Rancang Bangun Dashboard Monitoring Data Center Infrastructure Management (Dcim) Yang Terintegrasi Grafana Alert Dan Bot Telegram Dengan Arsitektur Kubernetes Studi Kasus Pt. Pelayaran Nasional Indonesia (Pelni)

1st Inaz Nadhira
Fakultas Ilmu Terapan
Universitas Telkom
Bandung, Indonesia
nadhirainaz@telkomuniversity.ac.id

2nd Muhammad Iqbal
Fakultas Ilmu Terapan
Universitas Telkom
Bandung, Indonesia
miqbal@telkomuniversity.ac.id

3rd Chandra Mahendra Putra
Pt. Pelayaran Nasional Indonesia
Bandung, Indonesia
chandra@pelni.co.id

Abstrak **- Proyek Akhir ini bertujuan mengembangkan sistem monitoring terintegrasi untuk aplikasi Data Center Infrastructure Management (DCIM) dalam lingkungan kubernetes. Sistem menggunakan prometheus untuk menampilkan metrik, grafana untuk visualisasi data, serta grafana alerts dan bot telegram untuk notifikasi real-time. Metode penelitian mencakup analisis kebutuhan, perancangan dashboard grafana, konfigurasi prometheus, penyesuaian grafana alerts, dan integrasi dengan bot telegram. Uji coba dan validasi dilakukan untuk memastikan fungsionalitas sebelum penerapan sistem. Pemeliharaan rutin dan peningkatan sistem dilakukan untuk memastikan kinerja optimal. Sistem monitoring ini, dapat memberikan pemantauan efektif dan notifikasi cepat, meningkatkan kinerja dan ketersediaan aplikasi DCIM, serta membantu organisasi dalam mengidentifikasi dan menangani masalah lebih cepat dan efisien.**

Kata kunci: **dashboard monitoring, DCIM, grafana alert, bot telegram, kubernetes**

I. PENDAHULUAN

Dalam era digital, perusahaan harus memperkuat infrastruktur teknologi informasi untuk mengelola data secara efisien, aman, dan terukur. PT. Pelayaran Nasional Indonesia (PELNI) memiliki *data center* yang menyimpan informasi penting seperti data pelayaran, jadwal kapal, data penumpang, dan logistik. Efektivitas pengelolaan dan pemantauan *data center* sangat penting untuk mengurangi risiko *downtime* dan memastikan performa optimal. PT. PELNI telah memperbarui sistem manajemen pusat data mereka menggunakan arsitektur *microservices* dan *platform* orkestrasi kontainer seperti Kubernetes, yang memungkinkan pengembangan dan implementasi aplikasi DCIM secara cepat dan fleksibel.

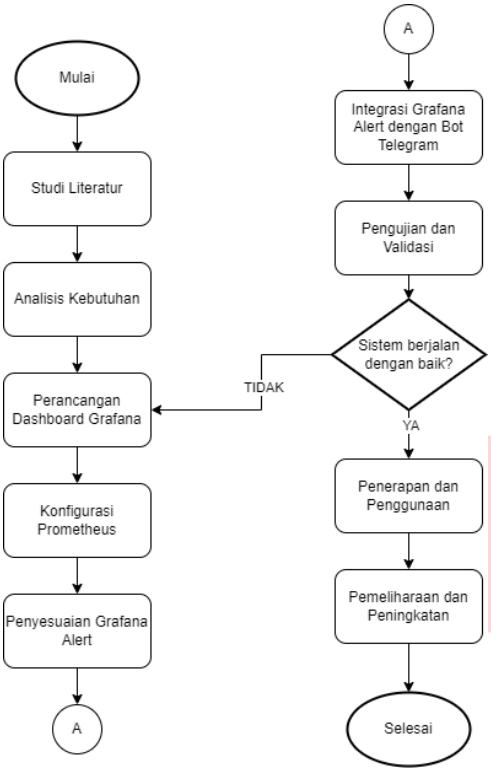
Proyek Akhir ini bertujuan mengembangkan sistem *monitoring* terintegrasi untuk aplikasi DCIM dalam lingkungan Kubernetes. Sistem ini menggunakan Prometheus untuk metrik, Grafana untuk visualisasi data, serta Grafana Alerts dan Bot Telegram untuk notifikasi *real-time*. Metode penelitian mencakup analisis kebutuhan, perancangan *dashboard*, konfigurasi, uji coba, dan validasi sistem. Pemeliharaan rutin dan peningkatan sistem dilakukan untuk memastikan kinerja optimal. Sistem ini akan meningkatkan efisiensi, ketersediaan, dan keandalan operasional *data center*, mendukung operasional perusahaan, dan meningkatkan kepuasan pelanggan.

