Journal of Applied Informatics and Computing (JAIC)
Vol.x, No.x, Juli 201x, pp. xx~xx
e-ISSN: 2548-6861

Implementasi Real-Time Availability Monitoring System dan Deteksi Anomali Keamanan Berbasis Mobile (Studi Kasus: PT Dwansoft Global Indonesia)

Prayongki Wijayandi[1]*, Nur Cahyono Kushardianto [2]**
*Teknik Informatika, Politeknik Negeri Batam
** Rekayasa Keamanan Siber, Politeknik Negeri Batam
prayongki.w@gmail.com [1], anung@polibatam.ac.id [2]

Article Info

Article history:
Received …
Revised …
Accepted …

Keyword:
Monitoring real time, deteksi anomali, brute force SSH, Telegram Bot API, Golang, React Native

ABSTRACT

PT Dwansoft Global Indonesia mengelola beberapa server production yang memerlukan pemantauan secara berkelanjutan untuk menjaga ketersediaan layanan dan stabilitas operasional. Berdasarkan hasil analisis kebutuhan Bersama mentor industry di PT dwnsoft, disepakati bahwa perancangan system ini difokuskan pada aspek Ketersediaan (Availability). Hal ini dikarenakan server perusahaan menjalankan layanan bisnis kritis klien, di mana setiap gangguan ketersediaan atau downtime akan berdampak langsung pada terhentinya operasional klien dan pelanggaran Service Level Agreement (SLA)
anomali seperti lonjakan beban CPU, serangan brute force pada SSH, dan peningkatan koneksi PostgreSQL secara cepat. Kondisi ini berpotensi menimbulkan downtime serta memperlambat penanganan insiden. Oleh karena itu, penelitian ini mengusulkan rancang bangun sistem monitoring ketersediaan server secara real-time berbasis mobile untuk membantu administrator dalam memantau kondisi server secara lebih efektif. Sistem dikembangkan menggunakan Golang pada sisi backend, React Native untuk aplikasi mobile, WebSocket untuk komunikasi data real-time, serta Telegram Bot API sebagai media peringatan dini. Sensor backend dirancang untuk mengekstraksi log sistem dan mendeteksi anomali keamanan secara otomatis, khususnya percobaan brute force SSH dan kondisi beban server yang tidak normal. Selain itu, sistem juga menyimpan data monitoring historis ke dalam PostgreSQL untuk keperluan evaluasi dan audit keamanan. Hasil penelitian menunjukkan bahwa sistem yang diusulkan mampu menampilkan metrik server secara real-time, mendeteksi anomali secara otomatis, dan mengirimkan notifikasi dengan cepat melalui Telegram. Sistem ini diharapkan dapat meningkatkan kecepatan respons, mengurangi beban pemantauan manual, dan menjaga ketersediaan server di PT Dwansoft Global Indonesia.

I. PENDAHULUAN

Dalam kerangka kerja keamanan informasi (CIA Triad), aspek Ketersediaan (Availability) merupakan salah satu pilar utama yang harus dijaga agar layanan sistem informasi dapat diakses oleh pengguna yang sah kapan pun dibutuhkan. Pada lingkungan production, server sering kali dihadapkan pada ancaman yang menargetkan terganggunya ketersediaan layanan, seperti serangan Denial of Service (DoS) yang menyebabkan lonjakan komputasi (High Load), hingga upaya peretasan akses melalui serangan Brute Force pada layanan Secure Shell (SSH).

PT Dwansoft Global Indonesia merupakan perusahaan teknologi yang mengelola dan memelihara berbagai server production untuk mendukung jalannya operasi bisnis klien. Berdasarkan hasil observasi selama pelaksanaan magang industri di perusahaan ini, ditemukan sebuah urgensi terkait proses pemantauan (monitoring) infrastruktur. Saat ini, metode pemantauan yang tersedia masih bersifat pasif, di mana administrator harus mengecek metrik secara manual. Pendekatan ini menjadi celah keamanan, terutama ketika terjadi lonjakan beban server atau serangan siber di luar jam kerja. Sering kali insiden baru disadari setelah layanan mengalami downtime, yang berpotensi melanggar Service Level Agreement (SLA) dengan klien.

e-ISSN: 2548-6861

Menjawab urgensi di PT Dwansoft Global Indonesia, diperlukan sebuah sistem pemantauan yang proaktif, otonom, dan dapat diakses secara mobile. Penelitian ini mengusulkan rancang bangun Real-Time Availability Monitoring System menggunakan Golang di sisi backend untuk pemrosesan background yang efisien, serta React Native untuk aplikasi mobile administrator. Sistem ini dirancang untuk mem-parsing log keamanan Linux secara mandiri guna mendeteksi serangan SSH Brute Force dan memantau anomali utilisasi CPU. Ketika anomali terdeteksi, backend akan langsung memicu Early Warning System (EWS) melalui Telegram Bot API. Implementasi ini diharapkan dapat memangkas waktu Incident Response secara drastis dan menjaga stabilitas server di PT Dwansoft Global Indonesia

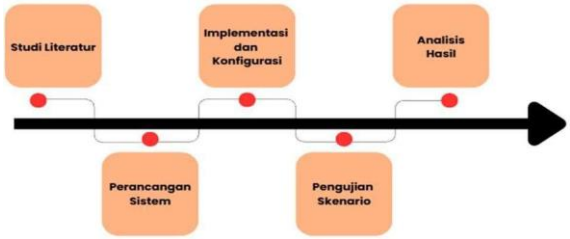
.

II. METODE

Penelitian ini menggunakan metode Design and Development (D&D) dengan pendekatan eksperimental kuantitatif untuk membangun dan menguji sistem monitoring real-time availability dan deteksi anomali keamanan berbasis mobile di PT Dwansoft Global. Pendekatan ini dipilih karena memungkinkan pengembangan sistem secara iteratif sambil melakukan pengujian fungsional dan performa pada setiap tahap, mulai dari analisis kebutuhan, desain arsitektur, implementasi kode, hingga validasi hasil di lingkungan production server perusahaan.

Pengujian dilakukan dengan simulasi beban normal dan serangan (brute force SSH, DoS PostgreSQL connection flood) untuk mengukur akurasi deteksi anomali, latensi WebSocket, waktu respons Telegram notification, serta konsumsi resource sensor Golang pada server target. Pengumpulan data kuantitatif dilakukan untuk mengevaluasi efektivitas sistem secara komprehensif. Parameter ukur yang digunakan meliputi waktu respons deteksi (durasi yang dihitung sejak anomali terjadi pada server hingga notifikasi peringatan berhasil diterima di aplikasi mobile, dengan target capaian waktu latensi di bawah 3 detik).

Dalam penelitian ini, batas maksimal 3 detik digunakan sebagai target waktu pengiriman notifikasi pada Early Warning System. Penentuan angka tersebut didasarkan pada pertimbangan dan pengamatan penulis terhadap kebutuhan kecepatan informasi dalam konteks monitoring server, serta referensi pribadi terkait praktik pemantauan insiden. Angka 3 detik ini tidak merujuk pada standar baku tertentu, melainkan dijadikan acuan kinerja pada penelitian ini agar notifikasi dianggap berhasil jika dapat diterima dalam rentang waktu yang sangat singkat dan masih mudah dicapai secara teknis.



