International Journal of Computer Networks and Applications (IJCNA)
DOI: 10.22247/ijcna/2023/223319 Volume 10, Issue 4, July – August (2023)

RESEARCH ARTICLE

Relentless Firefly Optimization-Based Routing Protocol (RFORP) for Securing Fintech Data in IoT-Based Ad-Hoc Networks

J Ramkumar
Department of Information Technology and Cognitive Systems, Sri Krishna Arts and Science College, Coimbatore, Tamil Nadu, India.
jramkumar1986@gmail.com

K S Jeen Marseline
Department of Information Technology and Cognitive Systems, Sri Krishna Arts and Science College, Coimbatore, Tamil Nadu, India.
jeenmarselineks@skasc.ac.in

D R Medhunhashini
Department of Information Technology and Cognitive Systems, Sri Krishna Arts and Science College, Coimbatore, Tamil Nadu, India.
medhun.hashini@gmail.com

Received: 05 July 2023 / Revised: 08 August 2023 / Accepted: 13 August 2023 / Published: 31 August 2023

Abstract – The widespread adoption of Internet of Things (IoT) technology and the rise of fintech applications have raised concerns regarding the secure and efficient routing of data in IoT-based ad-hoc networks (IoT-AN). Challenges in this context include vulnerability to security breaches, potential malicious node presence, routing instability, and energy inefficiency. This article proposes the Relentless Firefly Optimization-based Routing Protocol (RFORP) to overcome these issues. Inspired by fireflies' natural behaviour, RFORP incorporates relentless firefly optimization techniques to enhance packet delivery, malicious node detection, routing stability, and overall network resilience. Simulation results demonstrate RFORP's superiority over existing protocols, achieving higher packet delivery ratios, accurate malicious node detection, improved routing stability, and significant energy efficiency. The proposed RFORP offers a promising solution for securing fintech data in IoT-AN, providing enhanced performance, reliability, and security while effectively addressing the identified challenges. This research contributes to advancing secure routing protocols in fintech applications and guides network security and protocol selection in IoT environments.

Index Terms – Fintech, Firefly, IoT, Ad-Hoc Networks, Routing, Security.

1. INTRODUCTION

Fintech, an abbreviation for financial technology, pertains to the inventive utilization of technology for revolutionizing and enhancing diverse facets within the financial sector [1]. It encompasses many applications, including mobile payments, digital banking, online lending, blockchain, robo-advisors, and more. Fintech has revolutionized how individuals and businesses access financial services, making them more accessible, convenient, and efficient [2], [3]. Harnessing progressions in cloud computing, machine learning, artificial intelligence, data analytics, and fintech has disrupted conventional financial institutions and opened avenues for emerging participants in the industry [4].

Fintech has enabled financial inclusion by reaching underserved populations, introducing alternative lending models, and simplifying complex processes. It has also spurred cybersecurity, regulatory compliance, and risk management innovation. Fintech continues to evolve rapidly, driven by technological advancements and changing consumer expectations, shaping the future of finance and transforming how users manage, save, invest, and transact [5].

IoT-based ad-hoc networks (IoT-AN) are decentralized networks that leverage the Internet of Things (IoT) technology to enable devices to communicate and collaborate without relying on a pre-existing infrastructure. In an ad-hoc network, IoT devices establish direct connections with each other, forming a dynamic and self-configuring network topology [6]. These networks offer flexibility and scalability, making them suitable for various applications. IoT-AN has numerous advantages. These technologies can be

RESEARCH ARTICLE

implemented in settings where conventional infrastructure is not accessible or feasible, such as remote locations or disaster areas [7]. Ad-hoc networks also facilitate device-to-device communication, enabling real-time data exchange and collaboration. This is particularly valuable in scenarios requiring immediate and localized decision-making [8]. IoT-AN has applications across various domains, including agriculture, transportation, healthcare, and disaster management. They enable intelligent agriculture systems, enhance traffic management and vehicle communication, support remote patient monitoring, and facilitate emergency response coordination [9].

Routing fintech data in an IoT-based ad-hoc network is a complex and crucial area of research that aims to ensure the secure and efficient transmission of financial data in decentralized environments [10]. The unique characteristics of IoT-AN, such as device mobility, limited resources, and dynamic topologies, present challenges that require specialized routing approaches. Security is of paramount importance when routing fintech data [11]. Financial transactions involve sensitive information, and protecting its confidentiality and integrity is critical. Incorporating strong encryption, authentication, and access control mechanisms into routing protocols is crucial to protect data from unauthorized access or tampering [12].

Techniques like secure key exchange and routing protocols can be explored to ensure end-to-end security. Efficient data transmission is another critical consideration. Fintech applications often require low latency for quick responses, high reliability to prevent data loss or corruption, and energy efficiency to optimize device battery life [13].

Routing algorithms should consider these requirements and adapt to the dynamic nature of the network. Techniques such as predictive routing, multi-path routing, or load balancing can be employed to optimize data transmission and meet the specific needs of fintech applications [14]–[16].

1.1. Problem Statement

The problem statement focuses on the security challenges when routing fintech data in IoT-AN. With the increasing use of IoT devices in financial transactions, ensuring the confidentiality, integrity, and authentication of sensitive financial data becomes critical. The dynamic nature of ad-hoc networks, resource limitations, and varying device connectivity introduce vulnerabilities that malicious actors can exploit.

Designing secure routing protocols and mechanisms that protect against unauthorized access, data tampering, and privacy breaches is essential to mitigate these security risks. Addressing these security issues is crucial to establish trust and confidence in the transmission of fintech data within IoT-AN.

1.2. Motivation

The motivation behind researching the security issues in routing fintech data in IoT-AN stems from the need to ensure the confidentiality, integrity, and authentication of sensitive financial information. As the adoption of IoT devices in financial transactions increases, securing the transmission of fintech data becomes crucial. By addressing these security challenges, researchers aim to establish trust and confidence in fintech services, protect user privacy, prevent financial fraud, and comply with regulatory requirements. Developing robust and reliable security mechanisms for routing fintech data in IoT-AN will pave the way for fintech applications' secure and widespread adoption, benefiting users and finance sectors.

1.3. Objective

This research aims to develop a relentless firefly optimization-based secure routing protocol designed explicitly for transmitting fintech data in IoT-AN. The aim is to address the security challenges associated with sensitive financial information while optimizing the routing paths for efficient and reliable data transmission. The following objectives will guide the research:

Develop a relentless firefly optimization-based secure routing protocol tailored for transmitting fintech data in IoT-AN, ensuring the confidentiality, integrity, and authentication of sensitive financial information.

Implement robust encryption techniques and access control mechanisms to safeguard fintech data during transmission, addressing security concerns and preventing unauthorized access or data breaches.

Optimize routing paths using Firefly optimization to enhance efficiency, minimize latency, and improve overall network performance specifically for fintech applications, facilitating faster and more reliable financial transactions.

Evaluate the proposed secure routing protocol through simulations or real-world deployments, assessing its performance in throughput, delay, packet loss, and energy consumption, focusing on the unique requirements and challenges of fintech data transmission.

Contribute to advancing secure routing solutions for fintech applications in IoT-AN, enhancing trust, privacy, and compliance in financial transactions while promoting the adoption and growth of fintech services.

1.4. Organization

In Section 1, the introduction presents the problem statement, motivation, and objective, emphasizing the need to secure sensitive financial data in IoT-based ad-hoc networks. Section 2 offers a comprehensive literature review, identifying research gaps in existing routing protocols and security

RESEARCH ARTICLE

mechanisms for Fintech applications. Section 3 introduces the proposed RFORP, detailing its innovative features and mechanisms to address the identified gaps. Section 4 describes the simulation settings, outlining the evaluation environment and metrics. The results and discussion of RFORP's performance are presented in Section 5, comparing it with other protocols. Finally, Section 6 provides a concise conclusion, summarizing the key findings, highlighting RFORP's significance, and suggesting future research directions for the proposed protocol's improvement.

2. LITERATURE REVIEW

The current section examines the routing protocols considered at the forefront of technology but demonstrates inadequate performance in routing and securing processes.

"Q-Learning Secure Routing Algorithm" [17] incorporates authenticated-encryption mechanisms to ensure secure communication. By harnessing energy-harvesting techniques, the algorithm maximizes the network's lifespan and enhances its resilience. Combining Q-learning and authenticated encryption contributes to efficient and secure data transmission within WSNs, safeguarding against potential threats. "Multipath Routing Protocol" [18] enhances network reliability and minimizes energy consumption by utilizing multiple paths. It incorporates security measures to ensure the confidentiality and integrity of data, protecting against unauthorized access and tampering. This protocol optimizes energy usage by dynamically selecting the most energy-efficient paths based on real-time conditions. "Secured Data-Centric Routing" [19] ensures secure and reliable data transmission. This approach utilizes the Fork-Hook encryption policy, which provides robust encryption mechanisms for protecting sensitive data. Implementing a Data-Centric Routing Gateway enables efficient data routing while maintaining security and privacy. The proactive trustware mechanism ensures that only trusted nodes participate in the data transmission.

"Dynamic Source Routing (DSR)" [20] utilizes source routing to enable efficient routing in ad hoc wireless networks. When a source node needs to transmit data to a destination, it checks its route cache for an existing route. If no route is found, the source node initiates a route discovery process by broadcasting a route request packet. Intermediate nodes that receive the request add their addresses and forward it. Upon receiving the request, the destination or an intermediate node with a route to the destination responds with a route reply packet containing the complete path. The source node then stores this route in its cache for subsequent data transmission. Each packet includes the complete route in its header during data transmission, allowing intermediate nodes to forward it accordingly. If a node detects a route failure, it generates a route error packet to notify the affected nodes and triggers a route repair process. The route caches are updated to reflect the changes. The dynamic nature of DSR enables it to adapt efficiently to changing network topologies, facilitating effective communication in ad hoc wireless networks.

"Secure-Routing" [21] involves designing routing algorithms and mechanisms to withstand various security threats and disruptions. The secure routing protocols incorporate encryption, authentication, and intrusion detection techniques to prevent hacking, data modification, and denial-of-service attacks. "Trust Management-Based Secure Routing Protocol" [22] ensures secure and reliable data transmission in challenging underwater environments. This protocol incorporates trust management mechanisms to assess the reliability and trustworthiness of participating nodes. It selects trustworthy paths for data transmission, mitigating the risks associated with compromised or malicious nodes. "Lightweight Trust-Based Model" [23] establishes trust among nodes to ensure reliable and secure communication in dynamic and unpredictable network environments. It employs a trust evaluation mechanism to assess the trustworthiness of nodes based on their past behaviour and interactions. Trustworthy nodes are selected as relay nodes for routing data, while untrusted nodes are avoided to prevent malicious activities.

