

Implementasi Dan Optimasi Lightweight Intrusion Detection System Berbasis Snort Pada Server Berspek Rendah

Reza Raditya[#], Yusuf Muhyidin[#], Bayu Prasetyo[#]

[#] Program Studi Teknik Informatika, STT Wastukencana, Purwakarta, Indonesia
E-mail: [rezaraditya\[at\]sttwastukencana.ac.id](mailto:rezaraditya[at]sttwastukencana.ac.id), [yusufmuhyidin\[at\]sttwastukencana.ac.id](mailto:yusufmuhyidin[at]sttwastukencana.ac.id),
[bayuprasetyo\[at\]sttwastukencana.ac.id](mailto:bayuprasetyo[at]sttwastukencana.ac.id)

ABSTRACTS

The digitalization of village public services into e-government systems increases efficiency but exposes critical infrastructure to cyber threats such as SQL Injection, Cross-Site Scripting (XSS), and Brute Force. Village servers typically operate on low-specification hardware (1-2 vCPU, 2-4 GB RAM), making heavy security solutions infeasible. This study aims to design, implement, and evaluate a lightweight layered security architecture by integrating an optimized Intrusion Detection System (IDS) using Snort on a low-spec server for e-government services (Case Study Sapa Warga System). We propose a 3-layer architecture (1) Application Layer (PHP Native, MySQL) with input sanitization, (2) Detection Layer (Snort 3.1 with Community Rules optimized from 42,000 to 3,250 active rules relevant to web attacks), and (3) Monitoring Layer (custom dashboard for real-time alerts). The testbed topology consists of Victim Server (Ubuntu Server 22.04, 2 vCPU Intel Xeon 2.4GHz, 2GB RAM, 20GB SSD, Apache 2.4), Attacker Machine (Kali Linux 2024.2, 4GB RAM), and Admin Client. Security testing involved 1,500 malicious payloads, 500 SQL Injection using SQLMap 1.8, 500 XSS payloads using XSSStrike, 300 Brute Force attempts using Hydra, and 200 Port Scanning using Nmap. Metrics measured were Detection Rate, False Positive Rate, Alert Latency, CPU and RAM usage. Snort achieved an average Detection Rate of 96.7% (SQL Injection 98.4%, XSS 95.2%, Brute Force 97.3%, Port Scanning 96.0%) with a low False Positive Rate of 2.1%. Average alert latency was 18.5 ms. Crucially, with rule optimization, CPU usage during peak attack (100 req/sec) was maintained at 23.4% average (peak 31.2%) and RAM usage at 312 MB, compared to 68.7% CPU without optimization. Compared to Suricata 6.0 in the same environment, Snort consumed 41% less RAM (312 MB vs 528 MB) although Suricata had 8% higher throughput. The proposed lightweight architecture proves that Snort, with targeted rule optimization, is highly feasible and efficient for securing e-government applications on resource-constrained village servers, providing real-time detection without significant performance degradation.

Manuscript received Jul 15, 2026;
revised Aug 8 2026; accepted Aug
10, 2026 Date of publication Sep
30, 2026. International Journal,
JITSI : Jurnal Ilmiah Teknologi
Sistem Informasi licensed under a
Creative Commons Attribution-
Share Alike 4.0 International
License



ABSTRAK

Digitalisasi pelayanan publik desa menjadi sistem e-government meningkatkan efisiensi namun membuka kerentanan terhadap ancaman siber seperti SQL Injection, Cross-Site Scripting (XSS), dan Brute Force. Server desa umumnya beroperasi pada spesifikasi rendah (1-2 vCPU, 2-4 GB RAM), sehingga solusi keamanan yang berat tidak dapat diterapkan. Penelitian ini bertujuan merancang, mengimplementasikan, dan mengevaluasi arsitektur keamanan berlapis yang ringan dengan mengintegrasikan Intrusion Detection System (IDS) berbasis Snort yang dioptimasi pada server berspesifikasi rendah untuk layanan e-government (Studi Kasus Sistem Sapa Warga). Kami mengusulkan arsitektur 3 lapis, (1) Application Layer (PHP Native, MySQL) dengan sanitasi input, (2) Detection Layer (Snort 3.1 dengan

