

RESEARCH ARTICLE

Pengembangan *System Monitoring* dan Visualisasi Data Penggunaan Sumber Daya Pada *Server* Menggunakan *Prometheus* Dan *Grafana* Pada *Database* Terintegrasi di PT. Dirgantara

Bagus Bintang Fauzi, Muhammad Iqbal* and Hendar Rusman

Fakultas Ilmu Terapan, Universitas Telkom, Bandung, 40257, Jawa Barat, Indonesia

*Corresponding author: miqbal.staff.telkomuniversity.ac.id

Abstrak

Dalam Pengembangan *system* infrastruktur jaringan dan *server*, *Engineer Development* membutuhkan sebuah *system* untuk menjaga kualitas dan kestabilan performa *server* secara berkelanjutan, dalam hal ini *system monitoring* dan *alerting* sangat diperlukan untuk mendeteksi dan merespon gangguan pada *server* dengan cepat. Proses pengembangan akan meliputi beberapa tahapan instalasi dan konfigurasi yaitu *Prometheus*, *Node Exporter*, *Grafana* sebagai *dashboard* visualisasi data penggunaan CPU, *disk*, *memory* dan *network*, lalu terakhir adalah mengintegrasikan *Prometheus* dengan *system* alertingnya. Tujuan dari pembuatan *system monitoring* and *alerting* adalah memudahkan tim IT di PT. Dirgantara Indonesia untuk memantau performa *server* sebanyak 128 Vmware dalam satu *dashboard* yang dijalankan secara *real-time* dan menganalisis beban kerja *system* agar dapat meningkatkan efisiensi dan mencegah potensi gangguan pada *server* sejak dini. Hasil akhir dari pengembangan ini adalah dapat merancang *system* yang mampu memberikan informasi data *source* penggunaan *server* secara akurat dan *real-time*, serta memberikan notifikasi pada saat terjadi masalah pada *server*. *system* ini diharapkan dapat mendukung pengelolaan infrastruktur *server* yang lebih efisien di lingkungan PT. Dirgantara Indonesia.

Key words: *Dashboard*, *Monitoring*, *Alerting*, *Prometheus*, *Node Exporter*, *Grafana*.

Pendahuluan

PT. Dirgantara Indonesia adalah perusahaan manufaktur pesawat terbang terbesar di Asia Tenggara dan bagian dari holding BUMN DEFEND.ID. Untuk mendukung operasional sistem informasi dan infrastruktur *server*, perusahaan memiliki tim IT, termasuk divisi *Engineering Development* yang bertugas menjaga kestabilan *server* dan mencegah anomali. Saat ini, sistem monitoring *server* dan jaringan masih menggunakan *software* terpisah, seperti HP *Intelligent Management Center* (IMC) untuk jaringan, serta VCenter dan Netapp *Management* untuk penggunaan sumber daya di VMware. Sistem ini kurang efisien dan cukup mahal, sehingga dibutuhkan solusi monitoring baru yang lebih terintegrasi, akurat, dan mampu menampilkan data secara *real-time* dalam satu *dashboard*.

Selain itu, sistem juga perlu memberikan notifikasi jika terjadi gangguan. *Server* berperan penting dalam menyediakan akses data, dan pemantauan manual sering kesulitan mendeteksi masalah dengan cepat [1][2]. Untuk mengatasi kendala tersebut, dikembangkan sistem monitoring dan visualisasi yang efisien dan terintegrasi menggunakan *Prometheus*, *Grafana*, *Node Exporter*, dan Telegram. *Prometheus* mengumpulkan data dari *Node Exporter* dan menyimpannya dalam format *time-series*, sedangkan *Grafana* menampilkan visualisasi data

tersebut. Telegram digunakan untuk mengirim notifikasi otomatis 3060 detik setelah anomali terdeteksi, tergantung pada jaringan. Sistem ini terbukti efektif dalam mendeteksi dan merespons gangguan secara cepat [3], dan *Grafana* mendukung integrasi plugin dengan berbagai basis data [4]. Dengan keempat *tools* tersebut, tim IT dapat memantau penggunaan CPU, *disk*, *memory*, dan *network* secara *real-time*, serta mendapatkan notifikasi otomatis jika terjadi gangguan. Sistem juga memungkinkan analisis historis untuk mendukung pengambilan keputusan teknis yang lebih baik. Pengembangan ini diharapkan mampu menekan biaya, mengurangi ketergantungan pada *platform* berbayar, serta meningkatkan efisiensi dan keandalan infrastruktur *server* di PT. Dirgantara Indonesia.