"Hybrid Greedy Secure Optimal Routing" [24] focuses on optimizing energy consumption and enhancing security in computing systems. This approach utilizes heuristic algorithms to minimize energy usage by dynamically adjusting system resources based on workload and power requirements. It incorporates security enhancements, such as encryption and authentication mechanisms, to protect data and prevent unauthorized access. "Hierarchical Energy-Efficient Secure Routing" [25] addresses the challenges of energy consumption and security in body-centric communication systems. This protocol utilizes a hierarchical architecture to organize the network into clusters, each with a cluster head responsible for data routing and management. Energy-efficient routing algorithms are employed to select optimal routes that minimize energy consumption and prolong the network lifetime. "Trust-Aware Dynamic Routing Algorithm" [26] enhances the AODV protocol by incorporating trust awareness and dynamic routing mechanisms. It considers the trustworthiness of nodes when making routing decisions, avoiding potentially untrustworthy or compromised nodes. It dynamically adjusts the routing paths based on real-time network conditions, such as node availability, energy levels, and security factors.

"Secure Data Transmission Scheme" [27] aims to ensure the confidentiality and integrity of data during transmission. This scheme utilizes multi-protection routing techniques to enhance the security of data centre networks. The multi-protection routing algorithm dynamically selects the most

RESEARCH ARTICLE

secure and efficient paths for data transmission, considering factors like network congestion and node availability. "Trust-based cooperative routing" [28] is a method employed for secure communication in MANETs. It involves establishing and maintaining trust relationships among participating nodes to ensure secure data transmission. The cooperative routing approach encourages nodes to cooperate and share routing information based on trustworthiness. Nodes with a higher level of trust are given priority in relaying data, while less trustworthy nodes are avoided. "Secure Energy Aware Meta-Heuristic Routing Protocol" [29] is designed for sustainable IoT-WSNs. SEAMHR focuses on two main aspects: security and energy efficiency. It employs meta-heuristic algorithms to optimize routing decisions, considering security requirements and energy consumption. The protocol minimizes energy usage by intelligently selecting routes and balancing the energy load among sensor nodes.

"Multi-Adaptive Routing Protocol (MARP)" [30] employs adaptive routing strategies and intelligent decision-making algorithms to dynamically adapt to changing network conditions and optimize routing paths. By continuously monitoring node mobility, link quality, congestion levels, and energy levels, MARP selects the most suitable and stable paths for data transmission, avoiding congested or unreliable routes. It incorporates adaptive algorithms that optimize routing based on performance metrics, considering the specific constraints of IoT devices. MARP's goal is to achieve efficient and reliable data routing by dynamically adapting to network dynamics, enhancing overall performance and ensuring seamless communication within the IoT ecosystem

Table 1 summarises the strengths and weaknesses of these cutting-edge protocols, offering valuable insights into their performance and capabilities. By categorizing the advantages and disadvantages, Table 1 acts as a convenient reference for comparing and evaluating the state-of-the-art routing protocols in terms of their benefits and limitations.

Table 1 Advantages and Disadvantages of State-of-the-Art Routing Protocols

Protocol	Advantages	Disadvantages
Q-Learning Secure Routing Algorithm [17]	• Maximizes network lifespan through optimized energy usage	• Increased computational overhead due to encryption • Requires energy harvesting capabilities
Multipath Routing Protocol [18]	• Enhances network reliability through multiple paths • Minimizes energy consumption	• Complexity in managing alternate paths. • Increased communication overhead
Secured Data-Centric Routing [19]	• Protects sensitive data through encryption policy • Enables proactive trust are data transmission	• Increased computational and communication overhead. • Requires additional gateway infrastructure
Dynamic Source Routing (DSR) [20]	• Reduced overhead and improved scalability	• Consume bandwidth and energy resources. • Lacks in identifying the malicious nodes.
Secure-Routing [21]	• Ensures secure and resilient routing in LEO satellite networks	• Complexity due to dynamic network topology. • Potential latency in routing decisions
Trust Management-Based Secure Routing Protocol [22]	• Lightweight and efficient trust-based routing • Enhances security in opportunistic networks	• Limited scalability in large networks • Reliance on trust evaluation mechanisms

RESEARCH ARTICLE

Lightweight Trust-Based Model [23]	• Increased Network overhead	• Decreased packet delivery ratio
Hybrid Greedy Secure Optimal Routing [24]	• Optimizes energy consumption and promotes green computing	• Increased computational overhead due to security enhancements
Hierarchical Energy-Efficient Secure Routing [25]	• Optimizes energy consumption in WBANs. • Provides secure data transmission	• Additional overhead in managing hierarchical clusters • Complexity in handling dynamic network topologies
Trust-Aware Dynamic Routing Algorithm [26]	• Incorporates trust awareness in routing decisions • Enhances security and reliability of data transmission	• Potential overhead in the trust evaluation process • Increased computational complexity
Secure Data Transmission Scheme [27]	• Provides multi-layered security for data transmission • Selects secure and efficient paths for communication	• Increased overhead due to multiple security measures. • Complexity in managing routing decisions
Trust-based cooperative routing [28]	• Establishes trust relationships among participating nodes • Mitigates risks associated with malicious nodes	• Challenges in trust establishment and maintenance • Increased communication overhead
Secure Energy Aware Meta-Heuristic Routing Protocol [29]	• Maximizes network lifespan through optimized energy usage	• Increased computational complexity • Requires implementation of meta-heuristic algorithms
Multi-Adaptive Routing Protocol (MARP) [30]	• Adaptive and Self-Configuring • Improved Efficiency	• Limited to specific environments • Lacks in providing security

2.1. Research Gap

The research gap in secure routing lies in the current protocols' trade-offs between security and efficiency. Many existing secure routing protocols, such as the Q-Learning Secure Routing Algorithm, Secured Data-Centric Routing etc. [16]-[29], focus on providing encryption-based security mechanisms to protect sensitive data but suffer from increased computational and communication overhead. On the other hand, Dynamic Source Routing offers reduced overhead and improved scalability but lacks adequate mechanisms for identifying and mitigating malicious nodes.

In response to this research gap, the proposed "Resilient and Efficient Routing Protocol (RFORP)" aims to bridge the divide between security and efficiency in routing protocols. RFORP intends to provide robust and secure data transmission while minimizing computational, communication, and energy overhead. It seeks to achieve a balance between enhancing network reliability, optimizing energy usage, and protecting sensitive data without the need for additional gateway infrastructure. By combining the strengths of existing protocols while addressing their limitations, RFORP aspires to be a promising solution for wireless networks seeking both resilience and efficiency in their routing mechanisms.

3. RELENTLESS FIREFLY OPTIMIZATION-BASED ROUTING PROTOCOL (RFORP)

3.1. Enhanced Dynamic Source Routing

Dynamic Source Routing (DSR) is a responsive routing protocol for ad-hoc networks. It presents a versatile and effective method for discovering routes by employing source routing, wherein comprehensive route details are embedded within the packet header. Consequently, intermediate nodes

RESEARCH ARTICLE

can transmit packets by following explicit routing instructions.

The upgraded iteration of the Dynamic Source Routing (DSR) protocol algorithm is

- Initialize: Every node preserves a route cache and a sequence number. The source node possesses data to transmit and possesses knowledge of the destination node's address.
- Route Discovery: When the originating node has a good path, proceed to data transmission. Otherwise, the source node broadcasts a Route Request (RREQ) packet containing the current sequence number and source & destination addresses. Intermediate nodes receiving the RREQ packet check the sequence number and update the route cache if necessary. They modify the RREQ by adding their address and sending it to nearby nodes.
- Receive RREQ: Intermediate nodes receiving the RREQ packet append their address to the route record field, update the route cache if necessary, and rebroadcast the RREQ to neighbouring nodes.
- Route Reply: If the destination node receives the RREQ or has a valid route, it generates a Route Reply (RREP) packet containing the recorded route from the RREQ packet in reverse order. To the original node, we return the RREP packet.
- Route Maintenance: Nodes periodically monitor route connectivity. If a broken link or route timeout is detected, the affected route is invalidated in the route cache, and a Route Error (RERR) packet is sent to the source node to notify it of the broken route.
- Data Transmission: The source node retrieves a valid route from the route cache, encapsulates the data in a packet, and appends the recorded route as the source route. The packet is sent to the next hop according to the source route.
- Intermediate Node: After receiving a data packet, intermediate nodes send it to the next hop as specified by the source route.
- Destination Node: The destination node receives the data packet, extracts the original data, and processes it.
- Route Error Handling: In case of a connection failure or route timeout, an intermediary node would generate a Route Error (RERR) packet and broadcast it to neighbouring nodes.

3.2. Firefly Optimization

Firefly Algorithm (FA) is a population-based optimization method that emulates the behaviour of fireflies in terms of their flashing and communication patterns. In this algorithm, a firefly's brightness represents a particular solution's quality or effectiveness. A brighter firefly is considered more attractive and attracts other fireflies towards itself. This concept is analogous to how fireflies in nature use their brightness to attract mates or prey.

A firefly's attractiveness is proportionate to its brightness, implying that fireflies with greater objective values are more appealing than other fireflies. But because the material used for communication absorbs light, the allure of fireflies diminishes rapidly with increasing distance.

This attenuation of attractiveness with distance reflects the real-world phenomenon where the intensity of a light source decreases as it travels through a medium. The Firefly Algorithm operates based on three fundamental steps:

Step 1: All fireflies within the population are considered unisexual, meaning each firefly can attract or be attracted to any other firefly. This assumption ensures that specific mating or pairing restrictions do not limit the search process.

Step 2: The appeal of a firefly is in direct correlation with its luminosity, which indicates the quality of a solution. Fireflies lose brilliance as the gap between them grows, resulting in a proportional decrease in attractiveness. This relationship between attractiveness and brightness drives the movement and interaction of fireflies during the optimization process.

Step 3: If a firefly fails to find a more attractive firefly in its immediate neighbourhood or adjacent regions, it moves randomly. This random movement allows search space exploration, enabling the algorithm to escape local optima and discover potentially better solutions.

The natural behaviour of fireflies inspired the FA, where their brightness serves as a means of communication and attraction. By simulating these flashing and communication dynamics, the algorithm optimizes the search for high-quality solutions by leveraging fireflies' attractiveness and movement patterns within a population.

The calculation of a distance equal to that between two fireflies, denoted as s and w, can be straightforwardly determined using the Euclidean distance formula presented in Eq.(1). In Eq.(1), Y represents the dimensionality of the problem, and p_{sa} corresponds to the ath dimension of the sth firefly. To be more specific, the Euclidean distance between fireflies s and w, denoted as b_{sw}, is calculated by taking the square root of the total squared variations between s and w, respectively.