Gambar 1. Alur Penelitian

A. Tahapan Penelitian

Bagian ini menguraikan langkah-langkah penelitian yang dilakukan untuk merancang, membangun, dan menguji sistem Real-Time Availability Monitoring dan Deteksi Anomali Keamanan berbasis Mobile pada lingkungan simulasi PT Dwansoft Global Indonesia. Tahapan penelitian terdiri dari lima langkah utama yang dilakukan secara berurutan:

1. Studi Literatur

Di tahap ini, peneliti mempelajari literatur dan referensi yang berkaitan dengan sistem monitoring real-time, deteksi ancaman keamanan, dan teknologi yang akan digunakan dalam proyek ini. Yang dipelajari meliputi:

- Kajian literatur terkait implementasi Early Warning System (EWS) dan arsitektur komunikasi real-time berbasis protokol WebSocket untuk pemantauan ketersediaan infrastruktur IT
- Teori dasar seperti CIA Triad khususnya Availability, cara kerja WebSocket, fitur Golang goroutine, React Native, dan Telegram Bot API
- Cara teknis membaca log SSH dengan journalctl, memantau koneksi PostgreSQL pakai pg_stat_activity, dan mengambil data CPU/RAM dari /proc/stat
- Praktik terbaik menjaga keamanan server Linux dan batasan normal untuk mendeteksi serangan brute force atau DoS

Hasil dari studi literatur ini jadi acuan utama untuk merancang sistem, menentukan apa saja yang akan diuji, dan metrik apa yang digunakan untuk mengukur keberhasilan system

2. Rancangan Sistem

Tahap ini merupakan proses perancangan keseluruhan lingkungan sistem Real-Time Availability Monitoring dan Deteksi Anomali Keamanan berbasis Mobile yang akan dibangun sebagai dasar implementasi di PT Dwansoft Global Indonesia. Rancangan ini menjadi fondasi teknis untuk menguji bagaimana sistem dapat memantau ketersediaan server secara real-time dan mendeteksi anomali keamanan seperti brute force SSH dan DoS PostgreSQL secara otomatis melalui aplikasi mobile.

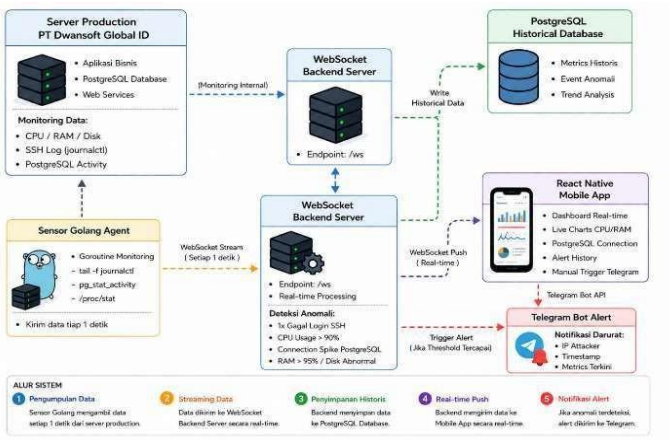
Arsitektur Jaringan:

- Server Production PT Dwansoft, server utama yang menjalankan aplikasi bisnis kritis, PostgreSQL database, dan layanan web. Semua aktivitas system (SSH login, PostgreSQL connection, CPU/RAM usage) dicatat dan dimonitor oleh sensor Golang
- Sensor Golang Agent, program golang berbasis goroutine yang berjalan di server production terdiri dari, [1]tail –f journalctl memantau pola “Failed password” SSH. [2]query pg_stat_activity setiap 1 detik untuk PostgreSQL metrics, [3] read/proc/stat untuk sistem CPU/RAM/disk usage
- Websocket Backend Server, server golang dengan endpoint /ws yang menerima data sensor, melakukan deteksi anomali (threshold brute force 1 kali / lebih dari 1 kali failed login, CPU diatas 90%), dan push real-time ke mobile client
- PostgreSQL Historical Database, berfungsi menyimpan metrik historis dan event anomali untuk keperluan evaluasi, penulurusan kejadian
- React Native mobile app, aplikasi mobile admin PT Dwansoft yang menampilkan real-time dashboard dan menerima websocket push notification

e-ISSN: 2548-6861

Model Pengelolaan Log:

- Pengumpulan Data real-time, sensor golang mengumpulkan SSH log, PostgreSQL metrics, dan sistem resource dari server production PT Dwansoft, kemudian stream via websocket ke backend server setiap detik
- Deteksi Anomali Otomatis, backend server melakukan analisis: [1]Brute force SSH, 1 kali gagal login langsung terkirim notifikasi alert, [2]DoS attack, CPU usage diatas 90 % atau postgreSQL connection spike, [3]Resource exhaustion, RAM diatas 95 % atau disk I/O abnormal, Hasil disimpan ke posgreSQL dan push ke mobile app.
- Notifikasi Darurat, websocket push data real-time ke React native app + trigger telegram bot API saat threshold tercapai, admin menerima alert dengan detail IP attacker, timestamp, dan metrics terkini
- Dashboard mobile real time, react native app menampilkan [1]Live charts CPU/RAM/Disk/PostgreSQL active-idle connection, [2]Alert history dan trend analysis 24 jam, [3]Manual trigger telegram notification



Gambar 2. Topologi Sismulasi Monitoring

Berdasarkan Gambar 2, alur kerja sistem dapat diringkas sebagai berikut

Server Production PT Dwansoft Global Indonesia:

- Server tersebut menjadi pusat utama tempat seluruh aktivitas sistem berlangsung, Server ini menjalankan aplikasi bisnis, database PostgreSQL, dan layanan web yang dipakai dalam operasional perusahaan. Dari server inilah sistem mengambil data penting seperti penggunaan CPU, RAM, disk, log SSH, dan aktivitas koneksi PostgreSQL untuk dipantau secara real-time.
- Sensor golang agent berjalan langsung pada server productioin untuk mengambil data secara otomatis dan berkala. Sensor ini membaca log SSH melalui journalctl, memantau aktivitas PostgreSQL melalui pg_stat_activity, dan memeriksa penggunaan resource server melalui /proc/stat, data yang dikumpulkan dikirim setiap detik agar kondisi server dapat dipantau tanpa jeda dan tanpa harus dicek secara manual.
- WebSocket Backend Server, menerima data dari sensor golang agent melalui koneksi websocket, data yang masuk kemudian di proses secara real-time untuk melihat apakah ada kondisi yang tidak normal, pada tahap ini sistem akan mendeteksi indikasi seperti brute force SSH, lonjakan penggunaan CPU, kenaikan koneksi PostgreSQL, serta penggunaan RAM atau disk yang berlebihan.
- PostgreSQL Historical Database, PostgreSQL historical Database berfungsi sebagai tempat penyimpanan data hasil monitoring dalam jangka waktu tertent, seluruh metrik server dan event anomali disimpan ke dalam database ini agar bisa digunakan kembali saat diperlukan, data historis tersebut membantu melihat riwayat kejadian, membaca pola perubahan, dan menjadi bahan evaluasi ketika terjadi gangguan pada server,

- React Native mobile app, digunakan oleh administrator untuk memantau kondisi server melalui perangkat mobile, Aplikasi ini menampilkan dashboard real-time yang berisi informasi CPU, RAM, disk, aktivitas PostgreSQL, dan riwayat alert. dengan ada nya aplikasi ini, administrator dapat mengetahui kondisi server kapan saja tanpa harus membuka server secara langsung
- Telegram Bot Alert. Berfungsi sebagai media notifikasi darurat secara otomatis ketika sistem mendeteksi anomali. Jika kondisi melewati batas normal, backend akan mengirimkan pesan berisi informasi penting seperti IP attacker, timestamp, dan metrik terkini. Pemilihan Telegram Bot API didasarkan pada efisiensi biaya karena layanannya sepenuhnya gratis, berbeda dengan WhatsApp Business API yang memungut biaya per sesi percakapan, sehingga berisiko membengkakkan biaya operasional perusahaan jika terjadi banyak log serangan. Notifikasi ini membantu administrator bertindak lebih cepat. Adapun alur kerja sistem secara berurutan adalah: [1] Server Production menghasilkan data aktivitas sistem, [2] Sensor Golang Agent mengambil data secara otomatis, [3] Data dikirim ke WebSocket Backend Server untuk diproses dan dianalisis, [4] Data historis disimpan ke PostgreSQL Historical Database, [5] Hasil pemantauan ditampilkan pada React Native Mobile app.

