

LAPORAN
MONITORING DOCKER CONTAINER DENGAN
PROMETHEUS DAN GRAFANA



Disusun Oleh:

2 D4 Teknik Informatika A (Kelompok 7)

Muhammad Qois Haidar (3122600001)

Raihan Eka Pramudya (3122600011)

Muhammad Arief S. W. (3122600015)

PROGRAM STUDI D4 TEKNIK INFORMATIKA
DEPARTEMEN TEKNIK INFORMATIKA DAN KOMPUTER
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA

2024

I. Abstraksi

Proyek ini bertujuan untuk membangun sistem monitoring container Docker menggunakan Prometheus, cAdvisor, dan Grafana. Sistem ini akan memantau performa dan kesehatan container Docker, serta mengirimkan notifikasi melalui bot Telegram jika terjadi kondisi abnormal. Implementasi sistem ini diharapkan dapat meningkatkan efisiensi dalam pengelolaan dan pemeliharaan container Docker.

II. Pendahuluan

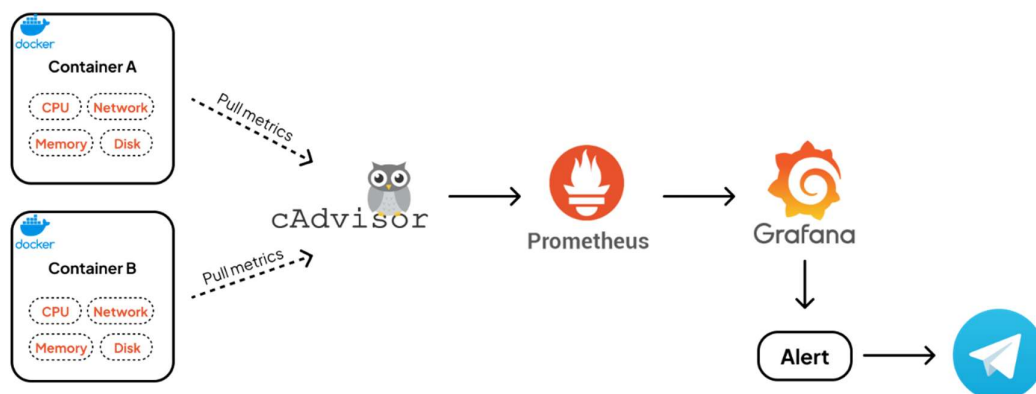
Container Docker adalah alat yang populer untuk mengemas aplikasi dengan dependensi mereka, memastikan bahwa aplikasi berjalan dengan konsisten di berbagai lingkungan. Untuk memastikan kinerja dan kesehatan container tetap optimal, diperlukan sistem monitoring yang baik. Proyek ini menggunakan cAdvisor untuk pengumpulan metrik dari container, Prometheus untuk menyimpan metrik, dan Grafana untuk visualisasi data. Selain itu, sistem ini juga diintegrasikan dengan bot Telegram untuk mengirimkan notifikasi alert.

III. Ruang Lingkup

Proyek ini mencakup:

- Instalasi dan konfigurasi Prometheus, cAdvisor, dan Grafana.
- Pengaturan bot Telegram untuk menerima notifikasi alert.
- Monitoring metrik performa utama seperti CPU, memori, dan jaringan untuk container Docker.
- Visualisasi metrik tersebut menggunakan Grafana.
- Pengaturan alert di Grafana yang dikirimkan melalui bot Telegram jika terjadi kondisi kritis pada container.

IV. Desain Sistem



- cAdvisor: Mengumpulkan metrik dari container Docker.
- Prometheus: Bertugas untuk mengumpulkan dan menyimpan metrik dari cAdvisor.
- Grafana: Menyediakan antarmuka visual untuk memantau metrik yang dikumpulkan oleh Prometheus.
- Telegram Bot: Mengirimkan notifikasi alert dari Grafana kepada tim/user melalui Telegram

V. Tim dan Tugas

- **Muhammad Qois Haidar**
 - Melakukan pengujian sistem untuk memastikan semua komponen berfungsi dengan baik.
- **Raihan Eka Pramudya**
 - Bertanggung jawab atas instalasi dan konfigurasi Prometheus, cAdvisor, dan Grafana.
 - Mengawasi keseluruhan proyek dan memastikan tujuan tercapai.
 - Menangani integrasi dengan bot Telegram.
- **Muhammad Arief S. W.**
 - Memastikan infrastruktur server berjalan dengan aman dan optimal.