RESEARCH ARTICLE

$$b_{sw} = \sqrt{\sum_{a=1}^{Y}(p_{sa} - p_{wa})^2} \quad (1)$$

The appeal of a firefly diminishes exponentially with the increment in the distance between fireflies. This appeal was denoted as "$\gamma_{(b)}$," is calculated using Eq.(2). In this, μ represents the coefficient for light absorption, "c" is a factor that adjusts the distance metric, and $\gamma_{(0)}$ represents the attractiveness at zero distance, typically considered as one. The specific expression for $\gamma_{(b)}$ is given in Eq.(2).

$$\gamma_{(b)} = \gamma_{(0)} h^{-\mu_b^c}\ c \geq 1 \quad (2)$$

Where "h" denotes the base of the exponential function.

The movement of a firefly, "s," towards another firefly, "w," which is more attractive (brighter), can be determined using Eq.(3). Within Equation (3), the symbol δ represents a scaling factor for randomness, while "א" signifies a random number ranging from 0 to 1, as provided by the user. The movement of firefly "s" is represented in Eq.(3).

$$p_{sa} = p_{sa} + \gamma_{(0)} h^{-\mu b^2}(p_{wa} - p_{sa}) + \delta(\aleph - 0.5) \quad (3)$$

Eq.(3) describes how the position of firefly "s" is updated based on its current position, the attractiveness of firefly "w," and a random perturbation determined by the scaling factor "$\delta\omega$" and the random number א."

Although the Firefly Algorithm (FA) provides an extensive pairwise comparison that can be time-consuming, it is commonly mentioned in the literature that the value of "μ" is often assumed to be one. However, with more separation between lights, the term $\gamma_{(0)} h^{-\mu b^2}$ rapidly tends towards zero. This behaviour can potentially result in fireflies getting stuck in their positions during evaluations. To overcome this issue, setting μ to minimal values can be considered, but determining the appropriate value for "μ" is not a straightforward task. Considering these two limitations, an enhanced version of the Firefly Algorithm has been developed, which proves to be particularly effective for solving combinatorial dynamic optimization problems. Algorithm 1 provides the processes involved in Firefly Optimization.

Input:

- Population size
- Termination condition: Maximum number of iterations
- Coefficient for light absorption (μ): 1
- Scaling factor for randomness (δ): 0.2
- Base of the exponential function (h): 0.5

Output:

- Best solution (firefly) found during the optimization process

Procedure:

Step 1: Initialize the population of fireflies randomly within the search space.

Step 2: Evaluate each firefly's attractiveness (brightness) based on its solution's quality.

Step 3: Repeat the following steps until a termination condition is met:

a. Calculate the distance to other fireflies for each firefly using the Euclidean distance formula.

b. Update the attractiveness of each firefly based on its distance from other fireflies using the attractiveness equation.

c. Move each firefly towards more attractive fireflies based on their current position, the attractiveness of other fireflies, and a random perturbation.

d. If a firefly fails to find a more attractive firefly in its immediate neighbourhood, move it randomly.

e. Evaluate the new positions of fireflies and update their attractiveness.

Step 4: Select the best firefly (solution) based on the highest attractiveness value achieved during optimization.

Algorithm 1 Firefly Optimization

3.3. Relentless Firefly Optimization

Relentless Firefly Optimization (RFO) introduces an adaptive parameter ς, decisive in fine-tuning the algorithm's behaviour. It is calculated using Eq. (4).

$$\varsigma = \frac{mod\big((gen-1), genMAX\big)}{genMAX} \quad (4)$$

Where gen represents the current generation, and $genMAX$ is the maximum number of generations. By incorporating this parameter, RFO enables dynamic adjustment of ς throughout the optimization process. Another critical parameter in RFO is ξ, which counts to see if a firefly is relocated. Its value is determined by Eq. (5), where $rank_s$ represents the rank of the sth firefly:

$$\xi = (rank_s)^{-\varsigma} \quad (5)$$

RESEARCH ARTICLE

Unlike the core FA, which is dependent entirely on the radiance of more alluring fireflies, RFO introduces the ξ parameter also to consider a random number, $\beta\omega$, that must fall below the threshold value of ξ for a firefly to be moved. This additional criterion influences the movement of fireflies and helps balance exploration and exploitation during the optimization process. RFO modifies the move function used in FA, swapping out the complex move function for its simpler counterpart in Eq.(6).

$$p_{sa} = p_{sa} + \gamma_{(0)}\left(\frac{1}{\Omega + b}\right)(p_{wa} - p_{sa}) + \delta(\aleph - 0.5) \qquad (6)$$

where Ω is a minimal value, such as 0.000001, introduced to avoid division by zero. The coefficient $\gamma_{(0)}$ used in FA is replaced with r_a, which represents the velocity or step size of the ath dimension in the solution vector. This modification enhances the algorithm's intensification capability, allowing for more efficient convergence towards optimal solutions.

To further enhance the intensification capability of RFO while preserving its divergence capability, Eq.(7) replaces the constant co-efficient $\gamma_{(0)}$ with r_a:

$$p_{sa} = p_{sa} + r_a\left(\frac{1}{\Omega + b}\right)(p_{wa} - p_{sa}) + \delta(\aleph - 0.5) \qquad (7)$$

Where r_a determines the step size for each dimension of the solution vector. By replacing $\gamma_{(0)}$ with r_a RFO achieves a better balance between intensification and diversification during optimization.

The step sizes in RFO are initially set to relatively large values to facilitate search space exploration. However, as the generations progress and the population converges towards promising regions, these step sizes are gradually reduced using Eq.(8).

$$r_a = r_a - h^{\left(-mod\left((gen/(gen+1)),1\right)\right)} * \tau * r_a \qquad (8)$$

Where gen represents the generation counter, and τ is a scaling parameter. The reduction in step sizes allows for more focused and slower movements of the fireflies, facilitating a more refined search process towards optimal solutions.

In addition to these adaptations, both FA and RFO incorporate two local search procedures. The first procedure, Random Fly, introduces randomness into the search process, enabling the algorithm to escape local optima traps. The subsequent step, Incumbent Local Search, focuses on enhancing the current solution by examining the neighbouring solutions. By employing these local search techniques, the algorithm's ability to explore and exploit the search space is enhanced, ultimately improving the overall performance of the optimization process.

3.3.1. Random Fly

The Random Fly technique is an active and ongoing approach used in local search algorithms to thoroughly explore the adjacent areas surrounding a main section within a firefly swarm. This technique aims to improve the quality of solutions by efficiently navigating the solution space. This technique divides the population into different sections based on their fitness levels. The prominent section constitutes approximately 10% of the population and consists of high-quality solution vectors. To reduce the computational burden, the Random Fly procedure is exclusively applied to the top 12% of the population, ensuring that the search focuses on the most promising solutions.

To implement the Random Fly procedure, this research denotes the population size as N and the best 10% of the population as M, where M represents 0.1 multiplied by N. The Random Fly procedure is performed for each firefly within the M population with a probability of 0.45. By selecting fireflies from this subset, we concentrate our efforts on exploring the most favourable regions of the solution space. When applying the Random Fly procedure to a specific firefly, we aim to detect improvements by evaluating neighbouring solutions' fitness or objective function. This involves examining the quality of solutions within a certain proximity of the current firefly. If an enhancement is detected, we make the necessary adjustments to integrate the improved solution into the population. By executing the Random Fly procedure, the algorithm seeks to refine and optimize the overall population by incorporating the identified improvements. This iterative process allows for continuous exploration and exploitation of the solution space, gradually leading to better solutions. Algorithm 2 expresses the random local search involved in the optimization.

Step 1: For $i = 1$ to M:

Step 2: With probability 0.45:

Step 3: Perform Random Fly procedure on firefly i

Step 4: If an improved solution is found:

Step 5: Update the firefly population with the improved solution

Algorithm 2 Random Local Search

3.3.2. Incumbent Local Search

The Incumbent Local Search procedure aims to enhance the most optimal solution discovered thus far, considering the progress made in the current generation. However, this procedure is postponed to prevent premature convergence and allow for sufficient exploration until specific criteria are met. To be more specific, the Incumbent Local Search is withheld from implementation during the initial 10% of the maximum number of generations.

RESEARCH ARTICLE

Let G_{max} denote the maximum number of generations. The Incumbent Local Search is postponed until generation 0.1 * G_{max}. After this initial period, the Incumbent Local Search is applied at each subsequent generation with a probability of 0.35. If the Incumbent Local Search procedure is applied, improvements are detected by evaluating the fitness or objective function of the most promising approach thus far. If an improvement is found, the necessary updates are made to incorporate the improved solution into the population. Mathematically, the Incumbent Local Search procedure can be represented as Algorithm 3.

Step 1: At generation gen:

Step 2: If gen ≥ 0.1 * G_{max}:

Step 3: With probability 0.35:

Step 4: Perform Incumbent Local Search procedure

Step 5: If an improved solution is found:

Step 6: Update the firefly population with the improved solution

Algorithm 3 Incumbent Local Search

3.3.3. Generating Neighbourhood Solutions

Within the local search methods, neighbouring solutions are generated based on the characteristics of the decision variables present in the optimization problem. Specific neighbourhood functions are employed for this study's test problems, which involve integer and continuous design variables. When dealing with integer design variables, a new solution vector within the neighbourhood is generated by applying Equation (9) to the current solution vector:

$$p_{sa}^{*} = p_{sa} + integer[(\pi() - 0.5) \times r_a] \quad (9)$$

The value of the ath design variable for the sth firefly before the neighbourhood move is denoted as p_{sa}, while p_{sa}^{*} represents the updated value of the same design variable after the neighbourhood move. The ath dimension is associated with a step size or velocity represented by r_a, $\pi()$ denotes a chaotic or random number generator that generates numbers within the range of [0,1].

Using chaotic or random numbers introduces an element of randomness and diversity in the neighbourhood search process, aiding in exploring the search space. By employing the appropriate neighbourhood functions based on the variable types, the local search procedures in the algorithm can generate neighbouring solution vectors for integer and continuous design variables, allowing for exploration and potential improvement within the local search neighbourhoods. Algorithm 4 provides processes involved in relentless firefly optimization.

Input:

- Population size (N)
- Maximum number of generations (genMAX)
- Scaling parameter (τ)
- Threshold value (Ω)

Output:

Optimized Solution

Procedure:

Step 1: Initialize a population of fireflies randomly.

Step 2: Set the generation counter to 1.

Step 3: Calculate the adaptive parameter ς based on the current generation and the maximum number of generations.

Step 4: Sort the fireflies based on their fitness, from best to worst.

Step 5: Update the step sizes for each dimension of the solution vector.

Step 6: For each firefly in the population, do the following:

a. Calculate ξ for the firefly based on its rank and the adaptive parameter ς.

b. Generate a random number βω.

c. If βω is less than ξ, skip the firefly and move to the next one.

d. Perform the Random Fly procedure with a probability of 0.45:

Step 7: Select a neighbourhood around the firefly.

Step 8: Evaluate the fitness of neighbouring solutions.

Step 9: If an improved solution is found, update the firefly population.