II. PERANCANGAN DAN MODEL SISTEM

Pada proyek akhir ini dilakukan perancangan sistem *monitoring* untuk aplikasi *Data Center Infrastructure Management* (DCIM) dalam lingkungan kubernetes. Pembahasan mencakup gambaran umum dan kebutuhan dalam perancangan sistem yang dibangun.

A. Flowchart Sistem Monitoring

Flowchart sistem monitoring ini menjelaskan tahapan dan alur proses perancangan *Dashboard Monitoring Data Center Infrastructure Management* (DCIM) yang terintegrasi Grafana Alert dan Bot Telegram dengan Arsitektur Kubernetes.



GAMBAR 2. 1
Flowchart Perancangan Sistem *Monitoring*

Tahapan dan alur proses perencanaan *Dashboard Monitoring* yang dilakukan melalui beberapa tahap.

1. Analisis kebutuhan: menentukan metrik yang perlu dipantau dan jenis notifikasi yang diinginkan untuk aplikasi DCIM dan infrastruktur Kubernetes.
2. Perancangan *dashboard*: merancang *dashboard* grafana untuk visualisasi metrik yang dikumpulkan oleh Prometheus, menentukan grafik dan panel yang dibutuhkan.
3. Konfigurasi Prometheus: mengonfigurasi Prometheus untuk mengumpulkan metrik dari aplikasi DCIM dan infrastruktur Kubernetes, menentukan target yang akan dimonitor.
4. Penyesuaian *alert rules*: Menyesuaikan aturan pemantauan dan *alert* di Grafana sesuai kebutuhan, menentukan kondisi pemicu notifikasi dan tindakan yang harus diambil.
5. Integrasi dengan Telegram: Mengintegrasi Grafana Alerts dengan bot Telegram untuk notifikasi instan dan *real-time*, mengonfigurasikan bot untuk menerima, mengirim pesan, dan menanggapi perintah pengguna.
6. Penerapan dan penggunaan: Menerapkan sistem *monitoring* dan notifikasi ke lingkungan produksi.
7. Pemeliharaan rutin: Melakukan pemeliharaan rutin, pemantauan kinerja, pemecahan masalah, serta menyesuaikan konfigurasi sesuai perkembangan aplikasi dan kebutuhan yang berubah.

B. Perancangan Sistem

Pada perancangan proyek akhir ini akan dirancang *Dashboard Monitoring Data Center Infrastructure Management* (DCIM) yang terintegrasi Grafana Alert dan Bot Telegram. Adapun beberapa konfigurasi dalam sistem ini, yaitu:

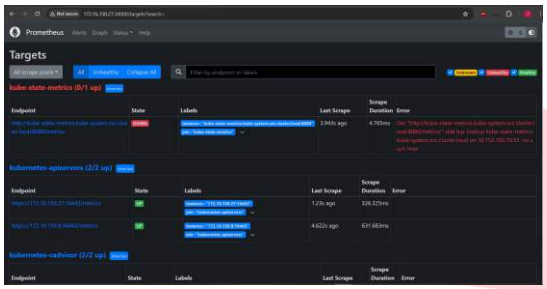
1. Prometheus Targets: target dalam Prometheus berfungsi untuk mengumpulkan data metrik dalam aplikasi Kubernetes, metrik yang dipantau termasuk CPU usage, *memory usage*, dan *network traffic*. Target dikonfigurasi dalam file ‘prometheus.yaml’ yang dapat mendefinisikan *endpoint* yang akan dimonitor.
2. Grafana Dashboard: halaman *dashboard* grafana memberikan tampilan visualisasi *real-time* dari metrik yang dikumpulkan sehingga dapat dengan mudah memahami status dan performa infrastruktur. *Dashboard* ini terdiri dari berbagai panel yang masing-masing menampilkan data dalam bentuk grafik, tabel, atau indikator lainnya. Konfigurasi *dashboard* dirancang di grafana dengan menambahkan panel dan mongonfigurasi sumber data (Prometheus) serta query yang digunakan untuk menampilkan data.
3. Grafana Alert Rules: grafana alert rules adalah aturan yang menentukan kondisi tertentu yang ketika terpenuhi akan memicu sebuah alert atau notifikasi. Konfigurasi aturan alert didefinisikan di dalam grafana dengan menggunakan query dan kondisi. Seperti, jika penggunaan CPU melebihi 80% selama lebih dari 1 menit, aturan alert akan memicu notifikasi.
4. Bot Telegram: bot telegram berfungsi untuk mengirim notifikasi instan ketika grafana alerts terpicu. Bot telegram dikonfigurasi dengan API telegram, dan grafana alerts diintegrasikan dengan bot ini.

III. HASIL DAN PENGUJIAN

Pada bab ini dipaparkan hasil dari perancangan sistem monitoring yang meliputi detail implementasi dan konfigurasi dari setiap komponen utama seperti Prometheus, Grafana, dan bot Telegram. Selain itu, disajikan juga hasil pengujian load testing untuk mengevaluasi kinerja sistem monitoring dalam kondisi beban tinggi. Pengujian ini mencakup analisis metrik performa seperti waktu respons, throughput, dan stabilitas sistem saat menghadapi lonjakan permintaan. Hasil pengujian tersebut digunakan untuk mengidentifikasi area yang perlu ditingkatkan dan memastikan bahwa sistem monitoring mampu memberikan performa optimal dalam lingkungan produksi yang dinamis.

A. Prometheus Targets

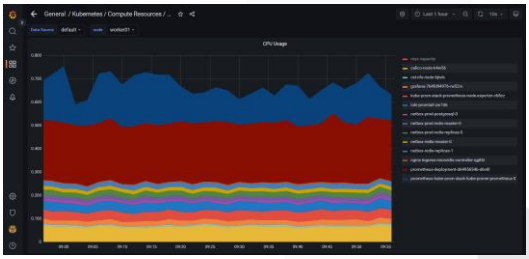
Pada halaman Prometheus targets terlihat bahwa Prometheus telah diaktifkan dengan sukses dan mampu menerima dan menyimpan data dari lingkungan Kubernetes. Hal ini menunjukkan bahwa konfigurasi Prometheus berjalan dengan baik, dan metrik-metrik penting seperti penggunaan CPU, memori, dan performa jaringan dari node dan pod Kubernetes berhasil dikumpulkan.



GAMBAR 3. 1
Prometheus Targets

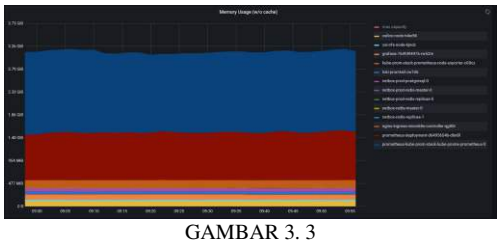
B. Dashboard Grafana

Gambar 3.2 merupakan panel yang menampilkan penggunaan CPU dari node atau pod aplikasi DCIM dan kubnernetes secara *real-time*. Yang memiliki fungsi untuk memantau beban kerja CPU untuk mengidentifikasi apakah ada penggunaan CPU yang tinggi atau tidak normal yang dapat mempengaruhi performa aplikasi.