VI. Tahapan Pelaksanaan

1. **Perencanaan:** Mendefinisikan tujuan dan ruang lingkup proyek, serta pembagian tugas.
2. **Instalasi dan Konfigurasi:**
 - Instalasi Docker, Prometheus, cAdvisor, dan Grafana.
 - Konfigurasi Prometheus untuk mengumpulkan data dari cAdvisor.
 - Konfigurasi Grafana untuk visualisasi data.
3. **Integrasi Bot Telegram:**
 - Membuat bot Telegram dan mendapatkan token API.
 - Konfigurasi Alertmanager untuk mengirim alert ke Telegram.
4. **Pengaturan Alert:** Membuat aturan alert di Grafana.
5. **Pengujian Sistem:** Melakukan pengujian untuk memastikan semua komponen berfungsi dengan baik dan alert dikirim dengan benar.
6. **Implementasi dan Peluncuran:** Mendemonstrasikan sistem monitoring yang telah berjalan.

VII. Implementasi

1. Instalasi Docker:

```
raihan@debian:~$ sudo docker --version
Docker version 26.1.3, build b72abbb
raihan@debian:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; preset: enabled)
   Active: active (running) since Sat 2024-06-01 16:27:06 WIB; 38s ago
     TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
      Main PID: 5863 (dockerd)
         Tasks: 10
        Memory: 35.2M
          CPU: 581ms
        CGroup: /system.slice/docker.service
               └─5863 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.█
```

2. Membuat direktori baru bernama Monitoring:

```
root@debian:/home/raihan# mkdir Monitoring
root@debian:/home/raihan# ls
Desktop  Documents  Downloads  Monitoring
```

3. Membuat file docker-compose.yml dan prometheus.yml:

```
root@debian:/home/raihan/Monitoring# vi docker-compose.yml
root@debian:/home/raihan/Monitoring# vi prometheus.yml
```

4. Instalasi kontainer Prometheus, Grafana, cAdvisor pada file docker-compose.yml:

```
version: '3.2'

services:
  prometheus:
    image: prom/prometheus:latest
    container_name: prometheus
    ports:
      - 9090:9090
    command:
      - --config.file=/home/raihan/Monitoring/prometheus.yml
    volumes:
      - ./prometheus.yml:/home/raihan/Monitoring/prometheus.yml:ro
    depends_on:
      - cadvisor

  grafana:
    image: grafana/grafana:latest
    container_name: grafana
    ports:
      - "3000:3000"
    volumes:
      - grafana-storage:/var/lib/grafana

  cadvisor:
    image: gcr.io/cadvisor/cadvisor:latest
    container_name: cadvisor
    ports:
      - 8080:8080
    volumes:
      - /:/rootfs:ro
      - /var/run:/var/run:rw
      - /sys:/sys:ro
      - /var/lib/docker:/var/lib/docker:ro
    depends_on:
      - redis

  redis:
    image: redis:latest
    container_name: redis
    ports:
      - 6379:6379

volumes:
  grafana-storage:
```

5. Konfigurasi prometheus pada file prometheus.yml untuk mengumpulkan metrik:

```
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: 'prometheus'
    static_configs:
      - targets: ['localhost:9090']

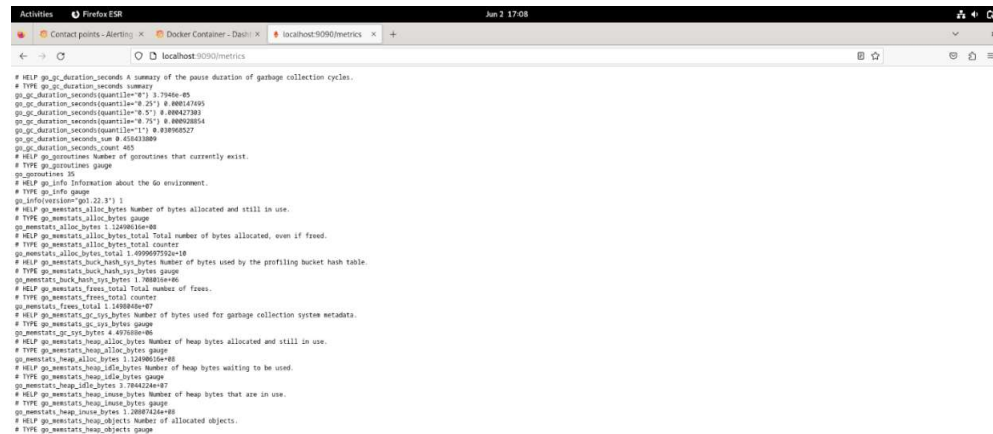
  - job_name: 'cadvisor'
    static_configs:
      - targets: ['cadvisor:8080']
```