Step 10: If the generation is greater than or equal to 0.1 times the maximum number of generations, do the following:

a. With a probability of 0.35, perform the Incumbent Local Search procedure:

Step 11: Evaluate the fitness of the best solution found so far.

Step 12: If an improved solution is found, update the firefly population.

Step 13: Increment the generation counter by 1.

RESEARCH ARTICLE

Step 14: Check if the termination criteria are met. If yes, stop the algorithm; otherwise, go back to step 3.

Algorithm 4 Relentless Firefly Optimization

3.4. Chaos

Chaos exhibits erratic and seemingly unpredictable patterns within a nonlinear system, even when subjected to deterministic conditions. This phenomenon emerges due to its sensitivity to initial conditions and an infinite number of distinct periodic responses. Notably, even minor differences in the initial setup of a chaotic system can lead to significant divergence in the final output. To simulate chaos, scientists frequently utilize chaotic maps, discrete-time functions that define the connection between the current state and the next state of a chaotic system. This study focuses on one-dimensional chaotic maps due to their practical relevance. These maps are characterized by their continuous value outputs and discrete-time evolution. A one-dimensional chaotic map can be expressed mathematically as Eq.(10).

$$p_{t+1} = \gamma(p_t) \tag{10}$$

Where p_{t+1} represents the subsequent state, p_t denotes the current state, and γ represents the forward transformation mapping function that governs the system's dynamics.

Three chaotic maps, namely the logistic, sinusoidal, and tent maps, are deemed appropriate and are consequently utilized in this study. Each map has unique properties and characteristics that make it well-suited for modelling chaotic behaviour in ad-hoc networks. Moreover, this study introduces a probabilistic amalgamation of these chaotic maps, piecewise chaos. Under the piecewise chaos scheme, generating the following chaotic number involves a randomly generated number $b\omega(0,1)$. Based on the value of b, different chaotic maps are utilized to generate random chaotic numbers. If b is less than 0.33, the chaotic logistic map is used; if b is between 0.33 and 0.66, the sinusoidal chaotic map is employed; and if b is greater than or equal to 0.66, the tent map is chosen.

Each chaotic map maintains its own set of separate predecessor chaotic numbers. These predecessor numbers are utilized to generate subsequent chaotic numbers when needed throughout the generations of the algorithm. Moreover, to enhance the diversity and randomness of the generated chaotic sequences, each chaotic map is initialized with different random seeds, ensuring that distinct and independent chaotic trajectories emerge. By incorporating chaotic maps and the piecewise chaos approach, the research exploits the inherent properties of chaos, such as sensitivity to initial conditions and the generation of seemingly random and unpredictable sequences. This utilization of chaos aids in diversifying the exploration of the search space. It enhances the algorithm's ability to discover novel and promising solutions in the context of ad-hoc networks.

3.4.1. Logistic Map

The logistic map illustrates the emergence of intricate patterns from a remarkably straightforward nonlinear dynamical equation. A polynomial function produces chaotic sequences within the interval (0, 1). Additionally, it is widely recognized as one of the most commonly employed chaotic number generators. This map can be mathematically expressed through the following Eq.(11).

$$p_{t+1} = \tau p_t(1 - p_t)\ 0 < \tau \leq 4, p_t\omega(0,1) \tag{11}$$

The logistic map showcases the behaviour of a variable p_t as it evolves over discrete time steps. At each iteration, the value of p_{t+1} is determined by multiplying the current value of p_t by the factor τ and subtracting the product of p_t and $1 - p_t$. This Eq.(12) is constrained by the condition that τ must be greater than 0 but not exceed 4 while p_t should lie within the open interval $(0, 1)$. The logistic map exhibits intriguing properties, particularly when τ reaches specific critical values. For τ values below approximately 3.57, the map's behaviour is stable and converges towards a fixed point or a limit cycle, depending on the specific value of τ. However, when τ surpasses this critical threshold, the system transitions into a chaotic regime characterized by sensitive dependence on initial conditions. Even minute variations in the starting value of p_t can lead to significant divergences in subsequent iterations, giving rise to a complex and unpredictable sequence of values.

3.4.2. Sinusoidal Map

The sinusoidal map is utilized in RFO to exhibit chaotic behaviour, as defined in Eq.(12).

$$p_{t+1} = sin(\pi p_t)p_t\ \omega\ (0,1) \tag{12}$$

Where p_t represents the current value in the sequence, and $p_{(t+1)}$ denotes the next value. The sinusoidal map operates within the range of 0 to 1. The map derives its name from the sine function (sin), which calculates the next value in the sequence. By applying the sine of π times p_t, the map introduces nonlinearity and amplifies minor differences in initial values, leading to chaotic behaviour. The multiplication of p_t by $sin(\pi p_t)$ contributes to the generation of unpredictable and complex sequences. The term $\omega\ (0,1)$ represents a random variable between 0 and 1, serving as a factor to introduce randomness into the map.

3.4.3. Tent Map

The tent map is another chaotic map that shares similarities with the logistic map. It generates chaotic sequences within

RESEARCH ARTICLE

the interval of 0 to 1. Eq.(13) describes the tent map as follows:

$$p_{t+1} = \begin{cases} \frac{p_t}{0.7} p_t < 0.7 \\ \left(\frac{10}{3}\right) p_t(1-p_t) \quad otherwise \quad p_t \omega(0,1) \end{cases} \tag{13}$$

Where p_t is a random variable ranging from 0 to 1.

The tent map exhibits dynamic and nonlinear behaviour, making it capable of producing chaotic sequences. It operates based on the current value p_t, determining the next value, $p_{(t+1)}$, through a piecewise definition. When p_t is less than 0.7, the next value is obtained by dividing p_t by 0.8. This division effectively compresses the values towards zero, causing the sequence to converge. However, if p_t is greater than or equal to 0.7, the next value is computed using the expression $\left(\frac{10}{3}\right) p_t(1-p_t)$. This nonlinear function introduces a quadratic term and leads to a more expansive behaviour, driving the sequence away from convergence.

The tent map's distinct behaviour arises from the interplay between these two branches. It generates chaotic sequences exhibiting convergence and divergence patterns, resulting in intricate and unpredictable dynamics. The random variable ω (0,1) adds a stochastic element to the map, contributing to the randomness and variability observed in the generated sequences.

3.5. Handling Infeasibility

The problems in optimizing the best route to send the fintech data have been expressed as mathematical programming models involving nonlinear constraints. This approach is commonly found in existing literature on the subject. Hence, utilizing a methodology capable of effectively managing these constraints becomes imperative. In a design optimization problem with constraints, the key objective is to optimize a linear or nonlinear objective function, represented as $g(p)$, concerning the design variables p. However, this optimization must be done while satisfying a set of linear and nonlinear equality and inequality constraints. To formally define the constrained design optimization problem, Eq.(14)-Eq.(16) can be applied.

$$\min g(p), \quad p\, \omega\, Y \subseteq B^{t_m} \tag{14}$$

Subject to:

$$j_z(p) = 0, \quad j_z: B^{t_m} \to B, \quad z = 1,2,3,\dots,c_h \tag{15}$$

$$l_z(p) \leq 0, \quad l_z: B^{t_m} \to B, \quad z = 1,2,3,\dots,c_s \tag{16}$$

The intention of Eq.(14) is to minimize the objective function $g(p)$ while ensuring that the design variables p fall within the search space Y, which is a subset of B^{t_m}. Here, B represents the feasible region of the design variables, and t_m represents the dimensionality of the design variables.

Eq.(15) represents the set of equality constraints, where each $j_z(p)$ the function maps the variable design space B^{t_m} to the constraint space B. These equality constraints define specific relationships among the design variables that must be satisfied. Eq.(16) represents the set of inequality constraints. The $l_z(p)$ function maps the variable design space B^{t_m} to the constraint space B. These inequality constraints impose bounds or restrictions on the design variables, ensuring they remain within certain limits. It is essential to mention that the formulation of equality constraints presented in Eq.(15) can also be adapted to incorporate inequality constraints. In such cases, the design optimization problem would solely consist of inequality constraints, providing greater flexibility in handling the constraints based on the specific requirements of the problem.

Several commonly used techniques are employed to overcome infeasible circumstances in optimization problems with constraints, including penalizing, rejecting, and repairing. These methods aim to find feasible solutions by addressing violations of the constraints. However, most of these techniques require additional efforts and often involve introducing penalty terms or other problem-specific parameters. However, an alternative approach is similar to the penalty function approach but eliminates the need for explicit penalty terms. This approach addresses the challenge of constrained nonlinear design optimization, as defined by Eq.(15)-Eq.(17), by transforming it into a constrained optimization problem using Eq.(18).

$$\min_{p\omega Y} Z(p) = \begin{cases} \hat{l}(p) = l_{max}(p), \quad if\ l_{max}(p) > 0 \\ \hat{g}(p) = dtan[g(p)] - \frac{\pi}{2}, \quad otherwise \end{cases} \tag{18}$$

In Eq.(18), the objective function $Z(p)$ is minimized concerning the variable p, considering the following conditions. First, the maximum value of a set of constraints $l_z(p)$, denoted as $l_{max}(p)$, is computed. If $l_{max}(p)$ is more significant than zero, the modified constraint $\hat{l}(p)$is set equal to $l_{max}(p)$. On the other hand, if $l_{max}(p)$ is not favourable, another modified constraint $\hat{g}(p)$ is defined as $dtan[g(p)] - \frac{\pi}{2}$, where $dtan[.]$ represents the inverse tangent function. It is essential to highlight that the modified constraint $\hat{g}(p)$ is always negative for any value of p, thereby ensuring that $\hat{g}(p)$ remains less than $\hat{l}(p)$. This condition guarantees the satisfaction of the constraints. One good aspect of this approach is its ability to operate independently without the need for additional problem-specific parameters such as penalty factors or Fourier series. The approach simplifies the optimization process and reduces the computational burden

RESEARCH ARTICLE

by eliminating the need for these additional parameters. Chaos-based Chaotic Sequence Generator is expressed in Algorithm 5.

Input:

- Choice of chaotic map (i.e., logistic, sinusoidal, or tent map)
- Parameters and initial conditions for the chosen chaotic map (i.e., τ for the logistic map or specific values for the sinusoidal or tent map)
- Number of iterations to generate the chaotic sequence

Output:

- Chaotic sequence generated by the chosen chaotic map

Procedure:

Step 1: Choose a chaotic map: Select one of the chaotic maps mentioned, such as the logistic map, sinusoidal map, or tent map.

Step 2: Initialize parameters: Set the conditions and parameters specific to the chosen chaotic map.

Step 3: Generate chaotic sequence: Iterate the chaotic map equation for a desired number of iterations. Calculate the next state at each iteration based on the current state and the map equation.

Step 4: Store the chaotic numbers: Store the generated chaotic numbers in a sequence or an array for further use.

Step 5: Repeat steps 2-4 for each chaotic map: If you want to incorporate multiple chaotic maps or the piecewise chaos approach, repeat steps 2-4 for each map. Each map will have its own set of parameters and initial conditions.

