International Journal on Perceptive and Cognitive Computing (IJPCC) Vol 12, Issue 2 (2026)
https://doi.org/10.31436/ijpcc.v12i2.705

# Debouncing-Based Automated Network Failover System with Multi-Channel Real-Time Notifications

Febrian Wahyu Christanto[1*], Samuel Vittorio Rivaldo[1], Firdaus Laia[2], Eryan Ahmad Firdaus[3]
[1]Department of Information Technology, Universitas Semarang, Semarang, Indonesia
[2]Department of Computer Science, Universitas Nias Raya, South Nias, Indonesia
[3]Department of Informatics, Indonesia Defense University, Bogor, Indonesia

*Corresponding Author: febrian.wahyu.christanto@usm.ac.id
(Received: 10th May 2026; Accepted: 17th June, 2026; Published on-line: 30th July, 2026)

***Abstract***— Internet connectivity failures in single-ISP environments may interrupt business operations, increase network recovery time, and reduce service availability. This research proposes a debouncing-based automated network failover system integrated with multi-channel real-time notifications to improve failover stability and operational responsiveness in small-to-medium business environments. The proposed system was implemented on a MikroTik RouterOS platform by integrating Netwatch, Scheduler, automated routing scripts, Telegram Bot API, and email notification services. Unlike conventional failover implementations that rely on single connectivity checks, the proposed approach incorporates a debouncing-based verification mechanism and edge-triggered notification logic to prevent false failover decisions, reduce route flapping caused by transient network disturbances, and minimize redundant notification delivery. The system development followed the PPDIOO (Prepare, Plan, Design, Implement, Operate, and Optimize) methodology to ensure a structured implementation and evaluation process. Experimental evaluation was conducted under real operational conditions by measuring failover recovery time, notification delay, failover stability, and client-side connectivity recovery during ISP transitions. The proposed system achieved an average failover recovery time of 4.2 seconds, while Telegram and email notifications recorded average delivery delays of 2.22 seconds and 8.61 seconds, respectively. Furthermore, connection tracking optimization improved effective client-side recovery by reducing session persistence issues following route switching. The implementation successfully reduced administrator intervention during network failures while improving service continuity, monitoring efficiency, and operational reliability. These results demonstrate that integrating debouncing-based failover control with edge-triggered multi-channel notifications provides an effective and practical solution for enhancing network availability and resilience in operational business networks.

***Keywords***—Failover System, Network Monitoring, MikroTik, Debouncing Mechanism, Real-time Notification

## I. INTRODUCTION

Reliable internet connectivity has become an essential requirement for modern business operations, particularly for organizations that depend on digital transactions, cloud-based services, and real-time communication. In Indonesia, internet penetration continues to increase significantly, reaching more than 80% of the national population in recent years [1], [2]. This growing dependence on internet infrastructure also increases the operational impact of network downtime, especially in business environments with limited redundancy mechanisms [3], [4]. Kampoeng Semarang, a souvenir and handicraft business center located in Semarang, Indonesia, relies heavily on internet connectivity to support daily operational activities. The existing network infrastructure utilizes a single Internet Service Provider (ISP) with a bandwidth capacity of 150 Mbps serving approximately 30 active users per day. However, the absence of an automated backup connection mechanism causes operational disruptions whenever the primary ISP experiences connectivity failure. Field observations indicated that network interruptions occurred between 20 and 40 times per month, with manual recovery processes requiring approximately 5 to 20 minutes depending on administrator response time and ISP handling.

In this research, network monitoring refers to the continuous observation of internet connectivity and network operational status to determine whether failover actions should be executed. The monitoring mechanism utilizes ICMP reachability tests implemented through MikroTik Netwatch, while repeated verification is performed using Scheduler to distinguish temporary packet loss from actual link failures. Operational monitoring parameters include host reachability, packet response status, failover recovery time, notification delay, and client-side connectivity recovery after ISP transitions.

However, practical failover implementation still faces several operational challenges. In real-world networking

environments, transient packet loss, unstable latency, and intermittent ISP connectivity may trigger false failover events if monitoring decisions rely solely on single connectivity checks [5], [6], [7]. Such conditions can produce route flapping behavior, where the network repeatedly switches between primary and backup links within short periods. Frequent route transitions not only degrade user experience but may also introduce session persistence problems, connection tracking inconsistencies, and delayed client-side recovery. Furthermore, repeated notifications generated during unstable conditions may reduce administrator responsiveness and increase operational noise. Therefore, failover systems require not only rapid switching capability but also stable decision-making mechanisms capable of distinguishing temporary disturbances from actual link failures [8], [9], [10].

To address these limitations, this research proposes a debouncing-based automated network failover system integrated with multi-channel real-time notifications. The proposed system introduces a debouncing mechanism using repeated verification through MikroTik Scheduler to prevent unstable failover decisions caused by temporary packet loss or intermittent connectivity fluctuations. In addition, an edge-triggered notification strategy is implemented to ensure that notifications are only transmitted during actual state transitions between UP and DOWN conditions, thereby minimizing redundant alerts [11]. The system was developed using the PPDIOO (Prepare, Plan, Design, Implement, Operate, and Optimize) framework to ensure structured deployment, operational validation, and optimization processes [12], [13], [14]. The proposed architecture integrates MikroTik Netwatch, automated routing scripts, Telegram Bot API, and email notification services to provide stable failover behavior and real-time operational awareness.

This research contributes to the field of network reliability and operational monitoring in three main aspects: (1) the implementation of a debouncing-based anti-flapping failover mechanism, (2) the integration of edge-triggered multi-channel notification delivery, and (3) operational evaluation in a real-world small-to-medium business environment. Experimental results demonstrate improvements in failover consistency, recovery responsiveness, and client-side connection stability during ISP transitions.

## II. Related Work

Several researches have implemented failover mechanisms and network monitoring systems to improve network availability and reduce downtime caused by ISP failures. Bustomi and Tahir [15] implemented a MikroTik-based failover system using a GSM backup connection to maintain service continuity during primary ISP disruptions. Similarly, Putra et. al. [16] applied a Netwatch-based failover mechanism to automatically switch traffic to a backup ISP when connectivity problems occurred.