Tinjauan Penelitian

Tinjauan Umum/ Deksripsi dan Skenario dan Perancangan Sistem

Seiring meningkatnya ketergantungan perusahaan pada sistem informasi dan layanan digital, kebutuhan akan monitoring dan *alerting* performa *server* yang andal dan terintegrasi menjadi krusial. Monitoring *server* adalah proses pengawasan komponen dan kinerja sistem, seperti penggunaan CPU, *memory* (RAM), aktivitas aplikasi (*disk*), dan

bandwidth jaringan (*network*). Sistem ini memberikan alert saat terjadi masalah pada protokol, serta memberi informasi spesifik secara otomatis kepada administrator jaringan [5]. Komponen penting dari monitoring adalah *alerting*, yaitu mekanisme peringatan otomatis ketika parameter melebihi *threshold* atau terjadi kondisi abnormal. Hal ini memungkinkan tim IT untuk merespons sebelum gangguan meluas.

Integrasi monitoring dan *alerting* bertujuan menciptakan pengawasan proaktif. Dalam implementasinya, digunakan *tools* seperti *Prometheus Alert Manager* yang mengelola aturan alert dan mengirim notifikasi melalui email, Telegram, atau sistem internal. Salah satu teknologi yang umum digunakan adalah *Prometheus*, *tool open-source* untuk monitoring dan *alerting* yang dikembangkan oleh *SoundCloud*. *Prometheus* menggunakan metode *time-series database* untuk menyimpan *metrics* dalam interval waktu tertentu dan mendukung bahasa kueri *PromQL*. Ia bekerja berdasarkan pull model, mengumpulkan data dari target melalui HTTP endpoint. Untuk visualisasi, digunakan *Grafana*, yang menyajikan data dari *Prometheus* dalam bentuk *dashboard* interaktif. Kombinasi keduanya banyak diadopsi industri karena mampu menyajikan data informatif secara *real-time* maupun historis.

Dalam penelitian ini, monitoring difokuskan pada empat parameter utama:

- Penggunaan CPU (beban kerja prosesor)
- Penggunaan *memory* (efisiensi RAM)
- Konsumsi *bandwidth* jaringan (aktivitas masuk dan keluar)
- Aktivitas aplikasi (*disk* dan sumber daya)

Data diperoleh dari *Node Exporter*, agen *Prometheus* yang berjalan di *server* target dan mengeksport metrik sistem operasi. Data disimpan oleh *Prometheus* dan divisualisasikan melalui *Grafana*. Untuk analisis lebih lanjut, data historis juga dicatat dalam *database* seperti *InfluxDB*, termasuk untuk keperluan *forecasting*. Dengan pendekatan ini, sistem tidak hanya memantau kondisi saat ini, tetapi juga memungkinkan analisis jangka panjang, evaluasi beban kerja, dan prediksi kebutuhan sumber daya di lingkungan *server* PT Dirgantara Indonesia.

Tinjauan Khusus 1 (Konsep Monitoring System)

Salah satu komponen penting dalam *system monitoring* adalah visualisasi data yang informatif dan *real-time*. Visualisasi ini bertujuan untuk mempermudah pemahaman terhadap kondisi *system* melalui representasi grafis dari data yang dikumpulkan. Dalam Karya Ilmiah ini, digunakan *Grafana* sebagai *platform* visualisasi karena kemampuannya yang fleksibel, mendukung berbagai jenis *datasource*, dan tampilannya yang interaktif serta mudah di *customize*.

Tinjauan Khusus 2 (Konsep Alerting)

Alerting merupakan komponen penting dalam sistem monitoring yang memungkinkan pengiriman notifikasi otomatis saat suatu metrik melebihi ambang batas tertentu. Dalam sistem ini, *Prometheus* bekerja sama dengan *Alertmanager* untuk menangani notifikasi tersebut. Salah satu media yang digunakan dalam penelitian ini adalah Telegram, karena sifatnya yang cepat, mudah diakses, dan mendukung bot otomatis [3]. Dengan mengintegrasikan *Alertmanager* dan Telegram Bot, sistem dapat mengirim pesan peringatan secara *real-time* kepada administrator ketika terjadi anomali seperti penggunaan CPU yang tinggi, memori hampir penuh, atau aplikasi yang tidak merespons. Hal ini memungkinkan tim IT di PT. Dirgantara Indonesia untuk melakukan tindakan mitigasi dengan lebih cepat dan efisien.

Tinjauan Khusus 3 (*Grafana*)

Grafana merupakan sebuah *platform open-source* terkemuka yang dirancang khusus untuk visualisasi dan pemantauan data berbasis web [6]. *Grafana* merupakan *platform open-source* yang digunakan

untuk memvisualisasikan dan memantau data berbasis web. *Grafana* mempunyai dukungan untuk menggunakan berbagai macam tipe *data-source* data seperti *Graphite*, *Prometheus*, *Elasticsearch*, *OpenTSDB*, dan *InfluxDB*. Seperti halnya *Prometheus*, *Grafana* juga merupakan salah satu proyek berbasis sumber terbuka yang sangat populer saat ini. *Prometheus server* akan mengambil data target pada interval waktu yang telah ditentukan pada konfigurasi dari target tertentu dan menyimpannya dalam bentuk basis *data time series*.

