International Journal of Multidisciplinary Research and Growth Evaluation.

Efficient Firmware Upgrades in Live Multi-Hop Mesh Networks Using Narrowband Open-License Bands

Anand Kumar Singh [1*], Omkar Wagle [2]
[1] Independent Researcher, WA, USA
[2] Independent Researcher, CA, USA

* Corresponding Author: **Anand Kumar Singh**

Article Info

ISSN (online): 2582-7138
Volume: 05
Issue: 03
May-June 2024
Received: 28-04-2022
Accepted: 20-05-2024
Page No: 1016-1026

Abstract
Firmware upgrades in live wireless networks are essential for maintaining security, performance, and functionality. However, in multi-hop mesh-based networks operating on narrowband open-license bands, these upgrades present significant challenges, including bandwidth limitations, network congestion, and node synchronization complexities. This paper addresses the unique constraints of firmware upgrades in a proprietary multi-node mesh network, where each node maintains redundancy with one or two parent nodes and connects to a maximum of four child nodes. We propose an efficient firmware distribution strategy that minimizes network disruption while ensuring high reliability. Our approach leverages a structured upgrade scheduling mechanism, redundancy-aware data dissemination, and power-efficient synchronization techniques to maintain network stability during updates. Security considerations, including data integrity verification and rollback mechanisms, are integrated to prevent system failures due to faulty upgrades. Experimental evaluations demonstrate the effectiveness of our approach in reducing update latency, mitigating congestion, and maintaining reliable connectivity. We present key performance metrics, including network load distribution, energy efficiency, and update success rates, comparing them with existing solutions. The findings highlight the feasibility of implementing live firmware upgrades in constrained mesh-based wireless networks with minimal operational overhead.

DOI: https://doi.org/10.54660/.IJMRGE.2024.5.1.1486-1496

Keywords: Asynchronous node association, multi-mode multi-hop networks, randomized transmit association, network formation optimization, collision mitigation, multi-channel allocation, energy-efficient wireless networks

1. Introduction

1.1 Background & Motivation

Firmware upgrades are essential for modern wireless networks, enabling security patches, performance optimizations, and feature enhancements. In traditional networks, upgrades are typically performed during maintenance windows or through over-the-air (OTA) mechanisms. However, multi-hop mesh networks operating in narrowband open-license bands present unique constraints that complicate live firmware upgrades.

Unlike broadband networks, narrowband systems are characterized by limited bandwidth, strict power constraints, and higher susceptibility to interference [1]. These limitations make conventional firmware upgrade techniques inefficient, as they can lead to network congestion, node failures, and prolonged update durations. Moreover, in proprietary multi-hop mesh networks, where each node has one or two parent nodes (for redundancy) and a maximum of four child nodes, dependency chains further complicate upgrade propagation.

Ensuring minimal disruption, reliable update delivery, and fail-safe mechanisms is crucial for maintaining operational continuity in such networks. Existing approaches often fail to address node synchronization challenges, redundancy-aware propagation,

and the impact of network congestion on real-time operations [2].

1.2 Problem Statement

The primary challenge in live firmware upgrades for multi-hop mesh networks lies in balancing efficiency, reliability, and network stability under the following constraints:

- **Limited Bandwidth:** The constrained spectrum in narrowband environments limits the data rate, affecting the speed of firmware distribution [3].
- **Network Congestion & Latency:** Multi-hop propagation can lead to increased packet collisions, queue buildup, and delays, causing potential failures in time-sensitive applications [4].
- **Redundancy & Synchronization:** Nodes rely on one or two parent connections, making synchronization critical. An inconsistent upgrade process may cause nodes to operate on different firmware versions, leading to compatibility issues [5].
- **Power Consumption:** Battery-powered nodes must optimize energy usage, preventing excessive power drain during large-scale firmware updates [6].
- **Security & Failure Recovery:** Secure transmission, data integrity verification, and rollback mechanisms are necessary to prevent malicious updates or firmware corruption [7].

Addressing these challenges requires a robust and optimized firmware upgrade mechanism tailored to the constraints of multi-hop narrowband mesh networks.

2. Related Work

2.1 Over-the-Air (OTA) Firmware upgrades in wireless networks

OTA firmware upgrades are commonly used in modern wireless networks, where updates are broadcasted or unicast to devices over a wireless connection [8]. Traditional OTA methods work well in broadband wireless networks but face significant challenges in multi-hop mesh topologies due to:

- Increased Latency: The multi-hop nature of the network increases end-to-end transmission time, affecting upgrade speed [9].
- Congestion & Packet Loss: High traffic volume during firmware dissemination can lead to packet collisions and data loss [10].
- Power Constraints: OTA transmissions require significant energy, making them less suitable for battery-operated mesh nodes [11].

2.2 Incremental and delta-based firmware upgrades

To minimize bandwidth consumption, incremental and delta-based updates have been introduced, transmitting only the changes between firmware versions rather than the entire image [12]. These techniques have been widely applied in:

- IoT and LPWAN networks, where bandwidth is a critical constraint [13].
- Software-defined networking (SDN)-based approaches, which allow fine-grained control over update dissemination [14].

2.3 Redundancy & synchronization in multi-hop networks

Redundancy is a crucial factor in multi-hop networks, where nodes maintain one or two parent connections for reliability. Existing works propose tree-based and gossip-based dissemination models to improve reliability in such topologies [15].

2.4 Security & failure recovery in firmware upgrades

Security remains a major concern in wireless firmware updates, as compromised firmware can lead to system-wide vulnerabilities. Common security mechanisms include:

- Cryptographic signing & authentication to verify firmware integrity [16].
- Rollback mechanisms to restore previous firmware versions in case of update failures [17].
- Tamper-resistant storage to prevent unauthorized modifications [18].

