





Article

Securing IoT: Mitigating Sybil Flood Attacks with Bloom Filters and Hash Chains

Iain Baird , Baraq Ghaleb , Isam Wadhaj *, Gordon Russell  and William J. Buchanan 

School of Computing, Engineering and the Built Environment, Edinburgh Napier University, Edinburgh EH10 5DT, UK; i.baird2@napier.ac.uk (I.B.); b.ghaleb@napier.ac.uk (B.G.); g.russell@napier.ac.uk (G.R.); b.buchanan@napier.ac.uk (W.J.B.)

* Correspondence: i.wadhaj@napier.ac.uk

Abstract: In the evolving landscape of the Internet of Things (IoT), ensuring the security and integrity of data transmission remains a paramount challenge. Routing Protocol for Low-Power and Lossy Networks (RPL) is commonly utilized in IoT networks to facilitate efficient data routing. However, RPL networks are susceptible to various security threats, with Sybil and flood attacks being particularly detrimental. Sybil attacks involve malicious nodes generating multiple fake identities to disrupt network operations, while flood attacks overwhelm network resources by inundating them with excessive traffic. This paper proposes a novel mitigation strategy leveraging Bloom filters and hash chains to enhance the security of RPL-based IoT networks against sybil and flood attacks. Extensive simulation and performance analysis demonstrate that this solution significantly reduces the impact of sybil and flood attacks while maintaining a low power consumption profile and low computational overhead.

Keywords: authentication; Bloom filter; hash chain



Citation: Baird, I.; Ghaleb, B.; Wadhaj, I.; Russell, G.; Buchanan, W.J.

Securing IoT: Mitigating Sybil Flood Attacks with Bloom Filters and Hash Chains. *Electronics* **2024**, *13*, 3467.

<https://doi.org/10.3390/electronics13173467>

Academic Editor: Zbigniew Kotulski

Received: 14 July 2024

Revised: 27 August 2024

Accepted: 29 August 2024

Published: 31 August 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The proliferation of the Internet of Things (IoT) has led to the integration of countless devices into networked environments, enabling unprecedented levels of connectivity and automation. However, this vast network of interconnected devices also introduces significant security challenges, particularly in the context of data transmission and network integrity. With the rapid expansion of the IoT and its deep integration into various aspects of society, these networks present a vast attack surface for malicious actors.

The limited capabilities of some devices might initially suggest a lower risk of data access due to restricted functionality [1]. However, it is crucial to recognize that even the most basic devices and networks require robust authentication and identity management for several key reasons encompassing both user-to-device and device-to-device communication.

Unfortunately, applying conventional authentication techniques such as passwords, certificates, biometrics, and two-factor authentication (2FA) to Internet of Things (IoT) settings presents numerous challenges. These challenges include insufficient computational resources for the complex operations involved, limited storage capacity on IoT devices, lack of precision in IoT biometric sensors, and difficulties in managing and distributing certificates across numerous devices [2,3].

The rapid growth of the IoT has coincided with a significant increase in the adoption of Routing Protocol for Low-Power and Lossy Networks (RPL) as the primary routing protocol for IoT networks [4,5]. While RPL may be the ideal choice for resource-constrained devices, its inherently trust-based model does not provide sufficient security guarantees in IoT environments, opening networks up to potential attacks such as sybil and flood attacks.

Sybil attacks pose a severe threat to network security in IoT environments [6]. In such attacks, a malicious actor creates a multitude of fake identities to disrupt critical processes

such as voting or resource allocation. Sybil attacks are particularly concerning in IoT contexts due to their large number of low-power devices with limited security capabilities [7]. These compromised identities can present a distorted view of the network topology, granting the attacker control over multiple nodes and potentially disrupting network operations. Furthermore, sybil attacks can be leveraged to launch Denial-of-Service (DoS) attacks by overwhelming the network with illegitimate traffic. These vulnerabilities highlight the critical need for robust authentication techniques to prevent unauthorized devices from impersonating legitimate ones and manipulating the network for malicious purposes.