Tabel I. Spesifikasi Server

Komponen	Spesifikasi
Perangkat Utama	KVM/QEMU Virtual Machine
Prosesor (CPU)	QEMU Virtual CPU @ 2.599GHz
Memori (RAM)	4 GB
Sistem Operasi Host	Ubuntu 24.04.4 LTS x86_64

Semua perangkat lunak di bawah merupakan open source dan production-ready, dipilih karena sudah teruji di lingkungan enterprise seperti infrastruktur PT Dwansoft Global Indonesia. Golang dipilih untuk performa tinggi dan konkurensi goroutine, React Native untuk cross-platform mobile development, dan PostgreSQL untuk reliability database production dengan connection pooling yang aman dari DoS attack.

Tabel II. Spesifikasi Source

Komponen	Nama
Database	PostgreSQL
Messaging	Telegram Bot API
WebSocket Library	Gorilla/websocket
System Monitoring	Journalctl, pg_stat_acivity, /proc/stat
Process Manager	Systemd
Autentikasi & Keamanan	SJSON Web Token (JWT) - Algoritma HS256 (Symmetric Key HMAC SHA-256)

3. Implementasi dan Konfigurasi

Pada tahap ini, seluruh rancangan sistem Real-Time Availability Monitoring dan Deteksi Anomali Keamanan berbasis Mobile direalisasikan secara teknis sesuai topologi dan alur yang telah disusun sebelumnya di lingkungan simulasi PT Dwansoft Global Indonesia.

Peneliti menyiapkan lingkungan pengujian berupa virtual machine pada platform virtualisasi KVM/QEMU yang berjalan di atas sistem operasi Ubuntu 24.04.4 LTS x86_64 yang digunakan, meliputi server production simulasi, sensor Golang agent, WebSocket backend server, dan PostgreSQL historical database. Pada server production dikonfigurasi sensor Golang agent yang memantau SSH log, PostgreSQL connection metrics, dan sistem resource usage agar data ketersediaan server dan potensi anomali keamanan dikirim secara otomatis ke WebSocket backend server.

WebSocket backend server kemudian diimplementasikan menggunakan Golang untuk menerima data sensor secara real-time, melakukan deteksi anomali berdasarkan threshold yang telah ditentukan (brute force SSH, CPU spike, connection flood), serta mengirimkan push notification ke React Native mobile app dan Telegram Bot, dengan spesifikasi VM dan pola aliran data yang konsisten untuk menjamin validitas pengujian

4. Pengujian Skenario

Setelah sistem Real-Time Availability Monitoring dan Deteksi Anomali Keamanan berbasis Mobile berhasil diimplementasikan pada lingkungan uji yang sama, peneliti menjalankan serangkaian skenario pengujian untuk mengukur performa dan akurasi masing-masing komponen sistem. Pengujian dilakukan dengan memberikan beban operasional normal dan simulasi serangan keamanan (brute force SSH, PostgreSQL DoS connection flood) ke server target, kemudian mencatat nilai latensi WebSocket, waktu deteksi anomali, tingkat keberhasilan pengiriman Telegram notification, serta penggunaan CPU sensor agent pada setiap skenario pengujian.

5. Anilisis Hasil

Pada tahap ini akan dilakukan analisis hasil pengujian sistem Real-Time Availability Monitoring dan Deteksi Anomali Keamanan berbasis Mobile dengan harapan sebagai berikut:

e-ISSN: 2548-6861

- Waktu Respons Cepat dari Deteksi hingga alert, Sensor Golang diharapkan dapat menangkap anomali dan mengirim notifikasi Telegram ke admin dalam beberapa detik saja, sehingga brute force SSH bisa terdeteksi sangat cepat dan DoS PostgreSQL sebelum database kewalahan.
- Sistem dilengkapi dengan fitur deteksi anomali Brute Force SSH yang beroperasi berdasarkan kebijakan Zero Tolerance (tanpa toleransi). Melalui pendekatan ini, satu kali kegagalan login (failed login) saja akan secara instan memicu pengiriman notifikasi peringatan dini. Kebijakan ketat ini diterapkan untuk membangun tingkat kesadaran penuh (security awareness) bagi administrator.
- Lonjakan Koneksi PostgreSQL Terdeteksi Dini. Kenaikan koneksi database yang tidak wajar diharapkan langsung terdeteksi sehingga tim bisa bertindak cepat sebelum aplikasi bisnis PT Dwansoft terganggu.
- Notifikasi Telegram selalu terkirim. bahkan jika ada gangguan jaringan sistem akan otomatis kirim ulang.
- Penggunaan resource ringan, sensor golang dan websocket backend diharapkan hanya memakan resource CPU dan RAM yang sangat kecil sehingga tidak mengganggu aplikasi bisnis utama PT Dwansoft
- Sistem berjalan stabil tanpa mati, dengan systemd auto-restart, sistem diharapkan bisa jalan terus 24 jam tanpa pernah berhenti meskipun ada error atau restart server
- Waktu tanggap ancaman jauh lebih cepat, yang dulu nya butuh 2 hari untuk sadar ada serangan (cek log manual), diharapkan jadi hanya beberapa menit saja berkat alert telegram dan dashboard mobile.

Analisis ini akan membuktikan sistem yang dirancang mampu menjamin ketersediaan server dan keamanan real-time untuk kebutuhan production PT Dwansoft Global Indonesia.

III. HASIL DAN PEMBAHASAN

A. Implementasi Arsistektur Sistem

Pada tahap implementasi, sistem Real-Time Availability Monitoring dan Deteksi Anomali Keamanan berbasis mobile diimplementasikan sesuai dengan rancangan yang telah dibuat sebelumnya. Arsitektur sistem terdiri dari beberapa komponen utama, yaitu server production PT Dwansoft Global Indonesia, sensor Golang Agent, WebSocket Backend Server, PostgreSQL Historical Database, dan aplikasi mobile berbasis React Native. Seluruh komponen tersebut saling terhubung untuk mendukung proses monitoring secara real-time, penyimpanan data historis, serta pengiriman notifikasi anomali.

B. Implementasi Sensor Golang Agent

Sensor Golang Agent dijalankan pada server production untuk melakukan pengambilan data metrik dan log secara berkala. Sensor ini bertugas membaca data penggunaan CPU, RAM, disk storage, aktivitas PostgreSQL, serta log autentikasi SSH melalui journalctl atau sumber log sistem lainnya. Data yang diambil kemudian dikirim ke WebSocket Backend Server dengan interval.

C. Implementasi WebSocket Backend Server

WebSocket Backend Server digunakan sebagai penghubung utama antara sensor dan aplikasi mobile. Server ini menerima data dari sensor Golang, melakukan pemrosesan data secara real-time, serta menerapkan logika deteksi anomali berdasarkan ambang batas yang telah ditentukan. Ketika nilai metrik melewati batas normal, sistem akan menandai kejadian tersebut sebagai anomali dan menyimpannya ke database historis.

D. Implementasi Penyimpanan Data Historis

Seluruh data metrik dan event anomali disimpan ke dalam PostgreSQL Historical Database. Penyimpanan ini bertujuan agar data dapat digunakan untuk evaluasi, penelusuran kejadian, dan analisis tren di kemudian hari. Data yang disimpan meliputi.

E. Implementasi Aplikasi Mobile

Aplikasi mobile dikembangkan menggunakan React Native untuk menampilkan hasil monitoring secara real-time kepada administrator. Pada aplikasi ini tersedia dashboard yang menampilkan kondisi CPU, RAM, disk storage, koneksi PostgreSQL, histori alert, serta informasi anomali yang terdeteksi. Selain itu, aplikasi juga menyediakan fitur manual trigger untuk pengiriman notifikasi Telegram.

F. Implementasi Telegram Bot Alert

Telegram Bot API digunakan sebagai media notifikasi otomatis ketika sistem mendeteksi anomali. Notifikasi yang dikirim berisi informasi seperti jenis anomali, timestamp, IP source, serta nilai metrik saat kejadian. Dengan adanya fitur ini, administrator dapat segera mengetahui kondisi tidak normal tanpa harus terus memantau dashboard aplikasi.

G. Hasil Tampilan Sistem

Berdasarkan hasil implementasi, sistem berhasil menampilkan data monitoring server secara real-time pada aplikasi mobile. Dashboard menampilkan informasi CPU, RAM, disk storage, serta koneksi PostgreSQL yang diperbarui sesuai interval pengiriman data.

