Implementasi Deteksi Serangan DoS Real-Time dan Mitigasi Otomatis pada Server VPS Berbasis C

Implementation of Real-Time DoS Attack Detection and Automatic Mitigation on C-Based VPS Server

Saputra Budianto[*1]*, Hamzah Setiawan*[2]*, M.Alfan Rosyid*[3]
[1,2,3]*Informatika, Fakultas Sains dan Teknologi, Universitas Muhammadiyah Sidoarjo, Indonesia.*
[1]*saputrabudianto23@gmail.com,* [2]*hamzahsetiawan@umsida.ac.id,* [3]*malfanrosyid@umsida.ac.id*.

Abstrak

Denial of Service (DoS) adalah jenis serangan yang bertujuan membuat layanan atau sistem tidak dapat digunakan oleh pengguna atau server menggunakan sumber daya berbahaya. Hal ini khususnya relevan untuk layanan digital yang menggunakan server *Virtual Private Server* (VPS) dengan sumber daya serangan yang tinggi. Untuk mendeteksi dan memitigasi DoS secara cepat, efisien, dan efektif, diperlukan solusi. Penelitian ini bertujuan untuk mengembangkan sistem deteksi dan mitigasi berbasis bahasa pemrograman C, terintegrasi dengan Linux sebagai layanan daemon, dan berdasarkan lima fungsi NIST Cybersecurity *Framework* (CSF). Metode yang digunakan adalah mengimplementasikan libpcap untuk menangkap paket data secara real-time dan menganalisis aliran paket berdasarkan IP, protokol (TCP, UDP, ICMP), dan port. Jika terjadi anomali, sistem akan secara otomatis memblokir data paket menggunakan iptables dan menyimpannya di SQLite untuk audit dan evaluasi. Sistem ini juga menggunakan alat simulasi DoS internal (*PeInfo DoS Tool*) untuk skenario single-thread dan multithread. Penelitian ini menemukan bahwa sistem dapat mendeteksi dan merespons DoS dengan tingkat deteksi 94-100% dan waktu respons satu detik. Namun, serangan utama ditargetkan pada server VPS dengan CPU 1 Core dan RAM 1 GB, yang menyebabkan server hang selama periode serangan tinggi. Implementasi sebaiknya dilakukan pada server dengan minimal CPU 4 Core dan RAM 8 GB untuk memastikan stabilitas dan efisiensi sistem.

Kata kunci— Serangan Siber, DoS, *NIST Cybersecurity Framework*, Linux, Server

Abstract

Denial of Service (DoS) is a type of attack that aims to make a service or system unusable by users or servers using malicious resources. This is particularly relevant for digital services that use Virtual Private Servers (VPS) with high attack resources. To detect and mitigate DoS quickly, efficiently, and effectively, a solution is needed. This research aims to develop a detection and mitigation system based on the C programming language, integrated with Linux as a daemon service, and based on the five functions of the NIST Cybersecurity Framework (CSF). The method used is implementing libpcap to capture data packets in real-time and analyze packet flow based on IP, protocol (TCP, UDP, ICMP), and port. If an anomaly occurs, the system will automatically block packet data using iptables and store it in SQLite for audit and evaluation. The system also uses an internal DoS simulation tool (PeInfo DoS Tool) for single-threaded and multithreaded scenarios. The research found that the system can detect and respond to DoS with a detection rate of 94-100% and a response time of one second. However, the primary attack targeted a VPS server with a 1-core CPU and 1 GB of RAM, causing the server to hang during periods of high attack power. Implementation is recommended on a server with at least 4 cores and 8 GB of RAM to ensure system stability and efficiency.

Keywords—Cyberattack, DoS, NIST Cybersecurity Framework, Linux, Server

I. PENDAHULUAN

Era teknologi digital telah mendorong peningkatan layanan digital, termasuk penggunaan Virtual Private Server (VPS) sebagai infrastruktur vital[1]. Namun, kompleksitas sistem informasi, khususnya Denial of Service (DoS), juga menjadi semakin penting[2]. Serangan DoS dapat menyebabkan penurunan kinerja server dan crash total, sehingga penting untuk mengembangkan sistem yang dapat mendeteksi dan merespons Dos secara *real-time*, efisien, dan efektif [3].