6. Mengecek kontainer:

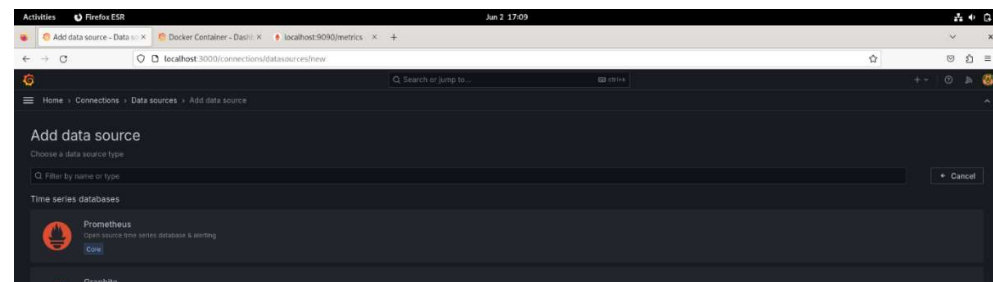
```
root@debian:/home/raihan/Monitoring# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
64f37411a638	prom/prometheus:latest	"/bin/prometheus --c..."	10 hours ago	Up 10 hours	0.0.0.0:9090->9090/tcp,
:9090->9090/tcp	prometheus				
c4f22a7ff36a	gcr.io/cadvisor/cadvisor:latest	"/usr/bin/cadvisor -..."	10 hours ago	Up 10 hours (healthy)	0.0.0.0:8080->8080/tcp,
:8080->8080/tcp	cadvisor				
f751eac3cd91	grafana/grafana:latest	"/run.sh"	10 hours ago	Up 10 hours	0.0.0.0:3000->3000/tcp,
:3000->3000/tcp	grafana				
ee27b2ba9b82	redis:latest	"docker-entrypoint.s..."	10 hours ago	Up 7 minutes	0.0.0.0:6379->6379/tcp,
:6379->6379/tcp	redis				

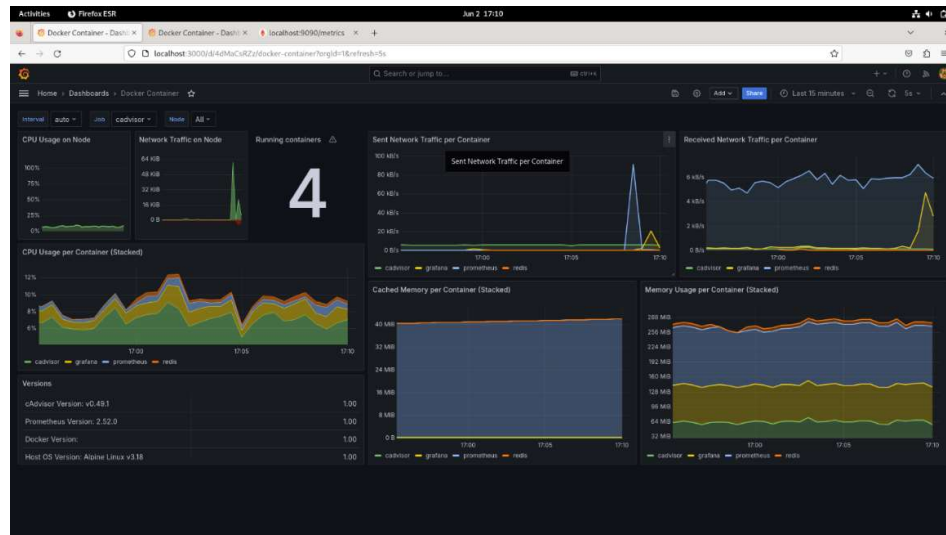
7. Mengecek metrics pada prometheus:



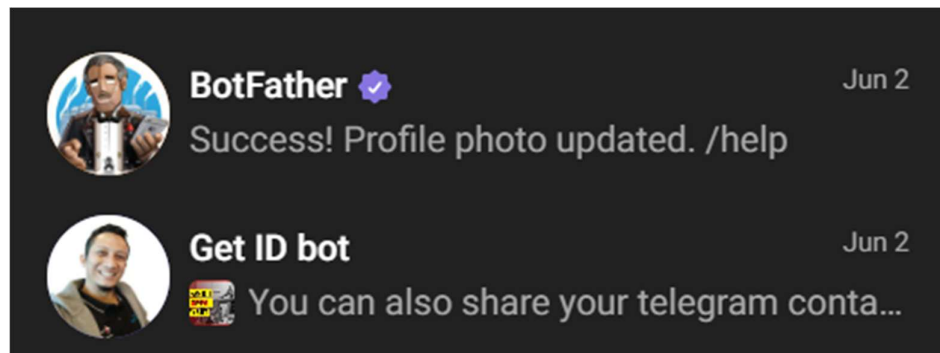
8. Menambahkan data source baru dari prometheus:



9. Tampilan kondisi kontainer pada dashboard grafana:



10. Membuat bot pada telegram dengan BotFather serta Get ID bot:



11. Konfigurasi Contact Points untuk telegram:

The screenshot shows the 'Contact points' configuration page in Grafana. The page title is 'Contact points' and the subtitle is 'Choose how to notify your contact points when an alert instance fires.' The 'Integration' dropdown is set to 'Telegram'. The 'BOT API Token' field is filled with 'Telegram BOT API Tokens'. The 'Chat ID' field is filled with 'Integer Telegram Chat identifier'. There are sections for 'Optional Telegram settings' and 'Notification settings'. At the bottom, there are buttons for 'Add contact point integration', 'Save contact point', and 'Cancel'.

12. Konfigurasi Notification Policy untuk alert:

The screenshot shows the 'Edit notification policy' dialog. It has a title bar with a close button. The main content area includes a 'Default contact point' dropdown set to 'grafana-telegram' with a link to 'Create a contact point'. Below is a 'Group by' section with a dropdown showing 'grafana_folder' and 'alertname'. The 'Timing options' section is expanded, showing 'Group wait' (30s), 'Group interval' (1m), and 'Repeat interval' (1h). At the bottom are 'Cancel' and 'Update default policy' buttons.