Kode sumber dari perangkat lunak *Prometheus* bisa diakses dan diunduh di halaman GitHub-nya [7]. Selain visualisasi data *real-time*, *Grafana* juga mendukung analisis historis yang memungkinkan administrator melacak tren penggunaan sumber daya dalam periode tertentu. Fitur ini sangat membantu dalam mendeteksi pola penggunaan yang dapat dijadikan dasar dalam optimasi kinerja *server* [8]. Dengan demikian, *Grafana* bukan hanya alat visualisasi, tetapi juga alat analisis yang membantu dalam pengambilan keputusan berbasis data.

Tinjauan Khusus 4 (*Prometheus*)

Prometheus adalah toolkit pemantauan dan peringatan *system* sumber terbuka yang awalnya dibangun di *SoundCloud*. Sejak awal pembuatannya pada tahun 2012, banyak perusahaan dan organisasi yang mengadopsi *Prometheus*, dan proyek ini memiliki komunitas pengembangan dan pengguna yang sangat aktif [7]. *Prometheus* mengumpulkan metrik dari data *resource*, baik secara langsung atau melalui *gateway push*. *Prometheus* menggunakan metrik untuk pekerjaan yang jangka pendek. Metrik mengumpulkan data yang diambil dari exporter yang telah di-*install* dan dapat digunakan untuk memberi peringatan [9]. File konfigurasi utama yang digunakan oleh *Prometheus* untuk menentukan bagaimana *system* melakukan pengambilan (*scraping*) data dari berbagai target monitoring adalah *Prometheus.yml*.

Dalam *Prometheus.yml*, pengguna dapat menentukan *job_name* yang merepresentasikan kelompok target tertentu, seperti *node-exporter*, kemudian menetapkan alamat IP atau *hostname* dari *server* yang akan dimonitor melalui bagian *targets*. Selain itu, konfigurasi ini juga dapat mencakup parameter tambahan seperti *label*, *timeouts*, dan *rule evaluation* untuk kebutuhan *alerting* (jika digunakan). File ini bersifat fleksibel dan mudah disesuaikan, sehingga sangat penting dalam arsitektur *Prometheus* untuk memastikan bahwa metrik yang relevan dapat dikumpulkan secara teratur dan akurat dari berbagai sumber data [10]. Contoh isi dari file *Prometheus.yml* sebagai konfigurasi penargetan *client*, sebagai berikut:

"global:

- *scrape_interval*: 15s # Interval pengambilan data dari target

scrape_configs:

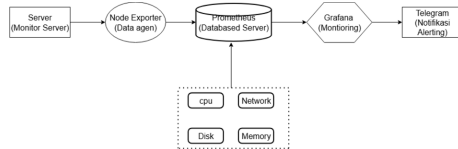
- *job_name*: 'node_exporter'
- static_configs:
- *targets*: ['localhost:9100'] # Ganti dengan IP VM atau *server*

Tinjauan Khusus 5 (*PromQL*)

PromQL (*Prometheus Query Language*) adalah bahasa *query* yang digunakan untuk mengambil dan memproses data dari *time-series database Prometheus*. *PromQL* memungkinkan pengguna untuk mengeksekusi berbagai jenis *query*, baik itu untuk melihat nilai saat ini, menghitung rata-rata, tren, maupun untuk membuat kondisi alert. Contoh *query* sederhana yang dapat dilihat pada Tabel 1. *Query* di atas digunakan untuk menghitung laju penggunaan CPU, *Disk*, *Memory*, dan *Network*. *PromQL* memiliki fungsi-fungsi seperti *rate()*, *avg()*, *sum()*, dan *max()* yang sangat berguna dalam analisis performa *system*.

Table 1. Query Sederhana

Nama Data	Query Dasar
CPU	node_cpu_seconds_total
Disk	node_filesystem_size_bytes
Memory	node_memory_MemTotal_bytes
Network	node_network_receive_bytes_total



Gambar 1. Blok System Monitoring

Tinjauan Khusus 5 (Node Exporter)

Node Exporter adalah komponen tambahan yang digunakan oleh *Prometheus*. Alat ini berfungsi untuk mengumpulkan data seperti penggunaan CPU, memori, disk I/O, jaringan, dan berbagai metrik lainnya yang berkaitan dengan kinerja infrastruktur [6]. Alat ini biasanya digunakan untuk mengumpulkan metrik seperti penggunaan CPU, *memory*, I/O, *network traffic* [11]. *Node Exporter* bekerja dengan mengekspos metrik melalui *HTTP endpoint*, biasanya di port 9100, yang kemudian di-scrape oleh *Prometheus*. Dalam implementasi ini, *Node Exporter* diinstal pada *server* yang ingin dimonitor. Setelah aktif, *Prometheus* dikonfigurasi untuk mengambil data dari alamat IP dan *port Node Exporter* melalui file *prometheus.yml*. Data yang dikumpulkan oleh *Node Exporter* bersifat granular dan sangat berguna untuk analisis performa *system* secara realtime melalui *dashboard Grafana*. Contoh endpoint metrik *Node Exporter*: <http://localhost:9100>.