Flood attacks, also known as Denial-of-Service attacks, aim to overwhelm a network or system with excessive traffic, rendering it unavailable to legitimate users [8]. The combination of a sybil attack and a flood attack is a particularly serious threat to IoT networks [9]. In this scenario, the attacker leverages the sybil attack to distribute the flood attack across a large number of seemingly legitimate nodes, making it extremely difficult to distinguish between legitimate and malicious traffic. This combined attack can severely degrade network performance, disrupt essential services and potentially compromising the overall security of IoT systems. Moreover, the weakened network defenses resulting from this attack create opportunities for further malicious activities, such as data breaches or unauthorized access attempts.

To address these challenges and ensure secure authentication in IoT environments, this paper proposes a novel authentication and flood mechanism integrating Bloom filters and hash chains. The proposed approach is evaluated through simulated performance testing to assess its effectiveness and efficiency, and the results demonstrate that this approach can provide a lightweight yet robust solution for securing IoT devices and networks by leveraging the space efficiency and rapid membership checking capabilities of Bloom filters alongside the tamper-evident properties of hash chains.

The rest of this paper is organized as follows: Section 2 establishes the foundation for the proposed approach. The section begins by introducing the RPL and its susceptibility to sybil and flooding attacks. Following this, we discuss two cryptographic constructs integral to our proposed solution, namely, Bloom filters and hash chains. Section 3 provides an overview of recent research efforts pertinent to our proposed solution while identifying their shortcomings. Section 4 describes our proposed solution, including its design principles and key features. Section 5 presents the results of our simulations, with a focus on evaluating the performance and effectiveness of the proposed solution. Finally, Section 6 summarizes the key findings and explores potential avenues for future research.

2. Background

This section establishes foundational knowledge essential for comprehending the proposed approach. It begins by introducing the Routing Protocol for Low Power and Lossy Networks (RPL), a critical component for routing data in resource-constrained environments. Following this, two cryptographic constructs are introduced that are crucial to the proposed solution, namely, Bloom filters and hash chains.

2.1. Overview of RPL

Routing Protocol for Low-Power and Lossy Networks (RPL) is a distance vector routing protocol specifically designed for IPv6 communication in Low-Power and Lossy Networks (LLNs) [5]. It facilitates the creation of a Destination-Oriented Directed Acyclic Graph (DODAG) by leveraging an objective function and a set of configurable parameters. This objective function guides the network in computing the 'best' path for data transmission, considering factors such as energy consumption, hop count, or other application-specific metrics.

RPL establishes a DODAG, a tree-like structure where all nodes have a single root, typically a border router or a collector node (LBR); routes are directed towards this root. This structure ensures loop-free routing and minimizes control overhead within the network. An example of a simple DODAG network can be seen in Figure 1.

