






Please cite the Published Version

Aguru, Aswani Devi, Pandey, Amrit, Erukala, Suresh Babu , Bashir, Ali Kashif , Zhu, Yaodong , Kaluri, Rajesh  and Gadekallu, Thippa Reddy  (2024) Reliable-RPL: A Reliability-Aware RPL Protocol Using Trust-Based Blockchain System for Internet of Things. IEEE Transactions on Reliability. ISSN 0018-9529

DOI: <https://doi.org/10.1109/tr.2024.3508652>

Publisher: Institute of Electrical and Electronics Engineers (IEEE)

Version: Accepted Version

Downloaded from: <https://e-space.mmu.ac.uk/637712/>

Usage rights:  [Creative Commons: Attribution 4.0](https://creativecommons.org/licenses/by/4.0/)

Additional Information: This is an author-produced version of the published paper. Uploaded in accordance with the University's Research Publications Policy

Enquiries:

If you have questions about this document, contact openresearch@mmu.ac.uk. Please include the URL of the record in e-space. If you believe that your, or a third party's rights have been compromised through this document please see our Take Down policy (available from <https://www.mmu.ac.uk/library/using-the-library/policies-and-guidelines>)

Reliable-RPL: A Reliability-Aware RPL Protocol Using Trust-Based Blockchain System for Internet of Things

Aswani Devi Aguru, Amrit Pandey, Suresh Babu Erukala, *Senior Member, IEEE*,
 Ali Kashif Bashir, *Senior Member, IEEE*, Yaodong Zhu, Rajesh Kaluri,
 and Thippa Reddy Gadekallu, *Senior Member, IEEE*

Abstract—Routing protocol for low-power and lossy network (RPL) is a routing protocol for resource-constrained Internet of Things (IoT) network devices. RPL has become a widely adopted protocol for routing in low-powered device networks. However, it lacks essential security features, including end-to-end security, robust authentication, and intrusion detection capabilities. Blockchain is a decentralized and immutable digital ledger that records transactions across multiple computers. It provides privacy, transparency, security, and trust. In this work, we proposed a blockchain-based reliable RPL protocol called reliable-RPL, which uses node reliability, link reliability, and relative trust scores of RPL-enabled IoT devices. The parent selection and network topology formulation are based on the proposed reliability-aware objective function. A lightweight ECC-based scheme performs registration, identification, and authentication of RPL-enabled IoT devices. The consistent topological updates from these authenticated IoT devices are used to secure routing paths in RPL-enabled networks. Using a modified trickle algorithm, we employed a reputation-based trust system that monitors and labels malicious nodes based on their reliable activities. The novelty of the proposed framework relies on integrating Contiki-NG (as fronted for IoT network simulation) and Hyperledger Fabric (as a backend for blockchain-based device authentication and trust-based attack resilience regarding rank, replay, sinkhole, and route poisoning attacks). The experimental evaluation of reliable-RPL has demonstrated its

effectiveness compared to state-of-the-art methods regarding significant performance metrics, including packet loss, routing overhead, and throughput on Hyperledger Caliper.

Index Terms—Blockchain, Internet of Things (IoT), routing protocol for low-power and lossy network (RPL), reliability, reputation, trust.

I. INTRODUCTION

THE routing protocol for low-power and lossy network (RPL) is a standardized routing protocol developed by the IETF for low-power and lossy networks (LLNs), such as Internet of Things (IoT) networks and wireless sensor networks (WSNs). It uses an IPv6-based directed acyclic graph (DAG) to represent network topology. Its features include multihop routing, energy efficiency, and robustness to network changes and failures. RPL supports unicast, multicast, and anycast traffic patterns. It is widely used in IoT and WSN deployments for its suitability for LLNs. RPL scales to large networks prioritizes energy efficiency, and ensures robustness in adverse conditions. It supports multihop routing and offers flexibility with different metrics for network optimization. Security mechanisms, including authentication and encryption, protect against attacks. RPL is ideal for various applications, including industrial automation, smart homes, and WSNs. Its key features address the challenges of LLNs, characterized by resource-constrained devices, low bandwidth, limited processing power, memory, and high packet loss.

Security in RPL communication is an active area of research. The literature contains a variety of works aimed toward securing RPL communication, which includes approaches based on acknowledgment, trust, location, mathematics, and specifications. Many of these efforts concern a few parameters and processes that fail to give a flawless solution to secure routing and are vulnerable to fundamental routing issues. Furthermore, many existing techniques have an inherent issue of being resource-demanding, which is counter-intuitive for resource-constrained IoT devices. For instance, Dvir et al. [1] employed Version and Rank authentication to prevent Ranks and Version attacks; nonetheless, it must address the solution's computationally expensive aspect. Conti et al. [2] presented software remote attestation solutions to prevent practically every software attack. However, the centralized form creates concerns, such as a single

Aswani Devi Aguru is with the Department of Computer Science and Engineering, SRM University, AP 522240, India (e-mail: aa720086@student.nitw.ac.in).

Amrit Pandey is with the SAP Labs India, Bangalore 560048, India (e-mail: mail.amritpandey@gmail.com).

Suresh Babu Erukala is with the National Institute of Technology Warangal, Warangal 506004, India (e-mail: esbabu@nitw.ac.in).

Ali Kashif Bashir is with the Department of Computing and Mathematics, Manchester Metropolitan University, M15 6BH Manchester, U.K., and also with the Centre for Research Impact and Outcome, Chitkara University Institute of Engineering and Technology, Chitkara University, Rajpura 140401, India (e-mail: Dr.alikashif.b@ieee.org).

Yaodong Zhu is with the Jiaying University School of Information Science and Engineering, Jiaying 314001, China (e-mail: zhuyaodong@163.com).

Rajesh Kaluri is with the School of Computer Science Engineering and Information Systems, Vellore Institute of Technology, Vellore 632014, India (e-mail: rajesh.kaluri@vit.ac.in).

Thippa Reddy Gadekallu is with the Division of Research and Development, Lovely Professional University, Phagwara 144411, India, and also with the Centre for Research Impact and Outcome, Chitkara University Institute of Engineering and Technology, Chitkara University, Rajpura 140401, India (e-mail: thippareddy@ieee.org).

point of failure. Although Airehrour et al. [3] provided an outstanding trust-based way to analyze routing paths in topology, it lacks a backup and recuperation mechanism to handle power loss and node greediness.

The significant challenge commonly identified while realizing secure RPL (SRPL) communication in IoT networks is the lack of reliability. This issue results in increased memory consumption due to continuous key exchange [4], energy depletion [5], decreased packet delivery ratio, DAO update rate during authentication [4], lower backup and recuperation of trust values [3], lower scalability [2], hardware dependency [6], failure against attacks [7], [8], inefficient routing paths [9] while parent selection and resource hungry schemes for attack detection using machine learning or deep learning. Addressing these research gaps is the principal motivation of our work.

In addition to the reliability, *authentication* is also a critical factor that ensures the trustworthiness of the IoT networks by limiting the involvement of untrustworthy and adversarial nodes in the network. The existing reliability-aware RPL protocol implementations by Haque et al. [10], Lahbib et al. [11], Nobakht et al. [12], Shahbakhsh et al. [13], and Shahbakhsh et al. [14] do not address the authenticity and trust scores of IoT devices while designing the objective functions (OFs). RPL's implicit authenticated mode performs the job but requires heavy computation and occasionally depends on per-installed keys. RPL needs a key management strategy for secure communication. Furthermore, the key management, distribution, and storage need additional memory resources. Due to the lack of confidentiality and integrity-preserving safeguards, nodes are vulnerable to eavesdropping, tampering, and other attacks on control messages. At least one of the techniques previously could make secure routing a reality; however, their absence creates a barrier to resolving security issues [15]. The foremost problems that this article addresses are as follows.

- 1) Lack of authenticity and trustworthiness of existing reliability-aware RPL protocols.
- 2) Routing overhead of RPL protocol in IoT networks.
- 3) Single point of failure of existing key management techniques.
- 4) Lack of backup and recuperation.

We address these research problems by designing a blockchain-enabled SRPL mechanism that mitigates potential cyberattacks by offering a lightweight, trust-based secure communication that is energy efficient, decentralized, and scalable to massive networks of IoT devices to secure routing paths.

This article proposes a reliability-aware RPL protocol for IoT networks using a trust-based blockchain system. Combined with the trust-based reputation model, this system effectively identifies malicious adversaries in the RPL-enabled IoT network. The blockchain network is designed on the fog layer and runs on the low-powered border routers (LBRs). The node registration on the blockchain network will eliminate various security concerns of the RPL-enabled IoT environment, including identification, authentication, scalability, and attack resilience. In addition, the proposed scheme also acts as a solid backbone for backup and recuperation of the LBR nodes. This feature shows the novelty of our framework to the existing SRPL protocol designs.

The *significant contributions* of the proposed framework are summarized as follows.

- 1) We design a blockchain-enabled reliability-aware RPL protocol that ensures node reliability through congestion control and energy efficiency and link reliability through optimal packet delivery ratio and signal strength in IoT networks.
- 2) We propose a trust-enabled trickle algorithm that detects inconsistencies in RPL-based data transmission using multiple trust score evaluations and reliability-aware objective function (ROF) for achieving attack resilience.
- 3) We develop chaincodes for device registration, ECC-based lightweight authentication, peer creation, and secure path updating on the Hyperledger Fabric test network.
- 4) We present a novel RPL protocol stack and blockchain network architecture integration for realizing SRPL routing paths in IoT networks. This amalgamation is implemented by integrating Contiki NG as the frontend for node simulation and Hyperledger Fabric as a backend for running the blockchain network.
- 5) We perform an informal security analysis of the proposed framework regarding attack resilience against various attacks. We analyze the performance of the proposed framework using the Cooja simulator, Wireshark, and Hyperledger Caliper regarding packet loss during attacks, routing overhead, power consumption, packet delivery ratio, and blockchain throughput which exhibit optimal results than state-of-the-art.