Community Rules yang dioptimasi dari 42.000 menjadi 3.250 rules aktif yang relevan dengan serangan web), dan (3) Monitoring Layer (dashboard kustom untuk alert real-time). Server Korban (Ubuntu Server 22.04, 2 vCPU Intel Xeon 2.4GHz, RAM 2GB, SSD 20GB, Apache 2.4), Mesin Penyerang (Kali Linux 2024.2, RAM 4GB), dan Klien Admin. Pengujian keamanan melibatkan 1.500 payload berbahaya, 500 SQL Injection menggunakan SQLMap 1.8, 500 payload XSS menggunakan XSSStrike, 300 percobaan Brute Force menggunakan Hydra, dan 200 Port Scanning menggunakan Nmap. Parameter yang diukur meliputi Detection Rate, False Positive Rate, Latensi Alert, serta penggunaan CPU dan RAM. Snort mencapai rata-rata Detection Rate sebesar 96,7% (SQL Injection 98,4%, XSS 95,2%, Brute Force 97,3%, Port Scanning 96,0%) dengan False Positive Rate rendah 2,1%. Rata-rata latensi alert adalah 18,5 ms. Secara krusial, dengan optimasi rules, penggunaan CPU saat serangan puncak (100 req/detik) terjaga pada rata-rata 23,4% (puncak 31,2%) dan penggunaan RAM 312 MB, dibandingkan 68,7% CPU tanpa optimasi. Dibandingkan Suricata 6.0 pada lingkungan yang sama, Snort mengonsumsi RAM 41% lebih rendah (312 MB vs 528 MB) meskipun Suricata memiliki throughput 8% lebih tinggi. Kesimpulan menunjukkan arsitektur ringan yang diusulkan membuktikan bahwa Snort dengan optimasi rules yang tepat sangat layak dan efisien untuk mengamankan aplikasi e-government pada server desa dengan keterbatasan sumber daya, memberikan deteksi real-time tanpa degradasi performa signifikan.

Keywords / Kata Kunci — *E-Government Security Architecture, Intrusion Detection System, Snort, Low-Spec Server, Rule Optimization, Web Security.*

CORRESPONDING AUTHOR

Reza Raditya
Program Studi Teknik Informatika, STT Wastukencana, Purwakarta, Indonesia
Email: rezaraditya[at]sttwastukencana.ac.id

1. PENDAHULUAN

Perkembangan teknologi informasi telah membawa perubahan yang signifikan dalam penyelenggaraan pelayanan publik di berbagai sektor, termasuk pemerintahan desa. Digitalisasi pelayanan publik menjadi salah satu strategi untuk meningkatkan kualitas pelayanan kepada masyarakat melalui pemanfaatan sistem informasi berbasis web. Implementasi layanan digital memungkinkan masyarakat memperoleh pelayanan secara lebih cepat, efektif, dan efisien tanpa harus datang langsung ke kantor desa. Selain meningkatkan efisiensi pelayanan, digitalisasi juga mendukung transparansi, akuntabilitas, serta pengelolaan data yang lebih terstruktur.

Salah satu bentuk implementasi pelayanan publik berbasis web adalah Website Sapa Warga Desa Cibungur yang dikembangkan sebagai media komunikasi antara masyarakat dan pemerintah desa. Sistem ini menyediakan layanan penyampaian aspirasi, pengaduan, serta pengajuan administrasi secara daring. Dengan adanya sistem tersebut, masyarakat dapat menyampaikan berbagai kebutuhan pelayanan kapan saja, sedangkan pihak desa dapat mengelola seluruh data pelayanan secara terpusat melalui dashboard administrator.