Gambar 3. Tampilan dashboard monitoring pada aplikasi mobile

Pada halaman utama dashboard monitoring, sistem wajib menambahkan koneksi WebSocket Secure dengan format wss://<IP_SERVER>:8080/ws?token= sebagai jalur komunikasi utama antara aplikasi mobile dan server backend. Penggunaan skema wss diterapkan agar seluruh data yang ditransmisikan selama proses monitoring dapat terlindungi melalui enkripsi, sehingga keamanan komunikasi tetap terjaga ketika data berpindah melalui jaringan.

Meskipun menempatkan token kredensial pada parameter URL (?token=) umumnya dihindari pada sistem web biasa, metode ini secara khusus dipilih karena merupakan standar yang paling stabil dan kompatibel untuk membuka koneksi WebSocket pada aplikasi mobile (React Native). Untuk menjamin metode ini tetap aman dari risiko penyadapan maupun kebocoran data, sistem mewajibkan penggunaan protokol WSS (WebSocket Secure) yang secara otomatis mengenkripsi seluruh jalur komunikasi, termasuk URL dan token di dalamnya.

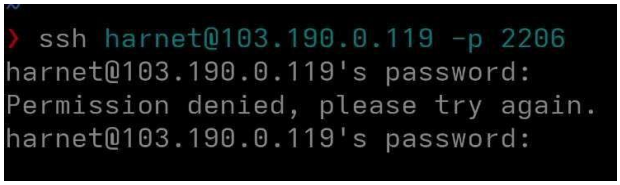
Selain perlindungan enkripsi tersebut, sisi backend Golang juga telah diprogram secara khusus agar tidak mencetak atau menyimpan aktivitas URL yang mengandung token ke dalam riwayat catatan server (journalctl), sehingga dipastikan tidak ada jejak kredensial yang tertinggal di dalam sistem.

Selain itu, pada parameter token digunakan mekanisme autentikasi berbasis JWT untuk memastikan bahwa hanya klien yang telah terotorisasi yang dapat terhubung ke endpoint WebSocket tersebut. Dengan penerapan token ini, risiko akses tidak sah, penyadapan data, maupun manipulasi informasi selama proses transmisi dapat diminimalkan. Adapun token JWT tersebut dapat digenerate melalui perintah go run main.go -generate-token, sehingga proses pembuatan token dapat dilakukan secara terstruktur sebelum digunakan pada halaman dashboard monitoring.

Gambar 4. Grafik CPU, RAM, disk, dan koneksi PostgreSQL

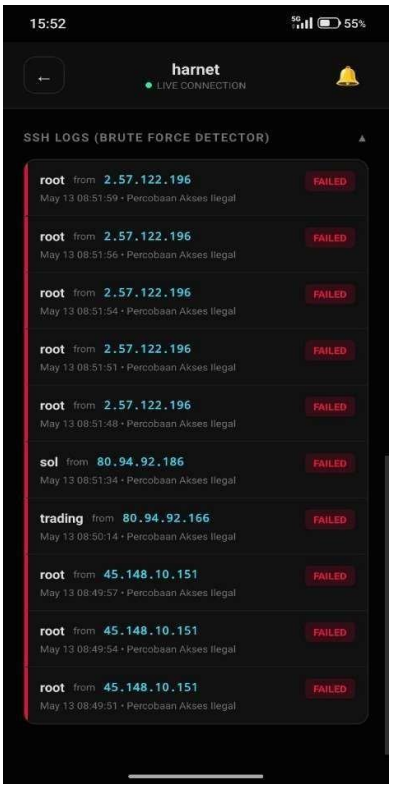
e-ISSN: 2548-6861

Pada gambar tersebut dapat dilihat tampilan halaman utama dashboard monitoring yang menampilkan informasi kondisi server secara ringkas dan real-time. Di dalam dashboard ini tersedia pemantauan terhadap beberapa parameter utama, yaitu CPU, RAM, disk storage, dan koneksi PostgreSQL. Keempat parameter tersebut ditampilkan dalam bentuk visual agar administrator dapat dengan mudah mengetahui kondisi resource server pada saat itu tanpa harus melakukan pengecekan secara manual melalui server. Nilai CPU digunakan untuk melihat tingkat beban pemrosesan yang sedang berjalan, nilai RAM menunjukkan penggunaan memori aktif, sedangkan disk storage memperlihatkan kapasitas penyimpanan yang masih tersedia maupun yang telah terpakai. Selain itu, koneksi PostgreSQL juga dipantau untuk mengetahui apakah terdapat lonjakan aktivitas database yang berpotensi mengganggu performa sistem. Dengan adanya tampilan ini, administrator dapat melakukan pemantauan kondisi server secara lebih cepat, efisien, dan terpusat dalam satu halaman dashboard.



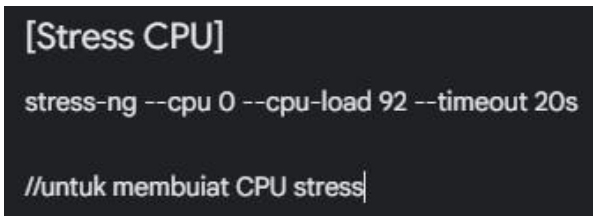
Gambar 5. Mencoba menyerang SSH

Pada bagian ini dilakukan pengujian terhadap mekanisme peringatan dini dengan mensimulasikan kesalahan autentikasi SSH sebanyak tiga kali berturut-turut menggunakan password yang tidak sesuai. Pengujian tersebut bertujuan untuk memastikan bahwa sistem mampu mengenali pola percobaan login gagal sebagai indikasi anomali keamanan. Dari hasil pengujian, sensor backend berbasis Golang berhasil mendeteksi aktivitas login yang gagal tersebut melalui pembacaan log autentikasi SSH, kemudian memprosesnya sebagai event anomali yang layak untuk dipicu menjadi alert. Selanjutnya, alert yang dihasilkan oleh sensor backend tersebut diteruskan ke Telegram Bot API sehingga notifikasi dapat diterima secara otomatis oleh administrator. Dengan demikian, hasil pengujian ini menunjukkan bahwa sistem telah berhasil menjalankan fungsi deteksi dan peringatan dini secara real-time ketika terjadi kesalahan password SSH yang berulang.



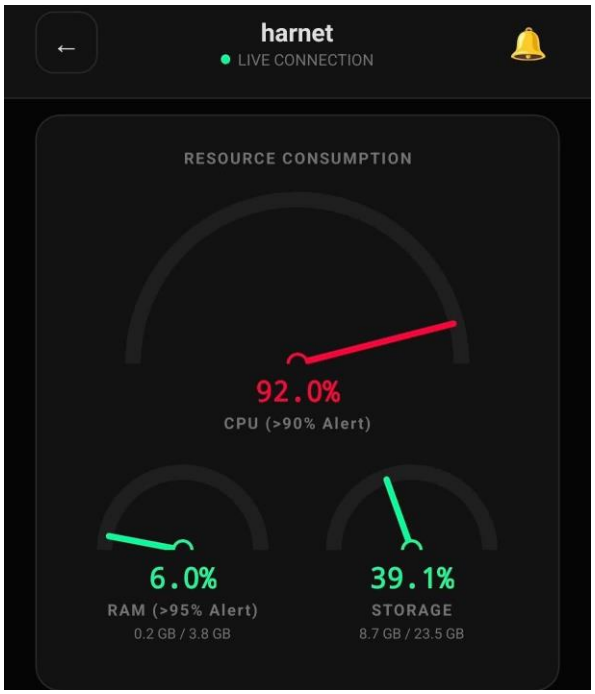
Gambar 6. Log anomaly brute force SSH

Pada bagian log anomali ini, sistem menampilkan hasil deteksi terhadap aktivitas autentikasi SSH yang terindikasi tidak normal. Pengujian dilakukan melalui percobaan login berulang dengan memasukkan kredensial yang tidak valid sebanyak tiga kali atau lebih sebagai bentuk simulasi serangan brute force SSH. Aktivitas tersebut dilakukan untuk menguji kemampuan sistem dalam mengenali pola percobaan akses tidak sah yang dilakukan secara terus-menerus terhadap layanan Secure Shell. Setiap percobaan login yang gagal akan tercatat pada log autentikasi dan diproses oleh sistem sebagai indikasi anomali keamanan. Apabila jumlah percobaan gagal telah memenuhi ambang batas yang ditentukan, sistem akan mengklasifikasikan aktivitas tersebut sebagai potensi serangan brute force dan menampilkannya pada halaman monitoring. Dengan demikian, mekanisme ini berfungsi sebagai deteksi awal terhadap upaya akses ilegal pada layanan SSH server.