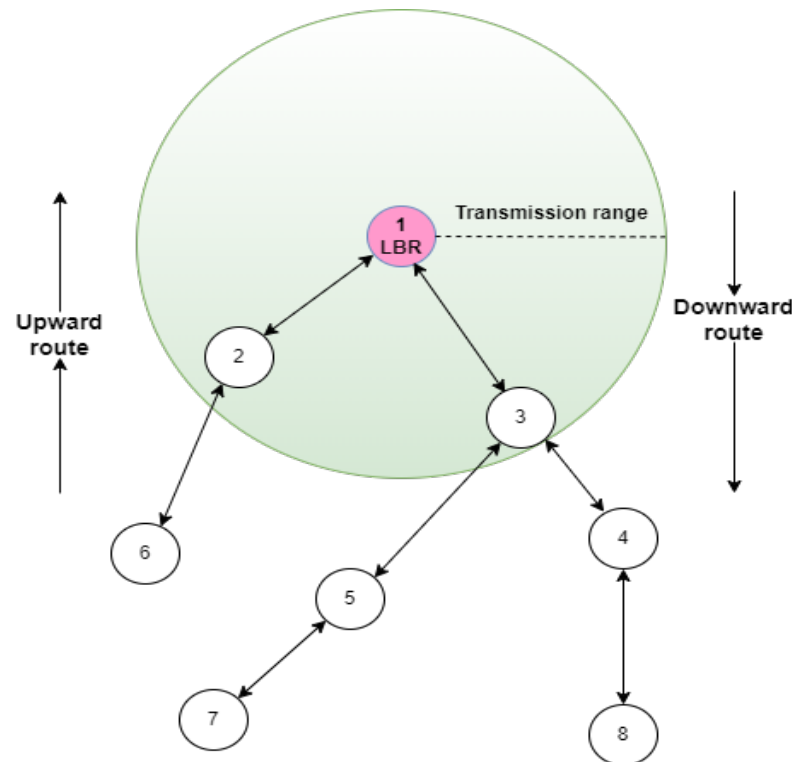


Figure 1. DODAG visualisation.

The RPL protocol utilises several control messages to create and maintain its topology. These messages, including DODAG Information Objects (DIOs) and Destination Advertisement Objects (DAOs), allow nodes to discover neighbouring nodes, learn about the DODAG structure, and advertise their own reachability towards the root. A third type of control message used in RPL is DODAG Information Solicitation (DIS). Nodes that have not yet joined the network and are seeking to establish themselves within the DODAG structure utilize DIS messages.

The RPL protocol leverages a key mechanism called the Trickle timer algorithm to optimize control message exchange and minimize energy consumption. This timer dynamically adjusts the interval between control message transmissions, balancing network responsiveness with energy efficiency. During normal operation, nodes gradually increase the interval between transmissions as the network stabilizes, reducing unnecessary control traffic and preserving battery life.

Despite its efficiency, RPL control messages can be abused by malicious actors [10,11]. One such vulnerability lies in the DIS message. Nodes that have not yet joined the network utilize DIS messages to discover existing DODAGs and request routing information. While this mechanism facilitates seamless node integration, it also presents an opportunity for Denial-of-Service (DoS) attacks [12].

A malicious node can exploit the network's discovery mechanism to launch a sybil-flood attack. This attack combines the creation of fake identities (sybil attack) with a flood of control messages (flood attack). Specifically, the attacker utilizes a large number of fake nodes in order to bombard the network with a continuous stream of DIS messages.

Upon receiving a DIS message, a neighboring node resets its Trickle timer to its fastest rate and immediately responds with a DIO message. This behavior, while intended to facilitate network discovery, can be manipulated by attackers. The constant barrage of DIS messages from the sybil nodes forces honest nodes to repeatedly reset their Trickle timers, leading to a surge in control message transmissions and a significant drain on network resources.

This excessive control traffic can overwhelm the network, causing several issues:

- Saturation: Communication channels become congested, hindering legitimate data transmission.
- Resource Depletion: Frequent control message exchange rapidly drains the finite battery resources of participating nodes.

The effectiveness of any flooding attack highlights the importance of incorporating security measures into RPL implementations. These measures can involve techniques for identifying and filtering out malicious messages, thereby protecting network integrity and ensuring efficient operation.

2.2. Bloom Filters

Bloom filters offer a space-efficient data structure that enables efficient query responses, making them ideal for scenarios where fast lookups are crucial [13]. This efficiency comes with a trade-off, however, in the possibility of false positives (FP), where a query element might be incorrectly identified as a member of the set. However, unlike false negatives (FN), which Bloom filters are guaranteed to avoid, false positives indicate a potential membership that needs further verification. FNs occur when a query element is erroneously reported as absent even though it is present in the dataset. This characteristic makes Bloom filters particularly valuable in scenarios where a certain level of FPs is tolerable or where secondary verification mechanisms can be employed to eliminate FPs post-filtering [14].

