JUPITER
Teknologi Informatika & Komputer
Vol. 6 No. 2 Agustus 2025
Institut Bisnis Muhammadiyah Bekasi
P-ISSN: 2798-3129 , E-ISSN: 2798-6187

SISTEM MONITORING JARINGAN DENGAN METODE *NETWORK DEVELOPMENT LIFE CYCLE* DI PT LINTAS TEKNOLOGI SOLUSINDO

Moh Ikbal Baihaqi[1], Slamet Raharjo[1,*]

[1]Teknik Informatika, Institut Bisnis Muhammadiyah Bekasi, ikbal20102001@gmail.com, sraharjo@gmail.com

ABSTRAK

Sistem monitoring jaringan sangat dibutuhkan untuk memudahkan dalam memantau kondisi perangkat jaringan secara *realtime* dan mendeteksi gangguan dengan cepat. Dengan menggunakan *Prometheus* memungkinkan sistem monitoring dapat mengambil data terkait kondisi perangkat *router* seperti *CPU*, status *interface* dan *network traffic* melalui *Simple Network Management Protocol (SNMP)*. Lalu *Grafana* akan menampilkannya kedalam bentuk visualisasi yang mudah dipahami. Penelitian menggunakan pendekatan metode *Network Development Life Cycle (NDLC)* yang tahapannya terdiri dari *analysis*, *design*, *simulation prototyping*, *implementation*, *monitoring* dan *management*. Hasil dari penelitian berupa tampilan *dashboard* dari *Grafana* yang dilengkapi sistem *alert* untuk mengirimkan notifikasi pada *Telegram* terkait kondisi jaringan. Adanya sistem monitoring jaringan diharapkan dapat membuat aktivitas pemantauan jaringan menjadi lebih mudah dan efisien serta mempercepat durasi *detection time* ketika terjadi gangguan pada jaringan.

Kata Kunci: *Bot Telegram, Grafana*, *NDLC*, *Prometheus*, Sistem Monitoring Jaringan

ABSTRACT

A network monitoring system is needed to make it easier to monitor the condition of network devices in realtime and detect disturbances quickly. Using Prometheus allows the monitoring system to retrieve data related to the condition of router devices such as CPU, interface status and network traffic via Simple Network Management Protocol (SNMP). Then Grafana will display it in a visualization form that is easy to understand. The research uses a Network Development Life Cycle (NDLC) method approach whose stages consist of analysis, design, simulation prototyping, implementation, monitoring and management. The results of the research are a dashboard display from Grafana which is equipped with an alert system to send notifications to Telegram regarding network conditions. It is hoped that the existence of a network monitoring system can make network monitoring activities easier and more efficient and speed up the duration of detection time when network disruption occurs.

Keywords: *Grafana, NDLC, Network Monitoring System, Prometheus, Telegram Bot*

PENDAHULUAN

Di era digital saat ini, keandalan dan kinerja jaringan internet menjadi sangat penting, terutama bagi penyedia layanan internet atau *Internet Service Provider (ISP)*. Hal ini disebabkan oleh meningkatnya permintaan pelanggan untuk akses internet yang stabil dan cepat. Penyedia layanan internet harus memastikan bahwa infrastruktur jaringan mereka selalu dapat diandalkan. Oleh karena itu, diperlukan adanya sistem

monitoring jaringan yang akurat dan terintegrasi dengan suatu aplikasi untuk memberikan peringatan ketika terjadi masalah pada infrastruktur jaringan.

Para penyedia layanan internet termasuk PT Lintas Teknologi Solusindo menghadapi banyak masalah dalam mengelola dan memantau jaringan mereka, seperti kecepatan akses internet yang lambat, *network traffic* yang penuh dan kondisi *downtime* akibat Fiber Optik *Cut* (*FO Cut*) maupun perangkat yang *hang* karena *resource* yang *overload*. Jika tidak ada sistem monitoring yang memadai, menemukan dan menyelesaikan masalah tersebut dapat menjadi sulit dan memakan waktu, yang pada akhirnya akan berdampak pada berkurangnya tingkat kepuasan pelanggan.

Untuk mengatasi masalah tersebut maka peneliti memiliki solusi yaitu membangun sistem monitoring jaringan dengan metode *Network Development Life Cycle (NDLC)*. Metode ini sangat cocok diterapkan pada pengembangan sistem jaringan komputer. Adapun *tools* monitoring jaringan yang digunakan yaitu *Prometheus* dan *Grafana*, karena perpaduan keduanya sangat cocok untuk dapat menyajikan data kondisi jaringan secara *realtime* dengan tingkat akurasi yang tinggi dan tampilan yang dinamis serta mudah dianalisa. Selain itu, diperlukan sebuah sistem *alert* untuk mengirimkan notifikasi ke *Telegram* agar staf *Network Operation Center (NOC)* tetap mengetahui kondisi jaringan ketika sedang tidak berada didepan layar monitor. Dengan menggunakan *Grafana* yang terintegrasi dengan *Telegram* diharapkan setiap gangguan yang terjadi dapat dideteksi lebih awal, sehingga waktu *detection time* dan waktu *troubleshooting* menjadi lebih cepat.