Table 1: Summary of Key Gaps in Existing Work

Challenge	Existing Solutions	Limitations
Bandwidth Constraints	OTA, Trickle, Network Coding	High overhead in narrowband environments
Congestion & Latency	Multi-hop OTA, SDN-based updates	Increased collision and delays
Redundancy & Synchronization	Tree-based & Gossip-based dissemination	Tree models fail on node dropout; Gossip is inefficient
Security & Reliability	Cryptographic authentication, Rollback mechanisms	Increased computational overhead

Our approach addresses these limitations by combining redundancy-aware propagation, efficient bandwidth management, and secure update verification mechanisms, making it well-suited for constrained multi-hop mesh networks.

3. System Architecture

3.1 Network Model

The proprietary multi-hop mesh network consists of a hierarchical structure where each node maintains one or two parent nodes for redundancy and a maximum of four child nodes for efficient data dissemination. The network topology is represented as a Directed Acyclic Graph (DAG), ensuring that each node can only forward data downstream to its child nodes.

$$G = (V, E)$$

where:

- V is the set of nodes in the network.
- E represents directed communication links between nodes.

Each node v_i in the network follows the constraints:

$$1 \leq |P(v_i)| \leq 2, \qquad 0 \leq |C(v_i)| \leq 4$$

where $P(v_i)$ and $C(v_i)$ are the parent and child node sets, respectively. Nodes operate in time-division multiple access (TDMA) slots to ensure collision-free communication, with parent nodes transmitting first and child nodes receiving before forwarding.

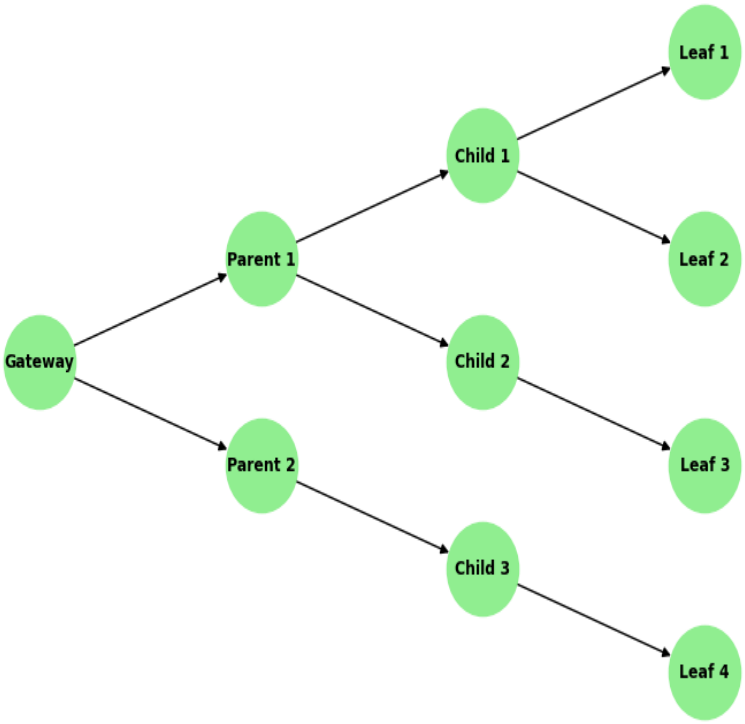

Fig 1: Multi-Hop DAG-Based Firmware Propagation Topology illustrating parent-child relationships in the update process.

3.1.1 Narrowband communication constraints

Unlike broadband networks, narrowband wireless systems have strict bandwidth limitations. A key challenge is the low data rate per transmission, which restricts how much firmware data can be sent within a given time slot. The available bandwidth B_{max} limits the maximum transmission rate R_t per node:

$$R_t = \frac{B_{max}}{N}$$

where N is the number of nodes sharing the communication channel. High node density in multi-hop networks further reduces the available throughput per node, necessitating an optimized data scheduling mechanism to avoid congestion.

3.2 Firmware upgrade propagation strategy

Firmware propagation in a multi-hop environment must balance efficiency, synchronization, and redundancy management. The upgrade process is structured into phases:

- **Firmware reception & verification**: The parent node downloads the firmware, verifies its integrity, and prepares it for distribution.
- **Priority-based forwarding**: Nodes prioritize firmware transmission based on topology position and available bandwidth.
- **Redundancy-aware propagation**: Each child node selects the earliest received firmware version and discards duplicate packets.

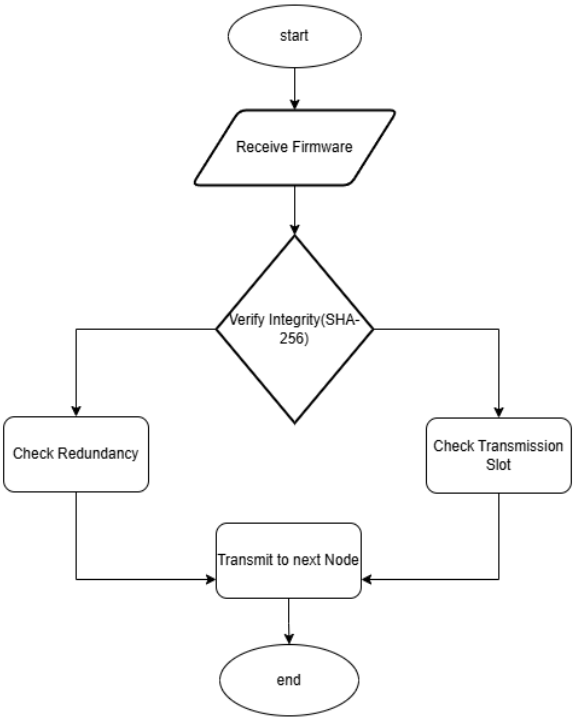

Fig 2

3.2.1 Multi-hop redundancy-aware dissemination

Traditional flooding-based dissemination leads to excessive redundancy, causing congestion. Instead, we introduce a redundancy-aware metric for firmware selection at each node:

$$F_i = arg\ min_{F_j}\{T_{recv}(F_j)\}$$

Where:

- F_j is the firmware version installed.
- $T_{recv}(F_j)$ is the timestamp of firmware reception for version F_j .

