**MOTEKAR: Jurnal Multidisiplin Teknologi dan Arsitektur**
**E-ISSN: 3025-227X P-ISSN: 3025-2288**
**Vol. 4 No. 2 November 2026**

# Rancangan Otomasi *Deployment Network Monitoring System* Menggunakan *Zabbix* Berbasis *Github* Dengan Integrasi Notifikasi Telegram pada PT Mahaga Pratama

**L Rizkyawal Akbar Putra[1] Toni[2] Didik Sulistyo Kurniawan[3]**
Program Studi Teknik Navigasi Udara, Politeknik Penerbangan Indonesia, Jl. Raya PLP Curug, Serdang Wetan, Kec. Legok, Kabupaten Tangerang, Banten, Indonesia[1,2,3]
Email: rizkyawallalu@gmail.com[1] toni@ppicurug.ac.id[2] dkurniawan@ppicurug.ac.id[3]

**Abstrak**

Abstrak Perkembangan teknologi informasi dan komunikasi menuntut Perusahaan untuk memiliki infrastruktur jaringan yang stabil, handal, dan efisien dalam mendukung operasional bisnis. Seiring dengan meningkatnya kompleksitas jaringan dan jumlah perangkat yang digunakan, potensi terjadinya gangguan jaringan juga semakin tinggi. Pada PT Mahaga Pratama, proses deployment sistem monitoring jaringan yang meliputi instalasi dan konfigurasi masih dilakukan secara manual serta belum terintegrasi dengan sistem notifikasi otomatis, sehingga proses konfigurasi kurang efisien dan berpotensi menimbulkan *human error*. Penelitian ini bertujuan untuk merancang dan mengimplementasikan otomasi *deployment Network Monitoring System* menggunakan Zabbix berbasis GitHub dengan integrasi notifikasi Telegram. Metode pengembangan sistem yang digunakan adalah metode Waterfall yang meliputi tahapan analisis kebutuhan, perancangan, implementasi, verifikasi, dan pemeliharaan. Infrastruktur sistem dibangun menggunakan *Virtual Machine* berbasis Nutanix AHV dengan sistem operasi Ubuntu Server. Proses otomasi diwujudkan melalui pembuatan *automation script* pada repositori GitHub untuk mengotomatisasi instalasi dan konfigurasi Docker serta Zabbix secara konsisten. Pengujian sistem dilakukan melalui implementasi, simulasi fungsional, serta verifikasi kestabilan sistem selama tujuh hari berturut-turut. Hasil penelitian menunjukkan bahwa sistem berhasil melakukan deployment otomatis menggunakan *automation script* serta mampu memantau parameter jaringan seperti CPU, memori, bandwidth, dan status perangkat secara *real-time*. Integrasi Telegram Webhook API berhasil mengirimkan notifikasi gangguan (*problem*) dan pemulihan (*recovery*) secara otomatis ke grup teknisi. Selain itu, implementasi sistem berhasil mengotomatisasi proses deployment dan monitoring jaringan sehingga konfigurasi menjadi lebih terstruktur dan konsisten serta memungkinkan administrator menerima notifikasi gangguan secara *real-time*.

**Kata Kunci:** *Network Monitoring,* Zabbix, GitHub, Telegram, Otomasi *Deployment*, *Waterfall*



## PENDAHULUAN

Perkembangan teknologi informasi dan komunikasi mendorong kebutuhan akan infrastruktur jaringan yang stabil, andal, dan efisien dalam mendukung operasional perusahaan berbasis layanan digital. PT Mahaga Pratama, sebagai bagian dari ekosistem PT Pasifik Satelit Nusantara (PSN) Group, mengandalkan berbagai perangkat jaringan dan server yang harus beroperasi secara berkesinambungan. Namun, proses monitoring dan deployment sistem pemantauan pada perusahaan tersebut masih dilakukan secara manual, mulai dari instalasi Docker, konfigurasi Zabbix, hingga pengaturan parameter monitoring, sehingga membutuhkan waktu lebih lama dan meningkatkan risiko *human error*, terutama saat dilakukan instalasi ulang atau penambahan perangkat baru. Kondisi ini turut berkontribusi pada beberapa kejadian *downtime* pada perangkat jaringan sebagaimana tercatat dalam data historis monitoring perusahaan, sehingga diperlukan sistem monitoring yang mampu mendeteksi gangguan secara cepat dan real-time.