II. RELATED WORK

This section presents the related work on RPL security frameworks existing in the literature. We have classified these mechanisms into four types based on the approach [16] viz. *Cryptography and mathematics-based methods* ([17], [9], [18]), *trust-based methods* ([3]), *specification-based methods* ([19], [2], [5], [6]), and *blockchain-based methods* ([20], [8]). The detailed insights into each work are presented as follows.

Mathematical- and cryptography-based security for RPL involves the application of mathematical algorithms and cryptographic techniques to ensure the integrity, confidentiality, and authenticity of data transmitted within the network. An inherent security mechanism in RPL is proposed by Raouf et al. [17], which is much more effective for external attacks. However, secure mode RPL uses significantly more energy with a drop-in PDR to 70% in case of internal attacks. An algorithm for secure parent selection based on a rank threshold value is presented by Iuchi et al. [9]. However, it results in an increased number of hops from each node to the destination-oriented directed acyclic graph (DODAG) root. An SRPL routing protocol incorporating rank threshold values and validation methods using the hash-chain technique is proposed by Glissa et al. [18]. However, this scheme results in an additional overhead due to excessive hashing algorithms.

Specification-based security for RPL involves enforcing security requirements and constraints based on the protocol

specifications. A mechanism to prevent insider DAO attacks, SecRPL, is presented by Ghalcb et al. [19]. However, this mechanism negatively affects the downward packet delivery ratio if the threshold is set to a low value, as the nodes will miss some critical DAO messages to build downward routing paths. A remote attestation method in RPL protocol to prevent software attacks is presented by Conti et al. [2]. However, this scheme only considers software-only attacks in an IoT environment and assumes hardware specifications. A trust anchor interconnection loop, called TRAIL, is presented by Perrey et al. [5] to validate the upward path of DODAG. However, it results in enhanced memory consumption. A trusted platform module (TPM) for storing digital keys, certificates, and passwords is presented by Seeber et al. [6] to safeguard against node tampering. This scheme is a hardware-based solution that makes it loosely coupled to the topology, in which physical access to a node can be gained and compromised.

Trust-based methods aim to enhance the security and reliability of communications within these networks by establishing and maintaining trust relationships between networks based on the node's activities. Airehrour et al. [21] proposed a simple trust calculation strategy to label malicious nodes in the network. This method fails in case of many imminent attacks. Thulasiraman and Wang [22] proposed a lightweight trust metric system for mobile-IoT networks performing routing over RPL. This system leads to heavy network overhead in case of increased mobility. Airehrour et al. [3] proposed a trust-based mechanism to evaluate and compute the trustworthy behavior of nodes in the network. This metric then facilitates a node to elect a neighboring node for routing; however, it must consider scalability and lack of backup and recuperation strategy for root nodes. A sinkhole detection scheme in RPL-based IoT networks, called SoS-RPL, is presented by Zaminkar et al. [23] which consists of two phases for ranking of RPL nodes and blocking the malicious nodes, respectively. The simulation results on NS3 have proven the efficiency of the protocol against sinkhole attack detection. A novel reliability-aware RPL protocol, called "reliability-aware adaptive RPL routing protocol" is proposed by Shahbakhsh et al. [13] for IoT networks. This protocol achieves reliability by evaluating various metrics regarding parent selection and stable path identification. The simulation of this protocol on Cooja ensures improved reliability on data exchange and reduced instability in RPL network topology. A hybrid security framework-enabled RPL protocol, called DSH-RPL, is proposed by Zaminkar et al. [24] for secure communication in IoT networks. Improvement in RPL reliability, detection of sinkhole attacks, blacklisting the malicious nodes, and encrypted data transmission are the four phases of this protocol. The simulation results have shown that the evaluation of DSH-RPL exhibits optimal performance in terms of various security parameters.

A *blockchain and ML-based RPL routing framework* is proposed by Sahay et al. [20]. However, this strategy may not be appropriate for systems that demand immediate responses to validate the generated IoT data. Ramezan et al. [8] proposed a novel routing protocol called BCR for routing in IoT networks. If the neighbors want to participate in the routing process, they can offer routes via them by paying *Route Offer Bond* in the smart contract address. In case of a malicious node offering the wrong

route, the node's address is added to the *Blacklisted_address* of the device. The BCR protocol, as proposed, eliminates the need for a central authority to authorize, add, or remove IoT devices. Unlike traditional centralized routing protocols, it does not rely on a secret key-sharing mechanism.

In addition, we carried out an extensive analysis of the optimal characteristics of the RPL protocol based on the survey presented by Shirvani et al. [25]. This survey is based on various trust-based routing schemes in RPL-based IoT networks to achieve blacklisting traffic from malicious devices, link reliability, enhanced congestion control, lightweight authentication, and improved scalability. This survey played a prominent role in identifying existing research problems and designing a novel RPL protocol with reliability, secure data transmission, scalability, trust, and decentralization. After performing a rigorous analysis of state-of-the-art techniques on RPL security frameworks (see Table I), we have proposed a trust-based blockchain network for SRPL routing while defending the rank attack, reply attack, and sinkhole attacks.

III. PRELIMINARIES

This section provides the preliminaries of the proposed framework.

A. RPL: Routing Protocol for Low Powered and Lossy Networks

RPL is a routing protocol for resource-constrained devices in LLNs, such as IoT deployments. It establishes and maintains routes between nodes to enable efficient communication within the network. The primary goal of RPL is to provide energy-efficient and reliable routing while accommodating the unique characteristics of low-power networks, including limited bandwidth, high loss rates, and constrained computational resources. RPL forms a DODAG to organize the network topology. The protocol utilizes a proactive approach, where nodes periodically exchange control messages called DODAG information object (DIO) to disseminate routing information throughout the network. RPL employs a rank-based mechanism to construct routes, where each node maintains a rank indicating its position in the DODAG. Nodes with higher ranks become parents to lower ranked nodes, forming upward routes toward the root of the DODAG. RPL also incorporates a trickle timer algorithm to regulate control message transmissions, minimizing overhead, and conserving energy. It also supports route optimization and repair mechanisms to adapt to network changes dynamically. Overall, RPL addresses the unique challenges of LLNs by providing efficient routing, scalability, and adaptability for diverse IoT applications. However, as IoT networks are resource-constrained, formulating reliable and energy-efficient RPL routing paths and trust-based parent selection are the major limitations in existing RPL protocol implementations.

B. Trickle Algorithm

The trickle algorithm is a key component of the RPL, designed for resource-constrained devices in IoT networks. Its purpose is to regulate the frequency of control messages exchanged

TABLE I
SUMMARY OF LITERATURE REVIEW

Ref.	Mechanism	Limitations	Rank attack	Version attack	Reply attack	Sinkhole attack
[17]	The inherent security mechanism of RPL, i.e., secure and authenticated modes that use preinstalled keys for secure control message transmission.	Decrease in PDR by 70%, increasing memory and power consumption due to cryptographic operations.	✓	✓	✓	✓
[9]	Parent selection with a higher rank than the threshold.	Leads to the path with a higher hop count by ignoring legitimate parents.	✓	✗	✗	✗
[18]	SRPL: Rank threshold value and route validation using hash-chain.	Act against all nodes during initialization of hash-chain, then increase computational overhead.	✗	✗	✓	✗
[19]	SecRPL: Limiting the number of DAO messages sent by the child to the parent.	Can lead to a poor downward packet delivery ratio, as important DAO updates can be missed.	✓	✓	✗	✓
[2]	SPLIT: Remote attestation of nodes by using root node as verifier.	Assume a lot of hardware specification and effective only against software-only attacks.	✓	✗	✓	✓
[5]	TRAIL: Using round trip messages from leaf to root to validate the upward path.	The scalable proposed solution of validating multiple nodes at a time increases memory overhead.	✓	✗	✓	✓
[6]	Embedding a separate TPM module in each node responsible for checking the software integrity of nodes.	Internal nodes can be compromised, which is not addressed in the mechanism; the TPM module is loosely coupled to the software.	✓	✓	✓	✓
[3]	SecTrust-RPL: Trust awareness of neighboring nodes to elect routing paths, and the trust decreases on each false or erroneous operation.	Decreases the trust value of legitimate nodes in case of resource depletion, as nodes might drop packets to conserve energy.	✓	✗	✓	✓
[20]	Using blockchain as a medium to share IoT datasets and using ML to detect routing attacks.	High block time is unsuitable for delay-intolerant systems.	✗	✗	✗	✗
[8]	Nodes use smart contracts to retrieve routing information from neighboring nodes.	No effective mechanism to validate routing information.	✓	✓	✓	✓
[23]	SoS-RPL: Attack resilience against sinkhole attacks using node rating and ranking.	Reliability aspects of RPL protocol are not addressed.	✗	✗	✗	✓
[13]	RAARPL: Reliability-aware parent selection with improved stability and reduced error rate.	Attack resilience is not addressed.	✗	✗	✗	✗
[24]	DSH-RPL: Encrypted data transmission through reliable RPL protocol against sinkhole attacks.	Inconsistencies and trust in data transmission are not addressed.	✗	✗	✗	✓

between neighboring nodes to ensure network stability while minimizing overhead. The algorithm operates based on “trickling” information throughout the network. Each node maintains a trickle timer that controls the transmission of control messages. The working of the trickle algorithm is presented in Algorithm 1. The trickle algorithm effectively balances the frequency of control message transmissions, reducing unnecessary overhead in the network while still ensuring that essential control information is propagated promptly. In our proposed mechanism, we have modified the trickle timer to multicast DIO messages containing the reputation values of the neighbor nodes. Formation components and variables in the trickle algorithm are minimum interval size (I_{\min}), maximum interval size (I_{\max}), redundancy constant (k), and current interval size (I), counter (c), random time (t), respectively.

C. Blockchain

Blockchain is a decentralized data storage and transfer system using nodes instead of a central authority [26], [27]. It enables secure and unalterable transaction records, benefiting

cryptocurrencies, smart contracts, supply chains, and digital identity. With cryptography and consensus, blockchain enhances industry transparency, efficiency, and trust [28], [29]. The consortium blockchain is a controlled network operated by multiple entities. Organizations form a consortium, collaborating to manage rules, validate transactions, and maintain the ledger [30]. This type offers strong privacy and security due to restricted network access [31].