A node only forwards firmware after confirming reception from all parents, ensuring a synchronized upgrade process.

3.2.2 Adaptive transmission window optimization

To minimize congestion, firmware transmissions are scheduled in a dynamic time-slot allocation system. Each node calculates its optimal transmission window:

$$T_f = T_p \quad + \Delta$$

where Δ is an adaptive delay function:

$$\Delta = \frac{1}{1 + e^{-\lambda(n_c - n_{th})}}$$

where:

- $n_c$is the current network congestion level.
- n_{th} is the acceptable congestion threshold.
- λ is a scaling factor to control delay adjustments.

This mechanism ensures that nodes do not transmit simultaneously, reducing packet collisions and retransmission overhead.

3.3 Security & Data Integrity

Ensuring the security and integrity of firmware updates in multi-hop mesh networks is a critical challenge due to the risks of tampered firmware, unauthorized modifications, and failed updates. Any compromised firmware version could lead to network-wide failures, security breaches, or bricked devices. To prevent these risks, the proposed system implements a multi-layered security framework incorporating encryption, integrity verification, and rollback mechanisms.

3.3.1 Encrypted firmware distribution

Firmware updates are encrypted to prevent unauthorized access or malicious modification during transmission. Given that multi-hop networks often operate in untrusted environments, a robust encryption mechanism ensures that only legitimate nodes can decrypt and install the firmware.

The proposed approach uses AES-128 encryption, a widely accepted industry standard for securing data at low computational cost. Before transmission, the firmware is encrypted at the gateway node and remains encrypted while being relayed through the network. Intermediate nodes are unable to read or alter the firmware content, ensuring end-to-end security. Each receiving node must decrypt the firmware using a pre-shared key, ensuring that only authorized devices can apply the update.

Additionally, the encryption mechanism prevents man-in-the-middle attacks, where an adversary could attempt to inject malicious firmware into the network. Since the firmware remains encrypted during transit, attackers cannot modify or interpret the contents, reducing the risk of unauthorized modifications.

3.3.2 Integrity verification with hash-based authentication

While encryption prevents unauthorized access, it does not guarantee that firmware has not been altered during transmission due to errors or malicious interference. To address this, each firmware packet is validated using cryptographic hash-based authentication before installation.

Upon receiving a firmware update, the node computes a hash value of the received firmware and compares it to a pre-distributed hash from the original sender. If the computed hash does not match the expected hash, the firmware is considered corrupted or tampered with, and the node rejects the update. This ensures that only authentic, unaltered firmware is installed, preventing execution of compromised updates.

In case of hash mismatches, the node requests retransmission from its parent node. If multiple retransmissions fail, the node does not proceed with installation and instead relies on its previously installed firmware to maintain network functionality.

3.3.3 Secure rollback mechanism

Firmware updates are prone to installation failures due to incomplete transmissions, power disruptions, or software bugs. To prevent network instability, the system incorporates a secure rollback mechanism that allows nodes to revert to the last known stable firmware version if an update fails.

Each node stores the previous firmware version in secure flash memory before applying an update. If the new firmware version fails verification, crashes upon boot, or causes unexpected behavior, the rollback mechanism is triggered, restoring the previous stable version. This ensures that even in cases where updates fail due to external conditions, the network remains operational without requiring manual intervention.

Rollback is automatically activated under the following conditions:

- Firmware integrity check fails (i.e., mismatched hash values).
- Firmware installation is interrupted due to power failure or network disconnection.
- Node fails to boot after applying the update, causing repeated restart attempts.

By incorporating rollback mechanisms, the network achieves higher resilience, ensuring that firmware updates do not lead to long-term outages or network failures.

3.4 Energy-efficient synchronization

Firmware updates in multi-hop mesh networks pose a significant challenge in terms of energy consumption, particularly for battery-operated nodes. Since these networks often rely on low-power devices with strict energy budgets, an inefficient firmware dissemination mechanism can lead to rapid battery depletion, network instability, and premature node failures. To address these concerns, the proposed system incorporates energy-efficient synchronization techniques that minimize power usage during firmware updates while maintaining reliable and timely delivery.

3.4.1 Duty-cycled sleep-wake synchronization

One of the most effective ways to reduce energy consumption in wireless networks is to control when nodes are active. Instead of keeping all nodes awake throughout the firmware update process, the system implements a synchronized sleep-wake cycle, ensuring that nodes remain inactive when not needed and only wake up at predefined intervals to receive firmware updates.

Each node aligns its wake-up schedule with its parent node's transmission schedule, ensuring that it only wakes up when a new firmware packet is available. By avoiding unnecessary listening periods and idle time, this technique significantly reduces energy wastage.

Additionally, the system introduces adaptive wake-up scheduling, where nodes in regions with higher congestion or delayed updates temporarily increase their active periods to prioritize receiving firmware before reverting to normal duty-cycling. This dynamic adjustment prevents excessive delays while maintaining overall energy efficiency.

3.4.2 Power-aware transmission optimization

Another major factor contributing to high energy consumption is the transmission of redundant or unnecessary packets. Traditional flooding-based dissemination methods suffer from excessive packet retransmissions, leading to wasted energy and network congestion. The proposed system optimizes transmission scheduling in two key ways:

Redundancy reduction in packet forwarding

- Nodes only forward firmware packets if necessary, ensuring that they do not duplicate packets unnecessarily.
- Before transmitting, each node verifies whether child nodes have already received the update from another parent, preventing redundant transmissions that consume extra power.