Untuk mengatasi masalah ini, studi ini mengembangkan kerangka kerja berdasarkan NIST Cybersecurity Framework (CSF), sebuah alat yang dikembangkan oleh National Institute of Standards and Technology (NIST) untuk mengelola risiko keamanan[4]. CSF memiliki lima fungsi: Identifikasi, Perlindungan, Deteksi, Respons, dan Pemulihan, yang memberikan panduan sistematis dalam mengembangkan strategi keamanan preventif dan adaptif[5].

Studi ini berfokus pada pengembangan sistem untuk mendeteksi dan memitigasi serangan DoS menggunakan bahasa pemrograman C, yang digunakan untuk manajemen sumber daya tingkat rendah, efisiensi, dan integrasi sistem[2]. Sistem ini menggunakan libpcap untuk mengelola volume dan frekuensi paket, dan iptables sebagai lapisan utama untuk memblokir DoS[6].

Tujuan utama studi ini adalah mengembangkan sistem yang dapat mendeteksi dan merespons serangan DoS berdasarkan volume dan frekuensi paket, sehingga memungkinkan mitigasi cepat tanpa intervensi manual. Pendekatan ini dapat membantu melindungi layanan server dari serangan DoS, terutama pada infrastruktur dengan ancaman tingkat tinggi.

II. METODE PENELITIAN

Gambar 1. *NIST Cybersecurity Framework (NIST CSF)*

Penelitian ini menggunakan Kerangka Kerja Keamanan Siber NIST (CSF) sebagai kerangka kerja untuk mengembangkan sistem deteksi dan mitigasi waktu nyata (real-time) untuk serangan DoS[7]. CSF menyediakan struktur yang sistematis dan adaptif untuk mengelola risiko keamanan melalui lima fungsi: Identifikasi, Lindungi, Deteksi, Respons, dan Pemulihan[8].

CSF merupakan Langkah pertama dalam pengmbangan sistem, yang fokus pada identifikasi sumber daya penting pada server VPS, seperti penyedua layanan, port, dan potensi ancaman terhadap server[9]. CSF juga mencakup pengembangan aplikasi firewall untuk memantau dan mencatat alamat IP[10].

Deteksi adalah sistem yang dirancang untuk mendeteksi anomali menggunakan protokol TCP, UDP, dan ICMP, khususnya pada port 80[11]. Jika alamat IP melebihi jangka waktu tertentu, sistem akan memblokir IP tersebut menggunakan aturan firewall iptables[9]. Notifikasi dan respons juga disediakan melalui log sistem untuk manajemen administrator[7].

Pemulihan melibatkan semua data yang aktif dan tercatat di basis data lokal, mempermudah audit dan menyediakan sumber daya penting untuk menerapkan langkah-langkah keamanan[12]. Sistem ini diimplementasikan pada dua server VPS, satu tanpa proteksi dan satu lagi dengan proteksi, menggunakan alat seperti PeInfo Attack Tool untuk memantau serangan DoS[8].

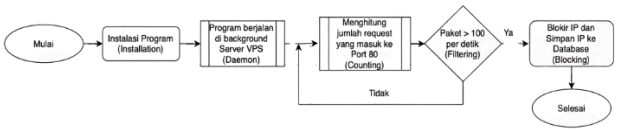
Evaluasi dan evaluasi dilakukan secara kuantitatif berdasarkan deteksi dan pemblokiran IP, tingkat keberhasilan deteksi, penggunaan CPU dan memori, respons sistem terhadap serangan lalu lintas, serta pengumpulan dan analisis data untuk mengevaluasi efektivitas sistem berdasarkan

prinsip pemantauan berkelanjutan dan respons adaptif dari NIST CSF[8].