Meskipun memberikan banyak manfaat, penggunaan aplikasi berbasis web juga menghadirkan tantangan baru dalam aspek keamanan informasi. Website pelayanan publik menjadi salah satu target serangan siber karena menyimpan berbagai data penting milik masyarakat maupun pemerintah desa. Serangan seperti SQL Injection, Brute Force Attack, Cross Site Scripting (XSS), maupun akses ilegal dapat menyebabkan kebocoran data, perubahan informasi, bahkan gangguan terhadap operasional sistem. Kondisi tersebut menunjukkan bahwa keamanan sistem informasi merupakan aspek yang tidak dapat dipisahkan dari proses pengembangan aplikasi berbasis web.

Salah satu pendekatan yang banyak digunakan untuk meningkatkan keamanan jaringan adalah penerapan Intrusion Detection System (IDS). IDS berfungsi memantau lalu lintas jaringan secara terus-menerus untuk mendeteksi aktivitas yang mengindikasikan adanya serangan terhadap sistem. Berbeda dengan firewall yang berfokus pada penyaringan akses jaringan, IDS memiliki kemampuan untuk mengenali pola serangan berdasarkan aturan (signature) maupun perilaku tertentu sehingga administrator dapat memperoleh peringatan secara dini ketika terjadi aktivitas yang mencurigakan.

Dalam penelitian ini digunakan Snort, yaitu perangkat lunak IDS berbasis open-source yang menerapkan metode signature-based detection. Snort dipilih karena memiliki kemampuan mendeteksi berbagai jenis serangan jaringan, fleksibel dalam pengelolaan aturan (rules), serta banyak digunakan dalam implementasi keamanan jaringan. Integrasi Snort pada Website Sapa Warga Desa Cibungur diharapkan mampu meningkatkan kemampuan sistem dalam memonitor aktivitas jaringan dan memberikan informasi mengenai upaya serangan secara real-time kepada administrator.

Beberapa penelitian terdahulu menunjukkan bahwa implementasi Snort efektif dalam mendeteksi berbagai serangan terhadap aplikasi web. Namun, sebagian besar penelitian masih berfokus pada simulasi lingkungan laboratorium atau hanya melakukan pengujian terhadap aplikasi uji seperti Damn Vulnerable Web Application (DVWA). Penelitian yang mengintegrasikan IDS secara langsung pada sistem pelayanan publik desa masih relatif terbatas. Oleh karena itu, penelitian ini menawarkan implementasi IDS Snort pada Website Sapa Warga Desa Cibungur sehingga tidak hanya membangun sistem pelayanan masyarakat berbasis web, tetapi juga mengintegrasikan mekanisme keamanan yang mampu mendeteksi aktivitas serangan terhadap aplikasi.

Berdasarkan permasalahan tersebut, penelitian ini bertujuan untuk mengimplementasikan Intrusion Detection System menggunakan Snort pada Website Sapa Warga Desa Cibungur guna meningkatkan keamanan sistem informasi desa. Selain membangun sistem pelayanan masyarakat secara daring, penelitian ini juga mengevaluasi kemampuan Snort dalam mendeteksi aktivitas serangan sehingga diharapkan dapat meningkatkan keandalan, keamanan, dan kualitas pelayanan publik berbasis web.