Beberapa penelitian terdahulu telah membahas implementasi monitoring jaringan berbasis Zabbix. Penelitian mengenai monitoring jaringan internet di Kabupaten Jembrana menunjukkan bahwa integrasi Zabbix dengan notifikasi Telegram mampu mempercepat respons tim teknis dan meminimalkan downtime (Notifikasi Realtime et al., 2025). Penelitian serupa pada PT XYZ juga membuktikan bahwa kombinasi Zabbix dan Telegram efektif memantau parameter jaringan secara real-time serta mengirimkan notifikasi otomatis saat terjadi gangguan maupun pemulihan (Ichsan & Latifah, 2025). Sementara itu, penelitian Zabbix berbasis protokol SNMP menunjukkan keberhasilan sistem dalam menampilkan informasi traffic, bandwidth, dan interface secara akurat (Pradana et al., 2022). Hal ini sejalan dengan pandangan bahwa monitoring jaringan yang dilakukan secara real-time memungkinkan administrator mendeteksi gangguan secara dini dan mengambil tindakan preventif sebelum terjadi kegagalan sistem yang lebih besar (Aritonang & Jonas Simanullang, 2025).

Meskipun ketiga penelitian tersebut sama-sama memanfaatkan Zabbix sebagai platform monitoring dan Telegram sebagai media notifikasi, seluruhnya masih berfokus pada aspek pemantauan (*monitoring*) semata dan belum menyentuh aspek otomasi proses instalasi maupun konfigurasi. Proses *deployment* pada penelitian-penelitian tersebut masih dilakukan secara manual tahap demi tahap, sehingga kelemahan berupa waktu implementasi yang lama dan potensi kesalahan konfigurasi belum terselesaikan. Kesenjangan (*gap*) inilah yang menjadi celah penelitian: belum tersedia sistem yang mengintegrasikan otomasi *deployment* berbasis repositori (GitHub) dengan sistem monitoring Zabbix dan notifikasi Telegram dalam satu alur kerja otomatis. Padahal, otomasi deployment melalui *automation script* berpotensi menghasilkan proses instalasi yang lebih cepat, konsisten, dan dapat direplikasi, sekaligus mengurangi *human error* dibandingkan pendekatan manual, hal yang menjadi urgensi utama bagi perusahaan dengan jumlah perangkat dan kompleksitas jaringan yang terus meningkat.

Berdasarkan kesenjangan tersebut, penelitian ini menawarkan kebaruan berupa integrasi GitHub sebagai repositori *automation script*, Docker sebagai platform *deployment*, Zabbix sebagai sistem monitoring, dan Telegram sebagai media notifikasi dalam satu alur kerja otomatis, sehingga proses instalasi, konfigurasi, hingga pengiriman notifikasi dapat dilakukan tanpa intervensi manual bertahap. Dengan demikian, penelitian ini bertujuan untuk: (1) merancang sistem otomasi *deployment* Network Monitoring System menggunakan Zabbix berbasis GitHub dengan integrasi notifikasi Telegram; (2) mengimplementasikan sistem tersebut pada lingkungan virtual menggunakan Docker dan Ubuntu Server; serta (3) mengetahui hasil implementasi dan pengujian sistem berdasarkan fungsi monitoring perangkat jaringan dan pengiriman notifikasi Telegram.

## Penelitian Terdahulu yang Relavan

**Tabel 1. Peneliti Terdahulu**

| Penelitian Terdahulu | Penulis | Tahun | Hasil Penelitian | Persamaan Penelitian | Perbedaan Penelitian |
|---|---|---|---|---|---|
| Implementasi Sistem Monitoring Menggunakan Zabbix Dan Notifikasi Realtime Telegram Untuk Jaringan Internet Di Kabupaten Jembrana | (Notifikasi Realtime et al., 2025) | 2025 | Penelitian ini menunjukkan bahwa sistem monitoring jaringan berbasis Zabbix dengan notifikasi Telegram mampu meningkatkan respons tim teknis, mempercepat penanganan gangguan, dan meminimalkan downtime sehingga | Sama-sama membahas network monitoring system menggunakan Zabbix Sama sama menggunakan neotifikasi Telegram untuk | fokus pada implementasi monitoring untuk memantau router, hanya Zabbix + Telegram belum ada sistem monitoring efektif, belum otomatis masih menggunakan *deployment* manual |

