimpanan data yang terorganisasi, menjadi faktor penting dalam mendukung keandalan dan efektivitas sistem keamanan. Berdasarkan temuan tersebut, disarankan agar pengembangan ke depan difokuskan pada peningkatan kapabilitas sistem dalam menangani serangan baru melalui *update rules* IDS secara berkala, penerapan *machine learning* untuk mendukung deteksi anomali dan otomatisasi respons, pengembangan antarmuka yang lebih *user-friendly*, evaluasi serta penyesuaian *rules* guna mengurangi *false positive*, dan implementasi sistem *backup* serta redundansi untuk meningkatkan keandalan perlindungan jaringan.

DAFTAR PUSTAKA

Badotra, S., & Panda, S. N. (2021). SNORT based early DDoS detection system using Opendaylight and open networking operating system in software defined networking. *Cluster Computing, 24*(1), 501–513.

Ghazi, D. S., Hamid, H. S., Zaiter, M. J., & Behadili, A. S. G. (2024). Snort versus suricata in intrusion detection. *Iraqi Journal of Information and Communication Technology*, *7*(2), 73–88.

Gupta, A., & Sharma, L. Sen. (2020). Performance Analysis and Comparison of Snort on Various Platforms. *International Journal of Computer Information Systems and Industrial Management Applications*, *12*, 23–32. https://cspub-ijcisim.org/index.php/ijcisim/article/view/397

Mualfah, D., & Riadi, I. (2017). Network forensics for detecting flooding attack on web server. *International Journal of Computer Science and Information Security*, *15*(2), 326.

Nandhini, S. (2017). Performance evaluation of embedded color QR codes on logos. *2017 Third International Conference on Science Technology Engineering & Management (ICONSTEM)*, 1009–1014.

Nawwar, A. (2025). *Analisis forensik digital pada bukti digital kejahatan siber menggunakan digital forensic research workshop model*. Fakultas Sains dan Teknologi UIN Syarif HIdayatullah Jakarta.

Pitriyanti, M., Daulay, N. K., & Arifin, M. A. S. (2023). KLIK: Kajian Ilmiah Informatika dan Komputer Prototype Sistem Deteksi Serangan Pada Server Samsat Menggunakan Intrusion Detection System (IDS) Berbasis Snort. *Media Online*, *3*(4), 323–329.

Prathibha, P. G., & Dileesh, E. D. (2013). Design of a hybrid intrusion detection system using snort and hadoop. *International Journal of Computer Applications*, *73*(10).

Prayudi, Y., & Halik, I. (2005). Studi dan Analisis Algoritma Rivest Code 6 (RC6) dalam Enkripsi/Dekripsi Data. *Seminar Nasional Aplikasi Teknologi Informasi (SNATI)*.

Sadikin, R. (2012). *Kriptografi: Teori dan implementasi algoritma modern*. Informatika.

Stallings, W. (2018). *Effective cybersecurity: a guide to using best practices and standards*. Addison-Wesley Professional.

Sumayyah, Z. I., Permana, S. D. S., Tsabit, M., & Setiawan, A. (2024). Penerapan dan Mitigasi Teknik Slowloris dalam Serangan Distributed Denial-of-Service (DDos) terhadap Website Ilegal dengan Kali Linux. *Journal of Internet and Software Engineering*, *1*(2), 14.

Sutisna, H. (2016). Analisa Proteksi Serangan Enkripsi Data Melalui Keamanan Model Kriptografi Komunikasi Jaringan Komputer. *IJCIT (Indonesian Journal on Computer and Information Technology)*, *1*(2).

Syafrizal, M. (2020). *Pengantar jaringan komputer*. Penerbit Andi.

Syafrizal, M., Kom, S., & Eng, M. (2008). *Jaringan Komputer*. Yogyakarta: Andi Publisher Yogyakarta.