TINJAUAN PUSTAKA

Sistem Monitoring Jaringan

Sistem monitoring jaringan atau *Nework Monitoring System (NMS)* adalah suatu sistem yang digunakan untuk memantau kondisi jaringan. Dengan memantau perangkat jaringan, *NOC* dapat dengan mudah mengetahui keadaan jaringan dengan menerima *alert*. *NMS* digunakan untuk melacak masalah jaringan, seperti gangguan pada *server* atau *overload*, serta koneksi jaringan dan perangkat lainnya (Prasetyo et al., 2019). Tahapan dalam sebuah sistem monitoring terbagi ke dalam tiga proses besar, yaitu (Prayogi et al., 2020):

a. Proses pengumpulan data untuk dilakukan monitoring.
b. Proses melakukan analisis terhadap data yang diperoleh.
c. Proses menampilkan data olahan monitoring.

Tujuan dari monitoring yaitu untuk menilai apakah tindakan yang dilakukan telah sesuai dengan rencana, menemukan masalah yang perlu ditangani segera, menilai apakah pola kerja dan manajemen yang digunakan telah tepat untuk mencapai tujuan, dan mengetahui hubungan antara tindakan dan tujuan (Wiriaatmadja & Ratama, 2022).

Exporter

Exporter memiliki fungsi untuk melakukan pengiriman data dari *node* yang akan dimonitoring ke *Prometheus server*. *Exporter* memungkinkan untuk mengekspor statistik dari sistem non-*Prometheus* dan mengimpornya ke *Prometheus*. Banyak macam *Exporter* yang dapat mengekspor metrik yang ada dari sistem pihak ketiga menjadi metrik *Prometheus* (Ramadoni et al., 2021).

Untuk pengambilan data, *Prometheus* akan mengakses metrik *Exporter*. Dalam monitoring jaringan *Exporter* yang digunakan yaitu *SNMP Exporter*. Apabila metrik

telah diakses, tahapan pengambilan data akan berjalan secara normal dan data akan disimpan dalam *database*. Pada dasarnya *Prometheus* tidak dapat mengambil data dari objek yang diawasi secara langsung. Perangkat lunak pihak ketiga yang disebut *Exporter* diperlukan untuk mengirimkan data ke sistem yang diawasi dan memungkinkan akses melalui *Hyper Text Transfer Protocol* (*HTTP)* yang biasanya digunakan *Prometheus* untuk mengambil data dari target (Harnanta et al., 2020).

Prometheus

Prometheus adalah aplikasi *open-source* yang bermanfaat untuk memantau ketersediaan dan performa jaringan komputer. *Prometheus* memiliki kemampuan untuk menghasilkan data metrik dengan menggunakan data *resource server* (Dira et al., 2022). *Prometheus* merupakan *tools* monitoring yang dirancang untuk memantau suatu sistem atau perangkat. *Prometheus* memantau objek tertentu yang dapat berupa *server* tunggal atau *endpoint* yang berkomunikasi melalui *Hyper Text Transfer Protocol (HTTP), Hyper Text Transfer Protocol Secure (HTTPS), Domain Name System (DNS), Transmission Control Protocol (TCP)*, dan *Internet Control Message Protocol (ICMP)*. *Prometheus* mengambil data target pada interval tertentu untuk mengumpulkan metrik target. Adapun karakteristik utama *Prometheus* adalah sebagai berikut (Rahman et al., 2020):

a. Model data multi dimensi dengan data *time-series* yang diidentifikasi oleh nama metrik dan pasangan kunci (*key*) atau nilai (*value*).
b. *Prometheus Query Language (PromQL)* yang merupakan bahasa permintaan yang fleksibel digunakan untuk mengambil data tertentu secara spesifik.
c. *Node server* tunggal bersifat otonom tanpa penyimpanan terdistribusi.
d. Pengumpulan *time-series* melalui protokol *HTTP*.
e. Mendorong *time-series* didukung melalui *gateway* perantara.
f. Didukung dengan visualisasi *dashboard web* seperti *Grafana*.

Tugas *Prometheus* yaitu mengumpulkan data metrik dari target yang mengekspor ke aplikasi yang telah ditetapkan sesuai keadaan saat itu. Kemudian, *Prometheus* mengevaluasi aturan ekspresi, menampilkan hasil, dan memberikan *feedback* yang sesuai jika kondisi diamati dengan benar (Darmawan, 2023). Beberapa komponen utama *Prometheus* yaitu *Prometheus server* yang mengelola dan menyimpan data, *Prometheus exporter* yang mengirim data dari *server* target ke *Prometheus server* dan *alert manager* yang mengatur notifikasi (Ramadoni et al., 2021).

Grafana

Grafana adalah perangkat lunak untuk visualisasi dan analisis data yang bersifat *open-source*. *Grafana* dapat bekerja untuk memvisualisasikan, mengingatkan, dan menjelajahi metrik yang disimpan. *Grafana* sebagai alat yang berperan mengubah data *Time Series Database (TSDB)* menjadi grafik dengan visualisasi yang dinamis dan indah (Rahman et al., 2020). *Grafana* mendukung berbagai *backend* penyimpanan untuk rangkaian data *time-series*. Setiap sumber data memiliki *query* editor unik yang disesuaikan untuk fitur dan kemampuan yang berbeda. Banyak basis data yang didukung antara lain yaitu *Graphite, Prometheus, InfluxDB, ElasticSearch, MySQL,* dan *PostgreSQL*. Selain itu, *Grafana* memungkinkan pembuatan *plugin* untuk diintegrasikan dengan berbagai basis data. *Grafana* memiliki beberapa fitur utama yang menjadi kelebihan dari *tools* monitoring ini seperti yaitu integrasi sumber data yang banyak,

visualisasi yang dinamis dan modern serta memiliki sistem *alert* yang terintegrasi ke banyak platform *message* (Darmawan, 2023).