In our proposed system, we have developed a consortium blockchain network of LBR nodes acting as organizations. These organizations have multiple peers that act as IoT nodes. The flow of information in the ledger is restricted to those LBR nodes that share the connection to the common channel and is closed to other participants. To become part of this channel, any LBR node has to register itself as an organization and get approval from the existing organizations.

1) *Consensus Mechanism*: The proposed architecture uses a consortium blockchain network that uses *Raft* consensus [32] mechanism for publishing blocks. Raft is often considered a lightweight consensus algorithm, which makes it suitable for running on LBR devices. Raft operates by electing a leader

Algorithm 1: Mechanism of Trickle Algorithm.

Require: Formation Components and variables
Ensure: Trickle activities

- 1: Set $c \leftarrow 0$; $I \leftarrow \{I_{\{min\}}, I_{\{min\}} \times 2^{I_{\{max\}}}\}$; $t \leftarrow [\frac{I}{2}, I)$, where $I \in \{I_{\{min\}} \times 2^n | n \in \mathbb{N}_0, n \leq I_{\{max\}}\}$;
- 2: **if** $Detect(Identical_{data})$ **then**
- 3: $c = c + 1$
- 4: **end if**
- 5: **while** t **do**
- 6: **if** $c < k$ **then**
- 7: $Allow(\text{Data transmission})$;
- 8: **else**
- 9: $Supress(\text{Data transmission})$;
- 10: **end if**
- 11: **end while**
- 12: **if** $Expiration(I)$ **then**
- 13: **while** $I < I_{\{max\}}$ **do**
- 14: $I = 2 \times I$;
- 15: **end while**
- 16: **end if**
- 17: **if** $Detect(Inconsistent_{data}) \ \&\& \ I > I_{\{min\}}$ **then**
- 18: $I = I_{\{min\}}$
- 19: **else**
- 20: $Return(\text{Trickle remains idle})$;
- 21: **end if**
- 22: $c \leftarrow 0$; $t \leftarrow [\frac{I}{2}, I)$;

node, which coordinates the other nodes in the system. The leader node receives client requests, updates the system's state, and replicates the updates to other nodes. If the leader fails, a new leader is elected through a leader election process. One of the ways that Raft achieves its lightweight design is through its use of leader election. In Raft, a single leader is responsible for coordinating the other nodes in the system, simplifying the replication process, and reducing the counts of transmissions needed to be exchanged.

D. Elliptic Curve Cryptography (ECC)

Elliptic curve cryptography is a public-key cryptographic algorithm that uses the algebraic structure of elliptic curves over finite fields to provide security. The algorithm generates a public and private key pair on an elliptic curve. The public key is derived from a point on the curve, and the private key is a random integer. Let E be an elliptic curve defined over a finite field F_p of prime order p , and let P be a point on E . The private key d is a random integer from the interval $[1, n - 1]$, where n is the order of P . The public key Q is the result of scalar multiplication of P by d : $Q = dP$. To encrypt a message, the sender chooses a random integer k from the interval $[1, n - 1]$ and generates a random point R on the curve: $R = kP$. The sender then calculates the encryption key K as follows: $K = Q + R = dP + kP = (d + k)P$. The message is then encrypted using K as the key. To decrypt the message, the recipient uses their private key d to perform scalar multiplication on K : $K' = dK = d(d + k)P = kdP + d^2P = kP + d^2P = R + Q$. The recipient can then recover the original

Algorithm 2: Signature Generation in ECDSA.

Require: Message m , Private key of sender dA
Ensure: Signature Key pair (R, S)

- 1: Calculate $e = Hash(m)$ { $Hash$ is the cryptographic hash function, such as SHA-2}
- 2: Choose a random integer k from $[1, P - 1]$
- 3: Calculate $R = x_1(mod N)$, where $(x_1, y_1) = k \times G$ { (x_1, y_1) is the curve point.}
- 4: **if** $R = 0$ **then**
- 5: Calculate e & Repeat the process.
- 6: **end if**
- 7: $S = k^{-1}(Hash(m) + dA \times R)mod(P)$
- 8: **if** $S = 0$ **then**
- 9: Calculate e & Repeat the process.
- 10: **end if**
- 11: Return (Signature key pair as (R, S))

message by using R to cancel out the encryption performed by the sender.

1) *Elliptic Curve Digital Signature Algorithm (ECDSA):* Digital signatures play a crucial role in blockchain technology, especially in the authentication of transactions. The nodes must provide proof of authorization while submitting a transaction, which is verified by every other node in the network. ECDSA uses ECC to generate digital signatures as key pairs for signing and verification purposes. Due to the many key advantages of ECC over other algorithms in public key cryptography, blockchain applications employ ECDSA for signing transactions and events. ECDSA uses the temporary key pairs to calculate the signature pairs R and S . R is x coordinator of the temporary public key. dA is the sender's private key. m is the message. P is the prime order of the elliptic curve (order of G). Qa is the sender's public key. $Hash()$ is the cryptographic hash function. A random point on the elliptic curve is chosen as the temporary private key k , and the public key is then derived by $P = k \times G$, where G is the elliptic curve base point. The transactions are signed using the signature generation algorithm in Algorithm 2. Later, the signed transactions are verified using the pairs R and S , and the public key is taken from Algorithm 3.

2) *ECC-Based Shared Key Exchange Scheme:* Elliptic curve Diffie–Hellman (ECDH) key exchange is a popular method for securing communication by creating a shared key between two parties. The ECDH key exchange works as follows.

IV. PROPOSED FRAMEWORK: RELIABLE-RPL

This section describes our proposed blockchain-based SRPL protocol, reliable-RPL, including system modeling, cross-layer integration, an SRPL path establishment, ECC-based lightweight authentication, and a trust-based modified trickle algorithm for attack resilience.

A. Assumptions

- 1) Blockchain network is hosted at the fog layer using LBRs. The orderer service is hosted as a server in the cloud layer.

Algorithm 3: Signature Validation in ECDSA.

Require: Private key of sender Qa
Ensure: Validation of signature.
 1: **if** $R, S \in [1, P - 1]$ **then**
 2: Return (“Signature is valid”)
 3: **end if**
 4: Calculate $e = Hash(m)$ {*Hash* is the hash used in signature generation algorithm}
 5: $P = S^{-1} \times Hash(m) \times G + S^{-1} \times R \times Qa$
 6: Calculate $w = S^{-1}(modP)$; $u_1 = e \times w(modP)$; $u_2 = R \times w(modP)$
 7: Calculate $(x_1, y_1) = u_1 \times G + u_2 \times Qa$ {The curve point (x_1, y_1) is the sum of two scalar multiplications.}
 8: **if** $R = x_1(modP)$ **then**
 9: Return (“Signature is valid”)
 10: **end if**

Algorithm 4: Key Generation Algorithm in ECDH.

Require: Public information of numbers n and G , secret numbers a, b .
Ensure: Generation of key pairs.
 1: Private key of A , $A_{pri} = A$; Private key of B , $B_{pri} = b$
 2: **if** $a \in [1, n]$ **then**
 3: Compute the public key for A as $A_{pub} = a \times G$.
 4: Generate the public key for B as $B_{pub} = b \times G$.
 5: **else**
 6: “Failed to generate keys”
 7: **end if**

Algorithm 5: Shared Key Calculation Algorithm in ECDH.

Require: $A_{pub}, A_{pri}, B_{pub}, B_{pri}$
Ensure: Shared key S
 1: For A : Secret Key(S) = $S = A_{pri} \times B_{pub} = a \times bG$.
 2: For B : Secret Key(S) = $S = B_{pri} \times A_{pub} = b \times aG$.

Algorithm 6: Key Exchange Algorithm in ECDH.

Require: Public keys A_{pub}, B_{pub}
Ensure: Key exchange between A and B
 1: Keys at A : A_{pub}, A_{pri} ; Keys at B : B_{pub}, B_{pri}
 2: **if** Flag == 1 **then**
 3: Keys at A : B_{pub} ; Keys at B : A_{pub}, B_{pri}
 4: “Successful execution of key exchange mechanism”
 5: **else**
 6: “Failed to exchange the keys”
 7: **end if**

- 2) LBR devices have sufficient memory and energy to run lightweight operations (ECC and Raft consensus) and peer services of blockchain networks.
- 3) The RPL protocol works in nonstoring mode.

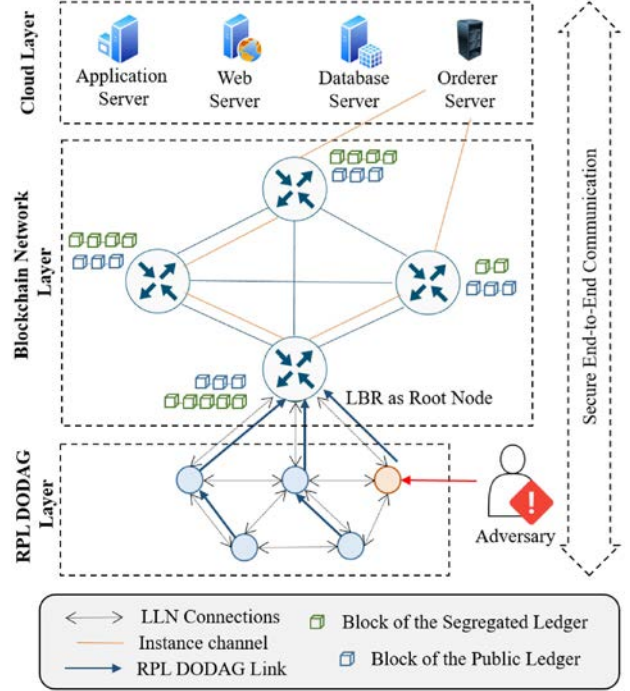


Fig. 1. Architecture of proposed reliable-RPL.