Tinjauan Khusus 5 (DDoS)

Distributed Denial of Service (DDoS) adalah salah satu bentuk serangan siber yang bertujuan untuk melumpuhkan layanan atau *system* dengan cara membanjiri *server*, jaringan, atau aplikasi dengan lalu lintas (*traffic*) yang sangat tinggi. Serangan ini dilakukan dari banyak sumber (terdistribusi), sehingga lebih sulit untuk ditangkal dibandingkan dengan serangan *DoS (Denial of Service)* biasa yang hanya berasal dari satu sumber. Serangan *Distributed Denial of Service (DDoS)* merupakan salah satu jenis serangan paling berbahaya yang dapat membuat layanan daring tidak dapat diakses oleh pengguna yang sah [12]. Dalam konteks *system monitoring* dan *alerting*, simulasi serangan *DDoS* digunakan sebagai metode pengujian untuk mengukur kemampuan *system* dalam mendeteksi lonjakan *traffic* atau penggunaan sumber daya secara abnormal [13]. Hal ini penting sebagai bentuk evaluasi performa dan efektivitas *system* dalam menghadapi potensi ancaman nyata di lingkungan *server* perusahaan.

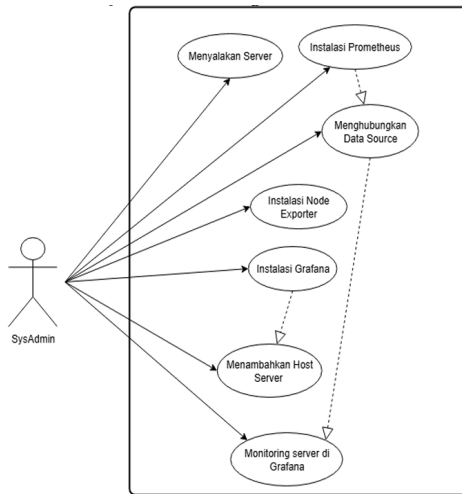
Metode Penelitian

Arsitektur *system monitoring* 1 ini dirancang dengan pendekatan *client-server* yang terintegrasi menggunakan *Prometheus* sebagai *database time-series* dan *Grafana* sebagai *platform visualisasi*. *System* ini bertujuan untuk memantau performa *server* secara *real-time* terhadap metrik penting seperti CPU, *Memory*, *Disk*, dan *Network usage* agar memudahkan admin dalam pengelolaan infrastruktur *server*. *System* ini terdiri dari empat komponen utama, yaitu:

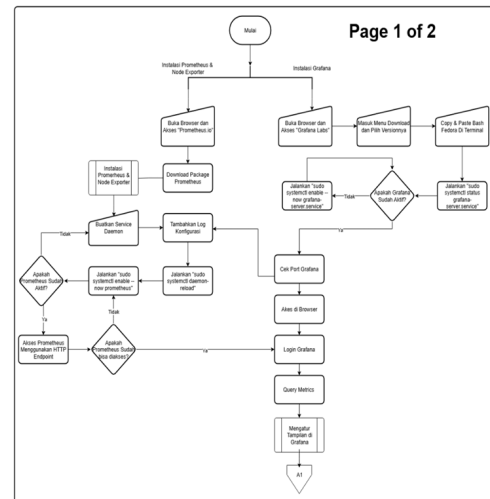
- Server (Monitoring Host)**
Server adalah perangkat utama yang dipantau dalam *system* ini. Di dalam *server* ini berjalan berbagai proses *system* yang mengonsumsi sumber daya seperti CPU, *memory*, *disk*, dan jaringan. Performa dari *server* ini menjadi objek utama dalam proses monitoring. *Server* ini juga menjadi tempat di mana *Node Exporter* diinstal untuk mengambil data metrik *system*.
- Node Exporter**
Node Exporter merupakan agen ringan yang berjalan di dalam *server* dan bertugas untuk mengumpulkan metrik performa seperti penggunaan CPU, *memory*, *disk I/O*, dan lalu lintas jaringan. *Node Exporter* menyediakan data dalam format yang kompatibel dengan *Prometheus* melalui *HTTP endpoint*. *Node Exporter* bertindak sebagai jembatan antara *system operasi server* dengan *Prometheus*, sehingga memungkinkan *Prometheus* untuk menarik (*scrape*) data secara periodik.
- Prometheus (Database Server)**
Prometheus berfungsi sebagai *server database timeseries* yang mengambil dan menyimpan data dari *Node Exporter* dalam bentuk *time-series*. *Prometheus* melakukan proses *scraping* terhadap data yang disediakan oleh *Node Exporter* sesuai interval waktu yang telah ditentukan. Selain itu, *Prometheus* juga mendukung *query* bahasa PromQL untuk melakukan agregasi, filter, dan manipulasi data metrik. Data yang sudah dikumpulkan akan digunakan untuk keperluan pemantauan maupun analisis lebih lanjut, termasuk peramalan (*forecasting*). data merupakan catatan atas kumpulan fakta dunia nyata yang mewakili objek seperti manusia, barang, hewan, konsep, peristiwa dan lain sebagainya yang diwujudkan dalam bentuk huruf, angka, simbol, gambar, teks, bunyi atau kombinasi lainnya [14].
- Grafana (Monitoring)**
Grafana adalah antarmuka visualisasi yang terhubung ke *Prometheus* sebagai sumber datanya. Melalui *Grafana*, administrator dapat membuat *dashboard* interaktif yang menampilkan grafik dan indikator performa dari *server* secara *real-time*. *Grafana* juga dapat dikonfigurasi untuk menampilkan hasil *forecasting* terhadap data metrik, seperti prediksi penggunaan CPU dalam 7 hari ke depan, dengan memanfaatkan integrasi *Python script* atau *plugin forecasting*. Ini memberikan nilai tambah dalam pengambilan keputusan dan perencanaan kapasitas *server*.
- Telegram (Notifikasi Alerting)**
Telegram Berfungsi sebagai *platform penerima alert* dari *Grafana* pada saat sumber daya *server* sudah mencapai batas *threshold* yang sudah ditentukan oleh administrator berdasarkan kebutuhan yang diinginkan. *System alerting* ini sangat penting dalam menjaga performa *server* dan dapat mengetahui aktivitas di dalam *server* ketika ada anomali yang terjadi seperti lonjakan data atau terjadinya *downtime*, sehingga administrator dapat mengatasi masalah tersebut.

