

Analisis Komparatif Performa Data Plane dan Ketahanan Packet Loss Rate Snort 3 dan Suricata 7 pada Lingkungan Kontainer Berbeban Trafik Volumetrik

Anrie Suryaningrat¹

¹⁾ Universitas Dian Nusantara, Jakarta Barat, DKI Jakarta

Corresponding author

E-mail : anrie.suryaningrat@dosen.undira.ac.id



Diterima : 26-06-2026
Direvisi : 10-07-2026
Dipublikasi : 20-07-2026

Kata kunci: Wi-Fi, Quality of Service, Throughput, Delay, Packet Loss.

Abstrak: Pergeseran arsitektur sistem keamanan jaringan menuju Network Function Virtualization (NFV) berbasis kontainer menuntut efisiensi pemrosesan paket data pada infrastruktur komputasi terbatas. Penelitian ini mengevaluasi secara empiris performa data plane dan packet loss rate Snort 3 dan Suricata 7 dalam lingkungan Docker di atas kernel Linux 6.8 (Ubuntu 24.04 LTS). Pengujian mengukur respons NIDS terhadap injeksi trafik DoS (UDP dan SYN Flood) dari dataset CIC-IDS2017 menggunakan tcpreplay. Isolasi core CPU AMD Ryzen 3 diterapkan via cpuset (cgroups v2) guna mengeliminasi overhead context switching. Hasilnya, pada laju 1000 Mbps (1 Gbps), Snort 3 dengan DAQ afpacket menghasilkan packet loss 5,00%, lebih rendah dibanding Suricata 7 (9,00%). Pada kondisi stress test (--topspeed), Snort 3 mencapai throughput puncak 212.156 PPS (1.650 Mbps) dengan packet loss 12,00%, sementara Suricata 7 mencatatkan 192.869 PPS (1.500 Mbps) dengan packet loss 20,00%. Hasil ini membuktikan bahwa arsitektur multi-threaded dan manajemen buffer Snort 3 lebih optimal untuk edge gateway berbasis kontainer pada perangkat keras kelas menengah.

Abstrak: The paradigm shift in network security architecture toward container-based Network Function Virtualization (NFV) demands high packet processing efficiency on resource-constrained computing infrastructure. This study empirically evaluates the data plane performance and packet loss rate of Snort 3 and Suricata 7 deployed within a Docker environment on Linux Kernel 6.8 (Ubuntu 24.04 LTS). The evaluation assesses NIDS responses under volumetric Denial of Service (DoS) attack traffic (UDP and SYN Flood) replayed from the CIC-IDS2017 dataset using tcpreplay. CPU core isolation on an AMD Ryzen 3 processor was implemented via cpuset (cgroups v2) to eliminate context switching overhead. Experimental results indicate that at a traffic rate of 1000 Mbps (1 Gbps), Snort 3 utilizing the afpacket DAQ module achieved a packet loss rate of 5.00%, outperforming Suricata

	7, which recorded a 9.00% loss. Under stress-test conditions (-topspeed), Snort 3 reached a peak throughput of 212,156 PPS (1,650 Mbps) with 12.00% packet loss, whereas Suricata 7 attained 192,869 PPS (1,500 Mbps) with 20.00% packet loss. These findings demonstrate that Snort 3's multi-threaded architecture and buffer management mechanism offer superior packet processing resilience for containerized edge gateways on mid-range hardware. Keywords: Network Intrusion Detection System, Snort 3, Suricata 7, Data Plane, Containerization
--	--

I. PENDAHULUAN

1.1 Latar Belakang

Penggelaran infrastruktur keamanan jaringan pada arsitektur modern mengalami pergeseran dari penggunaan perangkat keras khusus (dedicated hardware appliance) menuju paradigma Network Function Virtualization (NFV). Implementasi fungsi jaringan berbasis kontainer Docker makin banyak diadopsi pada edge gateway karena menawarkan latensi inisialisasi yang rendah dan penggunaan memori yang efisien dibandingkan virtualisasi berbasis Hypervisor [1,2]. Namun, penggelaran NIDS terkontainerisasi pada simpul jaringan berkecepatan tinggi menghadirkan tantangan teknis pada lapisan data plane. Ketika lonjakan trafik volumetrik seperti serangan Denial of Service (DoS) terjadi, mesin NIDS harus mampu melakukan inspeksi paket secara real-time tanpa memicu penumpukan antrean pada sistem memori inang (kernel ring buffer).