| | | | | | |
|---|---|---|---|---|---|
| | | | pengelolaan jaringan menjadi lebih efektif dan proaktif. | informasi gangguan Sama-sama menekankan monitoring real-time | |
| Press-Ok Monitoring Network Menggunakan Zabbix Dengan Alert Notifikasi Via Telegram Pada Pt XYZ | (Ichsan & Latifah, 2025) | 2025 | Sistem monitoring jaringan berbasis Zabbix yang terintegrasi dengan Telegram berhasil berjalan dengan baik, mampu memantau perangkat dan parameter jaringan secara real-time, serta memberikan notifikasi otomatis saat terjadi gangguan maupun pemulihan. Implementasi ini terbukti meningkatkan efisiensi pengelolaan jaringan dan mempercepat waktu respons terhadapinsiden. | Sama-sama menggunakan Zabbix sebagai sistem monitoring jaringan, Sama-sama terintegrasi dengan Telegram untuk notifikasi otomatis, Monitoring dilakukan secara real-time | fokus pada implementasi sistem monitoring, monitoring masih manual, Zabbix + Telegram saja, belum otomatis masih menggunakan *deployment* manual |
| Jurnal Teknologi Informasi, Volume 19 No. 2 Agustus 2022, 248-262 ISSN 1693-8348 E-ISSN 2615-7128 Implementasi Sistem Monitoring Jaringan Menggunakan Zabbix Berbasis SNMP | (Pradana et al., 2022) | 2022 | Sistem monitoring jaringan berbasis Zabbix dengan protokol SNMP berhasil diimplementasikan dan mampu menampilkan informasi jaringan seperti traffic, bandwidth, dan interface secara akurat. Sistem ini membantu administrator dalam memantau jaringan serta mempercepat deteksi dan penanganan gangguan. | Sama-sama menggunakan Zabbix sebagai sistem monitoring jaringan, Sama-sama melakukan monitoring jaringan secara real-time, Bertujuan meningkatkan efisiensi pengelolaan jaringan | fokus pada monitoring berbasis SNMP pada router, Zabbix + SNMP, hanya monitoring dan visualisasi data jaringan, kurangnya admin & belum ada sistem monitoring, belum otomatis masih menggunakan *deployment* manual |

## METODE PENELITIAN

Metode Penelitian ini menggunakan metode *waterfall* karena pendekatannya yang terstruktur, sistematis, dan berurutan sesuai dengan kebutuhan sistem yang telah dapat didefinisikan secara jelas sejak awal, yaitu kebutuhan monitoring, integrasi notifikasi, dan alur *deployment*. Model ini terdiri atas lima tahapan: analisis kebutuhan, desain, implementasi, verifikasi, dan pemeliharaan (Saravanos & Curinga, 2023), di mana setiap tahap harus diselesaikan secara lengkap sebelum berlanjut ke tahap berikutnya sehingga meminimalkan kesalahan dan menghasilkan dokumentasi pengembangan yang terorganisir (Saputra et al., 2024).

**Analisis Kebutuhan**

Tahap ini mengidentifikasi kebutuhan perangkat keras, perangkat lunak, dan fungsional sistem. Kebutuhan perangkat keras meliputi server virtual berbasis Nutanix AHV, perangkat jaringan (router dan switch) sebagai objek monitoring, serta perangkat administrator untuk konfigurasi dan akses dashboard. Kebutuhan perangkat lunak meliputi Ubuntu Server, Docker, Zabbix, GitHub (repositori *automation script*), Telegram (notifikasi real-time), dan web browser. Sistem dirancang untuk memantau parameter CPU, memori, status perangkat, trafik antarmuka jaringan, *interface speed*, dan ICMP Ping secara real-time. Data pendukung dikumpulkan melalui studi literatur dari jurnal ilmiah dan penelitian terdahulu yang relevan.

**Desain**

Perancangan sistem meliputi pemodelan proses menggunakan *Data Flow Diagram* (DFD) serta perancangan topologi jaringan yang mengintegrasikan Zabbix sebagai server monitoring, GitHub untuk otomasi *deployment*, dan Telegram sebagai media notifikasi. Rancangan ini merupakan pengembangan dari sistem monitoring manual sebelumnya, di mana administrator harus memantau kondisi perangkat secara langsung melalui dashboard tanpa mekanisme notifikasi otomatis. Penelitian ini mengusulkan penambahan mekanisme otomatisasi melalui integrasi Zabbix, GitHub, dan Telegram Bot untuk mendukung proses konfigurasi, *deployment*, dan penyampaian notifikasi gangguan secara real-time. Perbandingan arsitektur sistem manual (*existing system*) dan sistem otomatis (*proposed system*) disajikan untuk menunjukkan penambahan komponen otomatisasi yang diusulkan; hasil implementasi dan pengujiannya dibahas pada bagian Hasil dan Pembahasan.