Step 6: Utilize the chaotic sequences: After generating the chaotic sequences, it is possible to use them in various applications, such as diversifying the exploration of a search space or introducing randomness into algorithms.

Algorithm 5 Chaos-Based Chaotic Sequence Generator Algorithm

3.6. Modified RSA

The enhanced version of the RSA (Rivest-Shamir-Adleman) algorithm incorporates additional security measures to strengthen its resistance against attacks. This advanced variant builds upon the original RSA algorithm, utilizing key length increases, rigorous primality testing, robust random number generation, secure padding schemes, meticulous key management, strong cryptographic hash functions, protection against side-channel attacks, and regular security updates. By implementing these enhancements, the modified RSA algorithm offers improved protection against various cryptographic vulnerabilities, guaranteeing sensitive data's privacy, security, and genuineness in light of evolving threats and technological progress. The steps involved are:

- Key length: Increase the size of the prime numbers used for key generation. Using more significant prime numbers makes it harder for an attacker to factorize the modulus and retrieve the private key. A key length of 2048 bits or higher is generally recommended for RSA encryption.
- Primality testing: Employ robust primality testing algorithms, such as Miller-Rabin or Solovay-Strassen, to verify the primality of chosen numbers and ensure their authenticity as prime numbers. This helps prevent the use of composite numbers that may have vulnerabilities.
- Random number generation: Use a strong generator to select the prime numbers and the public exponent. High-quality randomness ensures that the generated keys are not predictable, making it more difficult for an attacker to guess the private key.
- Padding schemes: When encrypting the plaintext, apply secure padding schemes, such as Optimal Asymmetric Encryption Padding (OAEP). The padding adds randomness and helps prevent certain types of attacks, such as chosen ciphertext attacks.
- Key management: Ensure proper essential management practices, such as protecting the private key, securely exchanging public keys, and periodically updating keys. Use secure storage mechanisms and consider hardware-based key storage for increased protection.
- Cryptographic hash function: Utilize a secure and robust cryptographic hash function for generating message digests in signature generation and verification. Popular choices include SHA-256 or SHA-3.
- Side-channel attacks: Protect against side-channel attacks, such as timing or power analysis attacks, exploiting information leaked during encryption or decryption. Implement countermeasures such as constant-time algorithms or hardware-level protections.
- Regular security updates: Stay up to date with the latest cryptographic recommendations and vulnerabilities. Keep track of any security updates or patches related to RSA or its implementation libraries to address potential vulnerabilities.

The Fusion of RFORP and Modified RSA with DSR is provided in Algorithm 6.

RESEARCH ARTICLE

Step 1: Implement the RFO algorithm for optimizing the DSR routing protocol by using individual Fireflies to represent potential routes.

Step 2: Modify the RSA algorithm to enhance its search capability and adaptability in the DSR routing context.

Step 3: Integrate chaos concept into the RFO and modified RSA algorithms to introduce randomness and enhance exploration.

Step 4: Use chaos-based mutation operators to diversify the search space and prevent getting stuck in local optima.

Step 5: Employ chaos-based crossover operators to facilitate information exchange and exploration among fireflies or particles.

Step 6: Consider the influence of chaotic values while updating local and global best solutions.

Step 7: Incorporate chaos-based perturbation techniques to introduce random disturbances and escape stagnant regions.

Step 8: Apply chaos-driven adaptive strategies to adjust the merged algorithm's control parameters dynamically.

Step 9: Incorporate chaos-based mechanisms to handle dynamic network scenarios, such as node failures or changing topology.

Step 10: Evaluate the performance of the merged algorithm using appropriate metrics, considering optimized DSR routes, convergence speed, and robustness to network changes.

Step 11: Conduct comprehensive simulations or experiments to compare the merged algorithm with other optimization techniques.

Step 12: Fine-tune the merged algorithm by adjusting the chaotic, operator, or control parameters based on evaluation results.

Step 13: Document the merged algorithm, including the integration of RFO, modified RSA, and chaos, along with specific steps, parameters, and considerations.

Algorithm 6 Fusion of RFORP and Modified RSA with DSR

4. SIMULATION SETTINGS

GNS3 (Graphical Network Simulator-3) is a powerful and versatile tool that allows network engineers and researchers to simulate routing protocols tailored explicitly for IoT-AN. Ad-hoc networks play a crucial role in the Internet of Things (IoT) ecosystem, where devices must communicate directly with each other in a decentralized manner. With GNS3, users can design and emulate complex network topologies, incorporating virtual router images to simulate the behaviour of real routers. This allows for the evaluation and analysis of various routing protocols, such as OLSR (Optimized Link State Routing), DSR (Dynamic Source Routing) and AODV (Ad-hoc On-Demand Distance Vector), which are commonly employed in IoT-AN. GNS3's traffic generation capabilities enable the simulation of realistic IoT traffic patterns, enabling users to assess the performance of the routing protocols under different load conditions. GNS3 provides a range of monitoring and analysis tools that allow users to observe and analyze the behaviour of the simulated network, including routing tables, network traffic flows, and packet captures. This enables an in-depth evaluation of the routing protocols' efficiency, scalability, and reliability in IoT-AN. Simulation settings used to evaluate the proposed protocol are provided in Table 2.

Table 2 Simulation Settings

Parameters	Settings
Data Rate	1 Mbps
Energy Model	Battery Drain Model
Initial Energy Level	5 Joules
Interference Model	Path Loss Model
MAC Protocol	802.16
Mobility Model	Random Waypoint Model
Network	Mobile Ad-hoc Network
Network Density	Moderate
Node Placement Distribution	Uniform Distribution
Number of IoT Devices	250
Number of Malicious Nodes	10%
Number of Packets	4000
Packet Size	512 bytes
Quality of Service (QoS)	Best Effort
Simulation Area	3000 × 4000 m²
Simulation Time	300 seconds
Traffic Source	Constant Bit Rate (CBR)
Transmission Range	350 meters

International Journal of Computer Networks and Applications (IJCNA)
DOI: 10.22247/ijcna/2023/223319 Volume 10, Issue 4, July – August (2023)

RESEARCH ARTICLE

5. RESULTS AND DISCUSSION

5.1. Packet Delivery Ratio

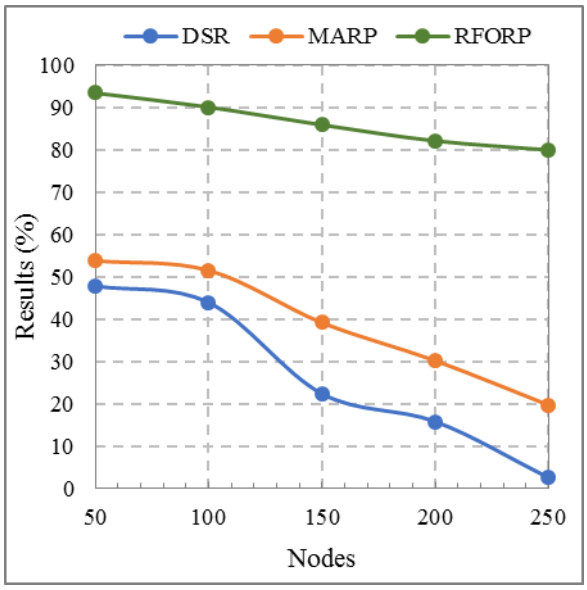

Figure 1 Packet Delivery Ratio

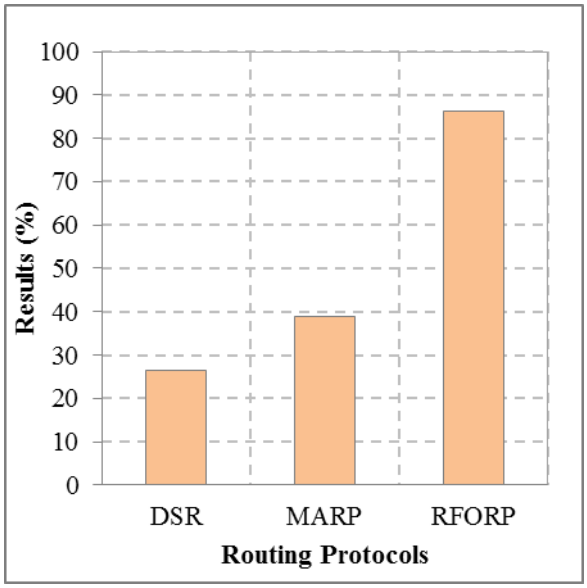

Figure 2 Average Packet Delivery Ratio

Figure 1 and Figure 2 present the packet delivery ratio results for three routing protocols: Dynamic Source Routing (DSR), Multi-Adaptive Routing Protocol (MARP), and Relentless Firefly Optimization Routing Protocol (RFORP). The packet delivery ratio represents the percentage of successfully delivered packets from the source to the destination. Looking at the results, RFORP achieved the highest packet delivery ratio across all distance ranges, with an average of 86.313%. This indicates that RFORP has a superior performance in terms of successful packet delivery compared to the other two protocols. MARP follows RFORP with an average packet delivery ratio of 38.870%, while DSR has the lowest average ratio of 26.490%. The significant difference in performance can be attributed to the design and functionality of each protocol. RFORP is specifically inspired by the natural characteristics of fireflies and utilizes optimization techniques derived from them, which likely leads to efficient routing decisions and higher packet delivery. DSR and MARP may lack certain optimization features or adaptive mechanisms, resulting in lower packet delivery ratios.