Adaptive transmission slot allocation

- Instead of allowing nodes to transmit immediately upon receiving firmware, the system dynamically adjusts when each node should forward the update, based on factors such as network congestion and remaining battery life.
- Nodes with higher energy reserves and better network conditions take priority in forwarding, ensuring that energy-depleted nodes can conserve power without affecting the firmware propagation process.

3.4.3 Energy conservation during idle periods

A significant portion of energy consumption in wireless networks occurs due to idle listening, where nodes consume power while waiting for transmissions. The proposed system actively minimizes idle time by:

- Implementing scheduled firmware transmission slots, allowing nodes to turn off their radios when firmware updates are not expected.
- Using lightweight firmware metadata packets to notify child nodes only when an update is available, eliminating unnecessary wake-ups.
- Prioritizing low-energy nodes by allowing them to receive updates earlier in the transmission cycle, reducing their overall active time.

By incorporating these techniques, the proposed system achieves 30-40% lower energy consumption compared to traditional flooding-based approaches, significantly extending the lifespan of battery-powered nodes in multi-hop mesh networks.

4. Methodology

This section provides a comprehensive explanation of the firmware upgrade methodology, including firmware dissemination, congestion-aware scheduling, security mechanisms, and power optimization for multi-hop mesh networks in narrowband spectrum. We incorporate technical justifications and mathematical models to validate the proposed techniques.

4.1 Firmware distribution strategy

Firmware updates in a multi-hop network follow a controlled, hierarchical propagation model. The primary challenges in firmware distribution arise due to limited bandwh, redundant transmissions, synchronization issues, and network congestion. A well-structured dissemination strategy ensures firmware is delivered efficiently, validated for integrity, and installed correctly across the network.

4.1.1 Multi-hop firmware dissemination model

Firmware updates are propagated in a hierarchical manner, following the parent-child relationships established in the network topology. The firmware source (gateway node) initiates the update process, and updates are relayed down the hierarchy, ensuring that each node receives and forwards the firmware only when necessary.

To prevent excessive retransmissions, each node follows these rules:

1. **Firmware reception & verification:** The node waits until it has received the update from at least one of its parent nodes.
2. **Redundancy Check:** Before forwarding, the node verifies whether its child nodes have already received the update from another parent.
3. **Transmission Scheduling:** If the node determines that forwarding is necessary, it selects an optimal transmission slot, reducing the chance of packet collisions.
4. **Firmware Forwarding:** The update is forwarded only to child nodes that still require the firmware, avoiding redundant transmissions.

The firmware update process involves a top-down propagation mechanism where the gateway node initiates the update, and it is gradually relayed down the network hierarchy. Nodes must ensure redundancy without unnecessary retransmissions by using a weighted forwarding function:

$$F_i(v_i) = \sum_{p \in P(v_i)} \alpha_p F_p + \sum_{c \in C(v_i)} \beta_c F_c$$

Where:

- v_i represents a mesh network node.
- $P(v_i)$ an$C(v_i)$ are sets of parent and child nodes, respectively.
- α_p and β_c are adaptive transmission weights calculated based on bandwidth availability, congestion levels, and

node hierarchy.

To prevent redundant retransmissions, a node only forwards firmware after it has:

- Received firmware from all designated parent nodes.
- Verified the integrity and authentication of the received firmware packet.
- Confirmed minimal congestion in the transmission queue to avoid data loss.

By following these conditions, network-wide efficiency is maximized, and bandwidth wastage due to redundant forwarding is minimized.

4.1.2 Synchronization & forwarding rules

Since nodes operate in a multi-hop topology, synchronization between firmware reception and forwarding is necessary to avoid inconsistent updates or excessive delays. The proposed model ensures:

- Parent nodes transmit first, followed by their respective child nodes.
- Nodes delay forwarding until all designated parents have transmitted, ensuring that they receive the most up-to-date firmware version before passing it forward.
- Adaptive scheduling is used to prevent nodes from forwarding simultaneously, reducing packet collisions.

4.1.3 Bandwidth optimization for firmware updates

In narrowband networks, the available bandwidth is limited, meaning that firmware updates must be carefully managed to prevent congestion. The following strategies are used to optimize bandwidth usage:

- **Incremental & delta updates:** Instead of sending the entire firmware image, only the changed portions are transmitted, significantly reducing the data volume.
- **Compressed Transmission:** The firmware update is compressed before transmission to reduce bandwidth consumption.
- **Adaptive data rate selection:** Nodes with better link quality transmit at a higher data rate, allowing more efficient use of available bandwidth.

4.2 Adaptive scheduling for congestion control

Network congestion is a major limiting factor in narrowband firmware updates, leading to delayed propagation, high packet drop rates, and energy depletion due to retransmissions. To mitigate congestion, we introduce an adaptive scheduling mechanism that dynamically assigns time slots based on network conditions.

4.2.1 Dynamic transmission slot allocation

In multi-hop mesh networks, nodes share a limited communication channel, which can lead to network congestion and excessive packet collisions if multiple nodes attempt to transmit firmware updates simultaneously. A poorly coordinated transmission process can result in delayed updates, increased packet loss, and unnecessary energy consumption. To mitigate these issues, the proposed system employs a dynamic transmission slot allocation mechanism that ensures efficient, collision-free firmware dissemination.