Telegram

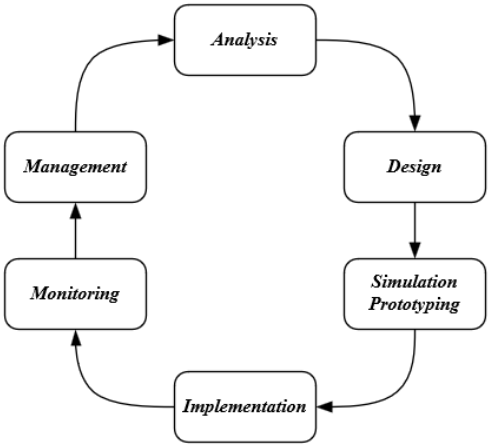
Telegram adalah platform komunikasi cepat yang sangat penting untuk menyampaikan pesan mendesak atau penting. Salah satu alat yang paling efektif untuk menyampaikan informasi dengan kecepatan tinggi adalah platform ini. Adanya *bot* adalah salah satu fitur terbaik. *Telegram Bot* adalah akun *Telegram* khusus yang dirancang untuk mengirimkan pesan secara otomatis, dan pengguna dapat berinteraksi dengan *bot* melalui pesan pribadi atau grup menggunakan pesan perintah (Ishaq & Firmansyah, 2023).

Telegram mempunyai dua jenis *Application Programming Interface (API)* yang dapat dimanfaatkan untuk kebutuhan monitoring, yaitu (Fernando et al., 2020):

a. *Bot API*
   Bot API memungkinkan pengembang menghubungkan *bot* ke sistem *Telegram*. *Telegram bot* adalah akun unik yang tidak membutuhkan nomor telepon dalam pengaturannya. Aplikasi ini berfungsi sebagai antarmuka yang memungkinkan kode berjalan di *server*.
b. *Telegram API*
   Telegram API terbuka sepenuhnya untuk semua pengembang yang ingin membuat aplikasi dengan platform *Telegram*. Ini memungkinkan pengembang untuk membangun klien *Telegram* yang diinginkan sendiri.

METODE PENELITIAN

Network Development Life Cycle (NDLC) adalah model untuk mendefinisikan siklus proses pembangunan atau pengembangan sistem jaringan komputer. Pencapaian tujuan strategi bisnis perusahaan saat ini dan di masa depan sangat bergantung pada keberhasilan penerapan *NDLC* dalam penyebaran informasi yang tepat dan akurat. Selama proses pengembangan sistem informasi bisnis, model *NDLC* memungkinkan perusahaan memiliki serangkaian arsitektur teknologi informasi jaringan yang efisien. Tahapan dalam metode *NDLC* seperti terlihat pada Gambar 1 terdiri dari *analysis*, *design*, *simulation prototype*, *implementation*, *monitoring* dan *management* (Lusi & Belutowe, 2023).



Gambar 1. Siklus *NDLC*

Peneliti menggunakan metode *NDLC* untuk membangun sistem monitoring jaringan yang akan peneliti jabarkan tiap tahapannya sebagai berikut:

a. *Analysis*
Pada tahap *analysis* dilakukan identifikasi permasalahan, kemudian dilanjutkan dengan analisis sistem berjalan untuk mengetahui bagaimana *Standard Operating Procedure (SOP)*, lalu dilakukan analisis sistem usulan untuk memberikan solusi terkait sistem monitoring jaringan yang ingin dibuat. Terakhir dalam tahapan ini dilakukan analisis kebutuhan perangkat keras *(hardware)* maupun perangkat lunak *(software)* yang akan digunakan.

b. *Design*
Tahap perancangan *(design)* merupakan tahapan dilakukannya perancangan sistem yang mendefinisikan sebuah gambaran bagaimana sistem bekerja dan bagaimana alur implementasinya. Tahapan ini terdiri dari perancangan topologi jaringan dan perancangan alur kerja sistem monitoring yang ingin diimplementasikan.

c. *Simulation Prototyping*
Pembuatan *prototyping* dilakukan sebagai simulasi sebelum melakukan implementasi aktual. Dengan begitu peneliti dapat mengetahui gambaran umum mekanisme kerja dari sistem monitoring jaringan yang akan dibangun. Peneliti membangun *prototyping* sistem ini pada lingkungan virtual menggunakan bantuan *software GNS3, VirtualBox* dan *VMware*. Pada *VirtualBox* akan dibuat sebuah *server Virtual Machine (VM)* berbasis *Linux Ubuntu* yang berfungsi sebagai *NMS Server* yang akan diinstal *Prometheus* dan *Grafana*. Pada *VMware* akan dibuat *GNS3 VM Server* berbasis *Linux Ubuntu* sedangkan untuk perangkat *router Mikrotik* juga akan disimulasikan menggunakan *Qemu* dengan menginstal *Mikrotik CHR*. Untuk melakukan simulasi sesuai desain topologi yang dirancang, peneliti menggunakan aplikasi *GNS3*.