Gambar 7. Mencoba melakukan stress CPU

Sensor Golang Agent dijalankan pada server production untuk melakukan pengambilan data metrik dan log secara berkala. Sensor ini bertugas membaca data penggunaan CPU, RAM, disk storage, aktivitas PostgreSQL, serta log autentikasi SSH melalui journalctl atau sumber log sistem lainnya. Data yang diambil kemudian dikirim ke WebSocket Backend Server dengan interval.



Gambar 8. Hasil output stress CPU

Pengujian ini berfokus pada simulasi lonjakan beban CPU (CPU stress test) di mana ambang batas peringatan diatur pada angka 90% sebelumnya menetapkan angka itu dikarenakan penyerang selalu melakukan mining server, yang Dimana hasil dari histori penyerang selalu diatas ambang 90%. Jika penggunaan CPU terdeteksi melewati batas limit tersebut, sistem secara instan akan mengirimkan notifikasi anomali resource ke Telegram. Pengujian tersebut bertujuan untuk mengetahui kemampuan sistem dalam mendeteksi kondisi beban CPU yang melebihi batas normal yang telah ditentukan. Berdasarkan hasil pengujian, ketika utilisasi CPU meningkat hingga melampaui threshold yang ditetapkan, sistem berhasil mengenali kondisi tersebut sebagai anomali dan menampilkan indikator metrik sesuai dengan parameter yang telah dikonfigurasi. Hasil ini menunjukkan bahwa mekanisme deteksi pada sensor backend dapat berfungsi secara efektif dalam mengidentifikasi peningkatan beban CPU secara otomatis dan real-time.

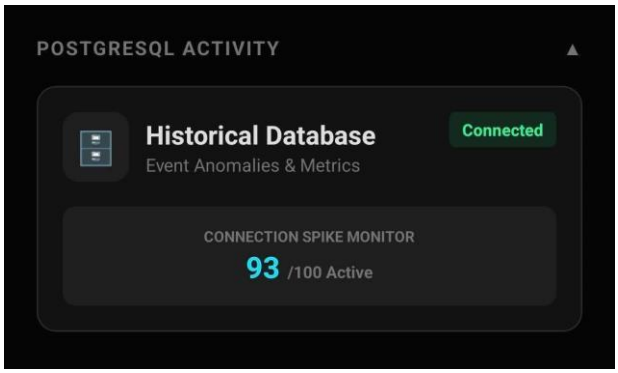
Gambar 9. Notifikasi telegram yang masuk saat threshold tecapai

Ditunjukkan notifikasi Telegram yang diterima ketika threshold deteksi anomali telah tercapai. Notifikasi ini merupakan hasil dari mekanisme early warning yang dirancang pada sistem untuk memberikan peringatan secara cepat kepada administrator ketika terjadi kondisi resource server yang tidak normal. Dalam contoh tersebut, sistem mendeteksi adanya anomali pada resource server dengan informasi yang ditampilkan meliputi IP attacker, timestamp kejadian, serta metrics terkini pada server bernama harnet.

e-ISSN: 2548-6861

Nilai metrik yang tercatat menunjukkan CPU usage sebesar 92%, RAM usage sebesar 6.0%, disk usage sebesar 39.1%. Berdasarkan hasil tersebut, sistem mengidentifikasi bahwa CPU load telah melampaui batas yang ditetapkan.

Notifikasi yang dikirimkan melalui Telegram berfungsi sebagai media peringatan dini agar administrator dapat segera mengetahui adanya kondisi tidak normal tanpa harus memantau dashboard secara terus-menerus. Dengan adanya informasi IP attacker dan timestamp, administrator juga dapat menelusuri sumber kejadian dan waktu terjadinya anomali secara lebih cepat. Selain itu, pencatatan metrik terkini pada notifikasi memberikan gambaran langsung mengenai kondisi server saat anomali berlangsung, sehingga proses analisis dan penanganan dapat dilakukan dengan lebih tepat. Mekanisme ini menunjukkan bahwa sistem telah berhasil menerapkan deteksi ambang batas secara real-time dan mampu mengirimkan peringatan otomatis ketika kondisi resource server berada di luar batas normal yang telah ditentukan.



Gambar 10. Hasil output testing connection spike

Pada pengujian ini dilakukan simulasi serangan berupa lonjakan koneksi pada PostgreSQL dengan menggunakan 92 koneksi aktif secara bersamaan. Pengujian tersebut dijalankan menggunakan perintah PGPASSWORD='password123' pgbench -c 92 -j 4 -T 20 -U yongki -h 127.0.0.1 siabas_db, yang bertujuan untuk mensimulasikan kondisi beban tinggi pada database server dalam waktu tertentu. Melalui pengujian ini, sistem diharapkan mampu mengidentifikasi peningkatan jumlah koneksi yang melebihi batas normal sebagai bentuk anomali pada layanan database. Hasil pengujian menunjukkan bahwa sensor backend berhasil mendeteksi adanya lonjakan koneksi tersebut dan mengklasifikasikannya sebagai DB connection spike. Kondisi ini kemudian dicatat oleh sistem sebagai event anomali sehingga dapat ditampilkan pada halaman monitoring serta disimpan ke dalam database historis untuk keperluan evaluasi lebih lanjut. Dengan demikian, pengujian ini membuktikan bahwa sistem mampu melakukan deteksi terhadap peningkatan koneksi PostgreSQL secara otomatis, sehingga potensi gangguan terhadap ketersediaan layanan database dapat diketahui lebih awal.

Gambar 11. Notifikasi telegram hasil testing connection spike

Ditunjukkan tabel data historis pada PostgreSQL yang menampilkan hasil monitoring server, termasuk informasi terkait lonjakan koneksi basis data. Dalam data tersebut, sistem mencatat adanya kondisi abnormal pada jumlah koneksi aktif ke PostgreSQL, yaitu sebanyak 92 dari 100 koneksi aktif. Kondisi ini menunjukkan bahwa jumlah koneksi database telah mendekati batas maksimum yang ditentukan, sehingga sistem mengidentifikasinya sebagai DB connection spike.

Pencatatan kondisi tersebut ke dalam tabel historis bertujuan untuk menyimpan riwayat kejadian anomali secara terstruktur, sehingga data dapat digunakan kembali untuk analisis, evaluasi, maupun penelusuran insiden di kemudian hari. Dengan adanya histori ini, administrator dapat mengetahui kapan lonjakan koneksi terjadi, seberapa besar tingkat kenaikannya, serta bagaimana korelasinya dengan performa server pada saat kejadian. Hal ini menunjukkan bahwa sistem tidak hanya mendeteksi anomali secara real-

time, tetapi juga mampu menyimpan bukti monitoring secara historis sebagai dasar pengambilan keputusan dan evaluasi keamanan sistem.

H. Hasil Pengujian

Berdasarkan hasil implementasi yang telah dilakukan, sistem monitoring server real-time berbasis mobile menunjukkan bahwa seluruh fungsi utama dapat berjalan sesuai dengan rancangan yang telah ditetapkan. Pada pengujian tampilan dashboard, sistem berhasil menampilkan data monitoring server secara langsung pada aplikasi mobile, meliputi informasi CPU, RAM, disk storage, serta koneksi PostgreSQL. Data tersebut diperbarui secara berkala sesuai dengan interval pengiriman yang telah dikonfigurasi, sehingga administrator dapat memantau kondisi server secara real-time tanpa harus melakukan pengecekan langsung ke server.