Use Case System Monitoring

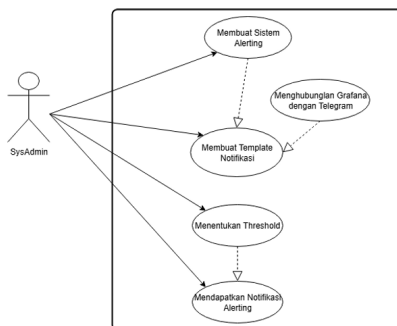
Gambar 2 *use case monitoring* menggambarkan alur kerja *system monitoring server* menggunakan *Prometheus* dan *Grafana* yang dilakukan oleh seorang *System*. Proses dimulai dari menyalakan *server*, menginstal *Prometheus*, *Grafana*, dan *Node Exporter*, hingga menghubungkan data source dan menambahkan *host* ke *Grafana*. Setelah itu, *System* dapat melakukan monitoring *server* secara visual melalui *Grafana*.



Gambar 2. Usecase Monitoring



Gambar 4. Instalasi System Monitoring Page 1 of 2



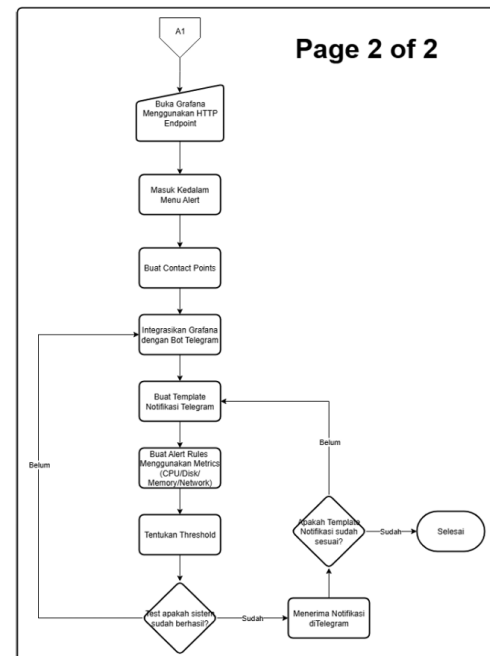
Gambar 3. Usecase Alerting

Use Case System Alerting

Gambar 3 *use case monitoring* ini menggambarkan alur kerja membangun *system alerting* untuk mendeteksi anomali atau kondisi tertentu pada *server*. *System* menghubungkan *Grafana* dengan *Telegram*, membuat template notifikasi, dan menentukan *threshold*. Jika kondisi yang ditentukan terpenuhi, *system* akan mengirimkan notifikasi secara otomatis melalui *Telegram*. Diagram ini menunjukkan pentingnya integrasi antar komponen dan otomatisasi untuk meningkatkan responsivitas dan efisiensi dalam pengelolaan infrastruktur TI.

Proses Pengerjaan (System Monitoring)

Flowchart pada gambar 4 "Page 1 of 2" menggambarkan tahapan instalasi dan konfigurasi sistem monitoring berbasis *Prometheus*, *Node Exporter*, dan *Grafana* untuk *server* di PT Dirgantara Indonesia. Proses dimulai dengan mengakses situs resmi masing-masing *tools*, mengunduh dan menginstal paket sesuai sistem operasi Fedora melalui terminal, lalu mengatur *service daemon* agar berjalan otomatis saat sistem menyala. Setelah *Prometheus* aktif dan dapat diakses melalui *HTTP endpoint*, proses dilanjutkan dengan instalasi *Grafana*, pengecekan status layanan, hingga akses ke *dashboard* melalui browser. Tahap akhir meliputi penambahan data *source* dari *Prometheus* ke *Grafana* dan visualisasi metrik seperti *CPU*, *memory*, *disk*, dan *network* dalam bentuk grafik interaktif.