B. System Model

The system model of the proposed reliable-RPL is depicted in Fig. 1. This architecture comprises three layers, namely *RPL DODAG layer*, *blockchain network layer*, and *cloud layer*.

1) *RPL DODAG Layer:* This layer consists of the IoT device network executing routing via RPL protocol over IPv6 suit. It comprises sensors, routers, and root nodes in a typical RPL topology. We have assumed that this layer runs the RPL protocol in nonstoring mode; hence, all the intermediary nodes do not store downward routing paths. Any request for routing goes through the sink (root), which is responsible for redirecting the message to the destination node in the network. The main role of the RPL DODAG layer is to propagate the neighbor nodes' relative and primary trust values to the root node so that the malicious nodes can be detected and a secure route path can be found.

Suppose the total number of packets delivered from node u to node v is $P(t)_d$. The total number of packets sent to node u from node v is $P(t)_s$. The time during the DODAG construction is t . The trust of each node in the RPL network can be calculated based on trusted parent node selection [33]

$$\text{PrimaryTrust} = \frac{P(t)_d}{P(t)_s}. \quad (1)$$

The retribution weight ρ is attached to the malicious node. Upon the consecutive malicious actions, the value of ρ will be incremented by a constant κ . The updation formula for retribution weight is mentioned in (2)

$$\rho_i = \rho_i + \kappa \quad (2)$$

$$\text{Relative Trust} = \frac{P(t)_s}{P(t)_s + \rho[P(t)_d - P(t)_s]} \quad (3)$$

$$\text{Threshold Trust} = \frac{1}{\text{Packet-count-beyond-threshold}}. \quad (4)$$

The parent selection will be based on the value of PrimaryTrust. The optimal parent is found based on the RelativeTrust if this value is equal. After the node joins the DODAG, the attacker node can be identified based on ThresholdTrust. The mutual trust among multiple nodes of RPL networks is formed using MutualTrust. When the trust score is reduced, the neighbor node sends an acknowledgment to the border router, and the information about the attacker node will be broadcast. The packets from the node will be dropped. This model minimizes energy consumption and unnecessary network traffic. Among many possible ways of DODAG construction, the optimal DODAG is formed based on the cost of both the link and nodes of RPL networks. The significant variables that decide the cost are the number of hops h , interference i , reliability r , and error rate e . So, the total cost of a path is determined using an OF as

$$\text{Cost}(\text{route}) = \alpha \times h + \beta \times i + \gamma \times r + \delta \times e. \quad (5)$$

The most remarkable factor among all these variables is the node's reliability r . In IoT networks, r is calculated based on node reliability nr and link reliability lr.

- 1) Evaluation of node reliability (nr): The reliability of an IoT node is evaluated in terms of node congestion and energy efficiency for the current node cn and parent node pn [34]. We consider two factors, namely buffer utilization BU and the remaining energy of the node EG, to analyze the node congestion and energy efficiency of the node, respectively. The node reliability of the current node nr(cn) depends on two factors, congestion and energy factor (cef), and reliability reduction factor (rrf). Equation (6) presents the evaluation of nr(cn) as follows:

$$\text{nr}(\text{cn}) = \max((\text{cef} \times \text{BU} + (1 - \text{cef}) \times \text{EG}), (\text{nr}(\text{pn}) \times \text{rrf})). \quad (6)$$

- 2) Evaluation of link reliability (lr): The reliability of the communication between the IoT nodes is evaluated in terms of expected transmission count (ETX) and radio signal strength indicator (RSSI) [35]. ETX indicates the data packet delivery between IoT nodes, RSSI indicates the received signal strength from pn to cn, MRSS indicates the maximum received signal strength by the cn. Equation (7) presents the evaluation of lr(cn) as follows:

$$\text{lr}(\text{cn}, \text{pn}) = \frac{1}{\text{ETX}(\text{cn}, \text{pn})} \times \frac{\text{RSSI}(\text{cn}, \text{pn})}{\text{MRSS}}. \quad (7)$$

- 3) The overall reliability factor r is $\text{DRI} \times \text{nr}(\text{cn}) + (1 - \text{DRI}) \times \text{lr}(\text{cn}, \text{pn})$, where DRI is the dynamic reliability index of the IoT network.

We define the ROF interims of r and RelativeTrust among cn and pn

$$\text{ROF}(\text{cn}) = \frac{P(t)_{\text{pn}}}{P(t)_{\text{pn}} + \rho[P(t)_{\text{cn}} - P(t)_{\text{pn}}]} \times (\gamma \times r). \quad (8)$$

TABLE II
CONTROL MESSAGES IN DODAG CONSTRUCTION

Control Message	Responsibility
DAG Information Object (DIO)	Contains information on RPL instances, configuration attributes, and sets of parents.
DAG Information Solicitation (DIS)	A node asks for DIO from an RPL node to join the network.
Destination Advertisement Object (DAO)	transmit upward traffic from the destination node to the root node.
DAO Acknowledgement (DAO-ACK)	Downward acknowledgment from the root node to the destination node after DAO is received.

The path with minimal cost and highest reliability is selected for the transmission. In DODAG, every node's rank is calculated by the ROF based on its level from the root node. The lower value of ROF indicates a higher probability of the node being selected as the parent. The construction and maintenance of DODAG are carried out using control messages described in Table II.

Based on the proposed ROF, the trickle algorithm reduces the overhead caused by control messages during DODAG construction and maintenance. It includes two steps as follows.

- 1) Transmission suppression: A node suppresses its transmission if a sufficient number of messages are captured in its range.
- 2) Resolving inconsistencies in DAG: If any change in the network is detected, the trickle increases the transmission of control messages.
 - 2) *Blockchain Network Layer*: The blockchain network is deployed over the fog network of LBRs. The LBRs run as independent peers on our consortium blockchain network. The chaincode or smart contract is deployed over the network. All the organizations (LBR) are running over one channel, sharing one public ledger and one segregated one. As shown in the system model, the orange line shows the connection of all the organizations with the orderer in the cloud layer.

Fig. 2 shows the registration, authentication, and registration mechanism of a new LBR device wanting to join the Hyperledger fabric network. The notations used in the sequence diagram are TS: Time stamp, PubK: Public key, and PriK: Private key. As shown in the sequence diagram, the fog device first sends a registration request to the blockchain network already running on the network of fog devices. These fog devices comprise all the LBR in the network, including the orderer hosted in the cloud. After the authentication mechanism is complete, the adding peer invokes the DeviceChaincode to add this node to the device ledger. Cryptomaterial, along with the device ID assigned by the blockchain network, is sent back to the new LBR device. Finally, the LBR device submits a request to join the instance channel and operates as a new DODAG root.

- 3) *Cloud Layer*: All the communication regarding device registration to sharing changes in the routing information happens via the external network supported by the cloud layer. The orderer organization responsible for running the consensus mechanism is also hosted on the cloud layer as an independent server. The cloud layer enables a seamless connection between all the organizations running different networks and varying protocol suits.

Algorithm 7: Proposed Trust-Enabled Trickle Algorithm.

Require: Formation Components and variables

Ensure: Trust-enabled Trickle activities

```

1: Set  $c \leftarrow 0$ ;  $I \leftarrow \{I_{min}, I_{min} \times 2^{I_{max}}\}$ ;  $t \leftarrow [\frac{I}{2}, I)$ ,
   where  $I \in \{I_{min} \times 2^n | n \in \mathbb{N}_0, n \leq I_{max}\}$ ;
2: if  $Detect(Identical_{data})$  then
3:    $c = c + 1$ 
4: end if
5: while  $t$  do
6:   if  $c < k$  then
7:      $Allow(Data\ transmission)$ ;
8:   else
9:      $Supress(Data\ transmission)$ ;
10:  end if
11: end while
12: if  $Expiration(I)$  then
13:   while  $I < I_{max}$  do
14:      $I = 2 \times I$ ;
15:   end while
16: end if
17: while  $Detection-of-inconsistent-data$  do
18:    $u, v \leftarrow$  Nodes in the RPL network with same rank.
19:   Calculate reputation scores;
20:    $Return(PrimaryTrust = \frac{P(t)_d}{P(t)_s})$ 
21:   Include a retribution weight  $\rho$ ;
22:    $Return(RelativeTrust = \frac{P(t)_s}{P(t)_s + \rho[P(t)_d - P(t)_s]})$ 
23:   Reliability-aware objective function  $ROF =$ 
      $RelativeTrust \times \gamma \times r$ 
24:    $Return(ThresholdTrust =$ 
      $\frac{1}{Packet-count-beyond-threshold})$ 
25:   The lower reputation scores indicate higher
     inconsistencies in data
26: end while
27: if  $Detect(Inconsistent_{data}) \ \&\& \ I > I_{min} \ \&\&$ 
      $!ROF$  then
28:    $I = I_{min}$ 
29: else
30:    $Return(Trickle\ remains\ idle)$ ;
31: end if
32:  $c \leftarrow 0$ ;  $t \leftarrow [\frac{I}{2}, I)$ ;

```

C. Methodology

The methodology followed to establish secure route paths in the IoT environment is explained as follows.

Step-1: The LBR device first submits a registration request to the existing blockchain network hosted by the fog devices.

Step-2: The leader peer in the organization accepts the request and begins the authentication mechanism via the ECDSA algorithm. Upon successful authentication of the device, the private and public key for the new device is generated using an ECC-based key generation algorithm.