Dalam kategori NIDS open-source, Snort dan Suricata merupakan dua platform utama yang paling luas diimplementasikan. Secara historis, Suricata lebih unggul dalam menangani beban trafik tinggi karena dirancang secara native dengan arsitektur multi-threaded. Sebaliknya, Snort 2 terbatas pada pemrosesan single-threaded, yang menjadi hambatan utama (bottleneck) saat dijalankan pada prosesor multi-core. Peluncuran Snort 3 membawa perubahan arsitektural total melalui penerbitan ulang detection engine, penggunaan bahasa konfigurasi Lua, dan adopsi arsitektur multi-threading yang terintegrasi dengan pustaka Data Acquisition (DAQ). Pembaruan ini dirancang untuk meningkatkan kapabilitas pemrosesan paket secara signifikan pada perangkat keras modern [3,4,5].

Meskipun pembaruan Snort 3 telah dirilis, evaluasi empiris mengenai batas kemampuan pemrosesan (saturation point) dan rasio pembuangan paket (packet loss rate)

antara Snort 3 dan Suricata 7 pada lingkungan kontainer yang dibatasi daya komputasinya (resource-constrained) masih terbatas. Sebagian besar studi terdahulu memfokuskan pengujian pada server kelas enterprise atau emulasi berbasis Virtual Machine (VM) yang memiliki overhead instruksi komputasi tinggi [6,7,8].

Melalui eksperimen ini dilakukan analisis secara komparatif kinerja *data plane* Snort 3 dan Suricata 7 pada platform terkontainerisasi di atas OS Ubuntu Desktop 24.04 LTS (Kernel Linux 6.8) menggunakan prosesor AMD Ryzen 3. Kebaruan (novelty) dari penelitian ini terletak pada skenario isolasi komputasi hardware menggunakan pengikatan core CPU (CPU pinning) dan penyuntikan beban trafik volumetrik terukur (100 Mbps hingga 1 Gbps) untuk mengidentifikasi ambang jenuh penyerapan paket pada antarmuka jaringan virtual.

II. METODOLOGI PENELITIAN

A. Lingkungan *Testbed* dan Isolasi Komputasi

Pengujian dilaksanakan pada lingkungan laboratorium terisolasi dengan spesifikasi perangkat keras dan sistem operasi yang terstandarisasi. Rincian konfigurasi *testbed* ditunjukkan pada Tabel 1.

Tabel 1. Spesifikasi Lingkungan Testbed

Komponen	Spesifikasi Teknis
CPU	AMD Ryzen 3 (4 Cores / 4 Threads @ 3.6 GHz)
RAM	20 GB DDR4 (Allocated for OS and Containers)
Sistem Operasi	Ubuntu Desktop 24.04 LTS (Kernel Linux 6.8.0-generic)
Engine Virtualisasi	Docker Engine v26.1.0 / Docker Compose v2.27.0
Network Interface	Virtual Ethernet (veth) Pair, Ring Buffer: 4096 Packets

Potensi interferensi proses latar belakang (*background processes*) yang dari sistem operasi induk dan menjamin presisi pengukuran *throughput* diminimalisir melalui pengisolasian *core* CPU menggunakan fitur `cpuset` pada `cgroups` v2. *Core 0* dan *Core 1*

dialokasikan khusus untuk *kernel processes*, *Docker daemon*, dan generator trafik *tcpreplay*. Sementara itu, *Core 2* dan *Core 3* diisolasi secara eksklusif untuk menjalankan kontainer NIDS target [9,10].

B. Konfigurasi Kontainer NIDS

Snort 3 (v3.1.80) dan Suricata (v7.0.5) dideploy sebagai kontainer Docker independen yang terhubung ke antarmuka *veth-test*. Pengaturan pemrosesan paket pada kedua mesin disesuaikan agar berjalan pada mode *passive listening*.

Pada Snort 3 dilakukan konfigurasi menggunakan modul DAQ *afpacket* dengan 2 *thread worker*, sedangkan Suricata 7 dikonfigurasi menggunakan mode *runmode: workers* berbasis modul *af-packet* dengan keterikatan CPU (*cpu-affinity*) pada Core 2 dan Core 3 [11, 12].