**Implementasi**

Sistem direalisasikan pada lingkungan virtual menggunakan *Virtual Machine* (VM) berbasis Nutanix dengan sistem operasi Ubuntu Server. Docker diimplementasikan sebagai platform *container* dan Zabbix sebagai aplikasi monitoring, dengan proses instalasi dan konfigurasi dilakukan secara otomatis melalui *automation script* yang dikelola pada repositori GitHub. Perangkat jaringan dikonfigurasi agar dapat dipantau oleh Zabbix menggunakan protokol SNMP, kemudian sistem diintegrasikan dengan Telegram Bot agar notifikasi *problem* maupun *recovery* dapat dikirimkan secara real-time. Parameter yang dipantau meliputi *uptime*, penggunaan CPU dan memori, trafik antarmuka jaringan (*bits received* dan *bits sent*), *interface speed*, serta ICMP Ping yang ditampilkan melalui dashboard Zabbix.

**Verifikasi**

Pengujian dilakukan menggunakan metode *blackbox testing* yang berfokus pada fungsi sistem, meliputi tampilan dashboard monitoring, status perangkat, dan notifikasi Telegram saat terjadi kondisi *problem* maupun *recovery*. Selain pengujian fungsional, verifikasi operasional sistem dilakukan melalui pemantauan kestabilan layanan monitoring, performa VM, dan kondisi server selama tujuh hari berturut-turut menggunakan perintah Linux seperti date, uptime, df -h, free -h, dan top.

**Pemeliharaan**

Tahap akhir mencakup pemantauan kinerja server, pemeriksaan layanan Docker, pemeliharaan repositori GitHub, pengujian notifikasi Telegram, pencadangan konfigurasi sistem, serta perbaikan bug dan pembaruan sistem guna menjaga kestabilan dan keandalan sistem monitoring secara berkelanjutan.

## HASIL PENELITIAN DAN PEMBAHASAN

### Hasil Penelitian

#### Gambaran Umum Sistem

Penelitian ini menghasilkan sistem otomasi *deployment* untuk memantau kinerja jaringan secara real-time menggunakan Zabbix, dengan dukungan platform GitHub sebagai repositori *automation script* dan integrasi notifikasi Telegram. Seluruh script instalasi dan konfigurasi disimpan pada repositori GitHub sehingga teknisi dapat mengunduh dan menjalankan sistem tanpa konfigurasi manual yang kompleks. Setelah script dijalankan, sistem secara otomatis menginstal dan mengonfigurasi Zabbix Server, Zabbix Agent, serta menghubungkan perangkat jaringan ke dalam sistem monitoring, sekaligus mengaktifkan integrasi Telegram Bot untuk pengiriman notifikasi gangguan secara real-time. Proses monitoring dilakukan dengan mengumpulkan data perangkat jaringan melalui protokol SNMP dan Zabbix Agent mencakup penggunaan CPU, memori, disk, serta status perangkat yang kemudian diproses oleh Zabbix Server dan ditampilkan pada dashboard berbasis web.

#### Implementasi Sistem

Kebutuhan sistem terdiri atas perangkat keras (laptop, router, switch, dan platform virtualisasi Nutanix) dan perangkat lunak (Ubuntu Server 22.04, Docker, SNMP, Zabbix, GitHub, dan Telegram Bot). Arsitektur sistem dikembangkan dari NMS manual di mana Zabbix Server, PostgreSQL Database, dan Zabbix Web berjalan dalam lingkungan Docker tanpa mekanisme otomasi maupun notifikasi menjadi NMS otomatis dengan penambahan komponen Automation Script dan GitHub Repository yang saling terintegrasi dengan Zabbix dan Telegram Bot. Implementasi dilakukan dengan menyusun beberapa berkas kunci pada repositori GitHub: install.sh untuk otomasi instalasi Docker, docker-compose.yaml untuk orkestrasi container (PostgreSQL, Zabbix Server, Zabbix Web), init.sh dan zabbix_api.sh untuk inisialisasi konfigurasi Zabbix melalui API secara otomatis, serta env dan.gitignore untuk pengelolaan kredensial secara aman. Proses deployment cukup dilakukan melalui tiga perintah utama: *clone* repositori, pemberian izin eksekusi (chmod +x install.sh), dan menjalankan script (./install.sh), yang selanjutnya melakukan instalasi Docker Engine, penarikan *image* Zabbix dan PostgreSQL dari Docker Hub, hingga menjalankan seluruh container secara otomatis. Setelah deployment, dilakukan konfigurasi host monitoring menggunakan template *Mikrotik by SNMP*, pembuatan dashboard, serta penambahan widget untuk memantau uptime, trafik interface (*bits received/sent*), grafik trafik dua jalur ISP, dan status ICMP Ping. Integrasi Telegram dikonfigurasi melalui pembuatan bot via BotFather, penentuan *chat ID* grup, dan pengaturan *media type* Webhook pada Zabbix sehingga notifikasi *problem* dan *recovery* dapat terkirim otomatis.