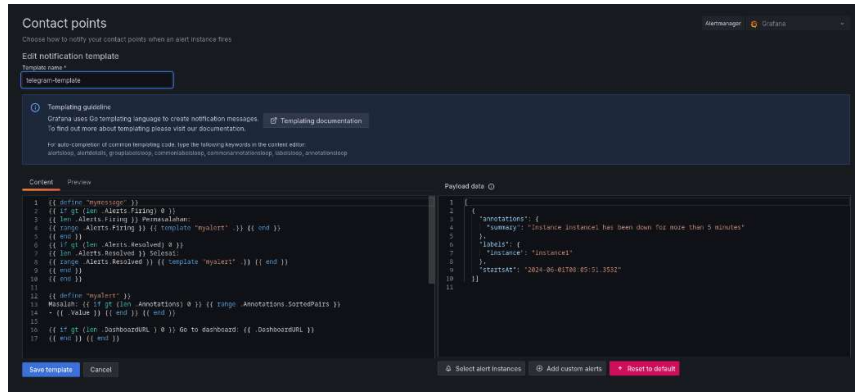
13. Konfigurasi alert rules untuk container yang mati:

The screenshot shows the 'Edit rule' dialog for a 'Container Alert'. It has a title bar and a 'Name' field set to 'Container Alert'. The '2. Define query and alert condition' section is active, showing a Prometheus query: `count(rate(container_last_seen{name=~".+"}[30s]))`. Below the query is a 'Rule type' section with 'Grafana-managed' selected. The 'Expressions' section shows a 'Threshold' condition set to 'Alert condition'.

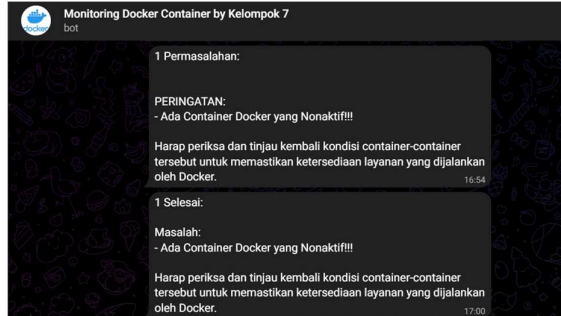
14. Konfigurasi alert rules untuk memory pada tiap container:

The screenshot shows the 'Edit rule' dialog for a 'Memory Alert'. It has a title bar and a 'Name' field set to 'Memory Alert'. The '2. Define query and alert condition' section is active, showing a Prometheus query: `sum(container_memory_usage{name=~".+"}) by (name) / 1024000`. Below the query is a 'Rule type' section with 'Grafana-managed' selected. The 'Expressions' section shows a 'Threshold' condition set to 'Alert condition'.

15. Konfigurasi template message untuk susunan teks telegram:



16. Percobaan ketika ada container yang mati:



VIII. Sistem Testing

1. Uji Fungsional:

- Verifikasi bahwa Prometheus mengumpulkan metrik dari cAdvisor.
- Pastikan Grafana menampilkan metrik yang dikumpulkan dengan benar.
- Uji pengiriman alert melalui bot Telegram dengan memicu kondisi (misalnya, mematikan salah satu kontainer docker).

2. Uji Integrasi:

- Pastikan integrasi antara cAdvisor, Prometheus, Grafana, dan bot Telegram berjalan dengan baik.
- Verifikasi pengiriman alert ke Telegram dan pastikan pesan alert sesuai dengan aturan yang ditentukan.

3. Uji Performa:

- Uji kinerja sistem monitoring dengan beberapa container untuk memastikan tidak ada bottleneck.

- Monitor penggunaan sumber daya (CPU, memori) oleh Prometheus, cAdvisor, dan Grafana.