Recent researches have emphasized that increasing internet dependency requires network infrastructures capable of maintaining high availability and service continuity under dynamic operational conditions. Howell [4] discussed internet ecosystem resilience, while Elsayed and Abohany [7] highlighted the importance of resilient enterprise network architectures to reduce operational disruption during connectivity failures. These studies underline the necessity of automated recovery mechanisms for maintaining business continuity. Previous studies have also investigated intelligent failover decision mechanisms. Anbalagan [8] proposed proactive failover automation for mission-critical systems, whereas Ryu et al. [9] introduced distributed failover controllers with embedded monitoring capabilities. Although these studies improve failover reliability, they do not specifically address route flapping caused by transient packet loss in MikroTik-based operational environments.

Other researches focused on monitoring and notification systems to improve administrator responsiveness. Zikra et. al. [17] developed a MikroTik-based monitoring system integrated with Telegram and email notifications to accelerate network issue detection. Vingestin et. al. [18] and Nurjanah et. al. [19] also utilized Telegram Bot integration for real-time monitoring notifications and demonstrated that Telegram-based alerts could improve operational awareness during network disturbances. Artificial Intelligence has also been explored to improve high-availability systems. Fotia et al. [10] reviewed AI-based approaches for improving system availability and fault tolerance. Nevertheless, most existing approaches focus on large-scale infrastructures and require more sophisticated architectures than those commonly deployed in small-to-medium business environments. Although previous researches successfully improved network monitoring and failover automation, most implementations still rely on single connectivity checks for failover decisions. This approach may cause route flapping during transient network instability and generate repetitive notifications under unstable conditions. In addition, previous works mainly focused on successful route switching without evaluating effective client-side recovery after ISP transitions. Notification effectiveness has also become an important aspect of operational monitoring systems. Li et al. [11] demonstrated that notification relevance significantly affects administrator responsiveness, indicating that reducing redundant alerts is essential for improving operational awareness.

https://doi.org/10.31436/ijpcc.v12i2.705

TABLE I
COMPARISON OF PREVIOUS RESEARCHES AND PROPOSED SYSTEM

| R | F | M | A | C | O |
|---|---|---|---|---|---|
| Hoesiin and Yuliandi [6] | ✓ | ✗ | ✓ | ✗ | ✗ |
| Anbalagan [8] | ✓ | ✗ | ✓ | ✗ | ✗ |
| Ryu et. al. [9] | ✓ | ✗ | ✓ | ✗ | ✗ |
| Bustomi and Tahir [15] | ✓ | ✗ | ✗ | ✗ | ✗ |
| Putra et. al. [16] | ✗ | ✓ | ✗ | ✗ | ✗ |
| Zikra et. al. [17] | ✓ | ✗ | ✗ | ✗ | ✗ |
| Vingestin et. al. [18] | ✗ | ✓ | ✗ | ✗ | ✗ |
| Nurjanah et. al. [19] | ✗ | ✓ | ✗ | ✗ | ✗ |
| Proposed System | ✓ | ✓ | ✓ | ✓ | ✓ |

**Legend:**
R: Research
F: Failover
M: Multi-channel Notification
A: Anti-Flapping/ Debouncing
C: Client-side Recovery Analysis
O: Operational Evaluation

Based on the comparison presented in Table 1, this research proposes a debouncing-based automated network failover system integrated with edge-triggered multi-channel notifications. The proposed system applies repeated verification using MikroTik Scheduler to improve failover stability and reduce false failover events caused by temporary disturbances. Furthermore, edge-triggered notifications are implemented to minimize redundant alerts, while client-side recovery behavior is evaluated to improve operational reliability in real-world business environments.

## III. METHODOLOGY

This research employed a methodology focused on the design, implementation, and evaluation of an automated network failover system integrated with multi-channel real-time notifications. The proposed system was implemented in the operational network environment of Kampoeng Semarang to improve network availability and reduce downtime caused by ISP connectivity failures.

### A. Research Framework

The research adopted the PPDIOO (Prepare, Plan, Design, Implement, Operate, and Optimize) framework to ensure a structured system development and evaluation process as shown in Figure 1.

The Prepare phase focused on identifying operational problems related to network downtime and connectivity instability in the existing infrastructure. The Plan phase involved determining system requirements, selecting supporting technologies, and preparing failover testing scenarios. During the Design phase, the network failover architecture, monitoring mechanism, and notification workflow were developed. The system design integrated MikroTik Netwatch, Scheduler, automated routing scripts, Telegram Bot API, and email notification services. Figure 2 shows the failover logic topology used in this research. The Implement phase focused on configuring the failover mechanism and integrating the notification system into the operational network environment.

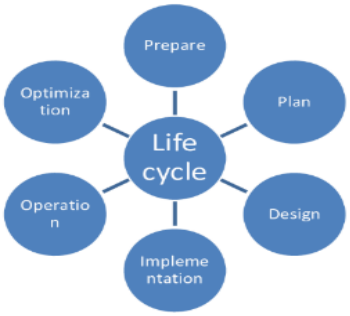

Fig. 1 PPDIOO Framework [12]

The Operate phase involved conducting failover testing, notification delay measurements, and operational monitoring under real network conditions.

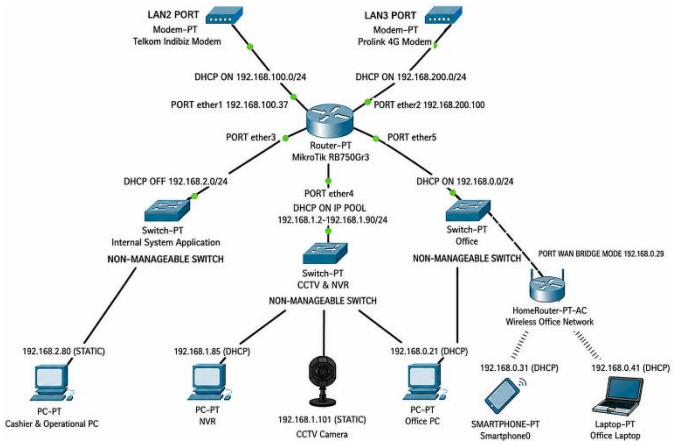

Fig. 2 Failover Network Topology

Finally, the Optimize phase focused on improving failover stability and client-side recovery behavior through connection tracking optimization and debouncing-based verification mechanisms to reduce route flapping during transient network disturbances.