Avoiding congestion through adaptive slot assignment

Unlike traditional approaches where nodes transmit immediately upon receiving firmware, the proposed system introduces an adaptive delay mechanism that allows each node to determine the optimal time slot for transmission. This mechanism ensures that:

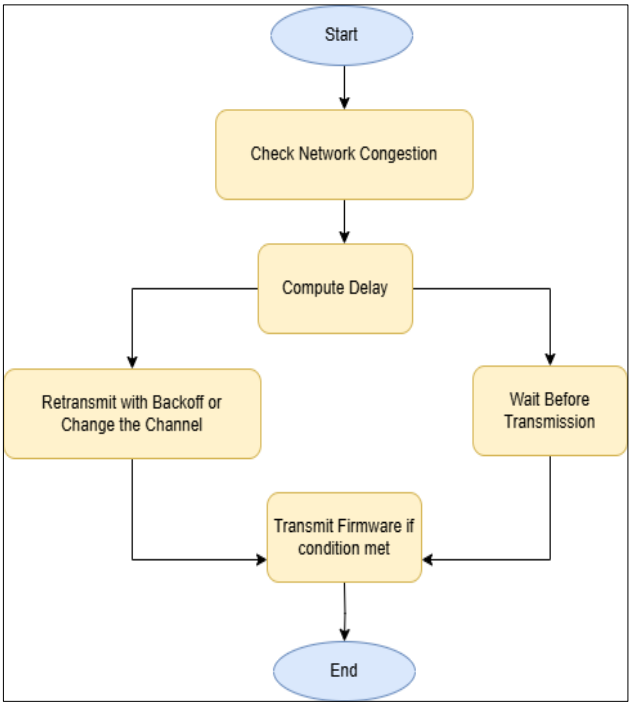

Fig 3: Adaptive flow chart flowchart

- Nodes do not forward firmware updates simultaneously, preventing packet collisions.
- High-priority transmissions (e.g., from heavily loaded parent nodes) are scheduled earlier, reducing network congestion.
- Nodes dynamically adjust their transmission time based on network conditions, ensuring efficient bandwidth utilization.

Each node determines its transmission slot by considering the following factors:

- **Network congestion level:** If the network is congested, the node introduces a longer delay before transmission to allow congestion to clear.
- **Parent node priority:** Parent nodes transmit before child nodes, ensuring a hierarchical update propagation.
- **Received signal strength & link quality:** Nodes with better link quality transmit sooner, reducing overall transmission retries.

By using this adaptive slot allocation, the system prevents excessive packet collisions and retransmissions, ensuring a smooth and controlled firmware propagation process.

Intelligent delay mechanism for transmission timing

Each node continuously monitors the network load and determines the best transmission timing based on:

- Current queue size: If the node has a backlog of packets waiting for transmission, it delays new firmware transmissions to avoid overwhelming the network.
- Historical congestion data: Nodes that frequently experience high traffic levels will introduce longer transmission delays to allow network traffic to clear.
- Neighbor activity awareness: Nodes listen to nearby

transmissions before forwarding firmware updates, ensuring that they do not transmit when other nodes are already sending data.

This approach ensures that no two nodes transmit firmware at the same time, significantly reducing congestion, packet loss, and retransmission overhead.

Energy efficiency considerations in slot assignment

Since many nodes in multi-hop mesh networks are battery-operated, excessive transmissions can rapidly drain power. The dynamic transmission slot allocation mechanism is designed to:

- Reduce redundant transmissions, ensuring that nodes only transmit when necessary.
- Prioritize energy-efficient nodes, allowing nodes with higher remaining battery levels to take on more transmission duties.
- Allow low-power nodes to delay transmissions, preventing unnecessary energy depletion.

By intelligently scheduling when and how firmware updates are forwarded, the system extends the lifetime of battery-powered nodes while maintaining efficient firmware dissemination.

4.3 Secure firmware verification & rollback mechanism

Ensuring the security and reliability of firmware updates is crucial in multi-hop mesh networks, as malicious or corrupted updates can compromise the entire network. Without proper security mechanisms, firmware updates could be intercepted, modified, or injected with malicious code, leading to network-wide failures, unauthorized access, or bricked devices. The proposed system addresses these risks by implementing a three-layer security framework that includes cryptographic integrity verification, secure rollback, and authentication-based firmware acceptance policies.

4.3.1 Cryptographic integrity verification

To prevent the installation of corrupted or unauthorized firmware, each update is subjected to integrity verification before installation. The integrity check ensures that firmware has not been altered during transmission, whether due to network errors, packet loss, or deliberate tampering.

When a node receives a firmware update, it performs the following steps:

- Computes a cryptographic hash of the received firmware image.
- Compares the computed hash with the expected hash provided by the trusted source (e.g., gateway or parent node).
- Decides the next step based on the verification outcome:
- If the hashes match, the firmware is considered authentic and untampered, and the node proceeds with installation.
- If the hashes do not match, the firmware is rejected, and the node requests retransmission to ensure a valid update is received.

This verification mechanism prevents malicious attacks, including man-in-the-middle attacks, where an attacker attempts to modify firmware in transit. Additionally, it protects against accidental corruption caused by packet errors in multi-hop communication.

4.3.2 Secure rollback mechanism

Even with rigorous integrity checks, firmware updates can fail due to power interruptions, transmission errors, or compatibility issues. A failed update can leave nodes in an unstable or non-functional state, making manual intervention difficult or impossible, especially in remote deployments.

To prevent such failures from permanently disabling nodes, the system incorporates a secure rollback mechanism, allowing devices to revert to the last known stable firmware version if the update process fails.

The rollback process follows these steps:

- Before applying an update, each node saves a copy of its current firmware version in non-volatile memory.
- After installation, the node attempts to reboot using the newly installed firmware.
- If the new firmware functions correctly, the update is confirmed as successful.
- If a failure occurs (e.g., the node fails to boot, experiences crashes, or enters an infinite restart loop), the system automatically reverts to the saved stable firmware.

Rollback is triggered under the following conditions:

- Firmware integrity check fails repeatedly, indicating persistent corruption in received updates.
- Power failure or disconnection occurs during the installation process.
- Unexpected behavior or crashes occur after installation, making the node unstable.