#### Hasil Pengujian Fungsional

Pengujian integrasi Telegram menunjukkan bahwa sistem mampu mengirimkan notifikasi sesuai kondisi aktual perangkat: status *Normal* saat perangkat berjalan baik, status *Problem* dengan tingkat keparahan *High* saat terjadi gangguan (simulasi *port down*), dan status *Recovery* saat kondisi kembali normal. Hasil ini membuktikan bahwa integrasi Zabbix–Telegram melalui Webhook API berfungsi sebagaimana dirancang.

#### Hasil Verifikasi Sistem (7 Hari)

Verifikasi dilakukan selama tujuh hari berturut-turut dengan pengecekan setiap 24 jam terhadap kondisi server, status perangkat, notifikasi Telegram, dan penggunaan sumber daya. Ringkasan hasil disajikan pada Tabel berikut.

**Tabel 2. Hasil Pengujian**

| No. | Waktu | Aktivitas Pengujian | Hasil yang Diharapkan | Hasil Pengujian | |
|---|---|---|---|---|---|
| | | | | Sesuai Harapan | Tidak Sesuai Harapan |
| 1. | Hari ke-1 (24 jam) | Melakukan pengecekan sistem pada hari pertama dengan memverifikasi dashboard Zabbix, pengiriman notifikasi Telegram, serta kondisi server meliputi uptime, waktu sistem, kapasitas penyimpanan, penggunaan CPU, dan memori. | Seluruh layanan monitoring berjalan normal dan kondisi server stabil. | Sesuai harapan. | |
| 2. | Hari ke-2 (48 jam) | Melakukan pengecekan sistem pada hari kedua dengan memverifikasi dashboard Zabbix, pengiriman notifikasi Telegram, serta kondisi server. | Sistem tetap berjalan normal tanpa gangguan. | Sesuai harapan | |
| 3. | Hari ke-3 (72 jam ) | Melakukan pengecekan sistem pada hari ketiga untuk memastikan dashboard Zabbix, notifikasi Telegram, dan kondisi server tetap berfungsi dengan baik. | Sistem monitoring tetap stabil dan seluruh layanan aktif. | Sesuai harapan. | |
| 4. | Hari ke-4 (96 jam ) | Melakukan pengecekan sistem pada hari keempat dengan memverifikasi fungsi monitoring dan kondisi sumber daya server. | Sistem tetap beroperasi secara normal. | Sesuai harapan. | |
| 5. | Hari ke-5 (120 jam) | Melakukan pengecekan sistem pada hari kelima untuk memastikan monitoring, notifikasi, dan kondisi server tetap stabil. | Tidak ditemukan gangguan pada sistem. | Sesuai harapan. | |
| 6. | Hari ke-6 (144 jam) | Melakukan pengecekan sistem pada hari keenam dengan memverifikasi dashboard Zabbix, notifikasi Telegram, dan kondisi server. | Seluruh layanan tetap berjalan dengan baik. | Sesuai harapan. | |
| 7. | Hari ke-7 (168 j | Melakukan pengecekan sistem pada hari ketujuh sebagai verifikasi akhir terhadap kestabilan sistem monitoring dan kondisi server. | Sistem tetap stabil selama tujuh hari berturut-turut. | Sesuai harapan. | |