### B. Data Collection Method

Data collection was conducted through observation, interviews, and literature review [20]. Observations were performed directly within the operational environment of Kampoeng Semarang to identify network behavior, downtime frequency, failover performance, and notification responsiveness before and after system implementation. Interviews were conducted with operational staff and network administrators to gather information regarding existing network management procedures, operational impacts of downtime, and administrator responses during ISP disruptions. The collected information was used to support operational analysis and system evaluation.

In addition, a literature review was carried out to obtain theoretical references and implementation approaches

https://doi.org/10.31436/ijpcc.v12i2.705

related to network failover systems, MikroTik Netwatch, real-time notifications, and network monitoring mechanisms. Previous researches and technical documentation were also used to identify evaluation metrics and implementation strategies relevant to this research.

### C. System Architecture

The proposed system utilized a dual-ISP architecture consisting of a primary ISP connection and a backup ISP connection managed by a MikroTik RouterOS device. Under normal conditions, all network traffic was routed through the primary ISP. When connectivity failure was detected on the primary connection, the failover mechanism automatically redirected traffic to the backup ISP. The architecture integrated MikroTik Netwatch for connectivity monitoring, Scheduler for repeated verification processes, automated routing scripts for failover execution, and Telegram Bot API and email notification services for notification delivery. Notification messages were transmitted automatically whenever network status transitions occurred between UP and DOWN conditions. This can be seen in Figure 3 below.

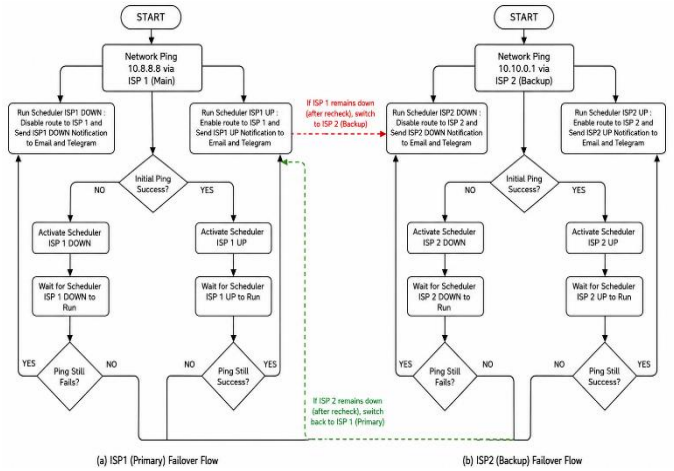


Fig. 3 Proposed Dual-ISP Failover System Architecture

The primary ISP served as the main operational internet connection, while the secondary ISP functioned as a backup link to maintain service continuity during connectivity failures. The router continuously monitored network reachability using connectivity verification mechanisms configured through Netwatch and Scheduler.

### D. Proposed Debouncing-Based Failover Mechanism

The proposed failover mechanism applied a debouncing-based verification strategy to improve failover stability and reduce false failover events caused by temporary packet loss or unstable latency conditions. Instead of relying on a single connectivity check, the system performed repeated verification using MikroTik Scheduler before determining whether the primary connection should be considered unavailable. When repeated monitoring results indicated that the primary ISP connection failed consecutively, the system executed automated routing scripts to activate the backup ISP route. Conversely, when the primary ISP connection returned to stable conditions, the system restored traffic routing to the primary link. The proposed debouncing-based failover workflow shown in Figure 4.

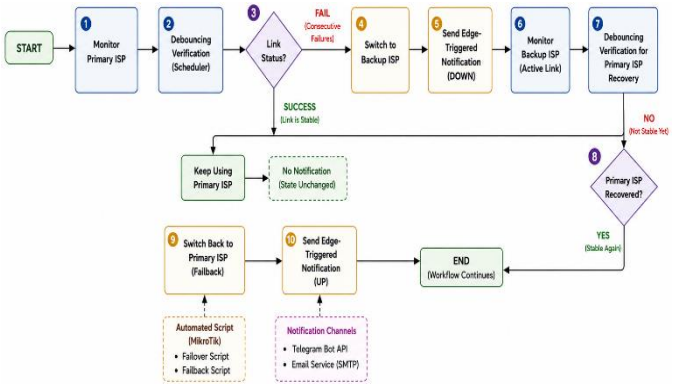


Fig. 4 Debouncing-based Failover Workflow

To reduce notification flooding during unstable network conditions, the system implemented an edge-triggered notification mechanism. Notifications were transmitted only when actual status transitions occurred between UP and DOWN states, preventing repetitive alerts under unchanged conditions. The failover process also incorporated connection tracking optimization to improve effective client-side recovery after ISP transitions. This optimization aimed to minimize session persistence issues and ensure that newly established client connections followed the currently active ISP route.

### E. Experimental Setup

The experimental environment consisted of a MikroTik RouterOS device connected to two ISP connections. The primary ISP provided a bandwidth capacity of 150 Mbps, while the secondary ISP functioned as a backup connection using a GSM-based internet network. Approximately 30 active users utilized the network during operational hours. System testing was conducted by intentionally disconnecting the primary ISP connection to observe failover behavior, recovery delay, notification delivery performance, and client-side recovery consistency. Multiple testing scenarios were performed to evaluate failover responsiveness under operational conditions.

Experiments were conducted repeatedly under different operational conditions to evaluate the consistency and stability of the proposed failover mechanism as shown in Table 2 below.

https://doi.org/10.31436/ijpcc.v12i2.705

TABLE II
EXPERIMENTAL ENVIRONMENT CONFIGURATION

| Component | Specification |
|---|---|
| Router Device | MikroTik RouterOS |
| Primary ISP | 150 Mbps Fiber Connection |
| Backup ISP | GSM-based Internet Connection |
| Notification Platform | Telegram Bot API, Email |
| Monitoring Mechanism | Netwatch + Scheduler |
| Active Users | Approximately 30 users |

Testing scenarios included ISP disconnection events, restoration events, notification delivery measurements, and client-side connectivity recovery evaluation after route transitions.