Gambar 5. Instalasi System Alerting Page 2 of 2

Proses Pengerjaan (System Alerting)

Flowchart "Page 2 of 2" ini menjelaskan tahapan konfigurasi *alerting* di *Grafana*, dimulai dari membuka *Grafana* melalui *HTTP endpoint*, masuk ke menu *Alert*, dan membuat *contact points*. Selanjutnya, *Grafana* diintegrasikan dengan bot *Telegram* untuk mengirim notifikasi. Pengguna kemudian membuat *template* notifikasi *Telegram*, menentukan *alert rules* berdasarkan metrik (*CPU*, *disk*, *memory*, *network*), serta menetapkan *threshold*. Setelah diuji, sistem akan mengirim notifikasi ke *Telegram*. Jika *template* dan notifikasi sudah sesuai, proses dinyatakan selesai; jika belum, dilakukan penyesuaian dan pengujian ulang. Ask ChatGPT

Table 2. Standarisasi Spesifikasi Server

Komponen	Spesifikasi
Storage	500 GB
CPU	8 Core
Ram	8 GB
Layanan Tambahan	Prometheus, Node Exporter, Grafana, dan Alertmanager.

Standarisasi Spesifikasi Server

Spesifikasi server berikut merupakan standarisasi internal yang dibutuhkan untuk mendukung *system monitoring* dan pengolahan data secara optimal, khususnya dalam mengelola metrik performa dari banyak server secara *real-time* terlihat pada tabel 2. PT. Dirgantara Indonesia memiliki total 128 server, di mana sekitar 50 server aktif digunakan untuk kebutuhan aplikasi, dan sekitar 4 hingga 6 server dalam kondisi tidak aktif. Sisanya difungsikan untuk pengolahan data internal perusahaan, termasuk proses komputasi, analisis, serta penyimpanan data penting. Server yang digunakan untuk *system monitoring* ini harus mampu menangani volume data metrik dari seluruh server yang aktif dan terhubung. Oleh karena itu, spesifikasi minimum seperti CPU 8 core dan RAM 8 GB diperlukan untuk menjamin proses pengumpulan dan pengolahan data berjalan lancar. Penyimpanan 500 GB digunakan untuk menyimpan *log system*, konfigurasi, serta data historis dari metrik server. Selain itu, koneksi jaringan yang stabil dan cepat diperlukan agar proses monitoring dan *alerting* dapat dilakukan secara *real-time* dan tanpa hambatan.

Hasil dan Pembahasan

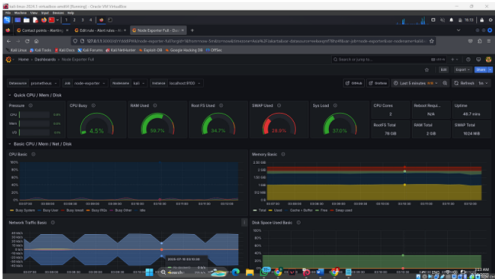
Implementasi *system monitoring* dan *alerting* ini dilakukan secara bertahap, dimulai dari proses instalasi *Prometheus* sebagai data scraper utama, pemasangan *Node Exporter* pada setiap server target, serta penyusunan *dashboard Grafana* sebagai media visualisasi data performa server [8]. *System* ini memungkinkan pemantauan metrik penting seperti penggunaan CPU, *memory*, *disk*, dan *bandwidth* jaringan secara *real-time*. Selain itu, *system* juga terintegrasi dengan fitur *alerting* melalui Telegram yang akan mengirimkan notifikasi otomatis apabila terjadi lonjakan performa, khususnya pada penggunaan CPU yang melebihi *Threshold* 75%.

Pengujian System Fungsional

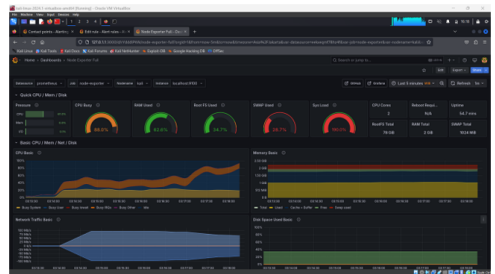
Adapun hasil dari pengujian *system* fungsional, hasil lengkapnya disajikan pada tabel 3.