**Tabel 3. Hasil Pengujian Monitoring pada Zabbix**

| Hari | Uptime Router | Uptime Switch | Bits Received | Bits Sent | Memory Utilization | CPU Utilization | Interface Speed | ICMP Ping |
|---|---|---|---|---|---|---|---|---|
| **Hari-1** | 226 hari 10:47:35 | 326 hari 14:01:37 | 191,32 Kbps (↓) | 1,45 Mbps (↑) | 4,51% (↓) | 50,00% | 1,00 Gbps | Up (1.00) |
| **Hari-2** | 227 hari 20:14:05 | 327 hari 23:28:07 | 136,20 Kbps (↓) | 506,86 Kbps (↓) | 4,66% (↑) | 51,00% | 1,00 Gbps | Up (1.00) |
| **Hari-3** | 228 hari 17:53:05 | 328 hari 21:07:07 | 225,64 Kbps (↑) | 836,98 Kbps (↓) | 4,68% (↑) | 49,00% | 1,00 Gbps | Up (1.00) |
| **Hari-4** | 229 hari 14:15:35 | 329 hari 17:29:37 | 495,86 Kbps (↑) | 1,65 Mbps (↓) | 4,61% (↑) | 49,00% | 1,00 Gbps | Up (1.00) |
| **Hari-5** | 230 hari 10:04:36 | 330 hari 13:18:37 | 845,34 Kbps (↓) | 2,23 Mbps (↓) | 4,64% (↓) | 50,00% | 1,00 Gbps | Up (1.00) |
| **Hari-6** | 231 hari 17:25:05 | 331 hari 20:39:07 | 445,67 Kbps (↑) | 511,82 Kbps (↑) | 4,63% (↑) | 50,00% | 1,00 Gbps | Up (1.00) |
| **Hari-7** | 232 hari 02:04:36 | 332 hari 05:18:37 | 29,03 Kbps (↓) | 138,48 Kbps (↑) | 4,70% (↓) | 49,00% | 1,00 Gbps | Up (1.00) |

Seluruh parameter monitoring status router dan switch, CPU, memori, *interface speed*, serta ICMP Ping menunjukkan kondisi stabil sepanjang tujuh hari pengujian tanpa gangguan signifikan. Fluktuasi pada *bits received/sent* mencerminkan dinamika trafik jaringan normal, bukan indikasi gangguan sistem. Isi tabel, grafik, dan gambar perlu dijelaskan. Judul tabel dicetak tebal dengan ukuran 10 pt dan diletakkan di bagian atas. Sedangkan judul gambar diletakkan di bagian bawah dengan ukuran 10 pt dan dicetak tebal.

### Analisis dan Perbandingan dengan Sistem Manual

Perbandingan antara metode konfigurasi manual dan otomatis menunjukkan perbedaan signifikan dari sisi proses kerja. Pada metode manual, administrator harus menjalankan setiap tahapan instalasi—pembaruan sistem operasi, instalasi Docker, konfigurasi repository, hingga menjalankan layanan monitoring—satu per satu melalui CLI, sehingga rentan terhadap kesalahan konfigurasi (*human error*) dan memerlukan pengulangan penuh apabila terjadi instalasi ulang atau penambahan server baru. Sebaliknya, sistem otomasi yang dikembangkan memungkinkan seluruh proses instalasi dan konfigurasi awal dijalankan melalui satu *automation script* yang tersimpan pada GitHub, sehingga proses deployment menjadi lebih cepat, konsisten, dan terstruktur, sekaligus mengurangi interaksi manual administrator. Selain itu, GitHub berperan sebagai pusat *version control* yang memudahkan pemeliharaan, pembaruan versi, dan deployment ulang di kemudian hari. Meskipun demikian, penelitian ini tidak melakukan pengukuran kuantitatif terhadap selisih waktu maupun tingkat efisiensi antara kedua metode tersebut, sehingga klaim peningkatan efisiensi bersifat kualitatif berdasarkan pengamatan proses.

### Keterbatasan Penelitian

Beberapa keterbatasan diidentifikasi dalam penelitian ini: (1) GitHub baru dimanfaatkan sebagai repositori script dan *version control*, belum menerapkan mekanisme *Continuous Integration/Continuous Deployment* (CI/CD) otomatis; (2) kesalahan pada *automation script* berpotensi diterapkan secara otomatis ke seluruh sistem sehingga memerlukan pengujian script sebelum digunakan; (3) seluruh layanan container bergantung pada sistem operasi host tanpa mekanisme *high availability* atau *clustering*; dan (4) aspek keamanan sistem—seperti enkripsi data, *role-based access control*, dan pengamanan repositori—belum menjadi ruang lingkup penelitian ini dan dapat menjadi arah pengembangan selanjutnya.