### F. Evaluation Metrics

Several evaluation metrics were used to evaluate the performance and operational reliability of the proposed debouncing-based failover system. The primary metric was failover recovery time, which measured the duration required for the system to redirect network traffic from the primary ISP to the backup ISP after connectivity failure detection. This metric was used to assess how quickly the proposed mechanism could maintain network availability during ISP disruptions. In addition, notification delay was measured to determine the responsiveness of the Telegram and email notification services by calculating the time difference between failover event occurrence and notification reception by the administrator.

The evaluation also focused on failover stability during transient network disturbances and repeated connectivity fluctuations. This metric was used to analyze the effectiveness of the debouncing verification mechanism in preventing false failover events and reducing route flapping behavior caused by temporary packet loss or unstable latency conditions. Furthermore, client-side recovery was evaluated to determine whether internet connectivity from the user perspective could be restored consistently after ISP transitions occurred. This evaluation was important because successful route switching at the router level does not always guarantee immediate recovery for connected client devices. The complete evaluation metrics in this research are presented in Table 3.

TABLE III
EVALUATION METRICS

| Metric | Description |
|---|---|
| Failover Recovery Time | ISP switching duration |
| Notification Delay | Notification transmission latency |
| Failover Stability | Resistance to route flapping |
| Client-Side Recovery | Effective recovery experienced by users |

The collected evaluation data were analyzed to determine the overall effectiveness of the proposed debouncing-based failover mechanism and edge-triggered multi-channel notification system in improving operational network reliability. The evaluation results were then used to assess system responsiveness, failover consistency, notification performance, and effective connectivity recovery under real operational conditions. In addition, repeated testing scenarios were conducted to observe system behavior during both failover and failback processes under varying network conditions. The analysis also aimed to identify whether the proposed debouncing mechanism could minimize unstable route transitions and reduce redundant notifications caused by temporary connectivity disturbances.

## IV. RESULTS AND DISCUSSION

This section presents the implementation results and operational evaluation of the proposed debouncing-based automated network failover system integrated with multi-channel real-time notifications. The discussion focuses on failover performance, notification responsiveness, failover stability, and client-side recovery behavior under real operational conditions at Kampoeng Semarang.

### A. System Implementation

The proposed failover system was implemented using MikroTik RouterOS with dual-ISP connectivity consisting of a primary ISP and a backup GSM-based ISP. The implementation integrated MikroTik Netwatch, Scheduler, automated routing scripts, Telegram Bot API, and email notification services to support automatic failover and real-time monitoring.

The debouncing mechanism was implemented through repeated verification using Scheduler to ensure that failover decisions were not triggered by temporary packet loss or intermittent network disturbances. In addition, edge-triggered notifications were configured to ensure that notifications were transmitted only during actual status transitions between UP and DOWN conditions.

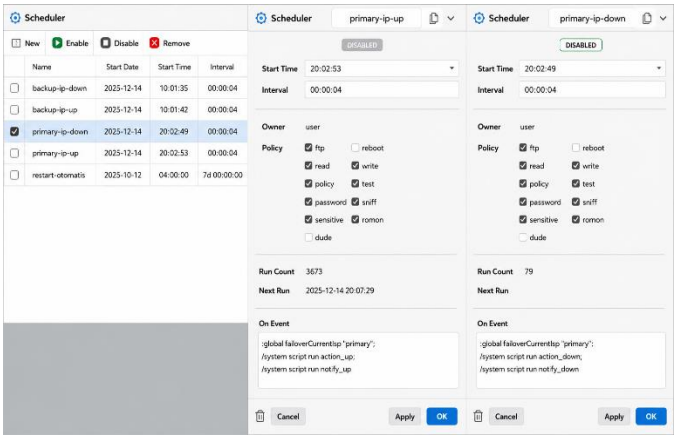

Fig. 5 Failover Scheduler Configuration

https://doi.org/10.31436/ijpcc.v12i2.705

Figure 5 presents the Scheduler configuration used to implement the debouncing-based failover verification mechanism in the proposed system.

### B. Operational Downtime Observation

Before the implementation of the proposed system, network recovery procedures were performed manually whenever the primary ISP experienced connectivity failure. Operational observations showed that network interruptions occurred repeatedly during business hours and required several minutes for administrator intervention and recovery. To illustrate the operational downtime conditions that motivated the implementation of the failover system, several downtime events recorded during the observation period are presented in Table 4.

TABLE IV
SAMPLE OPERATIONAL DOWNTIME EVENTS

| Date | Time DOWN | Time UP | Duration |
|---|---|---|---|
| 03/05/2025 | 09:08:05 | 09:08:25 | 00:00:20 |
| 03/05/2025 | 10:20:20 | 10:20:40 | 00:00:20 |
| 03/05/2025 | 10:20:50 | 10:21:10 | 00:00:20 |
| 03/05/2025 | 11:15:23 | 11:15:46 | 00:00:23 |
| 05/05/2025 | 09:46:20 | 09:46:40 | 00:00:20 |

Table 4 presents several operational downtime events observed before the implementation of the proposed automated failover system. The recorded incidents indicate that internet connectivity disruptions occurred repeatedly during operational hours and required manual intervention from network administrators to restore connectivity. The recovery duration varied depending on the type of disturbance, administrator response time, and ISP recovery process, resulting in service interruptions that affected daily operational activities. These findings demonstrate the necessity of an automated failover mechanism capable of reducing recovery dependency on manual handling while improving network availability and operational continuity.

### C. Failover Recovery Performance

The core evaluation focused on measuring the failover recovery time required to redirect network traffic from the primary ISP to the backup ISP after connectivity failure detection. The proposed debouncing-based mechanism successfully performed automatic failover within several seconds while maintaining stable routing behavior.

Multiple failover testing scenarios were conducted by intentionally disconnecting the primary ISP connection during operational conditions. The results showed that the proposed system achieved relatively consistent failover recovery performance with minimal variation between test scenarios.