Due to their rapid query times and space-efficient nature, Bloom filters have found widespread application in various domains. These data structures are particularly advantageous for tasks such as weak password detection, unique username identification, and suspicious URL caching within social media platforms. Their ability to conserve storage space makes them well suited for applications where managing large datasets is crucial.

Figure 2 visually depicts the operation of a simple Bloom filter. Initially, all bits within the filter array are set to zero, representing an empty set. During the insertion stage, elements are added to the Bloom filter.

Each element is inserted as follows:

- Hashing: The element undergoes hashing using multiple hash functions, typically denoted as K . This process maps the element to a set of (K) hash values.
- Modulo Operation: Each hash value is then subjected to a modulo operation with the size M of the Bloom filter array. This ensures that the resulting index falls within the valid range of the array (0 to $M - 1$).
- Bit Setting: The corresponding bit positions in the Bloom filter array, determined by the modulo operation results, are set to one.

Consider the example of Entry 1 in Figure 2. The entry is hashed using two different hash functions. These hash values undergo a modulo operation by 12. Consequently, positions 4 and 8 within the Bloom filter array are set to 1.

During a search operation, the queried element undergoes hashing with the same functions used for insertion, generating multiple hash values. These values are subjected to a modulo operation to obtain valid indices within the filter array.

A bitwise AND operation is then performed between the corresponding bit positions in the array and a bit vector of ones. If the resulting vector is equal to the search vector (as exemplified by Search 2 in Figure 2), it indicates a potential match. However, due to the probabilistic nature of Bloom filters, there is a chance of a false positive (the element might not actually be present; see the example of Search 3 in Figure 2). Conversely, any unset bit reveals the element's definitive absence (guaranteed true negative), as demonstrated by Search 1 in Figure 2.

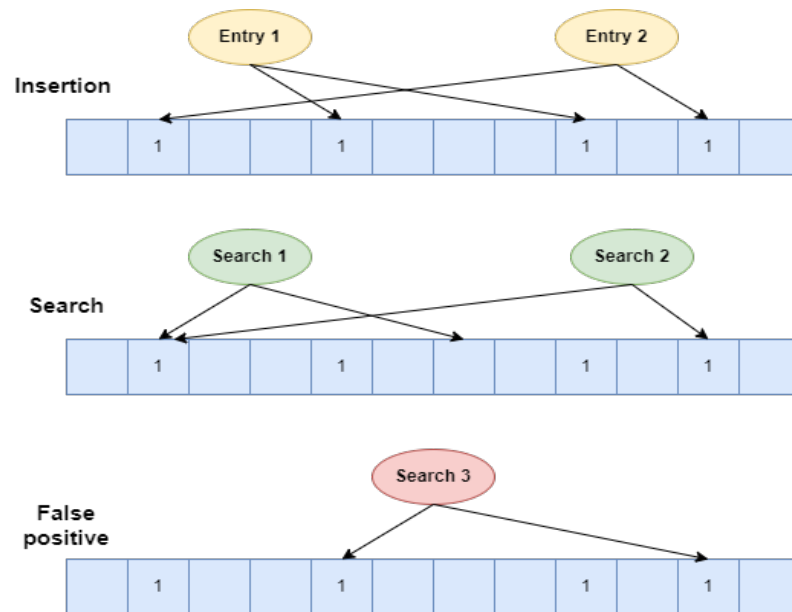


Figure 2. Simple Bloom filter operation.

The design and application of Bloom filters necessitates careful consideration of the False Positive Rate (FPR), as false positives can lead to unnecessary resource consumption, increased latency, and potentially compromised network security if not managed effectively [15]. To minimize the inherent tradeoff between query speed and accuracy, key parameters require thoughtful configuration. These parameters include:

- The size of the Bloom filter (N).
- The number of entries stored in the filter (M).
- The number of different hash functions employed (K).