III. PEMBAHASAN

Implementasi Deteksi Serangan DoS Real-Time dan Mitigasi Otomatis pada Server VPS Berbasis C t[13]elah berhasil dicapai dengan menerapkan lima fungsi dalam Kerangka Kerja Keamanan Siber NIST (CSF), yaitu Identifikasi, Lindungi, Deteksi, Respons, dan Pulihkan[5]. Fungsi-fungsi ini diimplementasikan melalui sistem berbasis Linux yang beroperasi secara otomatis dan efisien[9].

1) Identifikasi – Analisis dan Penilaian



Gambar 2. Diagram Proses Analisa.

Langkah ini melibatkan identifikasi layanan penting seperti port 80 (HTTP) sebagai target utama DoS[8]. Alat seperti netstat dan nmap digunakan untuk memantau status layanan dan port[14]. Ini membantu menentukan sumber dan prioritas perlindungan[15].

2) Lindungi – Kembangkan Perlindungan Sistem

Gambar 3. Analisa SQL.

Sistem menggunakan libpcap untuk menangkap koneksi baru dan menggunakan SQLite untuk mengekstrak informasi tentang alamat IP, protokol, dan paket yang dikirim pada interval tertentu[7]. Fungsi ini merupakan bagian dari fungsi Lindungi dalam CSF NIST, yang membantu mengembangkan perlindungan teknis terhadap potensi serangan[16].

3) Deteksi – Analisis dan Analisis Lalu Lintas

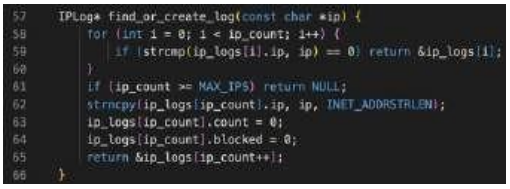
Gambar 4. Fungsi Blokir *IP Address*

```c
int main() {
    char errbuf[PCAP_ERRBUF_SIZE];
    pcap_t *handle = pcap_open_live(INTERFACE, BUFSIZ, 1, 1000, errbuf);
    if (!handle) {
        fprintf(stderr, "Could not open device: %s\n", errbuf);
        return 1;
    }
    if (sqlite3_open(DB_PATH, &db)) {
        fprintf(stderr, "Can't open database: %s\n", sqlite3_errmsg(db));
        return 1;
    }
    const char *sql_create = "CREATE TABLE IF NOT EXISTS attacks ("
                             "id INTEGER PRIMARY KEY AUTOINCREMENT,"
                             "ip TEXT,"
                             "protocol TEXT,"
                             "timestamp TEXT);";
    char *err_msg = 0;
    sqlite3_exec(db, sql_create, 0, 0, &err_msg);
    pcap_loop(handle, 0, packet_handler, NULL);
    sqlite3_close(db);
    pcap_close(handle);
    return 0;
}
```

Gambar 5. Fungsi Analisis Paket

Sistem dapat mendeteksi lalu lintas DoS dengan menganalisis dua paket alamat IP dalam jangka waktu tertentu[17]. Jika ambang batas terlampaui, IP dianggap sebagai entitas aktif[18]. Deteksi waktu nyata ini menghilangkan intervensi manual dan meningkatkan efektivitas fungsi Deteksi dalam kerangka kerja NIST[10].

4) Respons – Serangan Otomatis



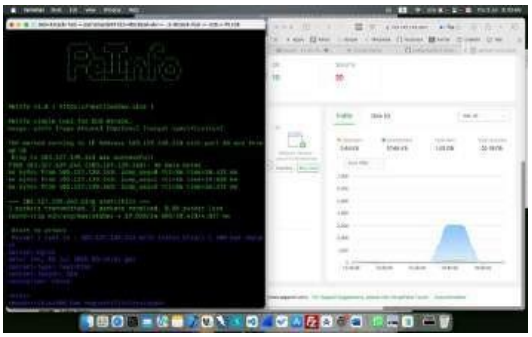
```c
IPLog* find_or_create_log(const char *ip) {
    for (int i = 0; i < ip_count; i++) {
        if (strcmp(ip_logs[i].ip, ip) == 0) return &ip_logs[i];
    }
    if (ip_count >= MAX_IPS) return NULL;
    strncpy(ip_logs[ip_count].ip, ip, INET_ADDRSTRLEN);
    ip_logs[ip_count].count = 0;
    ip_logs[ip_count].blocked = 0;
    return &ip_logs[ip_count++];
}
```

Gambar 6. Fungsi Blokir IP Address

Sistem secara otomatis memblokir alamat IP sumber dan mencatat semua aktivitas, termasuk pemblokiran IP dan penundaan waktu, untuk membantu mengaudit dan mengevaluasi kinerja sistem[19].