Step-3: The leader peer then invokes the *DeviceChaincode* by creating a transaction and submitting it to the orderer. Along with this, MSP certificates are generated by the organization's

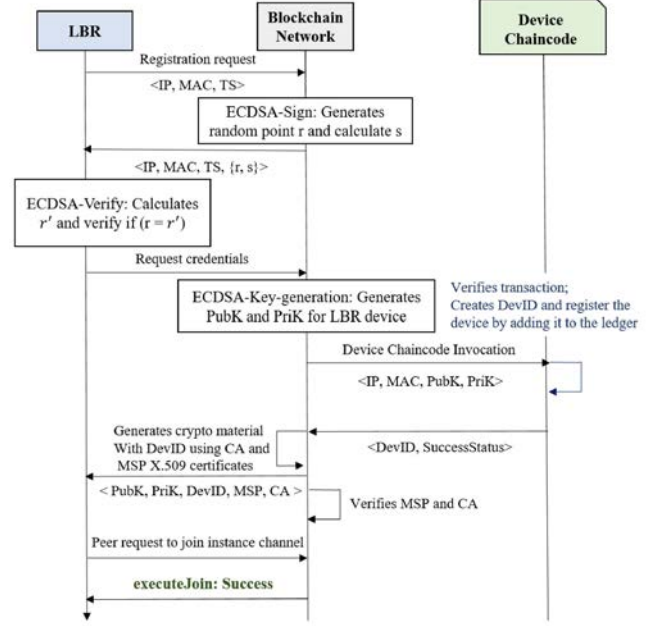


Fig. 2. Sequence diagram of LBR device registration, authentication, and peer update.

CA. DeviceChaincode then provisions a new device ID *DevID* for the peer.

Step-4: The LBR device is then sent back the cryptomaterials and DevID as a response by the blockchain network. On receiving the material, the LBR device sends a *executeJoin* request to join with the *instanceChannel* of the organization. Upon successful join request, the new LBR device now runs as an independent peer in the network with the shared public ledger and participates in the consensus algorithm.

Step-5: An IoT device registers itself to the LBR by sending the registration request to the nearby nodes.

Step-6: The LBR then completes the identification and authentication of the IoT device via ECDH and generates a shared key.

Step-7: LBR then registers this device on the ledger by invoking *DeviceChaincode*. The transaction is submitted and then run on the network for consensus. This chaincode effectively identifies edge and fog devices in the network.

Step-8: As a node, the IoT device acts like an adversary and disrupts routing paths. According to the evaluation of the (ROF), the node's probability of being selected as a parent is high if the value is low. In addition, the modified trickle algorithm then captures the decrease in the reputation trust value of the IoT device via a primary and relative trust (as presented in Algorithm 7). The request for a node with a low reputation value is submitted to the LBR.

Step-9: LBR, on receiving updates about the decreased reputation of a device, alters the routing paths in its routing table and publishes it as a transaction via *RoutingChaincode* to the blockchain network. In this way, the downward routing paths in the network are secured by sharing the secure routes over the blockchain network.

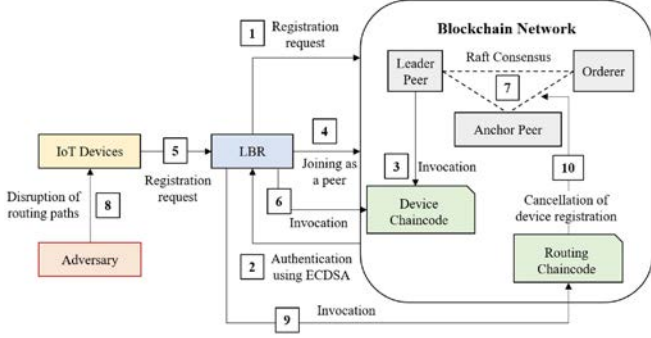


Fig. 3. Overall methodology for establishing secure route paths.

Step-10: Due to decreased reputation value, the malicious node is omitted while calculating a more secure route by individual LBRs. The route paths are also decentralized with the rest of the LBR; hence, any inconsistency detected by one LBR is propagated to the entire IoT network.

The overall methodology is based on the cross-layer integration of RPL-enabled IoT nodes and blockchain network. We proposed a novel integrated architecture by amalgamating the internal layers of RPL protocol and Hyperledger Fabric network (as presented in the following Section IV-C1).

1) *Cross-Layer Integration:* The reliable-RPL system involves orchestrating multiple software technologies, including a cross-layer integration of RPL protocol layers and blockchain network.

We have employed the Contiki-NG operating system that comes with the in-built implementation of the RPL-Lite protocol. Hyperledger Fabric is used to create a blockchain network hosted on the fog devices of any IoT network. In Fig. 4, we have shown the integration of Contiki-NG OS with the Hyperledger Fabric system in a layered manner.

Perception layer: In the bottom-most layer, the sensors and actuator nodes collect the data and send it upwards to the DODAG root. This layer is also called *perception layer* and comprises various network technology, which is the backbone of the communication in IoT.

Network layer: The next layer comprises our Contiki-NG operating system, which utilizes the various Contiki modules in order to achieve tasks such as registration, authentication, and identification of edge devices. This layer runs the RPL routing protocol on its top. The Contiki-NG operating system is modified to share the trust value and label the reputation values of the neighbor nodes. This layer also communicates to the blockchain network layer.

Blockchain network layer: Since the blockchain layer is also part of the LBR device (fog layer), the fog devices are responsible for registering and authenticating LBR devices looking to join the network. The blockchain layer provides services, such as LBR device registration and authentication, IoT device identification, registration, and authentication, and the consistent route path update over the entire network in a decentralized manner.

Cloud layer: The purpose of the cloud layer is to provide a communication backbone for communication between peers and

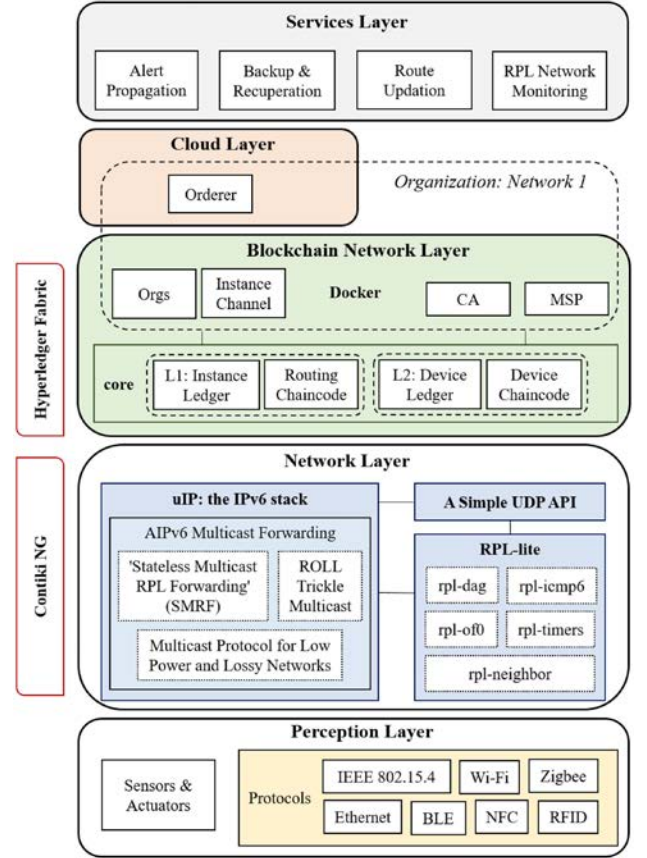


Fig. 4. Cross-layer architecture of RPL and blockchain network.

to host an orderer service that assists in running the consensus mechanism and validating transactions.

Application layer: Finally, the application of all the operations can be used in the application layer. Applications, such as network monitoring, route update visualizations, backup, recuperation, can be realized from this layer.

2) *Deployment of Chaincodes:* We have designed and deployed two chaincodes in our proposed model, both on the *instanceChannel* channel.

3) *Device Chaincode:* To maintain a consistent database of IoT devices and network topology in the network, we will use a chaincode related to storing each device's identity, authentication, and reputation values (see Algorithm 8). This chaincode will also have functions related to registering the device after LBR authenticates the device using our ECC-based key exchange mechanism, updating the device information, updating the reputation value, and deleting the device if it leaves the network. ρ is the base reputation value for any newly registered device on the blockchain network, as mentioned in the trust calculation.

4) *Routing Chaincode:* To preserve the updates in the routing topology and RPL-instance information of each LBR we have designed and implemented a routing chaincode, as presented in Algorithm 9. After receiving DAO messages from each new node that was recently registered or during the rank update among any node in the topology, the control message will trigger the

Algorithm 8: Device Registration.

Require: Registration Request $PubK, PriK, MAC_{ID}, IP$

Ensure: Device registration on Blockchain

- 1: $Dev_{ID} \leftarrow GenerateRandomDevID()$
- 2: **if** Device is LBR **then**
- 3: **Access Level** $\leftarrow 1$
- 4: **else**
- 5: **Access Level** $\leftarrow 0$
- 6: **end if**
- 7: **Device** $\leftarrow (Dev_{ID}, R = \rho)$
- 8: **DeviceChaincode** $\leftarrow network.getContract('deviceChaincode')$
- 9: **if** DeviceChaincode.execute('retriveDevice', Device) **then**
- 10: **Return** \leftarrow (Device already exist)
- 11: **else**
- 12: DeviceChaincode.submitTransaction('registerDevice', Device, Access Level)
- 13: **Response** \leftarrow New device registered on ledger
- 14: **Return** \leftarrow (New device registered, Dev_{ID})
- 15: **end if**

Algorithm 9: Downward Path Update Request.

Require: Old(current) and New downward paths

Ensure: Update instance information for the network

- 1: **Old** \leftarrow LBR(DevID).getRouteState()
- 2: **New** \leftarrow Path update request
- 3: **RoutingChaincode** \leftarrow network.getContract('routingChaincode')
- 4: **if** New route path is validated **then**
- 5: RoutingChaincode.submitTransaction(
- 6: 'updateDownwardPath', 'DevID', 'New')
- 7: **Return** \leftarrow (Downward Path updated in LBR(DevID))
- 8: **else**
- 9: **Return** \leftarrow (Invalid Update)
- 10: **end if**

creation of a new routing table in LBR. The routing chaincode will be responsible for executing the route change and consistent routing ledger that can be validated by each LBR in the network.

V. EXPERIMENTAL EVALUATION OF RELIABLE-RPL

In this section, we present Contiki-NG integration with Hyperledger Fabric and simulation of attack resilience against rank, sinkhole replay, and route poisoning attacks using Contiki-Cooja. Table III presents the simulation and experimentation parameters of reliable-RPL.