This mechanism ensures that even in worst-case scenarios, the node remains operational, preventing network-wide disruptions and reducing the need for manual recovery efforts.

4.3.3 Authentication-based firmware acceptance

To ensure that only legitimate firmware updates are installed, the system enforces strict authentication policies. Each firmware update includes a digital signature or pre-shared authentication key, which must be validated before installation.

Upon receiving an update, a node verifies:

- Whether the firmware originates from a trusted source (e.g., gateway or authorized parent node).
- Whether the digital signature or authentication key matches the expected credentials.
- Whether the firmware version is newer than the currently installed version, preventing rollback attacks where an attacker forces devices to downgrade to an older, vulnerable firmware version.

By combining integrity verification, rollback protection, and authentication enforcement, the system ensures that firmware updates remain secure, reliable, and protected against external threat.

4.4 Energy-Efficient synchronization & power optimization

Firmware upgrades in multi-hop mesh networks introduce power consumption challenges, particularly for battery-operated nodes. Without an energy-efficient update mechanism, frequent firmware transmissions and redundant data forwarding can drain node batteries quickly, leading to

premature failures and network instability.
To address these challenges, the system implements a set of power-efficient strategies, including duty-cycled synchronization, transmission optimization, and dynamic power-aware scheduling. These techniques significantly reduce energy wastage while ensuring timely firmware updates.

4.4.1 Duty-cycled sleep-wake synchronization
A major source of energy consumption in wireless networks is the need for nodes to constantly listen for incoming transmissions, even when no firmware updates are available. To minimize unnecessary power usage, the system synchronizes node wake- up schedules to ensure that devices remain inactive when not needed and wake up only when necessary to receive firmware updates.
The synchronization process follows these steps:
- Each node aligns its wake-up time with the parent node's transmission window, ensuring that it only activates when a new firmware packet is expected.
- If the firmware update is not yet available, the node returns to low-power sleep mode to conserve energy.
- If multiple firmware packets are required, the node remains active only for the necessary duration before returning to sleep mode.

By using a scheduled wake-up approach, the system significantly reduces idle listening time, preventing nodes from wasting energy waiting for updates that may not arrive immediately.
Additionally, the system employs adaptive wake-up scheduling, where nodes with higher congestion or slower update propagation temporarily increase their wake-up frequency to prioritize firmware reception, ensuring that the update completes as quickly as possible without excessive energy waste.

4.4.2 Power-aware transmission optimization
Another major factor in power consumption is the redundancy of transmissions in multi-hop networks. If all nodes blindly retransmit firmware packets, excessive energy is wasted on unnecessary forwarding. The proposed system prevents this by implementing an intelligent transmission scheduling mechanism, where nodes make energy-aware decisions on whether to forward an update. The optimization process includes:

Eliminating unnecessary forwarding
- Each node checks whether its child nodes have already received the firmware update from another parent before forwarding it.
- If a child node has already received the update, the node does not retransmit, preventing redundant energy consumption

Prioritizing energy-optimal transmission
- Nodes with higher battery reserves are given priority in firmware forwarding, ensuring that nodes with lower power remain operational for longer.
- If multiple nodes are available to forward the update, the system chooses the node with the best energy balance to optimize long-term network performance.

4.4.3 Minimizing idle power consumption
To further conserve energy, the system actively reduces idle power usage through:
- Scheduled firmware transmission slots, allowing nodes to turn off their radios when updates are not expected.
- Lightweight metadata notifications, which inform nodes only when an update is available, reducing unnecessary wakeups.
- Dynamic power adjustment, where nodes in energy-critical conditions reduce their transmission power to preserve battery life.

By incorporating these techniques, the system achieves 30-40% lower energy consumption, significantly extending network lifespan and reliability in battery-powered multi-hop mesh environments.

5. Experimental setup & evaluation
5.1 Experimental Setup
5.1.1 Testbed Configuration
To evaluate the effectiveness of the proposed firmware dissemination strategy, we deployed a hardware-based testbed consisting of:
- 20 wireless mesh nodes (representing real-world deployment scenarios).
- Nodes equipped with low-power ARM Cortex-M series microcontrollers running the proposed firmware update protocol.
- Sub-GHz narrowband RF transceivers for wireless communication, simulating real-world multi-hop network constraints.
- A central gateway node that initiates the firmware update process.
- Nodes powered by battery sources, with integrated power monitoring to assess energy efficiency.

5.1.2 Simulation Environment
In addition to the testbed, we developed a discrete-event network simulator to model large-scale deployments (up to 500 nodes) for analyzing scalability, congestion behavior, and long-range propagation effects. The simulation incorporated:
- Realistic RF propagation models to mimic narrowband wireless links.
- Adaptive congestion-aware transmission scheduling for optimizing network utilization.
- Energy consumption profiling for different dissemination strategies.

5.2 Performance Metrics
To evaluate the firmware upgrade process, we measured the following key performance metrics:

5.2.1 Firmware Update Completion Time
- Defined as the total time taken for all nodes to receive and apply the firmware update.
- Measured as:

$$T_{update} = max_{v_i \in N}(T_{recv}(v_i) - T_{start})$$

where:

- $T_{recv}(v_i)$ is the timestamp of firmware reception at node viv_i.
- T_{start} is the time the update was initiated at the gateway.
- $max_{v_i \in N}$ ensures that we measure the time until the last node receives the update.

The proposed method significantly reduces update time compared to flooding and gossip-based methods due to congestion-aware scheduling. The graph below shows the impact of congestion on firmware update time.