The probability of encountering a false positive during a search operation can be calculated using the following formula:

$$FPR = \left(1 - \left(1 - \frac{1}{M}\right)^{KN}\right)^K. \quad (1)$$

The Formula (1) calculates the FPR by considering the interplay between the following key parameters:

- **Array size (N):** A larger array size reduces the probability of collisions during hash function application. Collisions occur when different elements map to the same bit position in the Bloom filter array. With larger N , there are more available bit positions, leading to a lower FPR.
- **Number of entries (M):** As the number of entries stored in the filter increases, the probability of encountering a false positive also rises. This is because a higher M increases the likelihood of multiple elements sharing the same hash values, leading to more collisions and a higher FPR.
- **Number of hash functions (K):** Employing a greater number of hash functions disperses elements across the Bloom filter array more effectively. Each hash function maps the element to a potentially different bit position. More hash functions lowers the chance of collisions, and consequently reduces the FPR.

By adjusting the values of M , K , and N , developers can achieve a desired balance between the FPR and the space efficiency requirements of a specific application. A larger N offers a lower FPR but requires more space; conversely, a smaller N offers faster lookups but may lead to a higher FPR. Similarly, increasing K lowers the FPR but requires more

computation during insertion and querying. Finding the optimal configuration depends on the specific application's priorities for space efficiency, query speed, and tolerable FPR.

While standard Bloom filters offer significant advantages in terms of query speed and space efficiency, a key limitation is their inability to delete entries after insertion. This characteristic necessitates careful consideration of the data being stored and the potential membership changes throughout the application's life cycle. To address this limitation, extensive research has been conducted on extending the functionality of Bloom filters to allow removals without sacrificing performance [16].

Furthermore, efforts have been directed towards enhancing the space-efficient nature of Bloom filters. These advancements include the exploration of alternative data structures such as 2D and 3D arrays to potentially improve storage utilization [17].

2.3. Hash Chain

A hash chain is a cryptographically secure sequence of values that can be used to verify data integrity and ensure chronological order. A hash chain is created by applying a cryptographic hash function to an initial input, known as the seed, then iteratively applying the hash function to the resulting hash value [18]. The inherent one-way property of cryptographic hash functions makes them computationally impossible to reverse. This prevents derivation of the original seed or any preceding hash values from a given hash value in the chain, even in the event of collisions (where different inputs produce the same hash output).

A hash chain is a cryptographically secure sequence of fixed-size values constructed using three elements:

- Number of Hashes (x): This parameter defines the length of the chain, specifying the number of times a cryptographic hash function will be iteratively applied.
- Hash Function (h): This element identifies the specific cryptographic hash function used to generate the chain. Common choices include SHA-256, MD5 (although its use is discouraged due to security weaknesses), and other functions with well-defined collision resistance properties.
- Seed (k): This is the initial input value that serves as the starting point for the chain. The seed can be of any data type suitable for the chosen hash function.

The hash chain construction can be formally defined as follows:

- The initial hash value h^x is obtained by applying the hash function h to the seed k :

$$h^x = h(k). \quad (2)$$

- Starting from the second hash value, the hash function is applied iteratively to the previous hash value in the chain. This generates the remaining hash values:

$$h^{x-1} = h(h^x), h^{x-2} = h(h^{x-1}), \dots, h^1 = h(h^2). \quad (3)$$

- The final hash value in the chain h^0 is designated as the root hash. It is derived by applying the hash function to the second-to-last hash value:

$$h^0 = h(h^1). \quad (4)$$

To strengthen the security of transmitted messages, the proposed method incorporates a novel approach utilizing hash chains to generate a sequence of one-time passwords. This mechanism offers several advantages over alternative techniques such as Merkle hash trees.

In addition to its security benefits, the use of hash chains offers potential advantages in terms of both communication and computational efficiency. The sequential nature of hash chains allows for more compact representation of message authentication information compared to Merkle trees, resulting in reduced message size and lower transmission costs. Moreover, the computational overhead associated with hash chain generation is

generally lower than that of Merkle trees, making hash chains particularly suitable for resource-constrained IoT environments [19].