A. Hyperledger Fabric

Hyperledger Fabric [36] is a platform for building distributed ledger applications for business solutions. It has a modular architecture that allows different components, such as consensus and membership services, to be plug-and-play. As our proposed

TABLE III
SIMULATION PARAMETERS

Parameter	Specification
Operating System	Ubuntu 22.04
Blockchain Platform	Hyperledger Fabric and Hyperledger Caliper
Simulator	Contiki-ng Cooja v4.8
Transport Layer Protocol	UDP
Routing Protocol	RPL
PHY and MAC Layer	IEEE 802.15.4 and ContikiMAC
RPL Storing Mode	Non-Storing
Number of Motes	11 to 150 (for variation)

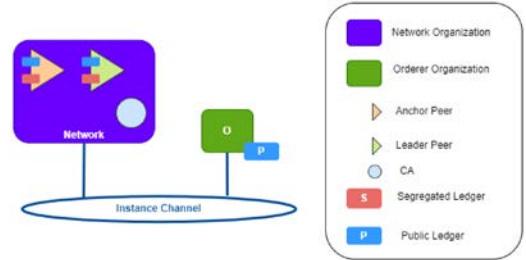


Fig. 5. Blockchain network model on Hyperledger Fabric.

system uses consortium blockchain, we will implement Hyperledger Fabric to create our blockchain network. We have used Hyperledger Fabric to create our consortium blockchain network. Fig. 5 shows the logical model of our Hyperledger Fabric system deployed and integrated with the Contiki-NG operating system. Our experimental fabric network consists of one organization that represents the network of LBRs called *Network1*. We have created a channel called *instancechannel* on this network to connect the peers to share a common public ledger. *Network* organization initially had a CA, and one peer as a leader joined it. A registration request by any LBR is authenticated by the leader peer, and on successful authentication, a new peer is created in this organization. Each peer hosts its own segregated and public ledger in CouchDB instances, which are simple key-value pair databases.

To test our framework, we have used *Docker* to host our blockchain network. Docker is an open-source platform that enables application deployment, scaling, and management automation through containerization. This technology offers a standardized container unit, encapsulating the software and its dependencies. Containers are self-contained and lightweight environments encompass all the components to run an application, such as code, runtime, libraries, system tools, and configurations. The Hyperledger Fabric model consists of four main elements: *assets*, *chaincode*, *ledger*, and *transactions*. Assets are anything that has value and can be exchanged over the network. Chaincode is the smart contract that defines the business logic and rules for asset manipulation. The ledger is the append-only record of all the transactions on the network. Transactions are the invocations of chaincode that result in state changes of assets. Participants in the Hyperledger Fabric network are divided into three roles: *clients*, *peers*, and *orderers*. Clients are applications that act on behalf of users to propose and endorse transactions. Peers are nodes that maintain the ledger and run the chaincode.

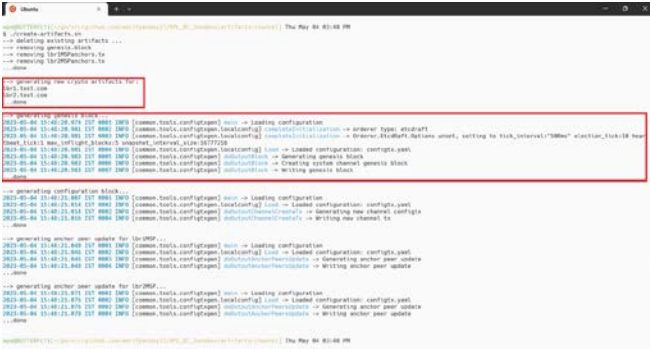


Fig. 6. Creation of genesis block.

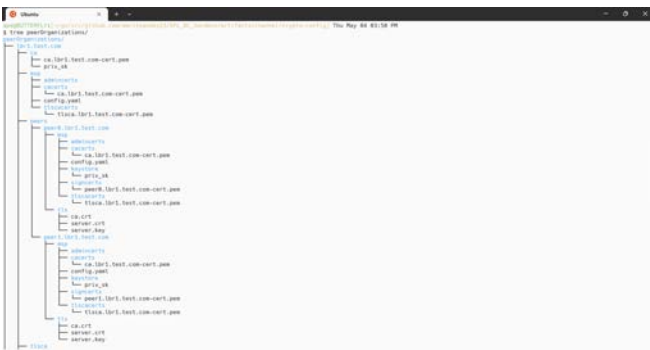


Fig. 7. Cryptomaterial for LBR1 organization.

Orderers are nodes that order transactions into blocks and broadcast them to peers. Peers and orderers can belong to different organizations enrolled through a *Membership Service Provider (MSP)*. All the organizations (LBR nodes) share a common public ledger by joining a channel. Each peer in the organization hosts two databases, namely *Transaction Log* and *World State*. Transaction log stores the history of updated transactions and blocks, whereas the World state stores the actual asset updates.

B. Blockchain Network Setup

The fog layer comprising the LBR devices runs the blockchain network, and we modeled this network on Hyperledger Fabric, as depicted in Fig. 5.

The blockchain network is responsible for registering, authenticating, and maintaining any new LBR device wanting to join as a peer. Since our network is a consortium blockchain network, employing an ECC-based authentication mechanism to allow only permissioned nodes to access the ledger becomes necessary. Fig. 6 shows the beginning of the system’s setup by creating genesis and configuration blocks via *configtx.yaml* file. First, the cryptomaterial for all the entities are created using the cryptogen tool provided by the Hyperledger Fabric platform.

Fig. 7 also shows the generation of cryptomaterial for LBR1 organization in terms of MSP identities, CA, and keys of various entities on the network done by the *cryptogen* binaries by utilizing the *crypto-config.yaml* file. The MSP will use these files

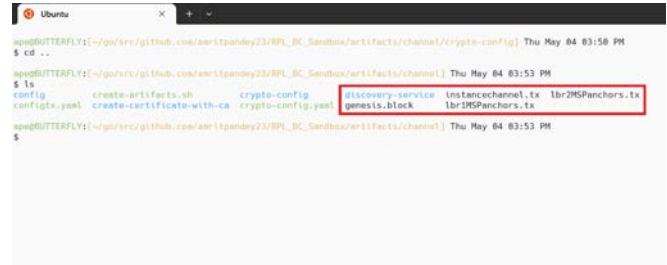


Fig. 8. Generated organization and genesis block binaries.

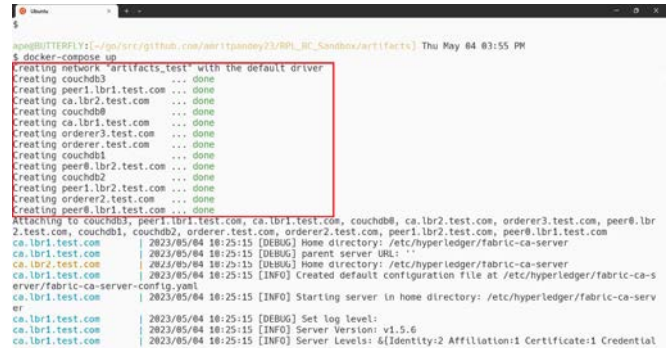


Fig. 9. Network up with docker.

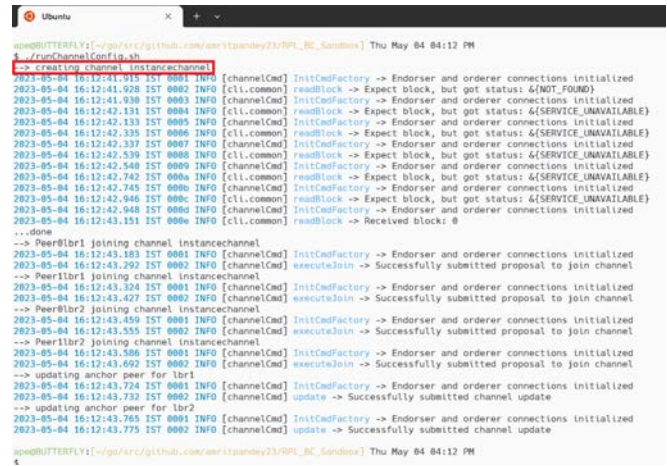


Fig. 10. “instanceChannel” channel creation.

to identify and authenticate peers and network entities. Fig. 8 depicts the generation of binaries.

Fig. 9 shows the running network in the docker container on which each entity is hosted. This network includes two peers, one CA, two instances of CouchDB (one for each peer), and three orderers (as used by the Raft consensus mechanism).

Fig. 10 shows the channel creation among the recently deployed Hyperledger entities. The *instanceChannel* that connects all network peers is granted by channel binary.

C. Chaincode Deployment

Chaincode in Hyperledger Fabric is responsible for implementing the business logic of smart contracts. It defines the rules and functions for reading from and writing to the blockchain

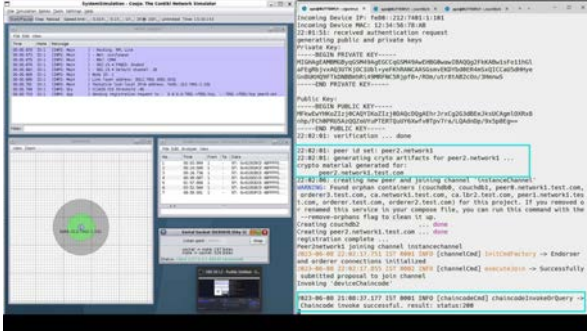


Fig. 16. Peer update on Hyperledger network.

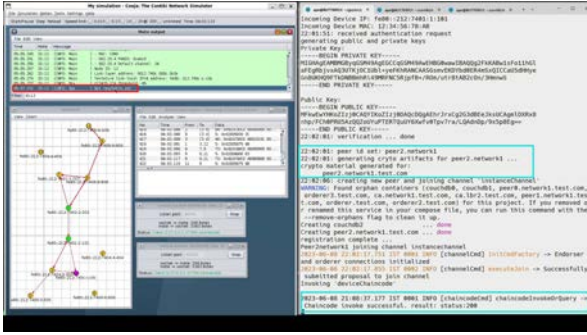


Fig. 17. Secure route path selection on detecting a malicious node.

to the orderer for validation of the transaction and committing it to the routing ledger.