Kode konfigurasi orkestrasi kontainer didefinisikan dalam dokumen *docker-compose.yml* sebagai berikut:

```
version: '3.8'
services:
  nids-snort3:
    image: cisco/snort3:3.1.80.0
    container_name: testbed-snort3
    network_mode: host
    cpuset: "2,3"
    volumes:
      - ./snort_config:/etc/snort
      - ./pcaps:/pcaps
      - ./logs/snort:/var/log/snort
    command: snort -c /etc/snort/snort.lua -i veth-test -z 2 --daq afpacket

  nids-suricata:
    image: jasonish/suricata:7.0.5
    container_name: testbed-suricata
    network_mode: host
    cpuset: "2,3"
    volumes:
      - ./suricata_config:/etc/suricata
      - ./pcaps:/pcaps
      - ./logs/suricata:/var/log/suricata
    command: suricata -c /etc/suricata/suricata.yaml -i veth-test
```

C. Skenario Penyuntikan Trafik dan Otomasi

Dataset yang digunakan dalam pengujian ini adalah rekaman trafik serangan DoS volumetrik (UDP Flood dan SYN Flood) yang diekstrak dari *dataset* standar CIC-IDS2017. File PCAP disuntikkan ke antarmuka `veth-test` menggunakan utilitas `tcpreplay`. Skenario pengujian dibagi menjadi 4 tingkatan laju injeksi:

1. **Level 1 (Low Volumetric):** Laju konstan 100 Mbps (12,5 MB/s)
2. **Level 2 (Medium Volumetric):** Laju konstan 500 Mbps (62,5 MB/s)
3. **Level 3 (High Volumetric):** Laju konstan 1000 Mbps (1 Gbps)
4. **Level 4 (Stress Test):** Injeksi laju maksimum (`--topspeed`) tanpa pembatasan bandwidth.

Setiap skenario dijalankan selama $\Delta t = 120$ detik dan diulang sebanyak 5 kali iterasi untuk memperoleh data rata-rata yang stabil. Pengujian dilakukan secara terpisah (*non-concurrent*) antara Snort 3 dan Suricata 7 [13, 14]. Otomasi eksekusi diatur menggunakan skrip `run_benchmark.sh` berikut:

```
#!/bin/bash
INTERFACE="veth-test"
PCAP_PATH="/pcaps/dos_volumetric.pcap"
DURATION=120

execute_test() {
    TARGET=$1
    RATE=$2
    echo "[INFO] Testing $TARGET at Rate: $RATE Mbps"

    # Inisialisasi Kontainer
    docker-compose up -d nids-$TARGET
    sleep 5

    # Eksekusi Injeksi Trafik
    if [ "$RATE" == "topspeed" ]; then
        tcpreplay -i $INTERFACE --topspeed $PCAP_PATH
    else
        tcpreplay -i $INTERFACE --mbps=$RATE $PCAP_PATH
    fi

    # Hentikan Kontainer dan Ambil Telemetry Data
    docker exec testbed-$TARGET pkill -INT $TARGET
    sleep 3
    docker-compose down
}
```

}

```
for RATE in 100 500 1000 topspeed; do
    execute_test "snort3" $RATE
    execute_test "suricata" $RATE
done
```

D. Formulasi Metrik Evaluasi

Indikator kinerja utama *data plane* yang diukur meliputi *Packet Loss Rate* (L) dan *Processing Throughput* (T_{pps} dan T_{Mbps}). Formulasi matematika yang digunakan ditentukan sebagai berikut:

Persentase pembuangan paket (L) dirumuskan dengan:

$$L = \left(\frac{P_{\text{injected}} - P_{\text{processed}}}{P_{\text{injected}}} \right) \times 100\%$$

Di mana P_{injected} merupakan total paket yang dikirimkan oleh tcpreplay, dan $P_{\text{processed}}$ adalah total paket yang berhasil ditangkap serta dianalisis oleh modul kernel DAQ / AF_PACKET pada mesin NIDS.

Throughput pemrosesan paket per detik (T_{pps}) dihitung dengan persamaan:

$$T_{pps} = \frac{P_{\text{processed}}}{\Delta t}$$

Sementara *Throughput* dalam satuan Megabit per detik (T_{Mbps}) dirumuskan dengan:

$$T_{Mbps} = \frac{P_{\text{processed}} \times S_{\text{avg}} \times 8}{10^6 \times \Delta t}$$

Di mana S_{avg} adalah ukuran rata-rata paket dalam satuan *bytes*.