d. *Implementation*
Tahap *implementation* akan menerapkan seluruh rancangan sistem yang sudah dibuat pada tahapan sebelumnya. Kegiatan ini diawali dengan menyiapkan kebutuhan perangkat keras maupun perangkat lunak dan menempatkan *server* yang akan menjadi *server* monitoring sesuai dengan rancangan topologi yang sudah dibuat. Setelah instalasi topologi selesai, selanjutnya peneliti melakukan instalasi dan konfigurasi *SNMP Exporter*, *Prometheus* dan *Grafana* pada *server* lalu melakukan *setup dashboard* monitoring pada *Grafana* dan melakukan pembuatan *bot Telegram* serta konfigurasi *alert* pada *Grafana*.

e. *Monitoring*
Pada tahapan *monitoring* dilakukan pengujian dan pengamatan terhadap sistem monitoring jaringan yang telah dibangun. Tujuannya agar dapat mengetahui apakah sistem monitoring jaringan sudah berjalan dengan baik atau perlu perbaikan. Metode pengujian yang akan digunakan adalah metode pengujian *Black-Box.* Metode pengujian *Black-Box* didasarkan pada pendekatan dimana sistem dianggap sebagai "kotak hitam" dan fokus utama pengujian adalah pada fungsionalitas eksternal tanpa memperhatikan struktur atau logika internal dari komponen sistem. Adapun pengujian dibagi menjadi empat bagian yaitu pengujian *SNMP Exporter*, pengujian *Prometheus*, pengujian *Grafana* dan pengujian *alert*.

f. *Management*
Pada tahap *management* dilakukan perawatan, pemeliharaan dan pengelolaan agar sistem monitoring jaringan yang telah dibangun dapat terjaga dan berfungsi dengan baik. Selain itu dilakukan juga analisis hasil dan evaluasi terhadap uji kelayakan sistem monitoring jaringan yang dibangun. Maka kebijakan terkait pemeliharaan dan evaluasi sistem perlu dibuat atau diatur oleh PT Lintas Teknologi Solusindo.

HASIL DAN PEMBAHASAN

Analisis Kebutuhan

a. Kebutuhan Perangkat Keras *(Hardware)*
Kebutuhan *hardware* untuk penelitian ini menggunakan perangkat laptop dengan spesifikasi seperti yang terdapat pada Tabel 1.

Tabel 1. Kebutuhan *Hardware*

Spesifikasi	**Keterangan**
System Model	*Lenovo Ideapad 330S*
Operating System	*Windows 11 Pro 64 bit*
Processor	*Intel(R) Core(TM) i5-8250U*
Memory	*16 GB*
Harddisk System	*SSD 256 GB*
Harddisk Storage	*1 TB SATA*
Network Interface	*VirtualBox Host-Only Ethernet Adapter, VMware Virtual Ethernet Adapter VMnet1, VMware Virtual Ethernet Adapter VMnet8*

Sumber: Hasil Penelitian (2024)

b. Kebutuhan Perangkat Lunak *(Software)*
Sistem monitoring jaringan yang dibangun membutuhkan perangkat lunak dengan spesifikasi seperti pada Tabel 2.

Tabel 2. Kebutuhan *Software*

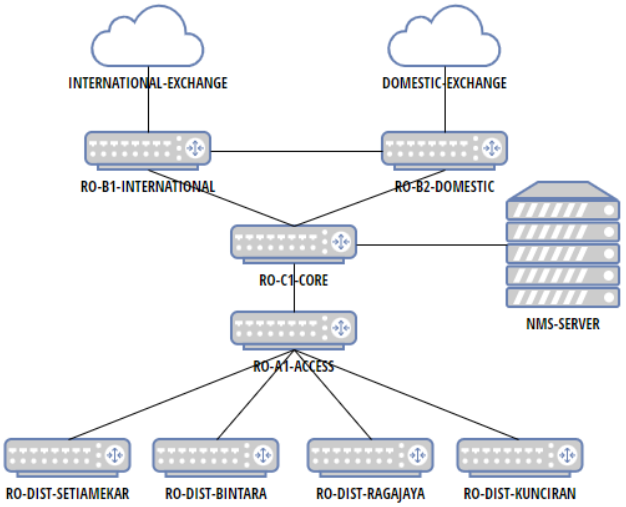
Spesifikasi	**Keterangan**
Operating System	*Linux Ubuntu 22.04.03, Mikrotik CHR 7.11.2 (stable)*
Software Virtualization	*VMware, VirtualBox*
Software Network Simulator	*GNS3*
Software Monitoring	*Prometheus, Grafana*
Software Messenger	*Telegram*

Sumber: Hasil Penelitian (2024)

Desain Topologi

Topologi jaringan yang digunakan pada PT Lintas Teknologi Solusindo menggunakan topologi *star*. Topologi *star* merupakan sebuah topologi yang memiliki model jaringan seperti bintang dengan letak *server* berada di bagian tengah sebagai *center*. Sementara perangkat komputer akan terhubung dengan *server* sebagai cabang dari *server* tersebut. PT Lintas teknologi Solusindo memiliki total 8 *router* dengan pembagian 4 *router* sebagai *core* dan 4 *router* sebagai *distribution*. Topologi jaringan dapat dilihat pada Gambar 2.