TABLE V
FAILOVER RECOVERY TIME FROM ISP 1 TO ISP 2

| Scenario (Date) | Failover Duration (seconds) |
|---|---|
| ISP 1 to ISP 2 (05-08-2025) | 4,0 |
| ISP 1 to ISP 2 (06-08-2025) | 4,0 |
| ISP 1 to ISP 2 (07-08-2025) | 4,0 |
| ISP 1 to ISP 2 (08-08-2025) | 5,0 |
| ISP 1 to ISP 2 (09-08-2025) | 4,0 |
| Average Recovery Time | 4,2 |

Table 5 presents the failover recovery time measured during the transition process from ISP 1 (primary connection) to ISP 2 (backup connection). The testing results show that the proposed debouncing-based failover mechanism successfully performed automatic route switching within a relatively consistent time range, achieving an average recovery time of approximately 4.2 seconds. The relatively small variation between testing scenarios indicates that the repeated verification process implemented through MikroTik Scheduler was able to maintain stable failover behavior while minimizing false failover events caused by temporary connectivity disturbances. The recovery duration was influenced by connectivity verification intervals, repeated Scheduler-based checks, and routing table update processes within MikroTik RouterOS, demonstrating that the proposed mechanism could maintain both failover responsiveness and operational stability during ISP disruptions.

### D. ISP Performance Evaluation

The performance of both ISP connections was evaluated to determine whether the backup ISP could maintain operational connectivity during failover conditions. Network speed measurements included download throughput, upload throughput, ping latency, and public IP verification. The evaluation results showed that the primary ISP provided significantly better throughput and lower latency compared to the backup ISP. However, the backup connection remained sufficient to maintain essential operational services during failover events.

TABLE VI
ISP PERFORMANCE AND PUBLIC IP VERIFICATION

| Path | Download (Mbps) | Upload (Mbps) | Ping (ms) | Public IP |
|---|---|---|---|---|
| ISP 1 (Main) | 146,0 | 23,9 | 101 | 36.80.254.36 |
| ISP 2 (Backup) | 31,4 | 19,9 | 120 | 114.10.8.72 |

Table 6 presents the performance evaluation results of both ISP connections, including download throughput, upload throughput, ping latency, and public IP verification. The results indicate that the primary ISP provided significantly better network performance compared to the backup ISP, particularly in terms of throughput capacity and lower latency, making it more suitable for daily operational

https://doi.org/10.31436/ijpcc.v12i2.705

traffic. However, the backup ISP was still capable of maintaining essential internet connectivity during failover conditions despite its lower performance characteristics. In addition, the different public IP addresses obtained during testing confirm that the failover mechanism successfully redirected network traffic to a separate ISP connection rather than maintaining local routing behavior, demonstrating that the proposed failover system operated correctly during ISP transition processes.

### *E. Notification Performance Evaluation*

The notification system was evaluated to measure the responsiveness of Telegram and email notifications during failover and failback events. Notification delay was calculated based on the time difference between failover event occurrence and notification reception by the administrator.

The experimental results demonstrated that Telegram notifications consistently achieved lower delivery delay compared to email notifications. Telegram provided near real-time operational alerts, while email notifications required additional processing time due to SMTP transmission and client synchronization processes.

TABLE VII
NOTIFICATION DELAY EVALUATION

| Scenario | Average Telegram Delay (seconds) | Average Email Delay (seconds) |
|---|---|---|
| ISP 1 to ISP 2 | 2,22 | 8,61 |
| ISP 2 to ISP 1 | 1,89 | 8,56 |

Table 7 presents the notification delay evaluation results for both Telegram and email notification services during failover and failback scenarios. The results show that Telegram notifications consistently achieved lower delivery latency compared to email notifications, with average delays of approximately 2 seconds, while email notifications required approximately 8–9 seconds to reach the administrator. This difference is primarily influenced by the lightweight API-based delivery mechanism used by Telegram, whereas email notifications require additional SMTP transmission and client synchronization processes before messages are received. The implementation of edge-triggered notifications also helped reduce redundant alert transmissions during unstable connectivity conditions, allowing administrators to receive more efficient and relevant operational status information during ISP transition events.

### *F. Notification Verification*

The functionality of the multi-channel notification system was further validated by observing notification messages received by administrators during failover events [21]. Notifications were generated automatically whenever actual status transitions occurred between UP and DOWN conditions. The notification mechanism successfully delivered operational status information through both Telegram and email platforms, improving administrator awareness regarding network disturbances and recovery events.

Figure 6 and Figure 7 presents the output of the Telegram and email notification system generated automatically during failover and failback events. The notifications provide real-time operational information regarding ISP status transitions, including DOWN conditions when the primary ISP becomes unavailable and UP conditions when connectivity is restored. The results demonstrate that the proposed multi-channel notification mechanism successfully delivered status updates through both Telegram Bot API and email services, thereby improving administrator awareness and enabling faster operational response during network disturbances. In addition, the implementation of edge-triggered notification behavior ensured that alerts were transmitted only during actual status changes, reducing redundant notifications and minimizing alert flooding during unstable network conditions.

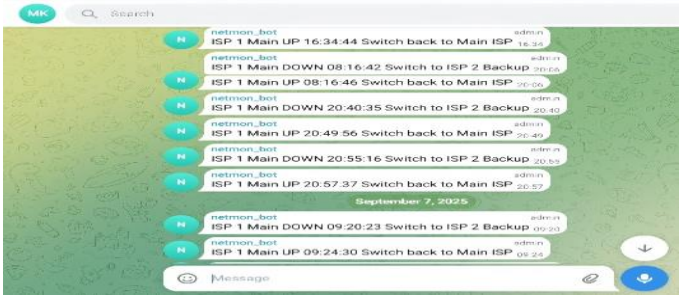

Fig. 6 Telegram Notification Output

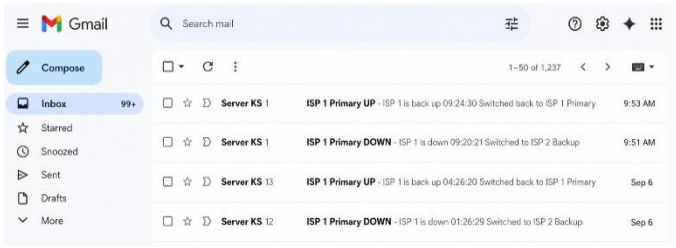

Fig. 7 Email Notification Output

### *G. Failover Optimization and Client-Side Recovery*

During operational testing, an important issue was identified in which several client devices experienced delayed internet recovery after failover transitions despite successful route switching at the router level. This condition was caused by connection tracking persistence and session stickiness behavior that maintained existing sessions on the previous ISP route.