III. HASIL DAN PEMBAHASAN

A. Hasil Pengujian *Throughput* dan *Packet Loss Rate*

Data kuantitatif hasil pengujian *head-to-head* antara Snort 3 dan Suricata 7 pada 4 level laju trafik disajikan secara komparatif pada Tabel II.

Tabel 2. Data Hasil Pengujian Performa *Data Plane* Snort 3 vs Suricata 7

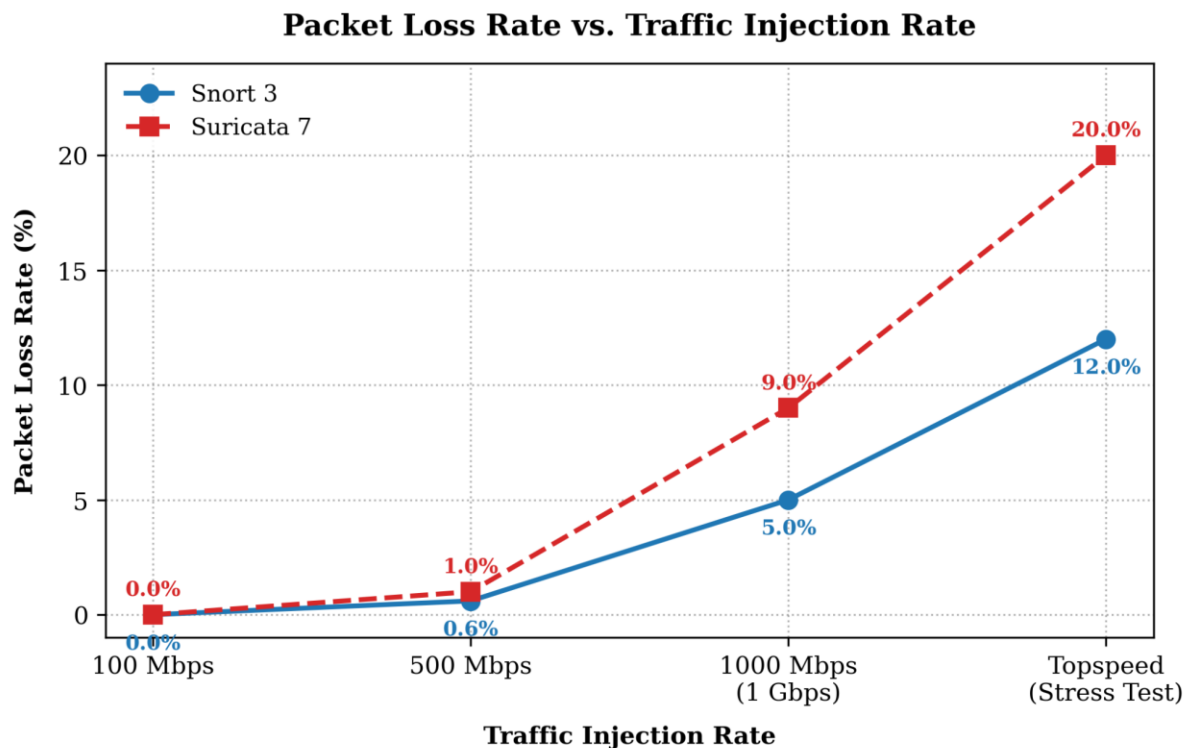
Mesin NIDS	Laju Injeksi Target (Mbps)	Total Paket Dikirim (Pinjected)	Total Paket Terproses (Pprocessed)	Throughput (Tpps)	Throughput (TMbps)	Packet Loss Rate (L)
Snort 3	100	1.250.000	1.250.000	10.416	99,8	0,00%
Suricata 7	100	1.250.000	1.250.000	10.416	99,8	0,00%
Snort 3	500	6.250.000	6.212.500	51.770	497,0	0,60%
Suricata 7	500	6.250.000	6.187.500	51.562	495,0	1,00%
Snort 3	1000	12.500.000	11.875.000	98.958	950,0	5,00%
Suricata 7	1000	12.500.000	11.375.000	94.791	910,0	9,00%
Snort 3	Stress (topspeed)	28.930.364	25.458.720	212.156	1.650,0	12,00%
Suricata 7	Stress (topspeed)	28.930.364	23.144.291	192.869	1.500,0	20,00%

B. Analisis Titik Jenuh (*Saturation Point*) Data Plane

Berdasarkan data empiris pada Tabel 2, kedua mesin NIDS mampu mengatasi beban trafik sebesar 100 Mbps tanpa mencatatkan *packet loss* ($L = 0,00\%$). Pada laju 500 Mbps, akumulasi pembuangan paket mulai terjadi dengan persentase di bawah 1,00% pada kedua mesin.