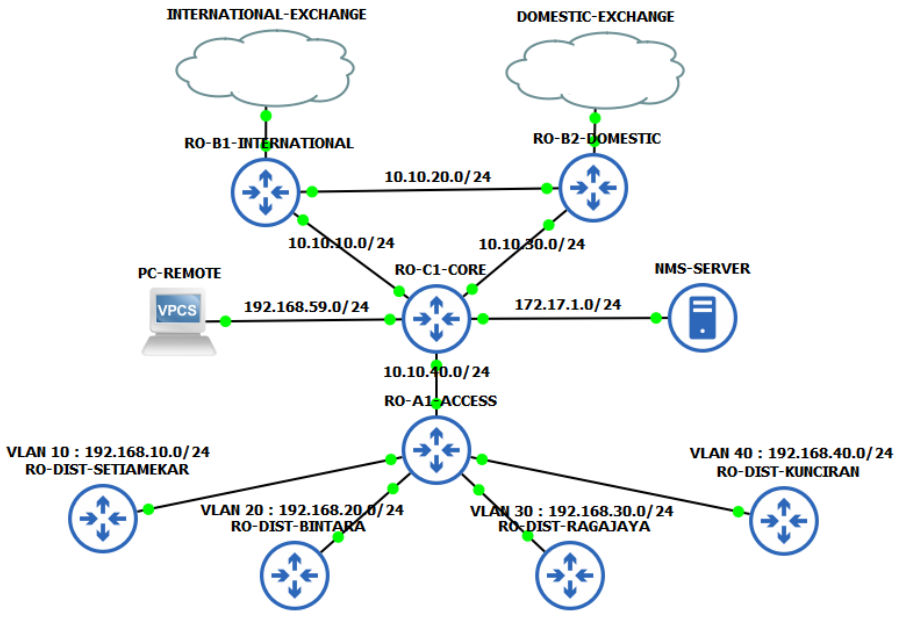
JUPITER
Teknologi Informatika & Komputer
Vol. 6 No. 2 Agustus 2025
Institut Bisnis Muhammadiyah Bekasi
P-ISSN: 2798-3129 , E-ISSN: 2798-6187



Gambar 2. Topologi Jaringan

Simulasi Prototipe

Simulasi prototipe menggunakan bantuan *software* khusus yang digunakan untuk melihat kinerja awal dari sistem monitoring jaringan yang akan dibangun sebelum melakukan *deploy* langsung ke tahap *production*. Pada tahap simulasi ini peneliti menggunakan bantuan *software GNS3* yang terintegrasi dengan *VirtualBox* dan *VMware*. *Prometheus* dan *Grafana* diinstal pada *server* yang berbasis *Linux Ubuntu 22.04.03* di dalam *VirtualBox*. Untuk perangkat *router Mikrotik* juga akan disimulasikan menggunakan *Qemu* dengan menginstal *Mikrotik CHR* didalamnya. Indikator hijau menandakan bahwa perangkat sudah aktif dan terkoneksi satu sama lain. *PC-REMOTE* berfungsi sebagai koneksi ke laptop fisik agar dapat melakukan *remote* ke *router Mikrotik* dan untuk mengakses *NMS-SERVER*.



Gambar 3. Simulasi *GNS3*

JUPITER
Teknologi Informatika & Komputer
Vol. 6 No. 2 Agustus 2025
Institut Bisnis Muhammadiyah Bekasi
P-ISSN: 2798-3129 , E-ISSN: 2798-6187

Konfigurasi

Pada tahap konfigurasi dilakukan instalasi *SNMP Exporter*, *Prometheus* dan *Grafana* pada *server*. Untuk mendapatkan data metrik dari *router* yang ingin dimonitoring perlu melakukan konfigurasi pada file *prometheus.yml* seperti yang terlihat pada Gambar 4.

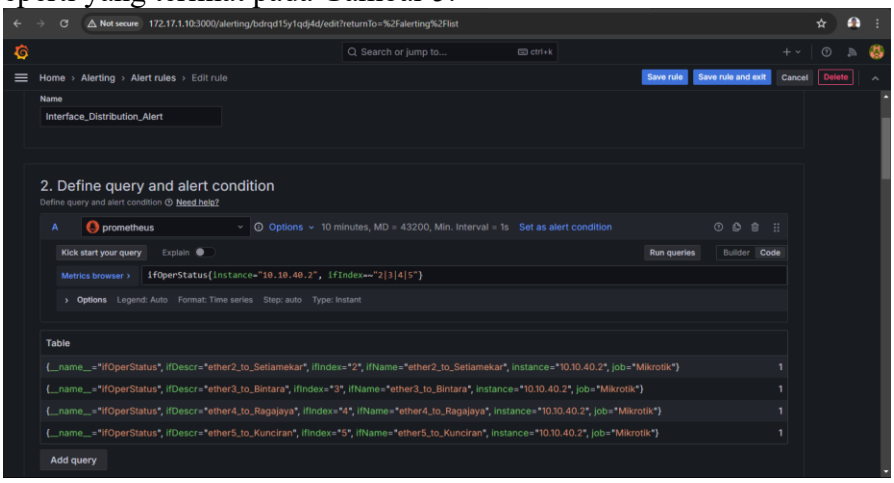
```
 GNU nano 6.2                                  /etc/prometheus/prometheus.yml
global:
  scrape_interval: 60s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
  evaluation_interval: 60s # Evaluate rules every 15 seconds. The default is every 1 minute.

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

  - job_name: 'Mikrotik'
    static_configs:
      - targets:
        - 10.10.10.1 # RO_B1_INTERNATIONAL
        - 10.10.30.1 # RO_B2_DOMESTIC
        - 172.17.1.1 # RO_C1_CORE
        - 10.10.40.2 # RO_A1_ACCESS
        - 192.168.10.2 # RO_SETIAMEKAR
        - 192.168.20.2 # RO_BINTARA
        - 192.168.30.2 # RO_RAGAJAYA
        - 192.168.40.2 # RO_KUNCIRAN
    metrics_path: /snmp
    params:
      module: [if_mib]
    relabel_configs:
      - source_labels: [__address__]
        target_label: __param_target
      - source_labels: [__param_target]
        target_label: instance
      - target_label: __address__
        replacement: 127.0.0.1:9116  # The SNMP exporter's real hostname:port.
```