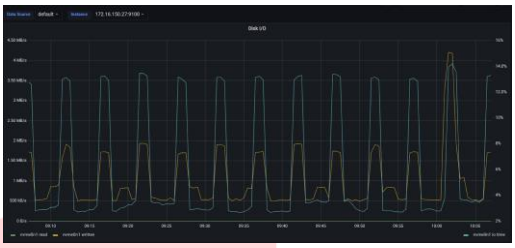
GAMBAR 3. 2
Dashboard Grafana CPU Usage

Gambar 3.3 merupakan panel yang menampilkan penggunaan memori dari node atau pod DCIM dan Kubernetes secara *real-time*. Yang berfungsi untuk memantau alokasi dan penggunaan memori untuk mendeteksi potensi masalah seperti kebocoran memori atau penggunaan memori yang tidak efisien.



GAMBAR 3. 3
Dashboard Grafana Memory Usage

Gambar 3.4 merupakan panel yang menampilkan aktivitas input/output disk dari node atau node DCIM dan kubernetes. Yang berfungsi untuk memantau aktivitas disk untuk mengidentifikasi bottleneck I/O atau masalah performa yang dapat mempengaruhi aplikasi.



GAMBAR 3. 4
Dashboard Grafana Disk I/O

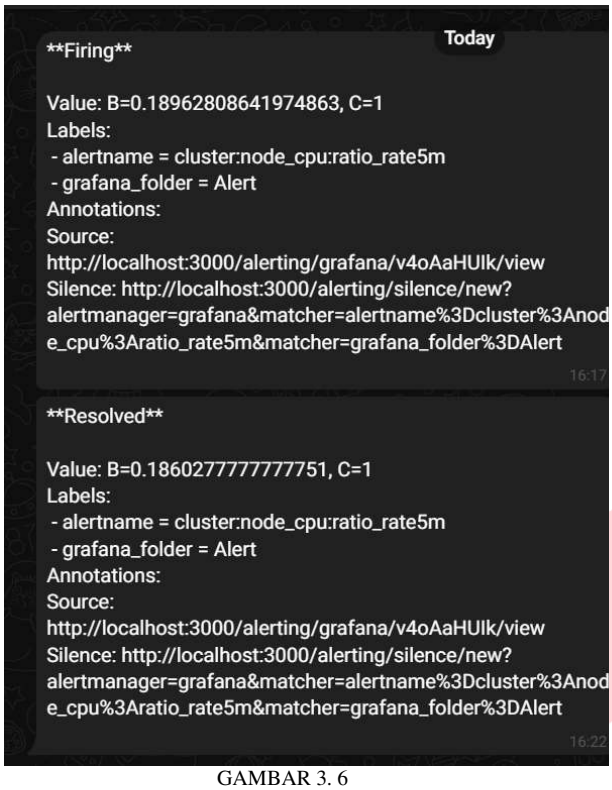
Gambar 3.5 merupakan panel yang menampilkan penggunaan ruang disk dari node atau pod DCIM dan Kubernetes. Yang berfungsi untuk memantau kapasitas disk untuk memastikan bahwa sistem tidak kehabisan ruang penyimpanan, yang dapat menyebabkan kegagalan sistem atau kehilangan data.

GAMBAR 3. 5
Dashboard Grafana Disk Space Usage5

C. Bot Telegram

Notifikasi melalui telegram bertujuan untuk memantau status layanan yang sedang berlangsung. Terdapat 3 status yang muncul sebelum grafana mengirim notifikasi pada telegram.