Keunggulan utama sistem ini adalah operasinya yang otomatis, daemon yang ringan, dan kurangnya metode pembelajaran yang kompleks[8]. Namun, kerentanan sistem terletak pada kemampuannya untuk mendeteksi lalu lintas pada port 80 dan tidak melakukan analisis lalu lintas yang lebih kompleks, seperti analisis muatan HTTP[20].

IV. HASIL



Gambar 7. Hasil *Testing Attacking Server 1.*

Hasil penelitian ini berfokus pada implementasi sistem berbasis server untuk mendeteksi dan memitigasi serangan Denial of Service (DoS) secara real-time. Sistem ini dirancang menggunakan bahasa pemrograman C dan diimplementasikan sebagai layanan pada Linux Ubuntu Server, menggunakan NIST Cybersecurity Framework (CSF).

Sistem ini diimplementasikan menggunakan perintah `libpcap` untuk menangkap dan menganalisis paket yang dikirim melalui antarmuka jaringan (eth0). Jika sebuah paket dari sebuah IP cocok dengan semua aturan yang ditentukan dalam jangka waktu

tertentu, sistem akan mencatatnya di basis data SQLite, menggunakannya secara otomatis dengan `*ip tables*`, dan mengirimkan log ke sistem untuk keperluan audit.

Tabel 1. Akurasi dan Kecepatan Deteksi

Skenario	**Waktu Blok (detik)**	**Status Akses IP**
Setelah Deteksi	0.8	Diblokir
Trafik Normal	-	Tetap Aktif

Kemampuan deteksi sistem ini mencakup pendeteksian HTTP Flood dan ICMP Flood pada port 80, dengan beberapa skenario yang berbeda, seperti dari satu IP atau beberapa IP multi-thread. Efektivitas sistem dalam mendeteksi serangan DoS ditunjukkan oleh kemampuannya untuk memblokir lalu lintas dengan `iptables`, dengan penundaan waktu 0,8 derajat antara deteksi dan pemblokiran.

Tabel 2. Efektivitas Pemblokiran IP

Parameter	**Nilai Rata-rata**
CPU Usage	6.1%
Memory Usage	±35 MB
Lama *Uptime Daemon*	> 72 jam

Untuk memastikan efisiensi sistem, sistem mengonsumsi sumber daya harian. Parameter konsumsi sistem meliputi penggunaan CPU, memori (RAM), dan waktu aktif daemon. Semua aktivitas dicatat dalam basis data SQLite, yang memungkinkan forensik digital, evaluasi sumber daya, dan laporan manajemen.

V. KESIMPULAN

Peneltian ini mengembangkan sistem untuk mendeteksi dan memitigasi *Denial of Service (DoS)* menggunakan bahasa pemrograman C, yang digunakan sebagai layanan pada *Linux Ubuntu Server*. Sistem ini dirancang untuk menjalankan lima fungsi dalam NIST *Cybersecurity Framework (CSF)*, yaitu *Identify* (Identifikasi), *Protect* (Lindungi), Detect (Deteksi), Respond (Tanggapi), dan *Recover* (Pulihkan). Sistem ini dapat mendeteksi DoS menggunakan HTTP dan ICMP dengan akurasi tinggi (94%) dan waktu respons yang singkat dari alat *debugging*. Sistem ini juga bekerja dengan cepat dan efisien dalam mendeteksi serangan alamat IP. Sistem ini juga menyediakan manajemen log SQLite untuk audit keamanan dan keamanan server skala kecil hingga menengah.