Gambar 4. Konfigurasi *Prometheus*

Setelah perangkat target memiliki status up pada Prometheus, maka selanjutnya melakukan query untuk mendapatkan data metrik seperti status interface, traffic dan CPU usage dan menampilkannya pada Grafana. Untuk pembuatan alert dilakukan dengan membuat bot Telegram terlebih dahulu dan melakukan konfigurasi pada alert Grafana seperti yang terlihat pada Gambar 5.



Gambar 5. Konfigurasi *Alert Grafana*

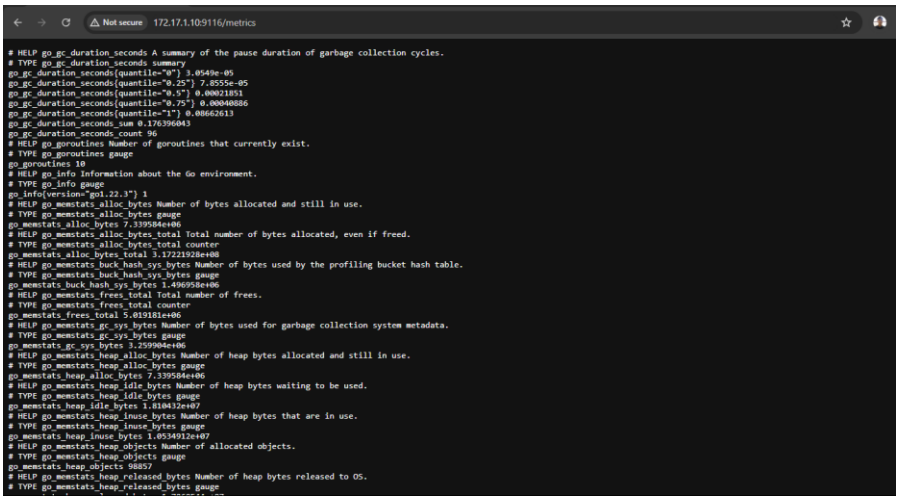
Pengujian

a. Pengujian *SNMP Exporter*

Pada pengujian ini *Exporter* yang akan diuji adalah *SNMP Exporter*. Pengujian dilakukan setelah tahap instalasi dan konfigurasi *SNMP Exporter* berhasil.

JUPITER
Teknologi Informatika & Komputer
Vol. 6 No. 2 Agustus 2025
Institut Bisnis Muhammadiyah Bekasi
P-ISSN: 2798-3129 , E-ISSN: 2798-6187

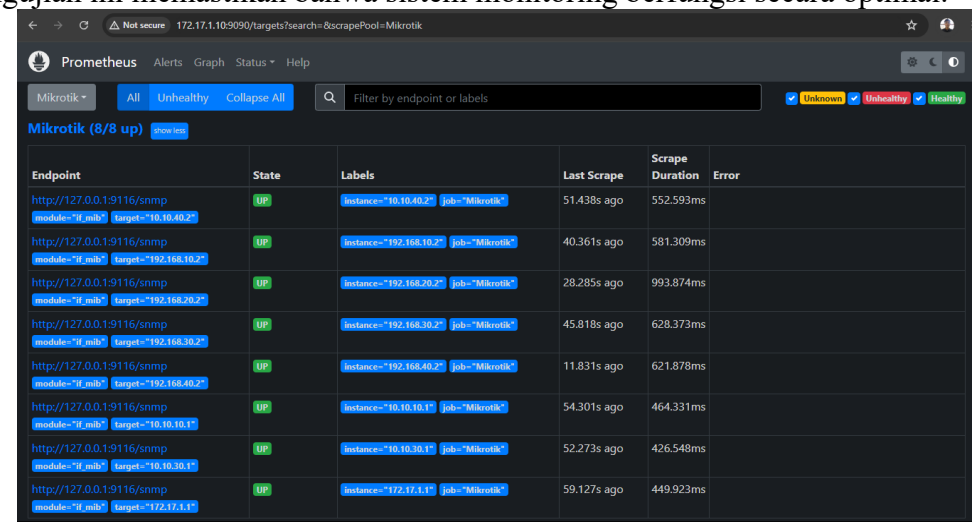
Pengujian ini untuk mengetahui apakah *SNMP Exporter* telah berhasil mengekspos metrik untuk pengambilan data pada target melalui protokol *SNMP*. Secara *default SNMP Exporter* berjalan di *port TCP* 9116. Apabila metrik telah berhasil diakses melalui *http://localhost:9116* maka untuk tahapan pengambilan data telah berjalan normal.