Perkembangan e-government di tingkat desa menghadapi dilema klasik. kebutuhan akan transparansi layanan berbanding terbalik dengan keterbatasan infrastruktur dan keamanan. Studi BSSN 2024 mencatat serangan terhadap layanan pemerintah daerah meningkat 38% YoY, dengan SQL Injection dan Brute Force sebagai vektor dominan. Penelitian terdahulu banyak mengimplementasikan Snort pada lingkungan laboratorium ideal dengan server high-spec dan aplikasi rentan buatan seperti Damn Vulnerable Web Application (DVWA). Research gap utama yang ditemukan yaitu pertama, perbandingan performa pada Low-Spec Environment, Belum ada studi komparatif Snort vs Suricata pada skenario server desa (2GB RAM) yang menjadi standar de facto di pemerintahan desa karena keterbatasan APBDes. Suricata dengan arsitektur multi-thread diklaim lebih performan, namun konsumsi memorinya tinggi. Kedua, masalah Rule Bloat, Implementasi default Snort Community Rules (>42.000 rules) menyebabkan CPU spike >65% dan false positive tinggi pada server rendah, menyebabkan service down. Belum ada strategi optimasi rules yang spesifik untuk konteks e-government desa yang hanya membutuhkan proteksi terhadap OWASP Top 10. Ketiga, arsitektur keamanan terintegrasi: Penelitian sebelumnya menempatkan IDS sebagai add-on terpisah, bukan sebagai bagian dari arsitektur berlapis (layered architecture) yang terintegrasi dengan dashboard admin desa. Untuk menjawab gap tersebut, penelitian ini berkontribusi Merancang arsitektur keamanan berlapis 3-lapis yang ringan untuk e-government desa, Melakukan optimasi rules Snort dari 42.000 menjadi 3.250 rules (hanya kategori web-attacks, sql, xss, brute-force, scan) dan Melakukan evaluasi kuantitatif dengan metrik Detection Rate, Latensi, dan Resource Consumption serta komparasi head-to-head Snort vs Suricata

2. METODOLOGI PENELITIAN

2.1. Arsitektur Sistem yang Diusulkan

Kami mengusulkan Layered Security Architecture yang terdiri dari tiga lapisan: Layer 1 - Application Layer: Sistem Sapa Warga (Fitur: Aspirasi, Pengaduan, Pengajuan) dibangun dengan PHP Native 8.1, MySQL 8.0, Apache 2.4. Dilengkapi input sanitization dan prepared statement, namun sengaja disisakan celah terbatas untuk pengujian IDS. Layer 2 - Detection Layer: Snort 3.1.65.0 berjalan dalam mode IDS (bukan IPS) agar tidak memblokir langsung. Snort ditempatkan pada interface eth0 yang sama dengan web server (tap mode). Optimasi: Hanya mengaktifkan rule kategori: community-sql-injection, community-web-attacks, community-web-xss, community-brute-force, community-scan. Thresholding: limit 5 event/sec per IP untuk mengurangi alert flood saat Brute Force. Layer 3 - Monitoring Layer: Log Snort (/var/log/snort/alert_json.txt) diparsing via Python script dan ditampilkan di dashboard admin (/admin/security-log) secara real-time dengan informasi: timestamp, classtype, src_ip, payload snippet.

2.2. Spesifikasi dan Topologi Pengujian Keamanan

Topologi yang digunakan adalah Star, 3 node dalam VirtualBox Host-Only Network 192.168.100.0/24. (1) Victim Server (Sapa Warga + Snort): OS: Ubuntu Server 22.04 LTS, vCPU: 2 Core Intel Xeon 2.40GHz, RAM: 2 GB DDR4, Storage: 20 GB SSD, Web Server: Apache 2.4.52, IDS: Snort 3.1.65.0, mode: snort -i eth0 -c /usr/local/etc/snort/snort.lua -A alert_json, Tools Monitoring: http, sar, tcpdump. (2) Attacker Machine: OS: Kali Linux 2024.2, RAM: 4 GB, Tools: Nmap 7.94 (Port Scanning), SQLMap 1.8.2 (SQL Injection), XSSStrike 3.1.4 (XSS), Hydra 9.5 (Brute Force). (3) Admin Client: Windows 11, Browser Chrome untuk akses dashboard.