Metode Pengujian Stress Testing

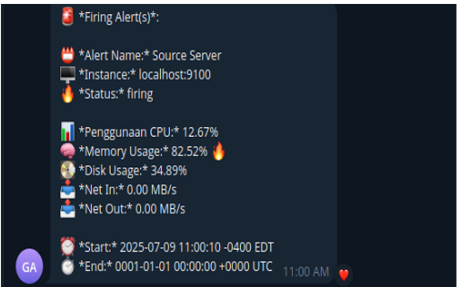
Metode pengujian pada gambar 6 & 7 yang digunakan dalam pengembangan *system* ini mencakup pengujian simulasi serangan DDoS (*Distributed Denial of Service*) ringan terhadap server target. Tujuan dari pengujian ini adalah untuk mengetahui sejauh mana *system monitoring* dan *alerting* yang dibangun mampu mendeteksi lonjakan performa, terutama pada penggunaan CPU, sebagai akibat dari serangan tidak normal yang membebani *system* secara berlebihan dalam waktu singkat [12]. Jika penggunaan CPU melebihi nilai ambang batas (*threshold*) yang telah ditentukan, yaitu 75%, maka *system alerting* yang terhubung dengan Telegram Bot API akan secara otomatis mengirimkan notifikasi kepada administrator [15]. Berikut adalah tabel 4 pengujian stabilitas *system monitoring* selama 5 menit dengan *refresh*



Gambar 6. Data Sebelum Pengujian



Gambar 7. Data Sesudah Pengujian



Gambar 8. Notifikasi Telegram

setiap 1 menit sekali. Pengujian ini dilakukan untuk mengukur ketahanan *system* saat menerima beban tinggi, seperti saat terjadi serangan DDoS. Selama pengujian berlangsung, *Prometheus* terus melakukan scraping data, dan *dashboard Grafana* tetap menampilkan data secara akurat tanpa *error*. *Alert* dari *Grafana* juga tetap aktif dan dikirim ke Telegram secara konsisten saat kondisi *threshold* tercapai. Hal ini menunjukkan bahwa sistem *alerting* berfungsi dengan baik dan responsif terhadap perubahan parameter sistem. Hasil pada Gambar 8 memperlihatkan notifikasi *alert* dari aktivitas pada server yang berhasil diterima melalui Telegram, menandakan integrasi berjalan lancar dan dapat diandalkan dalam mendeteksi serta memberi peringatan dini terhadap anomali sistem.

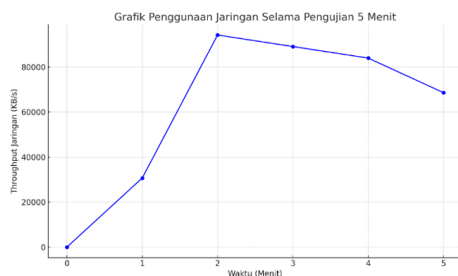
Hasil pengujian menunjukkan bahwa *system monitoring* dan *alerting* ini berjalan stabil, responsif, dan dapat memberikan peringatan secara tepat waktu, sehingga layak diterapkan dalam lingkungan operasional yang membutuhkan pengawasan *system server* secara *real-time*. Adapun grafik *traffic network* selama waktu pengujian untuk memudahkan dalam membaca tren, disajikan dalam bentuk visualisasi berikut. Grafik ini merepresentasikan perubahan *traffic* jaringan

Table 3. Pengujian Fungsional

No	Nama Pengujian	Tujuan Pengujian	Hasil Pengujian	Kesimpulan
1	Login ke <i>Dashboard Grafana</i>	Memastikan pengguna dapat login dengan kredensial default	Pengguna berhasil login ke <i>Grafana</i>	<i>System login</i> berfungsi dengan baik
2	Eksplorasi Panel CPU <i>Memory</i>	Melihat reaksi pengguna terhadap visualisasi performa	Pengguna dapat membaca grafik, mengganti waktu, melihat warna indikator	UI mudah dipahami, navigasi user-friendly
3	mulasi Serangan DDoS	Menguji apakah alert dikirim saat CPU melebihi threshold	Telegram menerima alert saat CPU 75%, <i>Grafana</i> menampilkan warna merah	Fitur alert dan visualisasi <i>textitthreshold</i> bekerja sempurna
4	Pengujian Notifikasi Telegram	Memastikan notifikasi diterima oleh pengguna ketika kondisi abnormal	Semua pengguna yang tergabung di grup Telegram menerima alert dari <i>Prometheus</i>	Integrasi Telegram dan alert <i>system</i> berjalan dengan baik

Table 4. Hasil Pengujian *System*

Menit Ke	CPU Usage (%)	Memory Usage (%)	Disk Usage (%)	Jaringan (Throughput)
0 (Awal)	4.5	59.7	34.7	37 KB/s
1	37.4	61.4	34.7	30MB/s
2	53.3	62.1	34.7	92MB/s
3	55.7	63.4	34.7	87MB/s
4	72.3	63.6	34.7	82MB/s

**Gambar 9.** Traffic Throughput

(dalam satuan MB/s) selama durasi pengujian lima menit, dengan interval pemantauan setiap satu menit sekali. Visualisasi ini membantu dalam mengidentifikasi pola lonjakan atau anomali *traffic* yang dapat mengindikasikan aktivitas mencurigakan seperti serangan DDoS, pada gambar 9.

Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian, sistem monitoring dan visualisasi performa *server* yang dikembangkan berhasil memenuhi seluruh tujuan yang ditetapkan. Sistem ini mampu memantau lebih dari 128 *Virtual Machine* di lingkungan PT Dirgantara Indonesia dengan metrik penting seperti CPU, *memory*, *disk*, dan *network* yang ditampilkan secara *real-time* melalui *dashboard Grafana*. Fitur *alerting* otomatis juga telah berfungsi dengan baik, mengirimkan notifikasi ke Telegram saat terjadi anomali melebihi *threshold*, memungkinkan tim IT melakukan mitigasi lebih cepat. Dalam pengembangannya, sistem memanfaatkan *tools open-source* seperti

Prometheus, *Node Exporter*, dan *Grafana* pada sistem operasi Linux Fedora untuk proses pengambilan data, penyimpanan *time series*, visualisasi, serta notifikasi. Secara keseluruhan, sistem ini mendukung operasional IT secara efisien, mudah diawasi, hemat biaya, dan sangat layak untuk diimplementasikan serta dikembangkan lebih lanjut.

Daftar Pustaka

- Harsono H. Faktor-Faktor yang Mempengaruhi Sistem Informasi Berbasis Komputer: Sistem Operasi, Server, dan Programmer (Literature Review Executive Support System for Business). *Jurnal Manajemen Pendidikan dan Ilmu Sosial*. 2022;3(2).
- Tasrif E, Huda A, Saputra HK, Mubai A. Design of Server Performance Monitoring Application Integrated Administration Service System in Electronic Engineering Department. In: *Journal of Physics: Conference Series*. vol. 1387. IOP Publishing; 2019. p. 012029.
- Saputra MYE, Noprianto, Arief SN, Wijyaningrum VN, Syaifudin YW. Real-Time Server Monitoring and Notification System with Prometheus, Grafana, and Telegram Integration. In: *Proceedings of the 2024 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETIS)*; 2024. p. 1808-13.
- Febriana RM. Implementasi Sistem Monitoring Menggunakan Prometheus dan Grafana. In: *Seminar Nasional Telekomunikasi dan Informatika*. vol. 13; 2020. p. 164-9.
- Sulaeman YJ. Implementasi Network Monitoring dan Notifikasi Sistem di PT XYZ Menggunakan Zabbix. *Jurnal Instrumentasi dan Teknologi Informasi (JITI)*. 2022;4(1):1-7.
- Rasyidi B, Pratama F. Sistem Monitoring Server di PT. XYZ Media Indonesia Berbasis Grafana dan Prometheus. *MALCOM: Indonesian Journal of Machine Learning and Computer Science*. 2024;4(4):1456-65.
- Amirudin MZ, Fahmi R, Utami E, Mustafa MS. Evaluasi Penggunaan Prometheus dan Grafana untuk Monitoring Database MongoDB. *Jurnal Informatika Polinema*. 2021;7(2):43-50.
- Siddiqui I, Pandey A, Jain S, Kothadia H. Comprehensive Monitoring and Observability with Jenkins and Grafana: A Review of Integration Strategies, Best Practices, and Emerging Trends. In: *2023 7th International Conference on Computing Methodologies and Communication (ICCMC)*. IEEE; 2023. .
- Rahman D, Amnur H, Rahmayuni I. Monitoring Server dengan Prometheus dan Grafana serta Notifikasi Telegram. *JITSI: Jurnal Ilmiah Teknologi Sistem Informasi*. 2020;1(4):133-8.
- Nurrohman I. Perancangan dan Implementasi Sistem Pemantauan Jaringan Berbasis Prometheus dan Grafana di SMK Harapan Bangsa. *Sekolah Tinggi Teknologi Terpadu Nurul Fikri*; 2024.

-
11. Rishwan RM, Nurkifli EH, Solehudin A. Analisis Perbandingan Performa Server WebSocket dengan Menggunakan Payload JSON, Binary Serialization, dan Protobuf Menggunakan Metode Load Testing. *JATI (Jurnal Mahasiswa Teknik Informatika)*. 2025;9(1):706-12.
 12. Nugroho A. Analisis Keamanan Website Terhadap Serangan DDoS. *Jurnal Keamanan Informasi*. 2021;6(2):78-85.
 13. Ahmim A, Maazouzi F, Ahmim M, Namane S, Ben Dhaou I. Distributed Denial of Service Attack Detection for the Internet of Things Using Hybrid Deep Learning Model. *IEEE Access*. 2023;11:119862-75.
 14. Rachmadi T. *Sistem Basis Data*. vol. 1. Tiga Ebook; 2020.
 15. Putra MA, Sari R. Sistem Monitoring Serangan pada Mikrotik Berbasis Bot Telegram. *Jurnal Ilmiah Manajemen dan Informatika*. 2022;8(1):45-52.