1. Pending: Ini adalah status ketika layanan yang dimonitor berada dalam proses pending, ditandai dengan warna orange. Status ini menunjukkan bahwa pesan alert akan dikirim, tetapi masih memerlukan waktu beberapa saat sebelum pesan tersebut siap untuk dikirimkan.
2. Firing: Ini adalah status ketika layanan yang dimonitor telah mengirimkan pesan alert ke Telegram, ditandai dengan warna merah. Status ini menunjukkan bahwa pesan alert telah dikirimkan dan layanan sedang dalam keadaan bermasalah.
3. Normal: Ini adalah status ketika layanan yang dimonitor kembali ke kondisi normal, ditandai dengan warna hijau. Status ini menunjukkan bahwa masalah yang terjadi telah diatasi dan layanan kembali beroperasi seperti biasa. Ini juga dapat digunakan untuk membuat notifikasi baru yang menyatakan bahwa masalah telah teratasi (*resolved*).



GAMBAR 3. 6
Pesan Notifikasi pada Telegram

D. Pengujian *Load Testing*

Pengujian sistem dilakukan dengan menyusun rencana pengujian (*test plan*) yang komprehensif serta mempersiapkan *endpoint* pada API yang akan diuji. Pengujian ini mencakup *load testing* untuk memastikan bahwa sistem dapat menangani beban yang tinggi dan tetap berfungsi dengan baik.

Selama pengujian, metrik kinerja yang diamati meliputi throughout dan *error rate*. Hisal dari skenario load testing akan dianalisis untuk mengidentifikasi potensi masalah kinerja dan memastikan bahwa sistem dapat memenuhi kebutuhan operasional dalam berbagai kondisi beban. Berikut merupakan hasil dari pengujian throughput.

TABEL 3. 1
Pengujian Parameter Throughput

Endpoint	Parameter	Number of Thread			
		10	50	100	250
Prometheus Targets	Throughput (KB/s)	1,10	5,10	10,10	24,70
Grafana Login		1,10	5,00	10,00	23,60
Grafana Dashboard		1,10	5,20	10,20	23,50
Grafana Contact Poins		1,10	5,40	10,20	23,50

Tabel 3.1 menunjukkan grafik hasil pengujian throughput per detik. Troughput merupakan jumlah permintaan yang diproses per detik. Grafik ini memperlihatkan bahwa throughput meningkat seiring dengan jumlah permintaan sampel. Ini mengindikasikan bahwa sistem mampu menangani beban dengan baik, karena peningkatan throughput yang sejalan dengan jumlah permintaan sampel menunjukkan kinerja yang baik.

TABEL 3. 2
Pengujian Parameter Error Rate

Endpoint	Parameter	Number of Thread			
		10	50	100	250
Prometheus targets	Error (%)	0,0	0,0	0,0	0,0
Grafana Login		0,0	0,0	0,0	0,0
Grafana Dashboard		0,0	0,0	0,0	0,0
Grafana Contact Poins		0,0	0,0	0,0	0,0

Tabel 3.2 merpakan hasil pengujian terhadap error, dapat dilihat dengan tingkat kesalahan (error) 0%, hasil pengujian ini mengindikasikan bahwa tidak ada permintaan yang gagal selama pengujian. Ini menunjukkan kehandalan tinggi dari server dalam menanggapi semua permintaan dengan sukses. Dengan kata lain, meskipun jumlah thread bertambah, server tetap dapat mengelola dan memberikan respons dengan ukuran yang konsisten, serta mempertahankan kinerja yang stabil dan andal tanpa mengalami kesalahan.

IV. KESIMPULAN

Berdasarkan hasil perancangan, pengujian, dan analisis yang telah dilakukan, dapat disimpulkan bahwa aplikasi ini berhasil diimplementasikan pada setiap node Kubernetes dan dapat diakses dengan baik. Seluruh fitur dalam sistem aplikasi berjalan sesuai harapan, menunjukkan bahwa fungsi-fungsi tersebut beroperasi dengan baik hingga 100%. Selain itu, hasil dari pengujian load testing menunjukkan bahwa performa setiap *endpoint* API tetap optimal meskipun menghadapi lonjakan beban permintaan, dengan tingkat kesalahan (*error rate*) yang tetap rendah, yaitu 0,0% pada setiap proses. Kesimpulan ini menggambarkan bahwa sistem ini siap untuk digunakan secara produktif dan dapat mengatasi tantangan dalam lingkungan operasional yang dinamis.