D. Simulation of Attack Resilience by Reliable-RPL

1) *Attack Model*: According to the system model, IoT devices communicate in the RPL network using insecure channels. An adversary can intercept the control messages, reducing trust among LBR and RPL nodes. We consider the Dolev–Yao threat model, which allows an adversary to modify the DIO, DIS, and DAO messages in RPL networks. The implementation results of our proposed framework prove its ability to withstand rank attacks, replay attacks, sinkhole attacks, and route poisoning attacks.

2) Informal Security Analysis of Reliable-RPL on Contiki-Cooja:

- 1) *Rank attacks*: In a typical network topology represented by $G(V, E)$ where V is a set of nodes(motes) and E represents the connection between them, each V 's depth in the topology is referred to by its rank $R(V)$. The rank is a relative concept dependent on the OF in which the DODAG operates. In our rank attack simulation, we have made our malicious node advertise the correct rank for 6–12 s, after which it manipulates the control messages to decrease its rank. As was observed in the simulation, due to rank change, the nearby nodes updated their parent list and added the malicious node as the default parent. For the initial few minutes, a heavy loss of packet delivery was observed that drastically decreased the relative reputation value of the malicious node.
- 2) *Sinkhole attacks*: In the network topology represented by $G(V, E)$, V is a set of nodes(motes), and E represents

the link connections between them. Then, the number of packets delivered by intermediate nodes from the set of motes V_n is given by $P(t)_d$, and the total number of packets forwarded through the mote is given by $P(t)_s$. A node becomes malicious when it starts to drop packets through it except the control packets such that $P(t)_d \ll P(t)_s$. As a significant number of packets are dropped, then mote becomes a sinkhole. In our sinkhole attack simulation, the mote was programmed to drop simple “Hello” packets sent upwards in the topology every 2 s. It was observed that a significant drop in delivered packets led to a decrease in the reputation of the mote. The route paths were then switched to choose a more secure path.

- 3) *Route poisoning attacks*: As individual LBRs publish all the routing updates and information about RPL instances on the routing ledger, the ledger maintains the consistency of the routing topology. To simulate a route poisoning attack on the LBR, we first replaced one of the existing peers with an unauthenticated peer. It was found that the new peer failed to create a new transaction as the Hyperledger network threw the error. Second, we try to make an inappropriate update to the network by unlabeled a malicious node, i.e., increasing the trust reputation value, which was also rejected by the orderer as the DeviceChaincode transaction update was inconsistent with the world state data.
- 4) *Replay attacks*: In the network topology represented by $G(V, E)$, V is a set of nodes(motes), and E represents the link connections between them. An adversary triggering a replay attack is a malicious node that captures old control messages and retransmits them to create inconsistent topological changes. To prevent this, the timestamp on this control message is verified with the latest control message transmitted by the LBR. In our simulation, we created a malicious node transmitting old and new control packets at 4–6 s. Upon receiving old packets, the surrounding nodes decreased the primary trust value of the malicious node.

E. Performance Analysis

The proposed system has been analyzed and evaluated using the tools available in the Cooja simulator. Data capture included reception logs, radio transmission packets, and routing table updates. We have used Wireshark to evaluate the radio packets and find the packet drop ratio. The routing overhead metric was calculated by taking the combined runtime of the reliable-RPL system. Finally, we also evaluated our Hyperledger Fabric system using Hyperledger Caliper. The results are compiled below in the graphs shown in Figs. 18, 19, and 22. Fig. 18 shows the average packet loss during all the attacks. The results have been compared with the [3] system. It was observed that during attacks, such as rank and sinkhole, the packet loss is less as the reputation value of the adversary decreases rapidly. If the value of ρ was increased, as discussed in our modified trickle algorithm, then a significant increase in the packet drop was observed.

Fig. 19 shows the average routing overhead after integrating Hyperledger Fabric with Contiki-NG. With the number of increases in the nodes in the simulation, it was observed

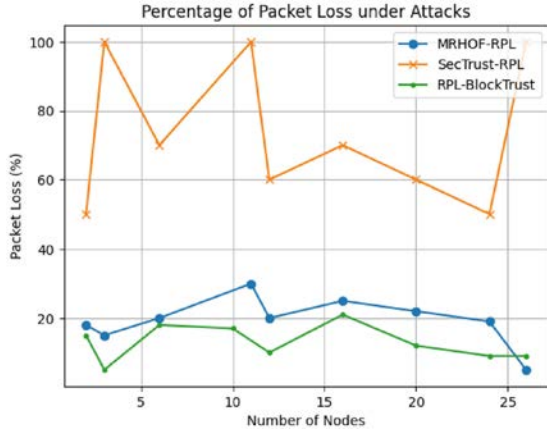


Fig. 18. Average packet loss during attacks.

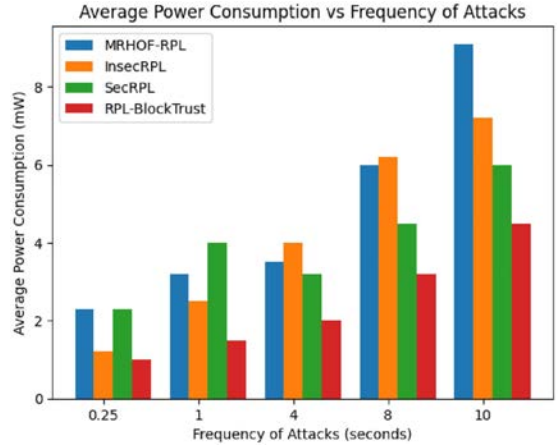


Fig. 21. Average power consumption.

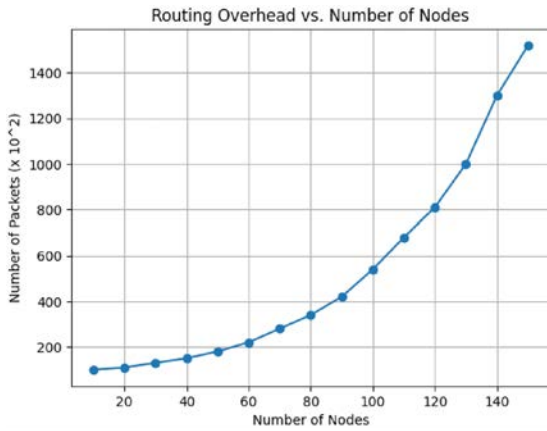


Fig. 19. Average routing overhead.

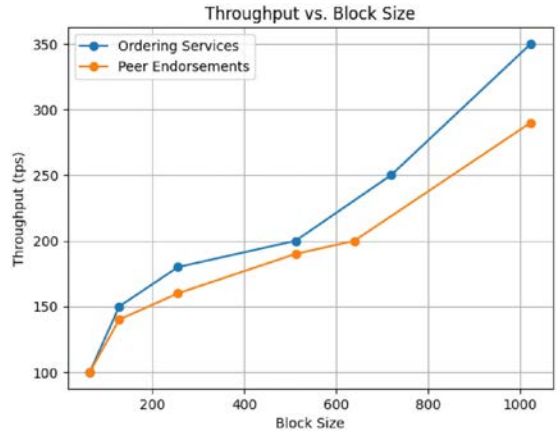


Fig. 22. Blockchain throughput.

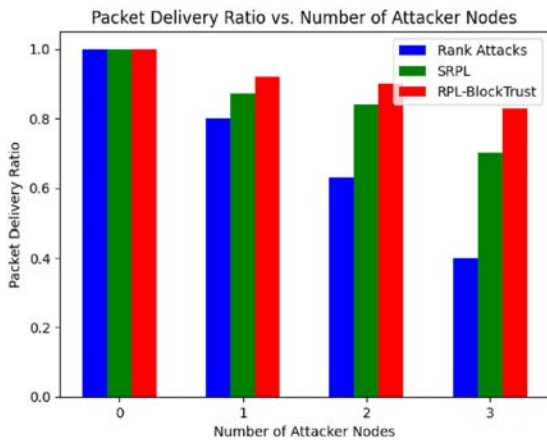


Fig. 20. Packet delivery ratio.

that routing overhead increases exponentially. This scenario is because as more nodes join the network, the number of route updates rises significantly. In the future, we will work to improve the efficiency of decreasing the routing overhead by employing efficient route update schemes.

Fig. 20 showcases the efficiency of “reliable-RPL” in maintaining higher packet delivery ratios even under malicious rank

attacks. It was observed that the system held a consistent packet delivery ratio compared to other mechanisms, even when increasing the number of attacker nodes in the network.

On an increasing number of attacks per second, it was observed that the reliable-RPL has a significant increase in power consumption; overall, it performed better than the rest of the mechanism regarding an increase in the power consumption ratio. Fig. 21 compares reliable-RPL with other mechanisms.

The standalone Hyperledger Fabric network’s performance was evaluated using Hyperledger Caliper. The registration and authentication of two fog nodes as peers on the network took 13 seconds, the highest among all the operations on the network. The throughput evaluation of the standalone Hyperledger Fabric blockchain network, which is evaluated as transactions per second, was also done, and it was observed that the throughput rate increases with an increase in the number of peers.

VI. CONCLUSION AND FUTURE WORK

In this article, a blockchain-based SRPL protocol, called reliable-RPL, is proposed to achieve trust, reliable, scalable, lightweight, and energy-efficient data transmission in RPL-enabled IoT networks. A consortium blockchain network of

LBRs runs a device chaincode for registration and ECC-based authentication of new peers and a routing chaincode for updating the routing information on the ledger of each peer. A trust-based trickle algorithm is proposed to detect data transmission inconsistencies based on node and link reliability through an ROF using various trust evaluations. A novel cross-layer architecture that amalgamates RPL and blockchain networks is proposed. The proposed protocol is resilient against rank, version, sink-hole, and replay attacks. In addition, the proposed mechanism also solves the problem of backup and recuperation in the LBRs by maintaining a consistent public ledger of RPL instances and routing information that can quickly restore the previous routing state of any LBR. We have implemented our model on an integrated simulation environment using Contiki-NG and Hyperledger Fabric. Attack resilience capabilities of the proposed reliable-RPL are simulated using Contiki-Cooja. In addition, we evaluated the performance of the proposed framework on Hyperledger Caliper. We proved the efficiency of reliable-RPL in terms of average packet loss, routing overhead, packet delivery ratio, average power consumption, and throughput.