Gambar 6. *SNMP Exporter*

b. Pengujian *Prometheus*

Setelah berhasil menyelesaikan instalasi dan konfigurasi *Prometheus*, peneliti melanjutkan dengan tahap pengujian untuk memastikan bahwa *Prometheus* telah aktif dan dapat menerima data dari *SNMP Exporter*. Pengujian ini penting untuk memastikan bahwa sistem berfungsi dengan baik dan data dapat ditangkap serta diolah oleh *Prometheus*. Langkah pertama dalam pengujian ini adalah membuka *browser* dan mengakses alamat *http://localhost:9090*. Jika proses instalasi dan konfigurasi berhasil, maka akan tampil antarmuka pengguna *Prometheus* yang menampilkan berbagai metrik dan data yang telah dikumpulkan. Jika antarmuka pengguna *Prometheus* muncul seperti yang diharapkan, ini menandakan bahwa *Prometheus* telah berjalan dengan baik dan siap digunakan. Dengan demikian, pengujian ini memastikan bahwa sistem monitoring berfungsi secara optimal.

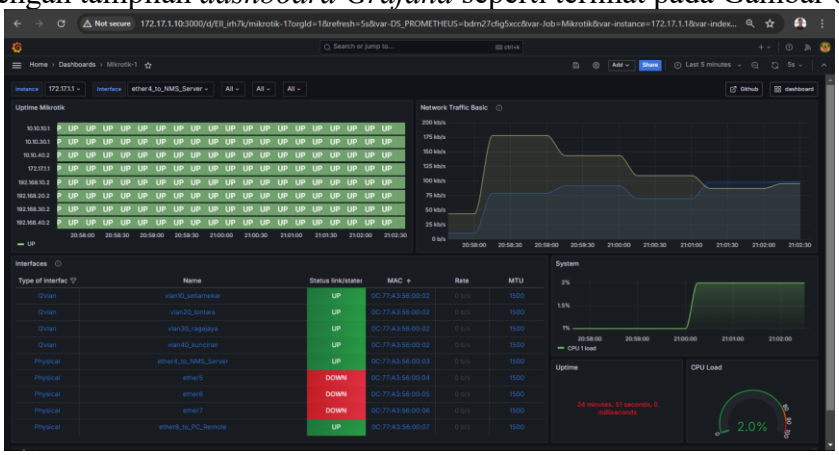


Gambar 7. *Prometheus UI*

JUPITER
Teknologi Informatika & Komputer
Vol. 6 No. 2 Agustus 2025
Institut Bisnis Muhammadiyah Bekasi
P-ISSN: 2798-3129 , E-ISSN: 2798-6187

c. Pengujian *Grafana*

Pada tahapan pengujian *Grafana* dilakukan pengecekan apakah *Grafana* dapat menampilkan data yang telah disimpan oleh *Prometheus* sebagai sumber data. Secara *default Grafana* berjalan diatas *port TCP* 3000. *Grafana* akan menampilan data sesuai dengan *query* untuk mendapatkan metrik tertentu dan jenis visualisasi yang dipilih. Apabila tidak terdapat eror, maka sistem monitoring dapat berjalan normal dengan tampilan *dashboard Grafana* seperti terlihat pada Gambar 8.

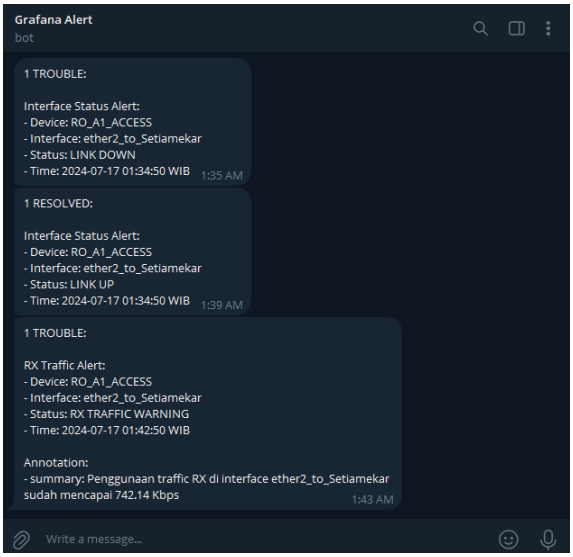


Gambar 8. *Dashboard Grafana*

d. Pengujian *Alert*

Pada pengujian ini apabila data atau nilai yang dimonitor memenuhi syarat dalam parameter *alert Grafana*, maka secara otomatis *alert* akan aktif dan mengirimkan notifikasi ke *Telegram*. Pada pengujian ini, parameter yang dibuat adalah sebagai berikut:

1. *Alert* akan aktif apabila status *interface* ke arah *router distribution* memiliki *value* 2 yang berarti *down*.
2. *Alert* akan aktif apabila *traffic RX/TX* ke arah *router distribution* mencapai batas *treshold* yang telah ditentukan.