2.3. Skenario dan Parameter Pengujian

Total serangan: 1.500 request berbahaya, dibagi menjadi: S1: SQL Injection (500 payload) -> sqlmap.py -u "http://192.168.100.10/aspirasi.php?id=1" --batch --risk 3, S2: Cross-Site Scripting (500 payload) -> xssstrike --url http://192.168.100.10/search?q=test, S3: Brute Force Login (300 percobaan) -> hydra -l admin -P rockyou.txt 192.168.100.10 http-post-form "/login.php:username=^USER^&password=^PASS^:gagal", S4: Port Scanning (200 scan) -> nmap -sS -A 192.168.100.10. Parameter yang diukur: Detection Rate = (Terdeteksi / Total Serangan) * 100%, False Positive Rate = (False Alert / Total Traffic Normal 1000 req) * 100%, Alert Latency: selisih waktu

paket masuk (tcpdump) dan alert tercatat (ms), CPU & RAM Usage: diukur dengan sar -u -r 1 selama serangan 60 detik pada rate 10, 50, dan 100 req/sec.

2.4. Komparasi Snort vs Suricata

Untuk validasi, kami mengulang skenario S1-S4 dengan Suricata 6.0.8 dengan konfigurasi default ET-Open rules (optimasi serupa) pada server korban yang sama setelah Snort dimatikan.

3. HASIL DAN PEMBAHASAN

3.1. Hasil Optimasi Rules

Tanpa optimasi (42.000 rules): CPU idle 34.2%, saat serangan 100 req/sec CPU 68.7%, RAM 412 MB, banyak false positive (11.4%). Dengan optimasi (3.250 rules): CPU idle 8.1%, saat serangan 100 req/sec CPU rata-rata 23.4% (puncak 31.2%), RAM 312 MB, False Positive turun menjadi 2.1%. Penurunan 81% jumlah rules menghasilkan efisiensi CPU 65.9%.

3.2. Performa Deteksi Snort

TABEL 1. Hasil Pengujian Deteksi Snort Teroptimasi

Jenis Serangan	Total Payload	Terdeteksi	Detection Rate	Avg Latency	Catatan
SQL Injection	500	492	98.4%	16.2 ms	8 payload bypass (obfuscation base64)
XSS	500	476	95.2%	19.8 ms	Gagal deteksi DOM-based XSS
Brute Force	300	292	97.3%	15.4 ms	Terdeteksi setelah 5 percobaan via threshold
Port Scanning	200	192	96.0%	22.6 ms	Nmap stealth -sS terdeteksi
Rata-rata	1500	1452	96.7%	18.5 ms	False Positive Rate 2.1%

Rata-rata Detection Rate 96,7% dengan latensi 18,5 ms dan False Positive Rate 2,1% (21 alert dari 1000 traffic normal).

3.3. Komparasi Snort vs Suricata pada Low-Spec Server

TABEL 2. Komparasi Snort vs Suricata (100 req/sec attack)

Metrik	Snort 3.1 (Optimasi)	Suricata 6.0 (Optimasi)	Analisis
Detection Rate Avg	96.7%	97.9%	Suricata unggul 1.2% karena multi-pattern matcher
CPU Usage Avg	23.4%	28.9%	Snort lebih ringan 19% (single-thread lebih efisien di 2 vCPU)
RAM Usage	312 MB	528 MB	Snort 41% lebih hemat, kritis untuk server 2GB
Throughput Drop	4.2%	2.1%	Suricata lebih baik menjaga throughput
Alert Latency	18.5 ms	12.1 ms	Suricata lebih cepat 6.4 ms

Suricata memang lebih cepat dan sedikit lebih akurat, namun konsumsi RAM 528 MB hampir 26% dari total RAM server 2GB, berisiko OOM-killer saat traffic normal tinggi. Snort dengan arsitektur single-thread justru lebih cocok untuk server desa berspesifikasi rendah karena footprint memori rendah dan CPU masih di bawah 35% saat serangan puncak.

3.4. Dampak terhadap Layanan E-Government

Selama pengujian serangan 100 req/sec, response time aplikasi Sapa Warga tanpa IDS: 145 ms, dengan Snort aktif: 187 ms (peningkatan 42 ms / 28.9%), masih di bawah batas toleransi 2 detik untuk e-government. Tidak terjadi downtime.