To improve effective client-side recovery, optimization was performed by clearing connection tracking sessions and

https://doi.org/10.31436/ijpcc.v12i2.705

adjusting failover handling mechanisms to ensure that newly established client connections followed the currently active ISP route.

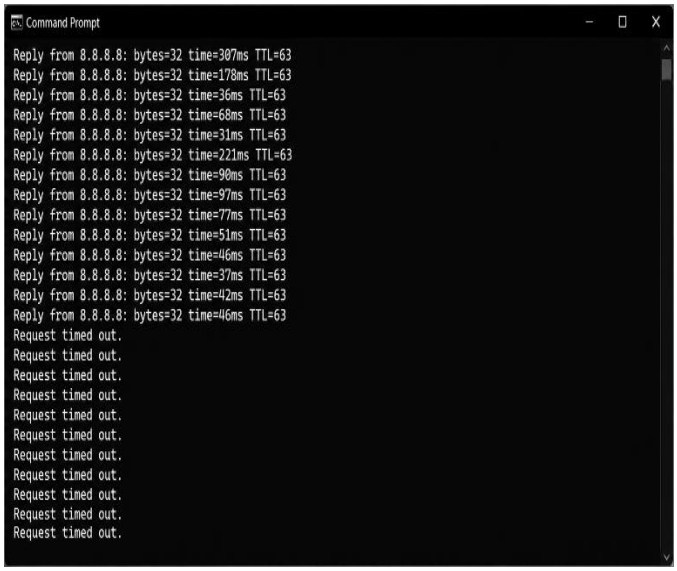

Fig. 8 Client Connectivity Issue Before Optimization

Figure 8 illustrates the client-side connectivity issue observed before the optimization process was applied to the proposed failover system. Although the router had successfully performed failover routing from the primary ISP to the backup ISP, several client devices still experienced temporary connectivity loss indicated by repeated request timeout responses during continuous ping testing. This condition was primarily caused by connection tracking persistence and session stickiness behavior, where existing client sessions remained associated with the previous ISP route even after routing changes had been executed. The results demonstrate that successful failover at the router level does not always guarantee immediate recovery from the client perspective, highlighting the importance of connection tracking optimization to improve effective client-side recovery during ISP transition processes.

After optimization was applied, additional testing showed that client-side connectivity recovery became significantly more stable during ISP transitions. The optimization successfully minimized delayed recovery behavior and improved effective internet accessibility from the user perspective.

Figure 9 presents the client-side connectivity condition after optimization was applied to the proposed failover system. The continuous ping results show that internet connectivity was restored more consistently during ISP transition processes, with significantly reduced request timeout occurrences compared to the condition before optimization. This improvement was achieved through connection tracking optimization and failover handling adjustments that ensured newly established client sessions followed the currently active ISP route. The results indicate that the optimization process successfully improved effective client-side recovery, minimized session persistence issues, and enhanced overall connectivity stability during failover and failback operations.

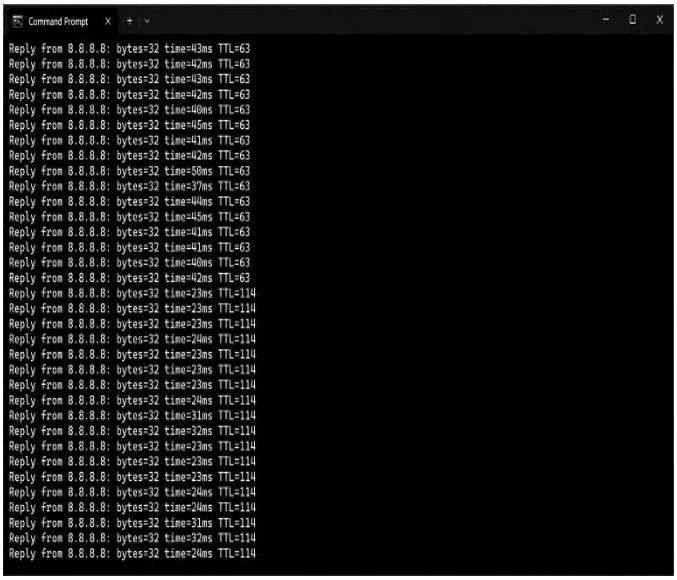

Fig. 9 Client Connectivity Recovery After Optimization

### *H. Optimization Impact Summary*

The optimization process improved not only router-level failover behavior but also effective client-side recovery consistency during ISP transitions. The proposed debouncing mechanism, edge-triggered notifications, and connection tracking optimization collectively improved operational reliability under real-world network conditions.

The Table 8 summarizes the impact of the optimization process applied to the proposed failover system. The results indicate that the implemented optimizations successfully improved failover stability, client-side recovery consistency, and overall operational reliability during ISP transition processes. Before optimization, several issues such as unstable route transitions, session stickiness, delayed client recovery, and repeated request timeout occurrences were still observed during failover events. After optimization, the system demonstrated more stable failover and failback behavior, reduced route flapping, improved connection tracking handling, and more consistent internet recovery from the client perspective. In addition, the implementation of edge-triggered notifications reduced redundant alert transmissions during unstable network conditions, thereby improving monitoring efficiency and administrator awareness.