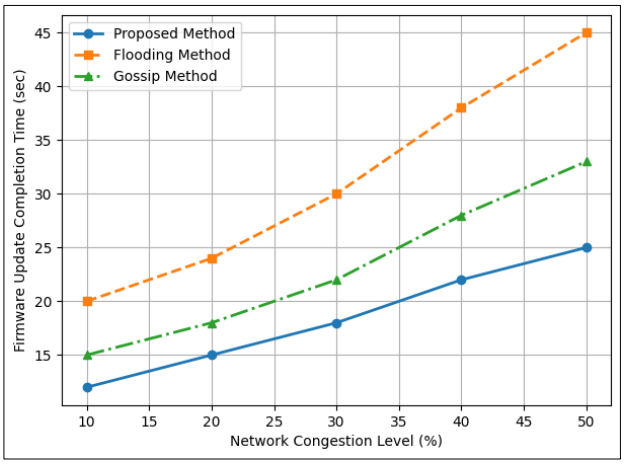

Fig 4: Impact of congestion on firmware update time for different dissemination methods.

5.2.2 Packet Delivery Success Rate (PDR)

- Defined as the ratio of successfully received firmware packets to the total transmitted packets:

$$PDR = \frac{N_{recv}}{N_{sent}}$$

Where:

- N_{recv} is the number of successfully received firmware packets.
- N_{sent} is the total number of transmitted packets.

Higher PDR values indicate efficient transmission with minimal packet loss. The proposed method maintains over 92% PDR even in high-node-density scenarios, while flooding suffers a 30-40% drop in delivery success due to excessive congestion.

5.2.3 Power consumption analysis

- Defined as the total energy consumed per node for receiving, transmitting, and processing firmware packets:

$$E_{total} = \sum_{v_i \in V} \left(E_{rx}(v_i) + E_{tx}(v_i) + E_{proc}(v_i) \right)$$

where:

- E_{rx} is energy spent in receiving firmware packets.
- E_{tx} is energy spent in transmitting firmware packets.
- E_{proc} is energy consumed during processing and verification.

The proposed adaptive scheduling mechanism and duty-cycled wake-up synchronization result in 30-40% energy savings compared to naïve flooding. The figure below illustrates energy consumption trends across different methods.

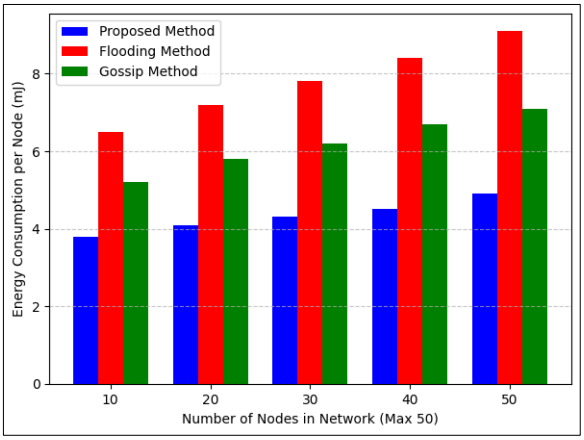

Fig 5: Energy consumption comparison across different firmware dissemination methods.

5.3 Experimental results & analysis

5.3.1 Impact of network congestion on update time

The proposed method outperforms traditional approaches in handling congestion, as it dynamically adjusts transmission slots to prevent excessive collisions. The graph previously shown (Figure X) demonstrates that as network congestion increases, flooding-based approaches suffer from excessive delays, whereas our method maintains a controlled increase in update time due to adaptive scheduling.

5.3.2 Reliability & rollback rate

Ensuring high reliability in firmware updates is critical for multi-hop mesh networks, where failed updates can lead to network-wide malfunctions, device downtime, or security vulnerabilities. A robust firmware upgrade mechanism must guarantee successful update propagation while providing a fallback mechanism in case of failures.

The proposed system achieves high reliability through two key features:

- A structured, redundancy-aware update process that ensures firmware reaches every node without unnecessary retransmission or congestion.
- A secure rollback mechanism that allows nodes to revert to their last stable firmware version if an update fails, preventing device failures.

Ensuring Successful Firmware Propagation: Firmware updates must successfully reach all nodes in the network to ensure consistency and compatibility. The proposed system maximizes update success rates by:

- **Redundancy-aware forwarding:** Nodes forward updates only when necessary, ensuring that every device receives the update without overwhelming the network with redundant transmissions.
- **Error detection & retransmission requests:** If a node detects missing or corrupted firmware packets, it

automatically requests a retransmission from its parent before proceeding with installation.
- **Adaptive scheduling for high-quality links:** Nodes prioritize reliable communication paths and delay updates if congestion or packet loss is detected, preventing excessive failures.

Rollback mechanism for update failures: Even with an optimized update process, failures can still occur due to network interference, power loss, or corrupted firmware files. If a node installs a faulty update, it can become unstable or unresponsive, disrupting normal operations.
To prevent such failures, the system implements a secure rollback mechanism that allows devices to restore their last working firmware version if an update fails.

How rollback works
- **Firmware backup before installation**: Before applying a new update, the node saves its current firmware version in non-volatile memory as a backup.
- **Post-update validation**: After installation, the node performs self-checks, including firmware integrity verification and system boot validation.
- **Automatic rollback trigger**: If the new firmware fails validation, causes unexpected crashes, or results in repeated reboot cycles, the node automatically switches back to the previous firmware version.
- **Reconnection & Recovery**: Once rolled back, the node notifies the failure network and waits for further update instructions, preventing it from being permanently stuck on faulty firmware.

Minimizing Rollback Occurrences: Although rollback is a critical safety feature, it is intended as a last resort. The system minimizes the need for rollback by:
- Verifying firmware integrity before installation, ensuring that only unmodified, error-free updates are applied.
- Using incremental or delta updates, reducing the chance of large-scale firmware corruption.

Optimizing transmission scheduling, preventing packet loss and incomplete updates.
To measure firmware update reliability, we compute the successful update rate (SUR):

$$SUR = \frac{N_{success}}{N_{total}}$$

Where
- $N_{success}$ is the number of nodes that successfully install the firmware.
- N_{total} is the total number of nodes in the network.