In the future, we aim to explore the application of our model to provide security in real-time IoT applications, such as secure data processing and cloud communication. We also aim to complete the formal analysis of our system using BAN logic and the AVISPA tool.

REFERENCES

- [1] A. Dvir, T. Holczer, and L. Buttyan, "VeRA-version number and rank authentication in RPL," in *Proc. IEEE 8th Int. Conf. Mobile Ad-Hoc Sensor Syst.*, IEEE, 2011, pp. 709–714.
- [2] M. Conti, P. Kaliyar, M. M. Rabbani, and S. Ranise, "SPLIT: A secure and scalable RPL routing protocol for Internet of Things," in *Proc. 14th Int. Conf. Wireless Mobile Comput., Netw. Commun.*, IEEE, 2018, pp. 1–8.
- [3] D. Airehrour, J. A. Gutierrez, and S. K. Ray, "Sectrust-RPL: A secure trust-aware RPL routing protocol for Internet of Things," *Future Gener. Comput. Syst.*, vol. 93, pp. 860–876, 2019.
- [4] T. Winter et al., "RPL: IPv6 routing protocol for low-power and lossy networks," *RFC*, vol. 6550, pp. 1–157, 2012.
- [5] H. Perrey, M. Landsmann, O. Ugus, M. Wählisch, and T. C. Schmidt, "TRAIL: Topology authentication in RPL," in *Proc. Int. Conf. Embedded Wireless Syst. Netw.*, 2016, pp. 59–64.
- [6] S. Seeber, A. Sehgal, B. Stelte, G. D. Rodosek, and J. Schönwälder, "Towards a trust computing architecture for RPL in cyber physical systems," in *Proc. 9th Int. Conf. Netw. Service Manage.*, IEEE, 2013, pp. 134–137.
- [7] A. Sehgal, A. Mayzaud, R. Badonnel, I. Chrisment, and J. Schönwälder, "Addressing DODAG inconsistency attacks in RPL networks," in *Proc. 2014 Glob. Inf. Infrastructure Netw. Symp.*, IEEE, 2014, pp. 1–8.
- [8] G. Ramezan and C. Leung, "A blockchain-based contractual routing protocol for the Internet of Things using smart contracts," *Wireless Commun. Mobile Comput.*, vol. 2018, Art. no. 4029591.
- [9] K. Iuchi, T. Matsunaga, K. Toyoda, and I. Sasase, "Secure parent node selection scheme in route construction to exclude attacking nodes from RPL network," in *Proc. 21st Asia-Pacific Conf. Commun.*, IEEE, 2015, pp. 299–303.
- [10] K. F. Haque, A. Abdelgawad, V. P. Yanambaka, and K. Yelamarthi, "An energy-efficient and reliable RPL for IoT," in *Proc. IEEE 6th World Forum Internet Things*, 2020, pp. 1–2.
- [11] A. Lahbib, K. Toumi, S. Elleuch, A. Laouiti, and S. Martin, "Link reliable and trust aware RPL routing protocol for Internet of Things," in *Proc. IEEE 16th Int. Symp. Netw. Comput. Appl.*, 2017, pp. 1–5.
- [12] N. Nobakht, S. S. Kashi, and S. Zokaie, "A reliable and delay-aware routing in RPL," in *Proc. 5th Conf. Knowl. Based Eng. Innov.*, 2019, pp. 102–107.
- [13] P. Shahbakhsh, S. H. Ghafouri, and A. K. Bardsiri, "RAARPL: End-to-end reliability-aware adaptive RPL routing protocol for Internet of Things," *Int. J. Commun. Syst.*, vol. 36, no. 6, 2023, Art. no. e5445. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/dac.5445>
- [14] A. Seyfollahi, M. Mainuddin, T. Taami, and A. Ghaffari, "RM-RPL: Reliable mobility management framework for RPL-based IoT systems," *Cluster Comput.*, vol. 27, pp. 4449–4468, 2024. [Online]. Available: <https://api.semanticscholar.org/CorpusID265488054>
- [15] Z. Xuemin et al., "Self-organizing key security management algorithm in socially aware networking," *J. Signal Process. Syst.*, vol. 96, pp. 369–383, 2024. [Online]. Available: <https://api.semanticscholar.org/CorpusID:270078601>
- [16] A. Raouf, A. Matrawy, and C.-H. Lung, "Routing attacks and mitigation methods for RPL-based Internet of Things," *IEEE Commun. Surv. Tut.*, vol. 21, no. 2, pp. 1582–1606, Apr.–Jun. 2019.
- [17] A. Raouf, A. Matrawy, and C.-H. Lung, "Secure routing in IoT: Evaluation of RPL's secure mode under attacks," in *Proc. 2019 IEEE Glob. Commun. Conf.*, IEEE, 2019, pp. 1–6.
- [18] G. Glissa, A. Rachedi, and A. Meddeb, "A secure routing protocol based on RPL for Internet of Things," in *Proc. 2016 IEEE Glob. Commun. Conf.*, IEEE, 2016, pp. 1–7.
- [19] B. Ghaleb, A. Al-Dubai, E. Ekonomou, M. Qasem, I. Romdhani, and L. Mackenzie, "Addressing the DAO insider attack in RPL's Internet of Things networks," *IEEE Commun. Lett.*, vol. 23, no. 1, pp. 68–71, Jan. 2019.
- [20] R. Sahay, G. Geethakumari, and B. Mitra, "A novel blockchain based framework to secure IoT-LLNS against routing attacks," *Computing*, vol. 102, no. 11, pp. 2445–2470, 2020.
- [21] D. Airehrour, J. Gutierrez, and S. K. Ray, "Securing RPL routing protocol from blackhole attacks using a trust-based mechanism," in *Proc. 26th Int. Telecommun. Netw. Appl. Conf.*, IEEE, 2016, pp. 115–120.
- [22] P. Thulasiraman and Y. Wang, "A lightweight trust-based security architecture for RPL in mobile IoT networks," in *Proc. 16th IEEE Annu. Consum. Commun. Netw. Conf.*, IEEE, 2019, pp. 1–6.
- [23] M. Zaminkar and R. Fotohi, "SoS-RPL: Securing Internet of Things against sinkhole attack using RPL protocol-based node rating and ranking mechanism," *Wireless Pers. Commun.*, vol. 114, pp. 1287–1312, 2020.
- [24] M. Zaminkar, F. Sarkohaki, and R. Fotohi, "A method based on encryption and node rating for securing the RPL protocol communications in the IoT ecosystem," *Int. J. Commun. Syst.*, vol. 34, 2020, Art. no. e4693.
- [25] M. Hosseini Shirvani and M. Masdari, "A survey study on trust-based security in Internet of Things: Challenges and issues," *Internet Things*, vol. 21, 2022, Art. no. 100640.
- [26] H. Jiang, M. Wang, P. Zhao, Z. Xiao, and S. Dustdar, "A utility-aware general framework with quantifiable privacy preservation for destination prediction in LBSs," *IEEE/ACM Trans. Netw.*, vol. 29, no. 5, pp. 2228–2241, Oct. 2021.
- [27] Y. Liu and Y. Zhao, "A blockchain-enabled framework for vehicular data sensing: Enhancing information freshness," *IEEE Trans. Veh. Technol.*, vol. 73, no. 11, pp. 17416–17429, Nov. 2024.
- [28] A. A. Laghari, A. A. Khan, R. Alkanhel, H. Elmannai, and S. Bourouis, "Lightweight-BIoV: Blockchain distributed ledger technology (BDLT) for internet of vehicles (IoVs)," *Electronics*, vol. 12, no. 3, p. 677, 2023. [Online]. Available: <https://www.mdpi.com/2079-9292/12/3/677>
- [29] J. Yang, K. Yang, Z. Xiao, H. Jiang, S. Xu, and S. Dustdar, "Improving commute experience for private car users via blockchain-enabled multitask learning," *IEEE Internet Things J.*, vol. 10, no. 24, pp. 21656–21669, Dec. 2023.
- [30] Y. Liu et al., "SS-DID: A secure and scalable Web3 decentralized identity utilizing multi-layer sharding blockchain," *IEEE Internet Things J.*, vol. 11, no. 15, pp. 25694–25705, Aug. 2024.
- [31] M. Waqas et al., "Botnet attack detection in Internet of Things devices over cloud environment via machine learning," *Concurrency Computation: Pract. Experience*, vol. 34, no. 4, 2022, Art. no. e6662. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.6662>
- [32] D. Huang, X. Ma, and S. Zhang, "Performance analysis of the raft consensus algorithm for private blockchains," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 1, pp. 172–181, Jan. 2020.
- [33] N. Djedjig, D. Tandjaoui, F. Medjek, and I. Romdhani, "New trust metric for the RPL routing protocol," in *Proc. 8th Int. Conf. Inf. Commun. Syst.*, IEEE, 2017, pp. 328–335.
- [34] Z. Fatima et al., "Mobile crowdsensing with energy efficiency to control road congestion in internet cloud of vehicles: A review," *Multimedia Tools Appl.*, vol. 83, pp. 1–26, 2023.
- [35] D. Airehrour, J. Gutierrez, and S. K. Ray, "A trust-aware RPL routing protocol to detect blackhole and selective forwarding attacks," *J. Telecommun. Digit. Economy*, vol. 5, no. 1, pp. 50–69, 2017.
- [36] E. Androulaki et al., "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proc. 13th EuroSys Conf.*, 2018, pp. 1–15.