Selanjutnya, pada pengujian deteksi anomali, sistem juga berhasil mengenali kondisi tidak normal pada server, khususnya pada lonjakan koneksi PostgreSQL. Ketika jumlah koneksi aktif melebihi ambang batas yang telah ditentukan, sistem secara otomatis mengklasifikasikan kondisi tersebut sebagai anomali dan mencatatnya sebagai event monitoring. Hasil ini menunjukkan bahwa mekanisme deteksi yang diterapkan pada backend mampu bekerja dengan baik dalam mengidentifikasi peningkatan beban database secara otomatis.

Pada pengujian pengiriman notifikasi, sistem berhasil mengirimkan peringatan melalui Telegram secara otomatis ketika anomali terdeteksi. Notifikasi tersebut berfungsi sebagai bentuk early warning agar administrator dapat segera mengetahui adanya kondisi abnormal pada server dan dapat melakukan penanganan lebih cepat. Dengan demikian, mekanisme peringatan dini ini dapat membantu mengurangi risiko keterlambatan dalam respon terhadap insiden.

Selain itu, pada pengujian penyimpanan data historis, seluruh data metrik dan event anomali berhasil disimpan ke dalam PostgreSQL Historical Database. Penyimpanan ini memungkinkan data monitoring untuk digunakan kembali sebagai bahan evaluasi, penelusuran kejadian, maupun analisis riwayat insiden di kemudian hari. Hal ini menunjukkan bahwa sistem tidak hanya berfungsi sebagai alat pemantauan real-time, tetapi juga sebagai media pencatatan histori monitoring yang terstruktur.

Terakhir, pada pengujian latensi data, sistem mampu menampilkan informasi monitoring dengan jeda yang masih berada dalam batas waktu yang ditentukan. Hal ini menunjukkan bahwa proses pengiriman dan penerimaan data berjalan secara cukup responsif, sehingga data yang tampil pada dashboard tetap relevan dengan kondisi server saat itu. Secara keseluruhan, hasil pengujian membuktikan bahwa sistem monitoring yang dibangun telah mampu menjalankan fungsi utamanya dengan baik, yaitu menampilkan data real-time, mendeteksi anomali, mengirim notifikasi, menyimpan histori, dan menjaga keterbaruan informasi pada aplikasi mobile.

Tabel III. Hasil list pengujian

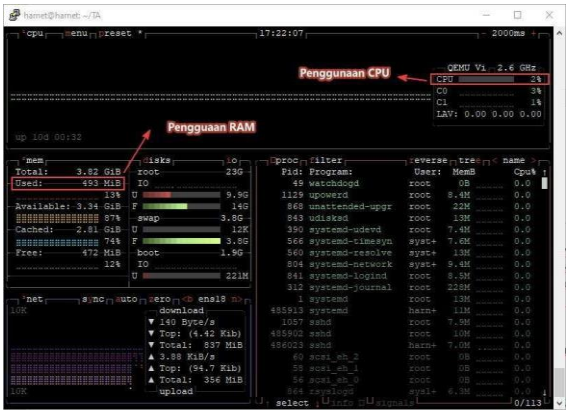
No.	Skenario Pengujian	Hasil yang diharapkan	Hasil Aktual	Status
1.	Menampilkan data monitoring server pada dashboard mobile .	Data CPU, RAM, disk storage, dan koneksi PostgreSQL tampil pada dashboard mobile.	Data monitoring berhasil ditampilkan pada dashboard mobile sesuai kondisi server.	Berhasil
2.	Mendeteksi anomali brute force SSH.	Sistem mendeteksi percobaan login gagal berulang dan menandainya sebagai anomali .	Sistem berhasil mendeteksi percobaan login gagal berulang pada log SSH dan mengklasifikasikannya sebagai anomali.	Berhasil
3.	Mendeteksi lonjakan koneksi PostgreSQL.	Sistem mendeteksi peningkatan koneksi PostgreSQL melebihi ambang batas yang ditentukan..	Sistem berhasil mendeteksi lonjakan koneksi PostgreSQL saat jumlah koneksi melebihi batas normal.	Berhasil
4.	Mengirim notifikasi Telegram saat anomali terjadi.	Notifikasi Telegram terkirim secara otomatis ketika sistem mendeteksi anomali.	Data monitoring dan anomali berhasil disimpan ke PostgreSQL Historical Database.	Berhasil
5.	Menyimpan data historis ke PostgreSQL.	Data metrik dan event anomali tersimpan ke database historis.	Data monitoring dan anomali berhasil disimpan ke PostgreSQL Historical Da	Berhasil

e-ISSN: 2548-6861

6.	Memastikan data tampil dengan latensi yang sesuai.	Data monitoring tampil dengan jeda yang sesuai dengan interval pengiriman.	Data monitoring tampil dengan latensi yang sesuai dan masih dalam batas waktu yang ditentukan.	Berhasil

Tabel IV. Hasil Pengukuran Sensor Golang Agent

No.	Kondisi Server Production	Penggunaan CPU	Penggunaan RAM
1.	Idle (Sensor Golang Nonaktif).	0-2%	499 MiB
2.	Berjalan Normal (Goroutine Aktif).	0-2%	526 MiB
3.	Stress CPU	90+ %	473 MiB
4.	Connection Spike	30+ %	606 MiB



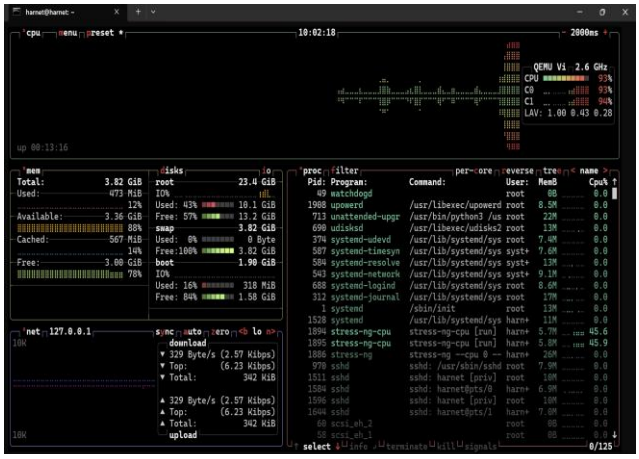
Gambar 12. Hasil pengukuran Sensor Golang tidak aktif

Saat aplikasi monitoring dan sensor Golang belum dinyalakan, penggunaan CPU server sangat rendah, yaitu di kisaran 0% sampai 2%, dengan RAM terpakai sebesar 499 MiB. Angka ini bisa dibilang sebagai modal awal (beban dasar) dari sistem operasi Ubuntu itu sendiri untuk menjalankan proses internalnya secara pasif, sebelum sistem monitoring kita mulai bekerja.

Gambar 13. Hasil pengukuran Sensor Golang aktif

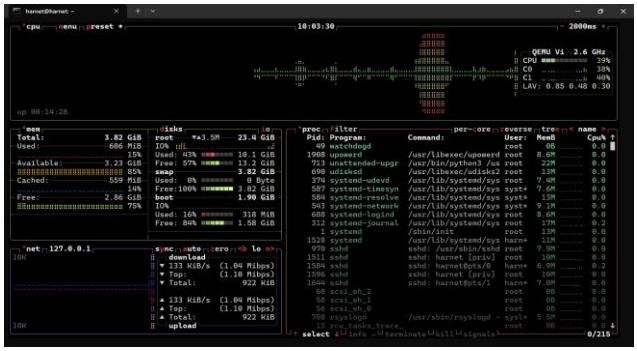
Saat fungsi sensor dan goroutine pada aplikasi Go mulai diaktifkan, penggunaan CPU ternyata tidak mengalami perubahan dan tetap stabil di angka 0% sampai 2%. Namun, ada kenaikan sedikit pada memori RAM sebesar 27 MiB, dari 499 MiB naik menjadi 526 MiB.