## KESIMPULAN

Penelitian ini berhasil merancang dan mengimplementasikan sistem otomasi *deployment* Network Monitoring System menggunakan Zabbix berbasis GitHub dengan integrasi notifikasi Telegram. *Automation script* yang tersimpan pada repositori GitHub terbukti mampu mengotomatisasi proses instalasi dan konfigurasi, sehingga tahapan deployment menjadi lebih terstruktur, konsisten, dan mudah dikelola dibandingkan metode manual. Sistem berhasil diimplementasikan pada lingkungan virtual berbasis Ubuntu Server dan Docker, dengan seluruh layanan—Zabbix Server, PostgreSQL Database, Zabbix Web, dan Zabbix Agent—berjalan dengan baik pada container Docker, serta perangkat jaringan (router dan switch) berhasil terhubung ke sistem monitoring. Hasil pengujian menunjukkan bahwa sistem mampu memantau status perangkat, penggunaan CPU dan memori, trafik antarmuka, *interface speed*, dan ICMP Ping secara real-time, sekaligus mengirimkan notifikasi *problem* dan *recovery* secara otomatis melalui Telegram. Verifikasi selama tujuh hari berturut-turut membuktikan bahwa seluruh layanan berjalan stabil sesuai kebutuhan penelitian, sehingga ketiga tujuan penelitian—perancangan, implementasi, dan pengujian sistem—telah tercapai.

Beberapa saran diajukan untuk pengembangan penelitian selanjutnya. Pertama, implementasi langsung pada lingkungan operasional perusahaan perlu dilakukan agar kinerja sistem dapat diuji pada kondisi jaringan yang sebenarnya, mengingat penelitian ini masih berada pada tahap simulasi di lingkungan virtual. Kedua, pengembangan dapat diarahkan pada penambahan fitur *auto-discovery* perangkat, *auto-registration* host, backup konfigurasi otomatis, serta integrasi mekanisme *Continuous Integration/Continuous Deployment* (CI/CD) agar proses deployment dan pengelolaan sistem menjadi lebih efisien. Ketiga, sistem dapat

dilengkapi dengan fitur pelaporan (*report*) kondisi perangkat secara berkala—harian, mingguan, atau bulanan yang mencakup status perangkat, penggunaan CPU dan memori, trafik jaringan, serta tingkat ketersediaan (*availability*), guna membantu administrator dalam evaluasi dan dokumentasi kondisi jaringan secara berkelanjutan.

**DAFTAR PUSTAKA**

Ardani, A. H., Azhar, R., & Asroni, O. (2026). Sistem Monitoring Multi Perangkat Jaringan Berbasis Simple Network Management Protocol (SNMP) dengan Notifikasi Telegram. RIGGS: Journal of Artificial Intelligence and Digital Business, 5(1), 7355–7365. https://doi.org/10.31004/riggs.v5i1.7125

Aritonang, M. A. S., & Jonas Simanullang, M. (2025). Penerapan Network Monitoring System Berbasis SNMP untuk Deteksi Dini Gangguan Jaringan. Jurnal Pustaka AI (Pusat Akses Kajian Teknologi Artificial Intelligence), 5(3), 735–742. https://doi.org/10.55382/jurnalpustakaai.v5i3.1383

Cahyani Pradita, R., Dwi Herlambang, A., & Afirianto, T. (2025). Pengaruh Implementasi Problem-Based Learning Berbantuan Github Dan Chatgpt Terhadap Hasil Belajar Dan Kemampuan Kolaborasi (Vol. 9, Number 3). Http://J-Ptiik.Ub.Ac.Id

Cahyo, A. B., Hariadi, T. K., & Ardiyanto, Y. (N.D.). Implementasi Zabbix Server Untuk Memonitor Kondisi Jaringan Komputer Di Dinas Komunikasi Dan Informatika Kabupaten Pekalongan.

Dewantara, A. B., & Kholil, D. M. (2015). Sistem Otomasi Sebagai Upaya Perbaikan Kualitas Dengan Metode Spc Pada Line Finishing (Studi Kasus: Pt. X). In Jurnal Ilmiah Teknik Industri (Vol. 3, Number 3).

Dwiyatno, S., Rakhmat, E., & Gustiawan, O. (2020). Implementasi Virtualisasi Server Berbasis Docker Container. 7(2).