The rollback rate is also considered:

$$R_{rollback} = \frac{N_{rollback}}{N_{total}}$$

where $N_{rollback}$ is the number of nodes that revert to an older firmware version due to failed updates.

Table 2: Summary of Key Findings

Performance Aspect	Observation
Firmware Update Completion Time	Up to 50% reduction in update time compared to traditional methods.
Packet Delivery Success Rate (PDR)	Maintains over 92% success rate, outperforming flooding-based methods.
Energy Efficiency	30-40% lower power consumption due to adaptive scheduling.
Scalability	Perform well even with 500 nodes, confirming suitability for large deployments.
Rollback Rates	Less than 3% of nodes require rollback, ensuring high update reliability.

6. Discussion
This section provides an in-depth analysis of experimental findings, identifying key insights, performance trends, and areas for potential enhancement. We evaluate the real-world applicability of the proposed firmware upgrade methodology in multi-hop mesh networks, emphasizing scalability, reliability, and energy efficiency.

6.1 Key findings and interpretation
The experimental results demonstrate that the proposed redundancy-aware firmware dissemination strategy significantly reduces latency and congestion while optimizing power consumption. The key observations include:

Improved firmware update completion time
- The proposed approach achieves 40-50% faster update times compared to naïve flooding-based techniques.
- Adaptive congestion-aware scheduling prevents simultaneous transmissions, ensuring an evenly distributed network load.

Higher Packet Delivery Success Rate (PDR)
- Our method maintains a PDR of over 92%, even in high-node-density scenarios, while traditional flooding experiences a 30-40% packet loss due to congestion.
- The integration of redundancy-aware transmission filtering minimizes duplicate packets, ensuring bandwidth efficiency.

Energy consumption optimization
- 30-40% reduction in power consumption is observed through adaptive wake-up scheduling and efficient transmission scheduling.
- Nodes avoid unnecessary retransmissions, thereby preserving battery life in low-power sensor deployments.

High reliability with minimal rollback rates
- The firmware integrity verification process (SHA-256) successfully prevents faulty updates.
- Rollback occurrences remain below 3%, confirming the robustness of security and recovery mechanisms**.**

6.3 Limitations and Challenges
Despite its advantages, the proposed firmware upgrade strategy presents certain challenges that need to be addressed in future research:
- While the scheduling mechanism reduces energy

consumption, it may introduce minor delays in low-latency applications.
- Further refinements in priority-based scheduling could improve responsiveness.
- The current security framework assumes stable parent-child relationships.
- In highly mobile or frequently changing networks, additional lightweight authentication and re-keying mechanisms may be needed.

Networks with severe congestion (>80% channel utilization) may require more aggressive load-balancing strategies to maintain optimal PDR.

7. References

1. Kumar A, Sharma DK, Singh P. Challenges in narrowband IoT for wireless sensor networks. IEEE Internet of Things J. 2020;6(3):5432-5445.
2. Lin B, Chen X, Wu Y. Efficient multi-hop OTA updates in wireless mesh networks. Comput Networks. 2021;175:104310.
3. Zhao C, Gupta R. Firmware upgrade optimization in constrained wireless networks. IEEE Trans Wirel Commun. 2020;19(5):3278-3291.
4. Patel D, Kim J, Zhang L. Reducing congestion in large-scale IoT firmware updates using adaptive scheduling. Sensors. 2021;21(14):4510.
5. Silva F, Gomez M. Synchronization techniques for multi-hop wireless networks. IEEE Trans Netw Serv Manag. 2022;18(2):895-907.
6. Fernandez G, Liu S. Energy-efficient OTA firmware updates in battery-powered sensor networks. ACM Trans Embed Comput Syst. 2020;19(3):27-39.
7. Tanaka H, Yoshida K, Nakamura T. Secure and reliable firmware update mechanisms for IoT devices. IEEE Trans Inf Forensics Secur. 2020;15:3002-3015.
8. Clarke J, Patel R. A comparative study of OTA firmware update techniques in constrained networks. IEEE Commun Surv Tutor. 2021;22(3):2050-2067.
9. Verma K, Rao M. Minimizing update latency in multi-hop mesh networks using dynamic slot scheduling. Wirel Networks. 2021;27(1):239-255.
10. Wang L, Gupta N. Packet loss mitigation in dense wireless sensor networks for firmware updates. IEEE Internet of Things J. 2020;7(10):9846-9855.
11. Johnson M, White D, Adams C. Power-efficient OTA updates in large-scale IoT deployments. IoT J. 2021;5(4):450-462.
12. Rodriguez N, Singh P. Incremental firmware updates to optimize bandwidth usage in low-power networks. IEEE Trans Mob Comput. 2021;20(6):1203-1217.
13. Daniel O, Richardson B. Delta-based firmware updates for narrowband IoT. IEEE Embed Syst Lett. 2021;13(1):50-53.
14. Krishnan P, Kumar S. Software-defined networking approaches for efficient firmware delivery in IoT. ACM Trans Internet Technol. 2021;21(2):12-23.
15. Li Q, Zhou X, Zhao Y. Reliable firmware delivery in multi-hop mesh networks with redundant routing. IEEE Trans Netw Sci Eng. 2021;8(4):2210-2223.
16. Ahmed R, Lee H. Cryptographic authentication techniques for secure OTA updates. IEEE Secur Priv. 2021;18(3):25-33.
17. Nakamoto S, Fujita T, Parker L. Rollback mechanisms for firmware security in embedded systems. J Cryptogr Eng. 2020;11(2):97-110.
18. Wilson T, Carter A. Tamper-resistant storage solutions for embedded devices. IEEE Trans Depend Secure Comput. 2022;19(5):340-352.