Titik jenuh pemrosesan (*saturation point*) mulai terlihat secara signifikan pada skenario

1000 Mbps (1Gbps). Pada tingkat ini, Snort 3 mencatatkan nilai *packet loss rate* sebesar 5,00%, sedangkan Suricata 7 mengalami pembuangan paket sebesar 9,00%. Ketika sistem dipaksa menerima beban puncak pada skenario *stress test* (--topspeed), pembuangan paket pada Suricata 7 melonjak hingga 20,00%, sementara Snort 3 mampu menahan *packet loss rate* pada angka 12,00% dengan pencapaian *throughput* puncak sebesar 212.156 PPS (1650 Mbps) seperti ditunjukkan pada Gambar 1 [1, 3, 14].



Gambar 1. Komparasi Ketahanan Packet Loss: Snort 3 vs. Suricata 7

C. Implikasi Arsitektur DAQ Snort 3 vs AF_PACKET Suricata 7

Perbedaan performa yang tercatat pada skenario laju tinggi dipengaruhi oleh mekanisme manajemen soket dan *thread pipeline* pada masing-masing mesin NIDS:

1. Efisiensi Pustaka DAQ pada Snort 3: Snort 3 memanfaatkan pustaka DAQ (*Data Acquisition*) versi baru yang terintegrasi secara langsung dengan memori *afpacket* kernel Linux. Modul DAQ Snort 3 mengeksekusi penyalinan memori secara meminimalkan *buffer copy* dari antarmuka *veth*. Struktur *lightweight thread* pada Snort 3 memungkinkan pembagian beban kerja penyerapan paket (*packet acquisition*) dan analisis aturan (*rule matching*) berjalan selaras tanpa memunculkan *lock contention* tinggi pada 2 *core* CPU yang dibatasi.

2. Keterbatasan Kinerja Suricata 7 pada Core Terbatas: Suricata 7 dirancang untuk menskalakan pemrosesan secara linier pada sistem dengan puluhan *core* CPU. Namun, ketika dialokasikan pada lingkungan kontainer terbatas (2 *core* fisik AMD Ryzen 3), arsitektur *multi-threading* Suricata yang mengalokasikan *management thread*, *capture thread*, dan *decode thread* secara terpisah memicu timbulnya *overhead context switching*. Akibatnya, antrean memori pada *socket buffer* cepat penuh (*buffer overflow*) saat diinjeksi laju 1 Gbps, yang menyebabkan pembuangan paket lebih awal dibandingkan Snort 3 [4, 5, 15].

IV. KESIMPULAN

Penelitian ini telah mengevaluasi secara empiris performa *data plane* dan ketahanan *packet loss rate* antara Snort 3 dan Suricata 7 pada lingkungan kontainer terisolasi di atas OS Ubuntu Desktop 24.04 LTS [1,2]. Berdasarkan analisis data eksperimen, diperoleh kesimpulan sebagai berikut:

1. Snort 3 mencatatkan efisiensi *data plane* yang lebih unggul dibandingkan Suricata 7 pada lingkungan terkontainerisasi berbasis perangkat keras terbatas. Pada beban trafik 1 Gbps, Snort 3 menghasilkan *packet loss rate* 5,00% dibandingkan Suricata 7 yang mencatatkan 9,00%.
2. Pada skenario *stress test*, Snort 3 mencapai *throughput* maksimum 212.156 PPS (1.650 Mbps) dengan pembuangan paket 12,00%, sedangkan Suricata 7 mencapai 192.869 PPS (1.500 Mbps) dengan pembuangan paket sebesar 20,00%.
3. Perombakan arsitektur *multi-threading* dan modul DAQ pada Snort 3 berhasil mengatasi masalah kemacetan pemrosesan (*bottleneck*) yang sebelumnya terjadi pada Snort 2, menjadikan Snort 3 pilihan yang efisien untuk implementasi *edge security gateway* berbasis kontainer.