Hal ini membuktikan bahwa aplikasi berbasis Golang yang diimplementasikan di PT Dwansoft Global Indonesia ini sangat efisien dan ringan. Ketika goroutine aktif tapi belum menerima trafik atau serangan, dia hanya memesan sedikit ruang di RAM untuk bersiap-siap. Karena statusnya masih menunggu (waiting), prosesor tidak perlu bekerja keras untuk oper-operan tugas (context switching), sehingga penggunaan CPU server tetap aman dan sangat hemat.



Gambar 14. Hasil pengukuran uji coba stress CPU

Kondisi kritis yang tervisualisasi pada gambar inilah yang menjadi acuan pengujian sistem peringatan dini. Ketika beban prosesor menyentuh angka tersebut, sensor Golang Agent yang beroperasi di latar belakang akan langsung mendeteksi bahwa utilisasi telah melewati ambang batas (*threshold*) 90% yang dikonfigurasi. Deteksi otomatis terhadap kondisi *resource exhaustion* ini kemudian secara instan memicu *backend* untuk mengirimkan pesan darurat ke Telegram administrator

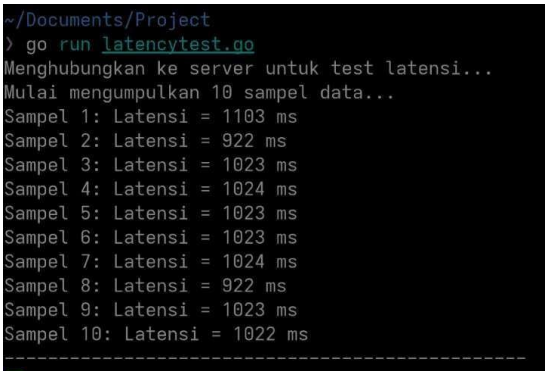


Gambar 15. Hasil pengukuran uji DB spike

Rekaman kondisi fluktuatif pada antarmuka pemantauan ini memvalidasi urgensi dari sensor backend yang dirancang. Sensor Golang tidak hanya bergantung pada metrik OS secara umum (seperti total CPU), melainkan secara spesifik melakukan query ke pg_stat_activity untuk menangkap momen kritis ketika batas koneksi aktif PostgreSQL terlampaui.

Tabel V. Hasil Pengukuran Latensi Transmisi Data

Segmen Jalur komunikasi	Rute transmisi data	Jumlah sampel	Latensi rata-rata	Keterangan
Waktu transmisi internal	Sensor agent ke Backend server	10	1010.90 ms	Sesuai Interval Sensor (1 Detik)
Waktu distribusi eksternal	Backend server ke Aplikasi mobile	10	1011.70 ms	Cepat & Stabil
Total latensi keseluruhan	Sensor agent ke Aplikasi mobile	10	1011.80 ms	Sangat Responsif



Gambar 16. Hasil pengukuran Sensor agent ke backend server

Segmen ini mengukur durasi waktu yang dibutuhkan untuk mengirimkan data metrik mentah (seperti utilisasi CPU, RAM, dan log keamanan) dari agen sensor yang beroperasi di Server Production menuju Backend Server utama.

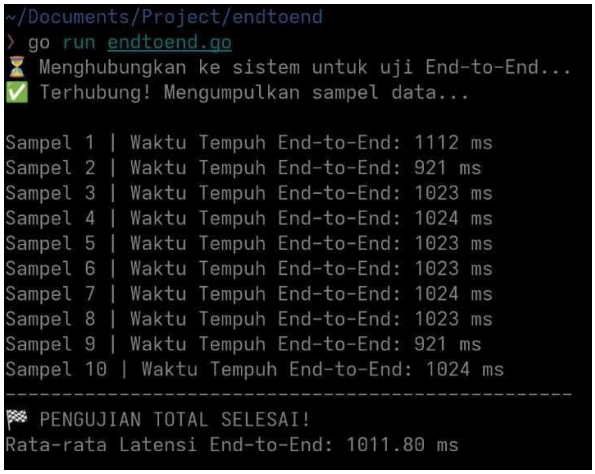
e-ISSN: 2548-6861

Bertujuan untuk mengevaluasi efisiensi pemrosesan internal pada tingkat server. Nilai latensi yang tinggi pada segmen ini akan mengindikasikan adanya hambatan (bottleneck) pada sisi server itu sendiri, misalnya akibat tingginya beban komputasi CPU atau kurang efisiennya algoritma pembacaan log pada agen pemantau.

Gambar 17. Hasil pengukuran backend server ke aplikasi mobile

Segmen ini mengukur durasi transmisi kelanjutan, yakni ketika Backend Server menyiarkan (push) data metrik tersebut melalui jaringan internet publik hingga mendarat di aplikasi mobile pada perangkat administrator.

Berfungsi sebagai parameter kontrol untuk mengevaluasi kualitas konektivitas jaringan eksternal (internet). Pemisahan metrik ini sangat penting untuk membuktikan secara empiris bahwa apabila terjadi keterlambatan (delay) pembaruan data di aplikasi, hal tersebut murni disebabkan oleh fluktuasi koneksi atau latensi Internet Service Provider (ISP) pada perangkat klien, bukan akibat malfungsi atau lambatnya sistem backend.



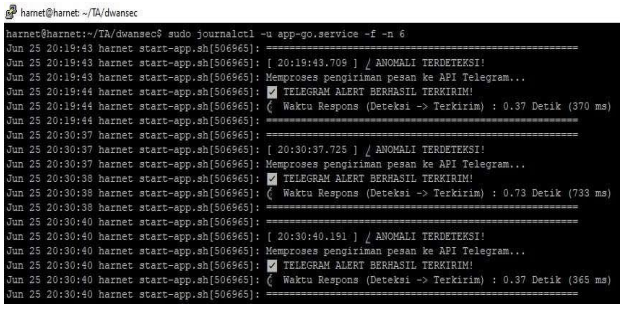
Gambar 18. Hasil pengukuran Sensor agent ke aplikasi mobile

Segmen ini merupakan akumulasi waktu perjalanan mutlak, yang dihitung dari titik milidetik pertama metrik direkam oleh Sensor Agent di Server Production, hingga data tersebut berhasil divisualisasikan pada layar aplikasi mobile.

Menjadi tolok ukur dan validasi utama terhadap klaim arsitektur pemantauan yang bersifat Real-Time. Metrik end-to-end ini memberikan bukti kuantitatif kepada penguji bahwa penggunaan arsitektur komunikasi WebSocket Bidirectional terbukti efektif dalam memangkas waktu tunggu data (overhead jaringan), sehingga sistem memenuhi standar operasional Early Warning System yang responsif dan presisi.

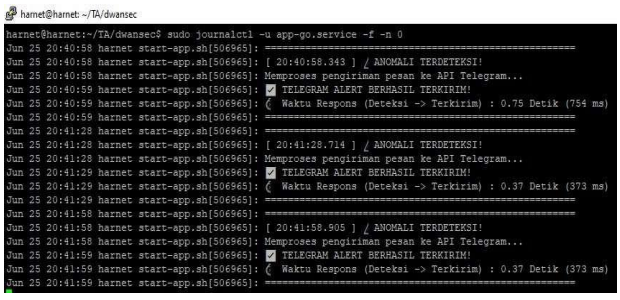
Tabel VI. Hasil Pengukuran Waktu Respons Notifikasi Telegram Alert

Simulasi penyerangan	Uji 1	Uji 2	Uji 3	Rata-rata detik
1 kali Gagal Login SSH	370 ms	733 ms	365 ms	0.49 detik
CPU Usage > 90%	754 ms	373 ms	373 ms	0.50 detik
Connection Spike PostgreSQL	755 ms	367 ms	395 ms	0.51 detik



Gambar 19. Hasil pengukuran waktu SSH

Hasil pengujian yang dilakukan sebanyak tiga kali repetisi mencatatkan waktu respons berturut-turut sebesar 370 ms, 733 ms, dan 365 ms. Dari ketiga sampel pengujian tersebut, diperoleh rata-rata kecepatan waktu respons pengiriman notifikasi sebesar 0.49 detik.