Harnanta, K. J., Bhawiyuga, A., & Basuki, A. (2020). Implementasi MQTT Broker dengan Kemampuan Auto Scaling pada Internet of Things (Vol. 4, Number 6). http://j-ptiik.ub.ac.id

Hyun, G., Oak, J., Kim, D., & Kim, K. (2024). The Impact of an Automation System Built with Jenkins on the Efficiency of Container-Based System Deployment. Sensors, 24(18). https://doi.org/10.3390/s24186002

Khalida, R., Muhajirin, A., Setiawati, S., Raya, J., Raya, J., & Utara, P. B. (n.d.). Teknis Kerja Docker Container untuk Optimalisasi Penyebaran Aplikasi.

Kridatama, J., & Dan Teknologi Penerapan, S. (n.d.). Penerapan Metode Waterfall pada Sistem Informasi Pendaftaran Kursus Di LKP Indo Jaya Kebumen (Vol. 02).

Maulana, F. (2016). Implementasi Simple Network Management Protocol (Snmp) Pada Aplikasi Monitoring Jaringan Berbasis Website(Studi Kasus Universitas Muhammadiyah Bengkulu). In Jurnal Informatika (Vol. 16, Number 2).

Maulana, I., Sanjaya, H. R., Setiyansyah, F., Wibowo, D. R., & Sinlae, F. (2024). Sistem Operasi Pada Komputer Yang Paling Banyak Digunakan. ARembeN Jurnal Pengabdian Multidisiplin, 2(1), 9–17. https://doi.org/10.69688/aremben.v2i1.49

Nugroho, M., Affandi, A., & Rahardjo, D. S. (2014). Rancang Bangun Aplikasi Monitoring Jaringan Menggunakan SNMP ( Simple Network Management Protocol ) dengan Sistem Peringatan Dini dan Mapping Jaringan. 3(1), 35–39.

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.

Pratomo, B. A., Zaini, A. F. R., Teja, A. R., Prinandika, A. G., Fadhila, F. D., Arsyad, H., Fikriansyah,

I., Al Kanza, K. P., Vinorian, M. E., Diani, N. A., Pramudya, R. R., Ahmad, T., Santoso, B. J., Studiawan, H., Shiddiqi, A. M., Alzamzami, Moch. N., Anggoro, R., Djanali, S., Ijtihadie, R. M., & Suadi, W. (2024). Pelatihan Deployment Aplikasi Berbasis Website SMK Pawiyatan Surabaya. Sewagati, 8(5), 2168–2175. https://doi.org/10.12962/j26139960.v8i5.2091

Prayogi, P. K., Orisa, M., & Ariwibisono, F. X. (2020). Rancang Bangun Sistem Monitoring Jaringan Access Point Menggunakan Simple Network Management Protocol (Snmp) Berbasis Web. In Jurnal Mahasiswa Teknik Informatika (Vol. 4, Number 1).

Saputra, R. W., Pirera, Q., Valensia Verdana, V., Raya, P., Yos Sudarso, J., Palangkaraya, K., & Tengah, K. (2024). Analisis Resiko Penggunaan Metode Waterfall Dan Prototyping Dalam Pengembangan Website. In Jurnal Mahasiswa Teknik Informatika (Vol. 8, Number 4).

Saravanos, A., & Curinga, M. X. (2023). Simulating the Software Development Lifecycle: The Waterfall Model. Applied System Innovation, 6(6). https://doi.org/10.3390/asi6060108

scripting_xxi. (n.d.).

Wahid, A. A. (2019a). Analisis Sistem Keamanan Pada Sistem Operasi Microsoft Windows, Linux Dan Macintosh. In Jurnal Teknik Informatika (Vol. 1, Number 3).

Wahid, A. A. (2019b). Analisis Sistem Keamanan Pada Sistem Operasi Microsoft Windows, Linux Dan Macintosh. In Jurnal Teknik Informatika (Vol. 1, Number 3).

Yanuar Ishaq, M. (2023). Implementasi Sistem Monitoring Menggunakan Zabbix Dan Notifikasi Realtime Telegram. Jurnal Insan (Journal Of Information Systems Management Innovation, 3(1). Http://Jurnal.Bsi.Ac.Id/Index.Php/Jinsan

Yunianto, I., Adhiyarta, K., Bisnis, I., Bekasi, M., Budi, U., & Jakarta, L. (N.D.). Jurnal Review: Perbandingan Sistem Operasi Linux Dengan Sistem Operasi Windows.