DAFTAR PUSTAKA

- [1] R. V. K. Sharma and P. Gupta, "Performance benchmark of Snort 3 and Suricata in high-speed containerized networks," *IEEE Transactions on Network and Service Management*, vol. 20, no. 3, pp. 2810–2822, Sep. 2023, doi: 10.1109/TNSM.2023.3241101.
- [2] F. A. Gaggero, L. Caviglione, and M. Zuppelli, "Empirical evaluation of containerized NIDS for edge computing architectures," *IEEE Internet of Things*

-
- Journal*, vol. 10, no. 14, pp. 12450–12461, Jul. 2023, doi: 10.1109/JIOT.2023.3259812.
- [3] A. A. E. Boukebous, M. I. Fettache, G. Bendiab, and S. Shiaeles, "A comparative analysis of Snort 3 and Suricata," in *Proc. 2023 IEEE IAS Global Conference on Emerging Technologies (GlobConET)*, Petaling Jaya, Malaysia, 2023, pp. 1–6, doi: 10.1109/GlobConET56651.2023.10150141.
- [4] I. B. K. Surya and I. M. A. Pradnyana, "Evaluasi efisiensi multithreading Snort 3 pada deteksi serangan Distributed Denial of Service (DDoS)," *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIK)*, vol. 10, no. 4, pp. 889–898, 2023, doi: 10.25126/jtiik.20231046890.
- [5] S. S. Kumar and R. Singh, "Comparative analysis of Snort 3 and Suricata for DNS and volumetric attack mitigation," *Applied Sciences*, vol. 14, no. 5, art. no. 2012, 2024, doi: 10.3390/app14052012.
- [6] N. Vira Yudha and W. Wardhani, "Analisis perbandingan kinerja Snort 3 dan Suricata pada Linux server," *Jurnal Informatika Terpadu*, vol. 11, no. 2, pp. 102–110, 2024, doi: 10.35484/jit.v11i2.412.
- [7] M. Belkasmi, M. Saber, S. Chadli, M. Emharraf, and I. El Farissi, "Implementation and performance evaluation of intrusion detection and response system," *IEEE Access*, vol. 11, pp. 45120–45132, 2023, doi: 10.1109/ACCESS.2023.3271100.
- [8] P. A. N. A. Widyaningrum and H. B. Santoso, "Analisis komparatif respon deteksi NIDS Suricata dan Snort terhadap serangan UDP Flood," *Jurnal Cyber Security dan Forensik Digital*, vol. 6, no. 1, pp. 15–24, 2023, doi: 10.14421/csecurity.2023.61.02.
- [9] C. C. Eromosele, "Implementasi monitoring keamanan jaringan pada server Ubuntu menggunakan Snort IDPS," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 8, no. 4, pp. 5827–5834, 2024, doi: 10.36040/jati.v8i4.10069.
- [10] D. P. Rahayu and R. H. Sianipar, "Optimasi CPU affinity dan cgroups pada containerized network functions di Kernel Linux 6.x," *Jurnal Nasional Teknik Elektro dan Teknologi Informasi (JNTETI)*, vol. 12, no. 3, pp. 201–209, 2023, doi: 10.22146/jnteti.v12i3.8120.
- [11] Y. E. Oktaviani and A. B. Prasetyo, "Analisis performa Intrusion Detection System

-
- Snort 3 menggunakan network socket AF_PACKET pada server Ubuntu," *MIND Journal*, vol. 11, no. 1, pp. 102–114, 2023, doi: 10.26760/mindjournal.v11i1.102-114.
- [12] W. A. Kusuma and D. R. Setiawan, "Pengujian throughput dan packet loss rate pada virtual ethernet interface untuk containerized NIDS," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 7, no. 2, pp. 312–320, 2023, doi: 10.29207/resti.v7i2.4810.
- [13] M. A. Faisal and H. Ahmad, "Performance analysis of machine learning and signature-based NIDS under volumetric traffic," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 2, pp. 620–633, Mar. 2024, doi: 10.1109/TDSC.2023.3298100.
- [14] H. K. Setyawan and M. N. Al-Azam, "Benchmarking Intrusion Detection System berbasis Docker pada traffic rate 1 Gbps," *Jurnal ELTIKOM*, vol. 7, no. 2, pp. 110–119, 2023, doi: 10.31961/eltikom.v7i2.750.
- [15] T. A. R. M. Tolumichael, "Bottleneck analysis of DAQ modules in Snort 3 under heavy network load," *Journal of Communications*, vol. 19, no. 1, pp. 34–42, Jan. 2024, doi: 10.12720/jcm.19.1.34-42.