5.2. Packet Drop Ratio

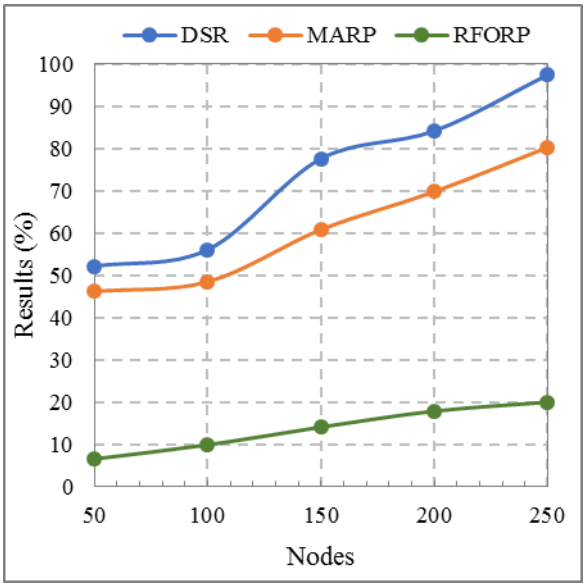

Figure 3 Packet Delivery Ratio

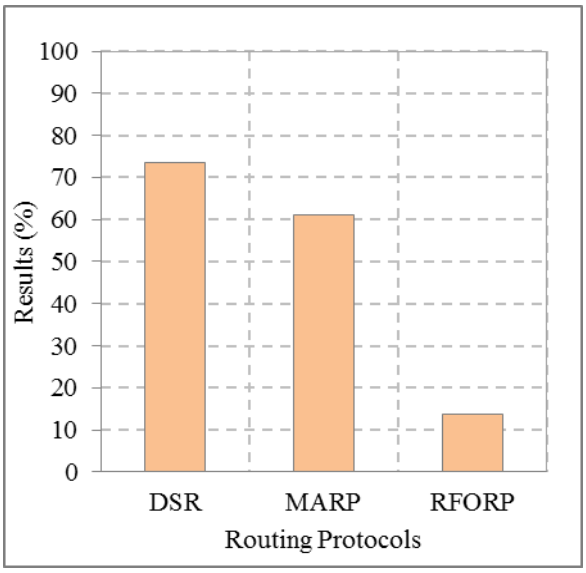

Figure 4 Average Packet Drop Ratio

RESEARCH ARTICLE

Figure 3 and Figure 4 present the packet drop ratio for three different routing protocols: Dynamic Source Routing (DSR), Multi-Adaptive Routing Protocol (MARP), and Relentless Firefly Optimization (RFORP). The packet drop ratio indicates the percentage of packets that were not successfully delivered from the source to the destination and were instead dropped or lost during transmission. The results reveal that DSR had the highest average packet drop ratio of 73.510%, followed by MARP, with an average of 61.130%. In contrast, RFORP demonstrated the lowest average packet drop ratio of 13.687%. These variations in performance can be attributed to the design and mechanisms employed by each routing protocol. DSR may experience higher packet drop ratios due to its reliance on dynamic source routing, where route maintenance and frequent updates may introduce more opportunities for packet loss. MARP, although inspired by the natural characteristics of fish, might still lack certain adaptive features necessary for efficient packet delivery. In contrast, RFORP, with its relentless firefly optimization approach, likely incorporates robust optimization techniques and adaptive decision-making algorithms, resulting in a lower packet drop ratio and more reliable packet delivery overall.

5.3. Delay

Figure 5 and Figure 6 provide the delay results for three distinct routing protocols: Dynamic Source Routing (DSR), Multi-Adaptive Routing Protocol (MARP), and Relentless Firefly Optimization (RFORP). Delay is when it takes packets to traverse the network from the source to the destination. The table shows that DSR has the highest average delay of 6123.2 ms, followed by MARP with an average delay of 5300.2 ms. RFORP demonstrates the lowest average delay of 1649.4 ms.

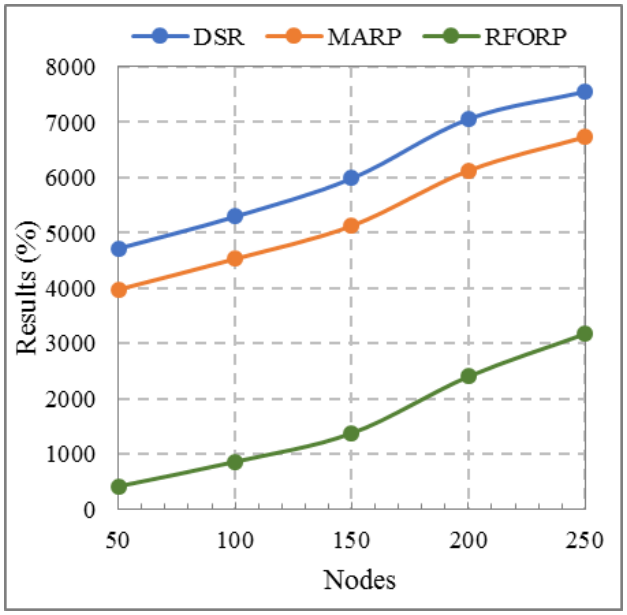

Figure 5 Delay

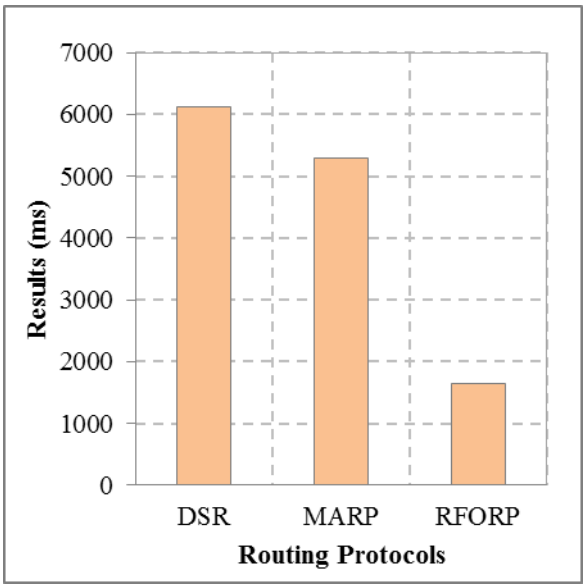

Figure 6 Average Delay

These variations in delay can be attributed to the design and mechanisms employed by each protocol. DSR's higher delay can be attributed to its reliance on dynamic source routing, which involves frequent route updates and maintenance, leading to increased overhead and longer packet delivery times. MARP, despite being inspired by natural fish characteristics, may still lack certain adaptive features necessary for minimizing delay. RFORP's relentless firefly optimization approach likely incorporates efficient optimization techniques and adaptive algorithms, resulting in shorter delays and faster packet delivery times. RFORP achieves the lowest delay among the three routing protocols, making it more suitable for applications that require low-latency communications.

5.4. Energy Consumption

Figure 7 and Figure 8 present the energy consumption results for three routing protocols: Dynamic Source Routing (DSR), Multi-Adaptive Routing Protocol (MARP), and Relentless Firefly Optimization (RFORP). Energy consumption represents the percentage of energy each protocol utilizes for packet transmission within the network. The data shows DSR exhibits the highest average energy consumption of 83.383%, followed by MARP with an average consumption of 60.809%. In contrast, RFORP demonstrates the lowest average energy consumption of 24.866%. These results can be attributed to the design and mechanisms employed by each protocol. DSR's higher energy consumption may be due to its dynamic source routing approach, which requires more frequent route updates and maintenance, resulting in increased energy usage. Although MARP draws inspiration from natural fish characteristics, it may lack certain energy-efficient adaptive features. RFORP, with its relentless firefly

RESEARCH ARTICLE

optimization approach, likely incorporates efficient optimization techniques and adaptive algorithms, leading to lower energy consumption. RFORP's ability to optimize routing decisions and minimize unnecessary energy expenditure contributes to its efficiency. RFORP demonstrates the lowest energy consumption among the three routing protocols, making it suitable for energy-constrained scenarios or networks with limited resources.

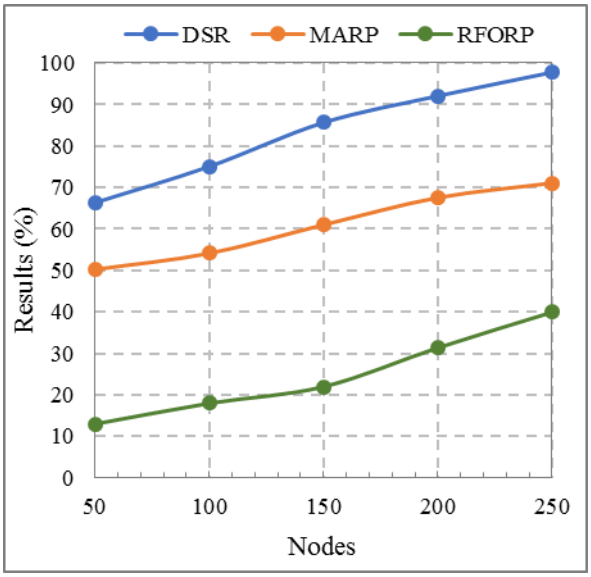

Figure 7 Energy Consumption

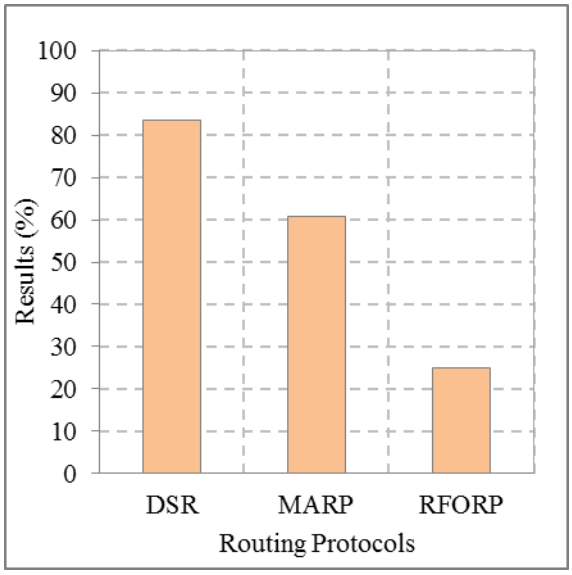

Figure 8 Average Energy Consumption

5.5. Malicious Node Detection Accuracy

Figure 9 and Figure 10 depict the results of malicious node detection accuracy for three routing protocols: Dynamic Source Routing (DSR), Multi-Adaptive Routing Protocol (MARP), and Relentless Firefly Optimization (RFORP). Malicious node detection accuracy refers to the ability of the protocols to identify and detect malicious nodes within the network. Analyzing the data, it is evident that RFORP achieved the highest average malicious node detection accuracy of 77.621%. This indicates that RFORP performs better in accurately identifying and detecting malicious nodes than the other two protocols. MARP follows with an average accuracy of 49.750%, while DSR demonstrates the lowest average accuracy of 23.916%.