Gambar 9. *Bot Telegram*

Pemeliharaan

Tahapan ini perlu dilakukan untuk mengelola dan membuat sistem monitoring jaringan yang telah dibuat dapat terjaga. Untuk memastikan bahwa sistem yang telah dibangun dan berjalan dengan baik dapat bertahan lama, maka kebijakan terkait pemeliharaan dan evaluasi sistem perlu dibuat atau diatur. Pada tahap ini dilakukan analisis dan evaluasi terhadap kelayakan sistem monitoring jaringan yang dibangun. Apabila terdapat eror dan kendala yang terjadi pada saat proses pengujian atau terdapat implementasi yang belum sesuai dengan kebutuhan dan rencana penelitian diawal, maka akan dilakukan perbaikan sistem.

PENUTUP

Simpulan

Berdasarkan pemaparan pada penelitian sistem monitoring jaringan yang telah peneliti uraikan, maka dapat ditarik kesimpulan bahwa:

1. Sistem monitoring jaringan dengan metode *Network Development Life Cycle* memberikan solusi untuk PT Lintas Teknologi Solusindo terkait kendala dalam memantau kondisi perangkat jaringan dan kendala dalam mendeteksi gangguan pada jaringan yang masih dilakukan secara manual yang bergantung pada aspek manusia.
2. Dengan *tools* monitoring *Prometheus* dan *Grafana* yang terintegrasi dengan *Telegram*, membuat durasi *detection time* saat terjadi gangguan menjadi lebih cepat sehingga proses *troubleshooting* tidak terlambat.

Saran

Untuk meningkatkan kualitas dan pembaruan pada sistem monitoring jaringan ini, maka peneliti mempunyai saran yaitu dengan membuat parameter atau konfigurasi agar sistem ini juga dapat memonitoring *log* pada *router Mikrotik* dan menambahkan platform untuk memberikan *alert* atau notifikasi melalui aplikasi *Whatsapp* yang lebih sering digunakan.

REFERENSI

Darmawan, R. T. (2023). Implementasi Grafana untuk Auto-Scaling pada NGINX Web-Server di Lingkungan Docker Container. *Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *7*(7), 3163–3167. https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/13051

Dira, S. R., Arif, M., & Ridha, F. (2022). Monitoring Kubernetes Cluster Menggunakan Prometheus dan Grafana. *Proceeding Applied Business and Engineering Conference*, *1*(1), 345–351. https://opac.lib.pcr.ac.id/index.php?p=show_detail&id=14600&keywords=

Fernando, N., Humaira, & Asri, E. (2020). Monitoring Jaringan dan Notifikasi dengan Telegram pada Dinas Komunikasi dan Informatika Kota Padang. *JITSI : Jurnal Ilmiah Teknologi Sistem Informasi*, *1*(4), 121–126. https://doi.org/10.30630/jitsi.1.4.17

Harnanta, K. J., Bhawiyuga, A., & Basuki, A. (2020). Implementasi MQTT Broker dengan Kemampuan Auto Scaling pada Internet of Things. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *4*(6), 1783–1792. http://j-ptiik.ub.ac.id

Ishaq, M. Y., & Firmansyah. (2023). Implementasi Sistem Monitoring Menggunakan

Zabbix Dan Notifikasi Realtime Telegram. *INSAN (Journal of Information Systems Management Innovation)*, *3*(1), 72–77. https://doi.org/https://doi.org/10.31294/jinsan.v3i2.2432

Lusi, D., & Belutowe, Y. S. (2023). Analisis Dan Implementasi Desain Jaringan Hotspot Berbasis Mikrotik Menggunakan Metode NDLC (Network Development Life Cycle) Pada Kantor Balai Pelaksanaan Jalan Nasional NTT. *Jurnal Teknologi Informasi*, *7*(1), 15–27. https://doi.org/https://doi.org/10.36294/jurti.v7i1.3454

Prasetyo, B., Budiman, E., & Putra, G. M. (2019). Implementasi Network Monitoring System (NMS) Sebagai Sistem Peringatan Dini Pada Router Mikrotik Dengan Layanan SMS Gateway (Studi Kasus: Universitas Mulawarman). *Prosiding Seminar Nasional Ilmu Komputer Dan Teknologi Informasi*, *4*(1), 6–10. https://e-journals.unmul.ac.id/index.php/SAKTI/article/view/2342/0

Prayogi, P. K., Orisa, M., & Ariwibisono, F. (2020). Rancang Bangun Sistem Monitoring Jaringan Access Point Menggunakan Simple Network Management Protocol (SNMP) Berbasis Web. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *4*(1), 192–197. https://doi.org/10.36040/jati.v4i1.2327

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. https://doi.org/10.30630/jitsi.1.4.19

Ramadoni, Amirudin, M. Z., Fahmi, R., Utami, E., & Mustafa, M. S. (2021). Evaluasi Penggunaan Prometheus dan Grafana Untuk Monitoring Database Mongodb. *Jurnal Informatika Polinema*, *7*(2), 43–50. https://doi.org/10.33795/jip.v7i2.530

Wiriaatmadja, M. F. J. S., & Ratama, N. (2022). Sistem Monitoring Jaringan Melalui Notifikasi Telegram Dengan Application Programming Interface (API) Menggunakan Netwatch Mikrotik Pada Jaringan. *OKTAL : Jurnal Ilmu Komputer Dan Sains*, *1*(6), 771–781. https://journal.mediapublikasi.id/index.php/oktal/article/view/192