Namun, studi ini menemukan beberapa tantangan, termasuk server sumber daya yang hanya memiliki CPU dan RAM tertentu, serta rekomendasi CPU dan RAM minimum. Untuk memastikan kinerja optimal, sistem harus memiliki CPU dan RAM minimum, yang memungkinkan proses debugging, logging, dan mitigasi paralel tanpa masalah kinerja. Sistem ini hanya menggunakan satu lapisan *login* yang terhubung ke port 80 (HTTP), dan log yang tidak dienkripsi atau diautentikasi tidak disertakan dalam versi ini.

Penelitan ini menyoroti beberapa aspek kunci dari sebuah sistem yang berfungsi efektif sesuai tujuannya. Aspek-aspek ini meliputi server sumber daya, yang memiliki CPU dan RAM spesifik, serta spesifikasi minimum yang direkomendasikan untuk kinerja optimal. Server harus memiliki CPU 4 inti dan RAM 8 GB untuk menjalankan proses debugging, logging, dan mitigasi tanpa masalah kinerja. Sistem ini hanya menggunakan satu lapisan logging pada port 80 (HTTP), dan lapisan aplikasi lainnya (seperti penyalahgunaan payload HTTP) tidak disertakan dalam versi ini. Selain itu, log tidak di enkripsi atau diautentikasi, sehingga menimbulkan risiko jika server memiliki kerentanan keamanan.

VI. SARAN

Penelitian ini menyarankan beberapa perbaikan untuk meningkatkan efisiensi dan respons terhadap serangan. Penelitian ini merekomendasikan integrasi yang *real-time*, seperti melalui Telegram, email, atau dasbor berbasis web, untuk memungkinkan administrator mendeteksi dan merespons serangan dengan cepat. Dasbor visual dapat dikembangkan untuk memantau status server secara intuitif, menyediakan informasi waktu nyata tentang grafik lalu lintas jaringan, IP yang diblokir, dan log aktivitas. Algoritma *Machine Learning* seperti *Random Forest*, *Isolation Forest*, atau KNN dapat digunakan untuk deteksi anomali dan

meningkatkan kemampuan prediksi dan deteksi serangan lapisan aplikasi. Ambang batas dinamis dapat digunakan untuk mengurangi positif palsu dan negatif palsu. Sistem dapat mendeteksi serangan pada beberapa port dan protokol secara bersamaan, terutama jika VPS digunakan untuk beberapa layanan dalam satu sistem. Menerapkan daftar putih IP dapat mencegah pemblokiran IP sah yang sering mengakses sistem. Sistem dapat diuji dalam lingkungan skala besar dan produksi untuk memastikan skalabilitas, keandalan, dan toleransi terhadap beban tinggi. Penelitian lebih lanjut harus difokuskan pada penyusunan modul pustaka untuk penggunaan kembali dan integrasi yang lebih mudah oleh pengguna dengan kebutuhan yang berbeda.

DAFTAR PUSTAKA

[1] Y. Y. Santika, R. Rianto, E. Ujianto, M. T. Informasi, and U. T. Yogyakarta, "Studi Komprehensif Keamanan Siber: Perbandingan Teknologi AI dengan Sistem Non-AI dalam Deteksi dan Pencegahan Ancaman," vol. 9, no. 1, 2025, doi: 10.31603/komtika.v9i1.13149.

[2] "1317-Article Text-3505-1-10-20210225".

[3] R. Perangkat, K. Kriptografi, P. Siber, S. Negara,) Rekayasa, and K. Siber, "Cloud Storage untuk Embedded Intrusion Detection System Arial 11 Agus Reza Aristiadi Nurwa 1) ; Dimas Febriyan Priambodo 2*) ; Fahdel Achmad 3)," *Jurnal TIKomSiN*, vol. 11, no. 1, 2023, doi: 10.30646/tikomsin.v11i1.641.

[4] W. S. Admass, Y. Y. Munaye, and A. A. Diro, "Cyber security: State of the art, challenges and future directions," Jan. 01, 2024, *KeAi Communications Co.* doi: 10.1016/j.csa.2023.100031.