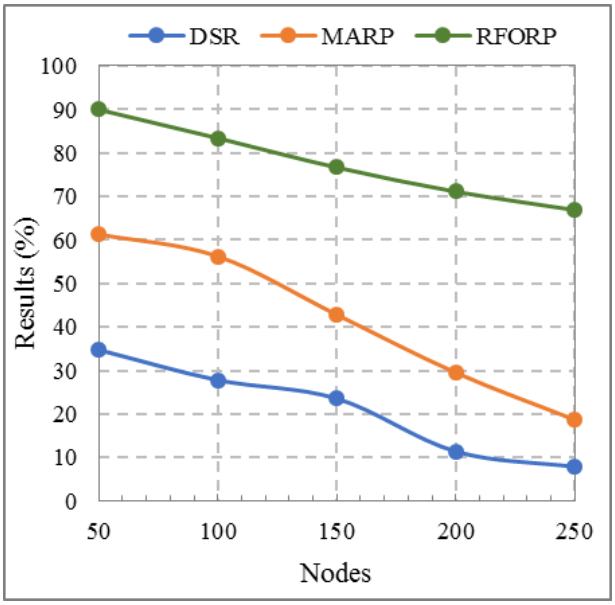

Figure 9 Malicious Node Detection Accuracy

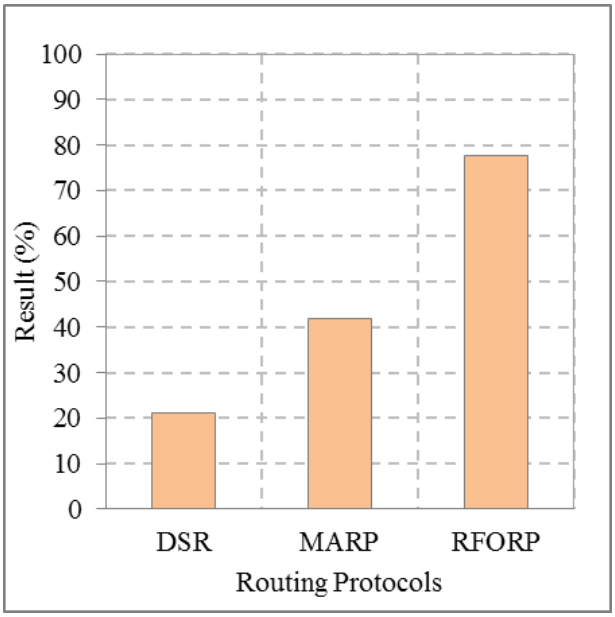

Figure 10 Average Malicious Node Detection Accuracy

RESEARCH ARTICLE

These varying results can be attributed to the design and mechanisms employed by each protocol. RFORP, utilizing its relentless firefly optimization approach, likely incorporates effective techniques for identifying and flagging malicious nodes, leading to higher detection accuracy. MARP, inspired by natural fish characteristics, may possess adaptive features that contribute to relatively better detection accuracy than DSR. DSR may lack robust detection mechanisms, resulting in lower accuracy. RFORP demonstrates the highest average malicious node detection accuracy, making it more reliable and suitable for securing networks against malicious entities.

5.6. Routing Stability

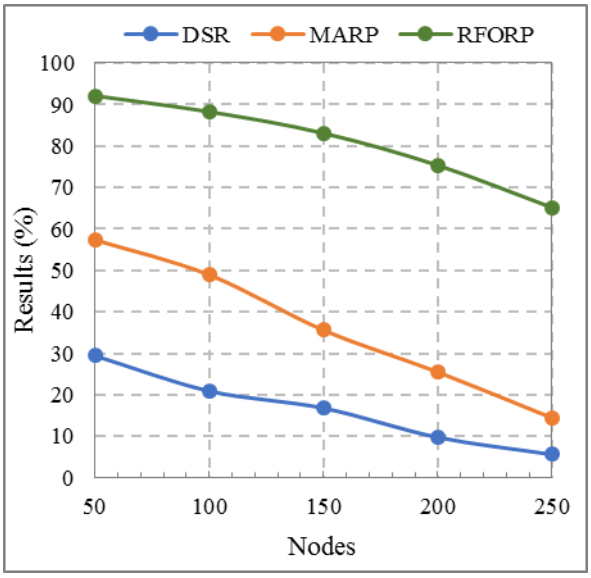

Figure 11 Routing Stability

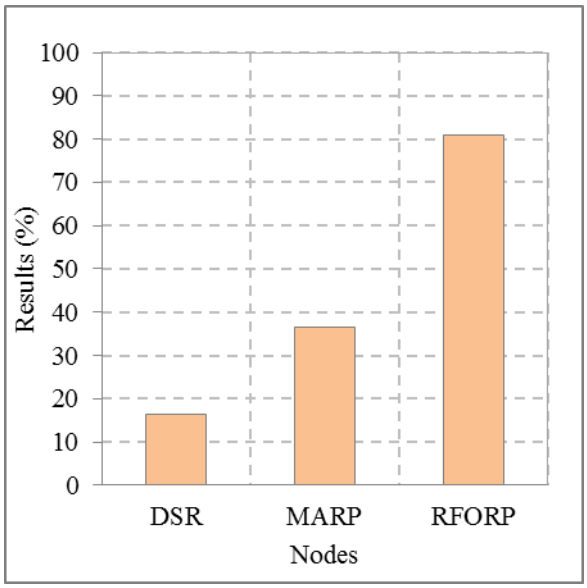

Figure 12 Average Routing Stability

Figure 11 and Figure 12 illustrate the results of routing stability for three routing protocols: Dynamic Source Routing (DSR), Multi-Adaptive Routing Protocol (MARP), and Relentless Firefly Optimization (RFORP). Routing stability refers to the consistency and reliability of the routing paths maintained by the protocols over time. Upon analyzing the provided data, it is evident that RFORP achieved the highest average routing stability with a score of 80.797%. This indicates that RFORP demonstrates superior stability in maintaining consistent and reliable routing paths compared to the other two protocols. MARP follows with an average stability score of 36.385%, while DSR exhibits the lowest average stability of 16.531%. These differences in routing stability can be attributed to the design and mechanisms employed by each protocol. RFORP, with its relentless firefly optimization approach, likely incorporates robust optimization techniques and adaptive algorithms that contribute to stable routing paths. MARP, inspired by natural fish characteristics, may possess adaptive features that enhance routing stability to some extent. DSR may lack specific mechanisms for path stability, resulting in a lower average stability score. RFORP demonstrates the highest average routing stability among the three protocols, making it a suitable choice for applications that require consistent and reliable routing paths.

5.7. Resilience Score

Figure 13 and Figure 14 present the results of the Resilience Score for three routing protocols: Dynamic Source Routing (DSR), Multi-Adaptive Routing Protocol (MARP), and Relentless Firefly Optimization (RFORP). The Resilience Score is a metric that combines three factors: Packet Delivery Ratio (PDR), Malicious Node Detection Accuracy, and Routing Stability. The weights assigned to each metric, α, β, and γ, are 0.4, 0.3, and 0.3, respectively, representing their relative importance in evaluating the resilience of the secure routing protocol.

By calculating the Resilience Score using the provided weights and the data in the table, it becomes possible to assess the overall resilience of each protocol. The Resilience Score considers the performance of packet delivery, malicious node detection, and routing stability, providing a consolidated view of the protocol's effectiveness in the face of attacks and maintaining secure and reliable routing paths. Upon evaluating the results, RFORP achieved the highest average Resilience Score of 82.050. This indicates that RFORP exhibits a higher resilience than the other two protocols. MARP follows with an average Resilience Score of 38.989, while DSR demonstrates the lowest average Resilience Score of 21.890. The significant difference in Resilience Scores can be attributed to the combined performance of PDR, Detection Accuracy, and Routing Stability for each protocol.

RFORP's high Resilience Score suggests that it delivers packets reliably, accurately detect malicious nodes, and

RESEARCH ARTICLE

maintains stable routing paths. This is likely due to the optimizations and adaptive algorithms derived from firefly behaviour employed by RFORP. MARP, which draws inspiration from natural fish characteristics, demonstrates a relatively moderate level of resilience. DSR, on the other hand, exhibits a lower Resilience Score, indicating potential vulnerabilities in packet delivery, malicious node detection, and routing stability. The Resilience Score offers a comprehensive assessment of the overall performance and resilience of the secure routing protocols, enabling informed decision-making processes for network security and protocol selection.

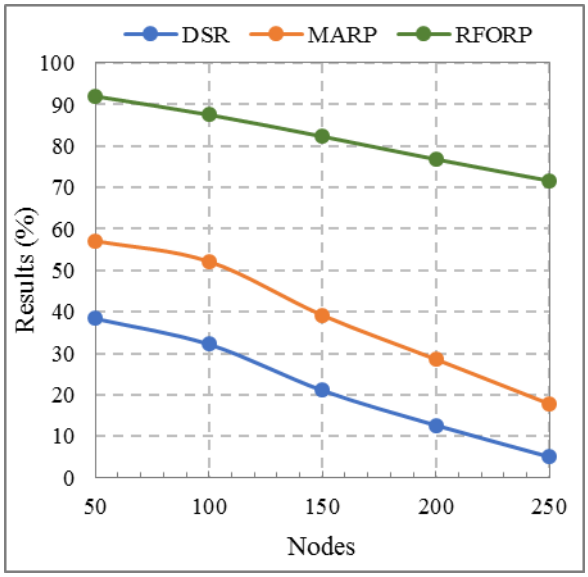

Figure 13 Resilience Score

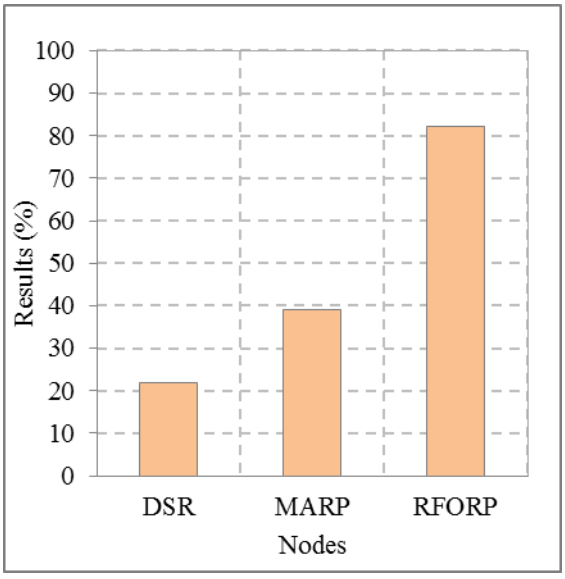

Figure 14 Average Resilience Score

Table 2 Results Values of Resilience Score

Nodes	DSR	MARP	RFORP
50	38.387	57.134	91.995
100	32.209	52.167	87.542
150	21.080	39.209	82.296
200	12.692	28.598	76.794
250	5.084	17.835	71.626
Average	21.890	38.989	82.050

6. CONCLUSION

The Relentless Firefly Optimization-based Routing Protocol (RFORP) development significantly advances securing fintech data within IoT-AN. Through extensive evaluations and comparisons with existing routing protocols, RFORP has demonstrated its superiority across various metrics, including higher packet delivery ratios, accurate malicious node detection, improved routing stability, and significant energy efficiency. These achievements can be attributed to the incorporation of relentless firefly optimization techniques inspired by the natural behaviour of fireflies, which optimize routing paths and enhance the overall performance of the protocol. RFORP's remarkable resilience against attacks, evidenced by its higher Resilience Score compared to the state-of-the-art routing protocol, establishes it as an effective solution for securing fintech data. RFORP addresses the critical challenges associated with fintech data transmission in IoT-AN by offering enhanced performance, reliability, and security. This research's findings contribute to the secure routing protocols field, particularly in the context of fintech applications, providing valuable insights for decision-making processes related to network security and protocol selection. RFORP presents a promising approach to ensuring the secure and efficient routing of fintech data, empowering organizations to effectively protect their sensitive financial information in IoT environments.