REFERENSI

[1] A. Tanuwijaya, H. Novianus Palit, and A. Noertjahyana, “Penerapan Microservices dan Amazon Elastic Container Service untuk Mendukung Scalability,” 2021.

[2] A. Nurul Huda and S. Kusumawardani, “Kubernetes Cluster Management For Cloud Computing Platform: A Systematic Literature Review Manajemen Klaster Kubernetes Pada Platform Komputasi Awan: Tinjauan Literatur Sistematik,” 2022.

[3] Editor, “Tentang PT Pelni,” pelni.co.id, 2019. [Online]. Available: https://www.pelni.co.id/tentang-kami. [Accessed 16 02 2024]
[4] T. Sinta Peringkat, berdasarkan S. Dirjen Penguatan RisBang Kemenristekdikti, A. Firmansyah, and N. Merlina, “Prediksi Pola Penjualan Tiket Kapal Pt. Pelni Cabang Makassar Menggunakan Metode Algoritma Apriori,” 2020.
[5] Editor, “Apa itu Kubernetes?,” Kubernetes.io, 2020. [Online]. Available: https://kubernetes.io/id/docs/concepts/overview/what-is-kubernetes/. [Accessed 16 02 2024]
[6] L. Widyawati, H. Santoso, and H. Budiman, “Analisa Penerapan Server Deployment Menggunakan Kubernetes Untuk Menghindari Single Of Failure,” 2021.
[7] M. Harris, “Data Center Infrastructure Management,” 2015. [Online]. Available: http://www.wiley.com/go/datacenterhandbook
[8] R. M. F. M. M. W. W. T. Muh. Usman C, “Aplikasi Sistem Monitoring Server Menggunakan Device Orange Pi berbasis Web Service Studi Kasus Pt. MNC Televisi Indonesia - MNC Group,” 2022.
[9] M. Dicky Syahputra Lubis *et al.*, “Membangun Router Pada Jaringan Komputer Menggunakan Ubuntu OS,” *Jurnal Teknik Informatika Kaputama (JTIK)*, vol. 4, no. 2, 2020.
[10] L. M. Alchuluq and F. Nurzaman, “Analisis Pada Arsitektur Microservice Untuk Layanan Bisnis Toko Online,” 2021.
[11] S. Saidah and R. Syaban, “Implementasi Arsitektur Microservices Pada Aplikasi Point Of Sale Toko Flyover Stiker,” 2023. [Online]. Available: https://flyoverstiker.online/,
[12] Editor, “NetBox Documentation,” docs.netbox.dev. [Online]. Available: https://docs.netbox.dev/en/stable/. [Accessed 17 02 2024]
[13] M. Fadlulloh and R. Bik, “Implementasi Docker Untuk Pengelolaan Banyak Aplikasi Web (Studi Kasus: Jurusan Teknik Informatika UNESA),” 2017.
[14] K. Thias Widagdo, I. Bayu, and Y. A. Susetyo, “Pemodelan Sistem Monitoring Sensor Curah Hujan Menggunakan Grafana,” 2018.
[15] D. Rahman and H. Amnur, “Monitoring Server dengan Prometheus dan Grafana serta Notifikasi Telegram,” 2020. [Online]. Available: http://jurnal-itsi.org
[16] R. Normadhoni, S. Putri Dewanti, W. Cahyo Namaskara, D. Yusfi Akhadi, and R. Fauzi, “Penggunaan Bot Telegram sebagai Announcemnt System dalam Dunia Parenting,” 2021. [Online]. Available: http://jurnalilmiah.org/journal/index.php/jet