Gambar 20. Hasil pengukuran waktu penggunaan CPU

Hasil pemantauan tersebut mencatatkan waktu respons penyampaian peringatan (alert) ke Telegram secara berturut-turut sebesar 754 ms, 373 ms, dan 373 ms. Berdasarkan pengujian repetitif tersebut, akumulasi waktu rata-rata yang dibutuhkan backend untuk menyadari anomali hingga berhasil mengirimkan notifikasi kepada administrator adalah sebesar 0.50 detik.

Gambar 21. Hasil pengukuran waktu Connection Spike PostgreSQL

Hasil pencatatan tersebut menunjukkan durasi waktu pengiriman peringatan sebesar 755 ms, 367 ms, dan 395 ms untuk masing-masing iterasi pengujian. rata-rata waktu yang dibutuhkan sistem dari titik deteksi anomali hingga pesan peringatan dini (alert) sukses terkirim ke Telegram administrator adalah sebesar 0.51 detik.

IV. KESIMPULAN

Berdasarkan hasil implementasi dan pengujian yang telah dilakukan, dapat disimpulkan bahwa sistem monitoring server real-time berbasis mobile berhasil dibangun sesuai dengan rancangan yang telah ditetapkan pada tahap perancangan sebelumnya. Sistem ini mampu menjalankan fungsi utama dalam memantau kondisi server secara langsung melalui perangkat mobile, sehingga administrator dapat memperoleh informasi mengenai keadaan server tanpa harus melakukan pengecekan secara manual melalui akses langsung ke server. Dengan adanya penerapan komunikasi data secara real-time, informasi yang ditampilkan pada dashboard dapat diperbarui dengan cepat sesuai dengan kondisi server yang sedang berjalan.

Selain itu, sistem yang dibangun juga berhasil mengimplementasikan mekanisme deteksi anomali keamanan yang berfokus pada aktivitas brute force SSH dan lonjakan koneksi PostgreSQL. Mekanisme ini memungkinkan sistem untuk mengenali kondisi yang tidak normal berdasarkan log autentikasi dan parameter koneksi database yang telah ditentukan sebelumnya. Ketika sistem mendeteksi adanya anomali, informasi tersebut tidak hanya ditampilkan pada dashboard monitoring, tetapi juga dicatat ke dalam database historis sehingga dapat digunakan sebagai data pendukung untuk evaluasi dan penelusuran kejadian di kemudian hari. Hal ini menunjukkan bahwa sistem tidak hanya berfungsi sebagai alat pemantauan, tetapi juga sebagai media pencatatan riwayat insiden keamanan.

e-ISSN: 2548-6861

Di samping itu, sistem juga telah berhasil mengintegrasikan notifikasi otomatis melalui Telegram Bot API sebagai bagian dari mekanisme peringatan dini. Integrasi ini memberikan kemudahan bagi administrator dalam menerima informasi secara cepat ketika terjadi kondisi yang tidak normal pada server, sehingga tindakan penanganan dapat dilakukan lebih awal sebelum gangguan berkembang menjadi insiden yang lebih besar. Dengan demikian, sistem monitoring yang dirancang tidak hanya membantu mempercepat proses deteksi, tetapi juga meningkatkan respons terhadap potensi gangguan pada server.

Secara keseluruhan, hasil penelitian ini menunjukkan bahwa sistem monitoring server real-time berbasis mobile yang dikembangkan telah memenuhi tujuan penelitian, yaitu menyediakan sarana pemantauan server yang lebih efektif, memberikan deteksi anomali keamanan secara otomatis, menyimpan data historis untuk keperluan evaluasi, serta mengirimkan peringatan dini melalui Telegram. Sistem ini diharapkan dapat membantu administrator PT Dwansoft Global Indonesia dalam menjaga ketersediaan layanan server dan serta meningkatkan kecepatan dalam proses monitoring serta penanganan gangguan.

UCAPAN TERIMA KASIH

Dengan memanjatkan puji syukur ke hadirat Allah SWT, penulis dapat menyelesaikan penyusunan laporan penelitian dan tugas akhir ini dengan lancar. Segala kemudahan, kelancaran, dan pertolongan yang diberikan-Nya menjadi alasan utama penulis mampu melalui seluruh proses penyusunan hingga selesai.

Ucapan terima kasih yang paling utama penulis sampaikan kepada kedua orang tua tercinta, yang selalu memberikan doa, dukungan, kasih sayang, serta pengorbanan yang tiada henti. Segala bentuk perhatian dan motivasi dari kedua orang tua menjadi kekuatan besar bagi penulis dalam menyelesaikan perjalanan akademik ini.

Penulis juga mengucapkan terima kasih yang sebesar-besarnya kepada Bapak Nur Cahyono Kushardianto selaku dosen pembimbing, atas arahan, waktu, ilmu, serta masukan yang sangat membantu selama proses perancangan, penyusunan, dan evaluasi penelitian ini. Ucapan terima kasih juga disampaikan kepada seluruh civitas akademika Program Studi Rekayasa Keamanan Siber, Politeknik Negeri Batam, atas dukungan dan fasilitas yang telah diberikan selama masa perkuliahan.

Selain itu, penulis turut menyampaikan apresiasi kepada manajemen serta seluruh tim di PT Dwansoft Global Indonesia yang telah memberikan kesempatan untuk melaksanakan magang profesional dan penelitian. Secara khusus, terima kasih penulis sampaikan kepada Bapak Ridwan, Ibu Nita, dan Bang Rifan atas bimbingan, arahan teknis, serta pengalaman berharga yang diberikan selama pelaksanaan magang.

Penulis juga mengucapkan terima kasih kepada teman-teman RKS yang telah memberikan semangat, bantuan, kebersamaan, serta dukungan selama masa perkuliahan dan penyusunan tugas akhir ini. Kehadiran teman-teman RKS menjadi bagian penting dalam menjaga motivasi penulis hingga proses penyelesaian laporan ini.

DAFTAR PUSTAKA

[1] Nauha, U. N., dkk. (2025). Implementasi Bot Telegram untuk Monitoring Jaringan Berbasis ICMP Real-Time. Ilmu Komputer UMMETRO, 5(2), 45-56.

[2] Ridho, M. R., dkk. (2025). Peningkatan Keamanan SSH Server Berbasis Linux melalui Implementasi Fail2Ban. Jurnal Penelitian Multidisiplin Bangsa, 1(12), 2206–2214.

[3] Rahman, D., Amnur, H., & Rahmayuni, I. (2020). Monitoring Server dengan Prometheus dan Grafana serta Notifikasi Telegram. JITSI: Jurnal Ilmiah Teknologi Sistem Informasi, 1(4), 133–138.

[4] Abdullah, S., dkk. (2022). Implementasi Bot Telegram untuk Monitoring Jaringan dengan Security Policy. BUSITI Jurnal, 8(3), 112-120.

[5] Hidayat, M. (2023). Sistem Monitoring Jaringan Realtime Berbasis Backdoor dan Telegram Notification. Jurnal Intech Ar-Raniry, 7(1), 34-42.

[6] Pratama, A., & dkk. (2024). Implementasi Monitoring Keamanan Jaringan Deteksi Brute Force SSH dan DDoS. JATI ITN, 12(2), 78-89.

[7] Thohir, M. H. (2023). Perbandingan WebSocket pada Komunikasi Aplikasi Mobile Monitoring Server. Repository UB, 13459, 50-65.

[8] Ridho, M. R., dkk. (2025). Peningkatan Keamanan SSH Server Berbasis Linux melalui Implementasi Fail2Ban. Jurnal Penelitian Multidisiplin Bangsa, 1(12), 2206–2214.

[9] Achsan, H. (2023). Implementasi Sistem Pemantauan Jaringan Real-Time Berbasis WebSocket dan Telegram Bot. Jurnal Teknik Informatika BSI, 478001, 1-15.

[10] Ahmad, R., Putra, D., & Sari, M. (2025). Comparative Performance Benchmarking of WebSocket Libraries on Real-Time Applications in Golang and Node.js. Jurnal SINKRON, 10(2), 100-112.