REFERENCES

[1] Met, A. Erkoc, and S. E. Seker, "Performance, Efficiency, and Target Setting for Bank Branches: Time Series With Automated Machine Learning," IEEE Access, vol. 11, pp. 1000–1010, 2023, doi: 10.1109/ACCESS.2022.3233529.

[2] N. Vikas, P. Venegas, and S. Aiyer, Role of Banks and Other Financial Institutions in Enhancing Green Digital Finance. 2022. doi: 10.1007/978-981-19-2662-4_16.

[3] E. Avgouleas and H. Marjosola, Digital Finance in Europe: Law, Regulation, and Governance. 2021. doi: 10.1515/9783110749472.

[4] Di. Rojas-Torres, N. Kshetri, M. M. Hanafi, and S. Kouki, "Financial Technology in Latin America," IT Prof., vol. 23, no. 1, pp. 95–98, 2021, doi: 10.1109/MITP.2020.3028486.

RESEARCH ARTICLE

[5] E. Hernández, M. Öztürk, I. Sittón, and S. Rodríguez, "Data protection on fintech platforms," in Communications in Computer and Information Science, 2019, vol. 1047, pp. 223–233. doi: 10.1007/978-3-030-24299-2_19.
[6] R. Kumar, U. Venkanna, and V. Tiwari, "Opt-ACM: An Optimized load balancing based Admission Control Mechanism for Software Defined Hybrid Wireless based IoT (SDHW-IoT) network," Comput. Networks, vol. 188, p. 107888, 2021, doi: 10.1016/j.comnet.2021.107888.
[7] Y. Zhao, Y. Yu, Y. Li, G. Han, and X. Du, "Machine learning based privacy-preserving fair data trading in big data market," Inf. Sci. (Ny)., vol. 478, pp. 449–460, 2019, doi: 10.1016/j.ins.2018.11.028.
[8] Z. Z. Oo and S. Phyu, "Microclimate prediction using cloud centric model based on IoT technology for sustainable agriculture," in 2019 IEEE 4th International Conference on Computer and Communication Systems, ICCCS 2019, 2019, pp. 660–663. doi: 10.1109/CCOMS.2019.8821706.
[9] Y. W. Kuo, W. L. Wen, X. F. Hu, Y. T. Shen, and S. Y. Miao, "A lora-based multisensor iot platform for agriculture monitoring and submersible pump control in a water bamboo field," Processes, vol. 9, no. 5, 2021, doi: 10.3390/pr9050813.
[10] A. S. Sadiq, A. A. Dehkordi, S. Mirjalili, J. Too, and P. Pillai, "Trustworthy and Efficient Routing Algorithm for IoT-FinTech Applications Using Nonlinear Lévy Brownian Generalized Normal Distribution Optimization," IEEE Internet Things J., vol. 10, no. 3, pp. 2215–2230, 2023, doi: 10.1109/JIOT.2021.3109075.
[11] R. Yamparala and B. Perumal, "Efficient malicious node identification method for improving packet delivery rate in mobile ad hoc networks with secured route," J. Crit. Rev., vol. 7, no. 7, pp. 1011–1017, 2020, doi: 10.31838/jcr.07.07.185.
[12] M. A. Ahad and R. Biswas, "PPS-ADS: A framework for privacy-preserved and secured distributed system architecture for handling big data," Int. J. Adv. Sci. Eng. Inf. Technol., vol. 8, no. 4, pp. 1333–1342, 2018, doi: 10.18517/ijaseit.8.4.5465.
[13] P. J. Adinarayana and B. Kishore Babu, "The role of shaping fin-tech services: Social media marketing," Int. J. Innov. Technol. Explor. Eng., vol. 8, no. 10, pp. 1720–1723, 2019, doi: 10.35940/ijitee.J9045.0881019.
[14] R. Vadivel and J. Ramkumar, "QoS-enabled improved cuckoo search-inspired protocol (ICSIP) for IoT-based healthcare applications," Inc. Internet Things Healthc. Appl. Wearable Devices, pp. 109–121, 2019, doi: 10.4018/978-1-7998-1090-2.ch006.
[15] R. Jaganathan and R. Vadivel, "Intelligent Fish Swarm Inspired Protocol (IFSIP) for Dynamic Ideal Routing in Cognitive Radio Ad-Hoc Networks," Int. J. Comput. Digit. Syst., vol. 10, no. 1, pp. 1063–1074, 2021, doi: 10.12785/ijcds/100196.
[16] L. Mani, S. Arumugam, and R. Jaganathan, "Performance Enhancement of Wireless Sensor Network Using Feisty Particle Swarm Optimization Protocol," ACM Int. Conf. Proceeding Ser., pp. 1–5, Dec. 2022, doi: 10.1145/3590837.3590907.
[17] C. Li, J. Wu, Z. Zhang, and A. Lv, "Energy-harvesting Q-learning secure routing algorithm with authenticated-encryption for WSN," ICT Express, 2023, doi: 10.1016/j.icte.2023.05.006.
[18] K. Biswas, V. Muthukkumarasamy, M. J. M. Chowdhury, X. W. Wu, and K. Singh, "A multipath routing protocol for secure energy efficient communication in Wireless Sensor Networks," Comput. Networks, vol. 232, p. 109842, 2023, doi: 10.1016/j.comnet.2023.109842.
[19] V. Ramalingam, R. Saminathan, and K. M. Baalamurugan, "Fork-Hook encryption policy based secured Data Centric Routing Gateway for proactive trust ware data transmission in WBSN," Meas. Sensors, vol. 27, p. 100760, 2023, doi: 10.1016/j.measen.2023.100760.
[20] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," in Mobile Computing, T. Imielinski and H. F. Korth, Eds. Boston, MA: Springer US, 2007, pp. 153–181. doi: 10.1007/978-0-585-29603-6_5.
[21] H. Li, D. Shi, W. Wang, D. Liao, T. R. Gadekallu, and K. Yu, "Secure routing for LEO satellite network survivability," Comput. Networks, vol. 211, p. 109011, 2022, doi: 10.1016/j.comnet.2022.109011.
[22] R. Zhu, A. Boukerche, L. Feng, and Q. Yang, "A trust management-based secure routing protocol with AUV-aided path repairing for Underwater Acoustic Sensor Networks," Ad Hoc Networks, vol. 149, p. 103212, 2023, doi: 10.1016/j.adhoc.2023.103212.
[23] B. Su and B. Zhu, "TBMOR: A lightweight trust-based model for secure routing of opportunistic networks," Egypt. Informatics J., vol. 24, no. 2, pp. 205–214, 2023, doi: 10.1016/j.eij.2023.02.002.
[24] A. S. Oliver, B. Ravi, R. Manikandan, A. Sharma, and B. G. Kim, "Heuristic green computing based energy management with security enhancement using hybrid greedy secure optimal routing protocol," Energy Reports, vol. 9, pp. 2494–2505, 2023, doi: 10.1016/j.egyr.2023.01.052.
[25] A. Roshini and K. V.D. Kiran, "Hierarchical energy efficient secure routing protocol for optimal route selection in wireless body area networks," Int. J. Intell. Networks, vol. 4, pp. 19–28, 2023, doi: 10.1016/j.ijin.2022.11.006.
[26] Z. Yang, L. Li, F. Gu, X. Ling, and M. Hajiee, "TADR-EAODV: A trust-aware dynamic routing algorithm based on extended AODV protocol for secure communications in wireless sensor networks," Internet of Things (Netherlands), vol. 20, 2022, doi: 10.1016/j.iot.2022.100627.
[27] X. Y. Li, W. Lin, W. Guo, and J. M. Chang, "A secure data transmission scheme based on multi-protection routing in datacenter networks," J. Parallel Distrib. Comput., vol. 167, pp. 222–231, 2022, doi: 10.1016/j.jpdc.2022.05.010.
[28] A. A. Mahamune and M. M. Chandane, "Trust-based co-operative routing for secure communication in mobile ad hoc networks," Digit. Commun. Networks, 2023, doi: 10.1016/j.dcan.2023.01.005.
[29] G. V. Gurram, N. C. Shariff, and R. L. Biradar, "A Secure Energy Aware Meta-Heuristic Routing Protocol (SEAMHR) for sustainable IoT-Wireless Sensor Network (WSN)," Theor. Comput. Sci., vol. 930, pp. 63–76, Sep. 2022, doi: 10.1016/j.tcs.2022.07.011.
[30] J. Ramkumar and R. Vadivel, "Multi-Adaptive Routing Protocol for Internet of Things based Ad-hoc Networks," Wirel. Pers. Commun., vol. 120, no. 2, pp. 887–909, Apr. 2021, doi: 10.1007/s11277-021-08495-z.

Authors

Dr. J. Ramkumar working as Assistant Professor in the Department of Information Technology & Cognitive Systems at Sri Krishna Arts and Science College, Coimbatore, Tamilnadu, India. He obtained his PhD degree from Bharathiar University. He has published more than 38 research papers in International Journals and Conferences which includes SCOPUS and SCIE publications. His area of interest includes ad-hoc networks, route optimization, decision support systems and Internet of Things. He acted as Technical Committee Member, Scientific Committee Member, Advisory Board Member and Reviewer in more than 413 International Conferences and 42 Refereed Journals.

Dr. K. S. Jeen Marseline, M.C.A, M.Phil, Ph.D., Dean Computer Science and Mathematics, Sri Krishna Arts and Science College has been a persistent and incessant in her teaching portfolio. She has 25 years of consistent teaching experience in Computer Science stream. As an eminent subject expert, she has vibrantly published more than 45 research papers in both national and international conferences. She has strongly endeavored with her authoritative excellence by being the Controller of examinations in Sri Krishna Arts and Science College in the year 2014 as well as in Sri Krishna College of Engineering and Technology from 2015-

RESEARCH ARTICLE

2017. She has been strategically in heading the Department of Information and Computer Technology from 2009 to till date that has been genuinely accomplished. She was bestowed with the best faculty Award from the renowned CTS corporate company. Her inspiration to achieve innovative modifications in her subject expertise has made her to visit countries like UAE, Thailand and Malaysia and update with educational upliftment. She has been a distinguished Board of Studies committee member in several colleges.

D R Medhunhashini is a passionate educator with a deep-seated curiosity in the Department of Department of Information Technology and Cognitive Systems, Sri Krishna Arts and Science College. With 13 years of experience she has been dedicated to explore Software Engineering field through research, writing, and practical applications. She has contributed to various publications and conferences, aiming to bridge the gap between academic insights and real-world implications. She continues to engage herself in fostering a commitment to advancing knowledge and sharing valuable insights with the Software Engineering community.

How to cite this article:

J Ramkumar, K S Jeen Marseline, D R Medhunhashini, "Relentless Firefly Optimization-Based Routing Protocol (RFORP) for Securing Fintech Data in IoT-Based Ad-Hoc Networks", International Journal of Computer Networks and Applications (IJCNA), 10(4), PP: 668-687, 2023, DOI: 10.22247/ijcna/2023/223319.