https://doi.org/10.31436/ijpcc.v12i2.705

TABLE VIII
OPTIMIZATION IMPACT SUMMARY

| Optimization Aspect | Before Optimization | After Optimization | Impact |
|---|---|---|---|
| Failover Stability | Occasional unstable route transitions during transient disturbances | More stable failover and failback behavior | Reduced route flapping |
| Client-Side Recovery | Several clients experienced delayed internet recovery | Faster and more consistent connectivity restoration | Improved effective recovery |
| Connection Tracking Behavior | Existing sessions remained attached to previous ISP route | Sessions followed the active ISP route correctly | Reduced session stickiness |
| Ping Test Results | Frequent request timeout occurrences during ISP transition | Minimal timeout occurrences during failover | Improved connectivity continuity |
| Notification Behavior | Potential redundant notifications during unstable conditions | Notifications sent only during actual state changes | Reduced alert flooding |
| Operational Reliability | Recovery still partially affected by unstable transitions | More reliable and consistent network operation | Improved operational continuity |

### *I. User Satisfaction Evaluation*

To evaluate operational impact from the user perspective, a user satisfaction survey was conducted after system implementation and optimization. The survey results indicated that users perceived improvements in connectivity stability, operational responsiveness, and downtime handling performance.

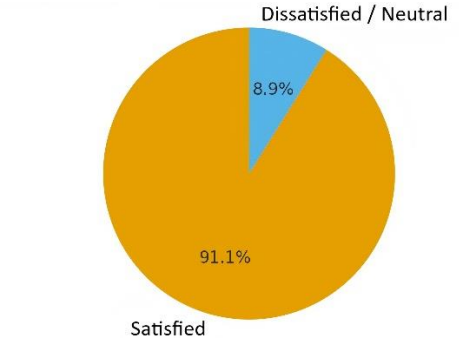

Fig. 10 User Satisfaction Survey Results

Figure 10 presents the results of the user satisfaction survey conducted after the implementation and optimization of the proposed failover monitoring system. The survey results indicate a high level of user satisfaction, with approximately 91.1% of respondents reporting positive perceptions regarding network stability, failover responsiveness, and operational reliability during internet disruptions. The improved satisfaction level suggests that the proposed debouncing-based failover mechanism and multi-channel notification system successfully enhanced service continuity and reduced the operational impact of ISP connectivity failures. These findings also demonstrate that the implemented optimizations contributed not only to technical performance improvements but also to better user experience in real operational environments.

### *J. Discussion*

The experimental results demonstrate that the proposed debouncing-based failover mechanism successfully improved both failover responsiveness and operational stability within a real-world networking environment. The implementation of repeated verification using MikroTik Scheduler effectively reduced false failover events caused by transient packet loss and unstable latency conditions, thereby minimizing route flapping behavior during ISP disruptions. Unlike conventional failover implementations that rely primarily on single connectivity checks, the proposed mechanism introduced a more stable decision-making process before route switching was executed. Despite the additional verification process, the system was still able to maintain relatively fast recovery performance with an average failover recovery time of approximately 4.2 seconds. Furthermore, the integration of edge-triggered multi-channel notifications improved operational monitoring efficiency by reducing redundant alerts during unstable connectivity conditions. Telegram notifications also demonstrated lower latency compared to email notifications, making them more suitable for real-time operational monitoring.

Another important finding of this research is the impact of connection tracking optimization on effective client-side recovery during ISP transition processes. Initial testing showed that successful failover at the router level did not always guarantee immediate internet recovery from the client perspective due to session persistence and connection tracking behavior. After optimization was applied, client-side connectivity recovery became significantly more stable with fewer request timeout occurrences during failover and failback operations. These results indicate that effective failover implementation should not only focus on automatic route switching but also

consider session handling and user-side recovery consistency. Overall, the proposed framework demonstrates that combining debouncing-based failover verification, edge-triggered notifications, and connection tracking optimization can improve network availability, operational reliability, and service continuity in small-to-medium business environments.

## V. Conclusion

This research successfully implemented a debouncing-based automated network failover system integrated with multi-channel real-time notifications using MikroTik RouterOS, Telegram Bot API, and email services within the operational environment of Kampoeng Semarang. The proposed system applied repeated verification through MikroTik Scheduler to improve failover stability and reduce false failover events caused by transient network disturbances. Experimental results showed that the system achieved an average failover recovery time of approximately 4.2 seconds, while Telegram and email notifications achieved average delivery delays of approximately 2 seconds and 8–9 seconds, respectively. The implementation of edge-triggered notifications also reduced redundant alert transmissions during unstable connectivity conditions, improving operational monitoring efficiency and administrator awareness.

In addition, the optimization process successfully improved effective client-side recovery by minimizing session persistence issues and enhancing connectivity stability during ISP transition processes. The user satisfaction evaluation further indicated that the proposed system improved perceived network reliability and operational continuity in real-world conditions. Overall, the proposed framework demonstrates that combining debouncing-based failover verification, automated route switching, and multi-channel notification mechanisms can improve network availability and operational resilience in small-to-medium business environments. Future work may include integrating centralized logging systems, adaptive monitoring thresholds, dashboard-based monitoring platforms, and support for multi-ISP scalability to further enhance system reliability and operational management capabilities.

## Acknowledgment

The research team extends its heartfelt gratitude to the Universitas Semarang, Universitas Nias Raya, and Indonesia Defense University for their invaluable support and contribution to this research.

## Conflict of Interest

Authors state no conflict of interest.

## Author (s) Contribution Statement

F.W.C: Conceptualization, Data curation, Formal analysis, Methodology, Project administration, Writing—original draft, and Writing—review and editing. S.V.R: Validation, Software, Visualization, Writing—original draft, and Writing—review and editing. F.L: Resources, Investigation, Supervision, Writing—original draft validation, and Writing—review & editing. E.A.F: Validation and Writing—review and editing.

## Data Availability Statement

The data that support the findings of this research are available from the corresponding author upon reasonable request.

## Ethics Statement

This research did not require ethical approval.

## References

[1] A. Abdurrahman, "Extending the IBCDE Framework to Explore Barriers and Drivers in Indonesia ' s Digital Economy," *J. Digit. Econ.*, vol. 4, no. August, pp. 123–143, 2025, doi: 10.1016/j.jdec.2025.08.003.