[5] R. Perangkat, K. Kriptografi, P. Siber, S. Negara,) Rekayasa, and K. Siber, "Cloud Storage untuk Embedded Intrusion Detection System Arial 11 Agus Reza Aristiadi Nurwa 1) ; Dimas Febriyan Priambodo 2*) ; Fahdel Achmad 3)," *Jurnal TIKomSiN*, vol. 11, no. 1, 2023, doi: 10.30646/tikomsin.v11i1.641.

[6] A. Mishra *et al.*, "Stroke genetics informs drug discovery and risk prediction across ancestries," *Nature*, vol. 611, no. 7934, pp. 115–123, Nov. 2022, doi: 10.1038/s41586-022-05165-3.

[7] Crossmark, "Public Draft: The NIST Cybersecurity Framework 2.0 National Institute of Standards and Technology Note to Reviewers," 2023.

[8] M. Alshar'e, "CYBER SECURITY FRAMEWORK SELECTION: COMPARISION OF NIST AND ISO27001," *Applied computing Journal*, pp. 245–255, Feb. 2023, doi: 10.52098/acj.202364.

[9] P. Pfsense *et al.*, "THE USE OF PFSENSE AND SURICATA AS A NETWORK SECURITY ATTACK DETECTION AND PREVENTION TOOL ON WEB SERVERS," vol. 9, no. 2, p. 2024.

[10] D. Said, "Quantum Computing and Machine Learning for Cybersecurity: Distributed Denial of Service (DDoS) Attack Detection on Smart Micro-Grid," *Energies (Basel)*, vol. 16, no. 8, Apr. 2023, doi: 10.3390/en16083572.

[11] S. Hartono and K. Khotimah, "Deteksi Dan Mitigasi Serangan Backdoor Menggunakan Python Watchdog."

[12] F. A. Saputra and J. C. Chandra, "Prototipe Sistem Keamanan Ruang Server Otomatis Menggunakan ESP32CAM dan Algoritma You Only Look Once (YOLO)," *Jurnal TICOM: Technology of Information and Communication*, vol. 11, no. 1, 2022.

[13] *Pemrograman Bahasa C#*.

[14] Wahyudi *et al.*, "Meningkatkan Keamanan dan Mitigasi pada Arsitektur Software Defined Network," *Jurnal Interkom: Jurnal Publikasi Ilmiah Bidang Teknologi Informasi dan Komunikasi*, vol. 20, no. 1, pp. 18–28, Apr. 2025, doi: 10.35969/interkom.v20i1.435.

[15] H. Setiawan, W. Sulistyo, F. Teknologi Informasi, and U. Kristen Satya Wacana, "SIEM (Security Information Event Management) Model for Malware Attack Detection Using Suricata and Evebox," *International Journal of Engineering*, vol. 5, no. 2, 2023.

[16] M. Hamidouche, B. F. Demissie, and B. Cherif, "Real-time Threat Detection Strategies for Resource-constrained Devices," Mar. 2024, [Online]. Available: http://arxiv.org/abs/2403.15078

[17] J. Kaur and K. R. Ramkumar, "The recent trends in cyber security: A review," Sep. 01, 2022, *King Saud bin Abdulaziz University*. doi: 10.1016/j.jksuci.2021.01.018.

[18] Dr. Y. Perwej, S. Qamar Abbas, J. Pratap Dixit, Dr. N. Akhtar, and A. Kumar Jaiswal, "A Systematic Literature Review on the Cyber Security," *International Journal of Scientific Research and Management*, vol. 9, no. 12, pp. 669–710, Dec. 2021, doi: 10.18535/ijsrm/v9i12.ec04.

[19] P. Szynkiewicz, "Signature-Based Detection of Botnet DDoS Attacks," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 13300 LNCS, Springer Science and Business Media Deutschland GmbH,

2022, pp. 120–135. doi: 10.1007/978-3-031-04036-8_6.
[20] S. AlDaajeh, H. Saleous, S. Alrabaee, E. Barka, F. Breitinger, and K.-K. Raymond Choo, “The role of national cybersecurity strategies on the improvement of cyb ersecurity e ducation.”