4. KESIMPULAN

Penelitian ini berhasil merancang arsitektur keamanan berlapis ringan untuk e-government desa dengan integrasi Snort yang dioptimasi. Kesimpulan spesifik: (1) Arsitektur 3-lapis (Application-Detection-Monitoring) terbukti efektif mengintegrasikan keamanan tanpa mengubah alur bisnis Sapa Warga. (2) Optimasi rules dari 42.000 menjadi 3.250 rules menurunkan penggunaan CPU sebesar 65.9% (dari 68.7% menjadi 23.4%) dan RAM tetap di 312 MB, menjadikannya layak untuk server 2 GB RAM. (3) Secara kuantitatif, Snort mencapai Detection Rate 96.7% dengan latensi 18.5 ms dan False Positive 2.1%, memenuhi kebutuhan deteksi real-time tingkat desa. (4) Dibandingkan dengan Suricata, Snort 41% lebih hemat RAM, menjadikannya pilihan superior untuk environment low-spec, meskipun Suricata unggul 1.2% dalam Detection Rate. Keterbatasan: Deteksi masih signature-based, sehingga zero-day attack tidak terdeteksi. Penelitian selanjutnya dapat mengintegrasikan WAF (ModSecurity) + Machine Learning anomaly detection untuk hybrid IDS.

UCAPAN TERIMA KASIH

Terima kasih kepada Desa Cibungur, STT Wastukencana, dan tim pembimbing.

REFERENSI

- [1] BSSN, Laporan Tahunan Keamanan Siber 2024.
- [2] M. Roesch, "Snort - Lightweight Intrusion Detection for Networks," LISA 1999.
- [3] OISF, "Suricata User Guide 6.0," 2022.
- [4] OWASP, "Top 10 Web Application Security Risks 2021."
- [5] Pressman, Software Engineering: A Practitioner's Approach, 9th ed.
- [6] R. S. Pressman and B. R. Maxim, Software Engineering: A Practitioner's Approach, 9th ed. New York, NY, USA: McGraw-Hill, 2020.
- [7] I. Sommerville, Software Engineering, 10th ed. Boston, MA, USA: Pearson, 2016.
- [8] M. Roesch, "Snort: Lightweight Intrusion Detection for Networks," Proceedings of the 13th USENIX Conference on System Administration, Seattle, WA, USA, pp. 229–238, 1999.
- [9] M. Cakra Adhana, A. Alanda, H. Amnur, and F. Kasmir, "Deteksi Malware pada File Executable Menggunakan Machine Learning Random Forest", jitsi, vol. 7, no. 2, pp. 176 - 182, Jun. 2026.
- [10] W. Stallings, Network Security Essentials: Applications and Standards, 7th ed. Boston, MA, USA: Pearson, 2021.
- [11] A. S. Tanenbaum and D. J. Wetherall, Computer Networks, 6th ed. Boston, MA, USA: Pearson, 2021.
- [12] B. Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C, 20th Anniversary ed. Indianapolis, IN, USA: Wiley, 2015.
- [13] P. Mell, K. Scarfone, and S. Romanosky, "A Complete Guide to Intrusion Detection and Prevention Systems," National Institute of Standards and Technology (NIST), Gaithersburg, MD, USA.
- [14] M. Szlenk, "Formal semantics and reasoning about UML class diagram," in Proceedings of the International Conference on Dependability of Computer Systems (DepCoS-RELCOMEX), Poland, 2006, pp. 51–59.
- [15] I. Riadi and A. Widodo, "Implementasi Intrusion Detection System (IDS) menggunakan Snort untuk mendeteksi serangan pada jaringan komputer," Jurnal Sarjana Teknik Informatika, vol. 2, no. 1, pp. 411–422, 2014.
- [16] D. Flanagan, JavaScript: The Definitive Guide, 7th ed., Sebastopol, CA: O'Reilly Media, 2020.