[2] R. E. Caraka *et al.*, "Evaluating Internet Accessibility and Equity in Indonesia using Integrated SUSENAS and OOKLA Data," *Discov. Sustain.*, vol. 7, no. 497, pp. 1–31, 2026, doi: 10.1007/s43621-026-02882-x.

[3] A. Djenna, S. Harous, and D. E. Saidouni, "Internet of Things Meet Internet of Threats : New Concern Cyber Security Issues of Critical Cyber Infrastructure," *Appl. Sci.*, vol. 11, no. 4580, pp. 1–30, 2021, doi: 10.3390/app11104580.

[4] B. Howell, "Beyond Infrastructure : Internet Ecosystem Resilience and the Public Good," *Telecomm. Policy*, vol. 49, no. 7, p. 102998, 2025, doi: 10.1016/j.telpol.2025.102998.

[5] F. Toriq and B. Santoso, "Effect of Load Balancing Bonding and Failover on Speed, Latency, Average, and Packet Loss," *J. Appl. Informatics Comput.*, vol. 8, no. 2, pp. 326–331, 2024, doi: 10.30871/jaic.v8i2.8276.

[6] Hoesiin and B. Yuliandi, "Optimizing Network Uptime Through Dual ISP Failover Implementation Using Netwatch," *J. Informatic, Educ. Manag.*, vol. 7, no. 2, pp. 336–349, 2025.

[7] W. M. Elsayed and A. A. Abohany, "Strengthening Network Resilience and Optimizing Resources in Large Enterprises: Innovative Infrastructure and Cybersecurity Strategies," *Cluster Comput.*, vol. 29, no. 252, pp. 1–31, 2026, doi: 10.1007/s10586-026-06075-z.

[8] B. Anbalagan, "Proactive Failover and Automation Frameworks for Mission-Critical Workloads : Lessons from Manufacturing Industry," *Int. J. Res. Appl. Innov.*, vol. 6, no. 1, pp. 8279–8296, 2023, doi: 10.15662/IJRAI.2023.0601004.

[9] C. K. Ryu, M. C. Lee, I. H. Hong, J. H. Park, J. D. Lee, and S. Y. Choi, "Heterogeneous PLC-Based Distributed Controller with Embedded Logic-Monitoring Blackbox for Real-Time Failover," *Electronics*, vol. 14, no. 4359, pp. 1–36, 2025, doi: 10.3390/electronics14224359.

[10] L. Fotia, R. Gaeta, F. Messina, D. Rosaci, and G. M. L. Sarné, "Artificial Intelligence for High-Availability Systems: A Comprehensive Review," *Computers*, vol. 15, no. 4, p. 231, 2026, doi: 10.3390/computers15040231.

[11] T. Li, J. K. Haines, M. F. R. De Eguino, J. I. Hong, and J. Nichols, "Alert Now or Never : Understanding and Predicting Notification

Preferences of Smartphone Users," *ACM Trans. Comput. Interact.*, vol. 29, no. 5, p. 39, 2022, doi: 10.1145/3478868.

[12] P. Petrov, I. Kuyumdzhiev, R. Malkawi, G. Dimitrov, and J. Jordanov, "Digitalization of Educational Services with Regard to Policy for Information Security," *TEM J.*, vol. 11, no. 3, pp. 1093–1102, 2022, doi: 10.18421/TEM113.

[13] R. Rayendra, A. K. Setyanegara, M. Raafi, and F. Tara, "Development of a Web-Based Server Monitoring System Using the PPDIOO Method," *Int. J. Adv. Technol. Eng. Inf. Syst.*, vol. 4, no. 2, pp. 207–224, 2025, doi: 10.55047/ijateis.v4i2.1690.

[14] A. Purwanto and B. Soewito, "Optimization Problem of Computer Network using PPDIOO," *ICIC Express Lett.*, vol. 15, no. 7, pp. 769–777, 2021, doi: 10.24507/icicel.15.07.769.

[15] Bustomi and M. Tahir, "Evaluation of PCC Load Balancing for Dual ISP Networks : Enhancing Throughput and Traffic Stability," *Edumatic J. Pendidik. Inform.*, vol. 10, no. 10, pp. 240–249, 2026, doi: 10.29408/edumatic.v10i1.34243.

[16] M. F. I. P. Putra, J. M. Parenreng, and M. Yahya, "Integration of Telkom ISP and 3 LTE Using PCC Method to Improve Internet Connection Stability," *Iota*, vol. 04, no. 02, pp. 244–256, 2024, doi: 10.31763/iota.v4i2.728.

[17] F. Zikra, Anwar, and I. Safar, "Monitoring Networks Based on Simple Network Management Protocol With Cacti Application and Telegram Notifications on Linux Ubuntu," *J. Teknol. Rekayasa Inf. dan Komput.*, vol. 9, no. 1, pp. 55–65, 2026, doi: 10.30811/jtrik.v9i1.8701.

[18] I. Vingestin, T. U. Kalsum, and Y. Mardiana, "The Design Of Network Monitoring System Using SNMP Protocol With Telegram Notification," *J. Media Comput. Sci.*, vol. 2, no. 1, pp. 93–100, 2022, doi: 10.37676/jmcs.v2i1.3441.

[19] S. Nurjanah, F. Sembiring, and R. R. Ayuningsih, "Integration of Telegram Bot and Uptime Kuma for Wi-Fi Network Monitoring using Mikrotik," *Pilar Nusa Mandiri J. Comput. Inf. Syst.*, vol. 20, no. 2, pp. 118–126, 2024, doi: 10.33480/pilar.v20i2.5535.

[20] B. P. Neupane, N. Dahal, R. K. Dhakal, M. K. Hasan, J. A. Villarama, and B. G. Fabros, "Data Collection Methods in Qualitative Research: Researchers' Reflections," *Front. Res. Metrics Anal.*, vol. 11, no. 1778160, pp. 1–9, 2026, doi: 10.3389/frma.2026.1778160.

[21] H. Zhang, R. Zhang, and J. Sun, "Developing Real-time IoT-based Public Safety Alert and Emergency Response Systems," *Sci. Rep.*, vol. 15, no. 29056, pp. 1–25, 2025, doi: 10.1038/s41598-025-13465-7.