While Merkle trees excel in certain applications, their suitability for generating sequential one-time passwords is limited. The inherent structure of Merkle trees does not naturally lend itself to the creation of ordered cryptographic values. Although adaptations could be made to accommodate this requirement, they would introduce additional complexity and potentially diminish the performance benefits often associated with Merkle trees [20]. By employing hash chains, our proposed system effectively addresses the critical challenge of message authentication while optimizing resource utilization in IoT networks.

3. Related Work

Ensuring secure authentication and defending against sybil and flooding attacks in resource-constrained IoT networks has been an active area of research in recent years. Researchers have explored various techniques and approaches to address these challenges while accounting for the computational and energy limitations of IoT devices. For instance, ref. [21] proposed liteSAD, a novel and secure mechanism for addressing sybil attacks in RPL-based IoT systems. liteSAD leverages Physical Unclonable Functions (PUFs) and a lightweight Bloom filter to enable distributed sybil attack detection, aiming to enhance security while maintaining efficiency compared to traditional methods that store complete node identities. The core of liteSAD involves the DODAG root generating a Bloom filter populated by hashing each legitimate node's identifier along with its unique PUF response. This Bloom filter is then distributed to all legitimate nodes through a new packet type called BF-DAO. Each node uses the filter to verify other nodes' PUF responses, enabling distributed sybil attack detection.

While liteSAD demonstrates promise, the paper lacks a comprehensive analysis of its scalability in large IoT networks. Specifically, the impact of network growth on the size and performance of the Bloom filter needs further exploration. Additionally, the trade-off between false positive rates and communication overhead as the network expands requires investigation. Finally, a detailed analysis of how the Bloom filter and PUF integration affects processing power, memory, and battery life of IoT devices is crucial for assessing the feasibility of liteSAD in resource-constrained environments.

The authors of [15] introduced a novel Bloom filter variant called enhanced Bloom filter (eBF) specifically designed to address intrusion detection challenges in resource-constrained IoT environments. Their proposed filter tackles the limitations of traditional Bloom filters regarding memory efficiency, processing speed, and accuracy in detecting malicious activities. A key innovation of the eBF lies in its two-dimensional structure. It utilizes two separate Bloom filters: the intrusion Bloom filter (iBF) stores intrusion data packet information, while the benign Bloom filter (bBF) stores benign data packet information. This design distinction helps to reduce false positives by ensuring that the same data packet is not present in both filters simultaneously. The system categorizes a packet as potentially benign, intrusion, or a false positive based on the responses from iBF and bBF. In cases where both Bloom filters return true for the same packet, indicating a potential false positive, the system employs a deep learning model for accurate classification. This model analyses the packet's features and determines whether it is an intrusion or benign. After classification by the deep learning model, the data packet is either stored in the iBF for future reference and blocking (if classified as an intrusion) or in the bBF for further processing (if classified as benign). This combined approach employing Bloom filters and deep learning intervention enables the system to effectively differentiate between malicious intrusion attempts and harmless activities in IoT devices, enhancing overall security and intrusion detection capabilities. While eBF presents a novel approach to enhancing intrusion detection in IoT through improved memory efficiency and accuracy, careful consideration of implementation issues and potential challenges is essential for successful integration into real-world IoT environments.

The study presented in [22] proposed a lightweight protocol for secure communication in smart homes. This protocol leverages a technique called a cumulative keyed-hash chain to achieve multiple security goals. One key feature is mutual authentication. The hash chain establishes a challenge–response mechanism that verifies the identity of both the device and the user attempting to communicate. This helps to prevent unauthorized access to the smart home system. The protocol utilizes a hash chain to fragment and verify the integrity of firmware updates for IoT devices. This ensures that the firmware remains unaltered during distribution. Additionally, the protocol utilizes the hash chain to dynamically update secret key values between communicating devices. This helps to prevent unauthorized access by ensuring that the shared secret is regularly refreshed.

While the cumulative hash chain shares some similarities with blockchain technology in terms of chaining hash values together, it is not considered a blockchain due to differences in the structure, decentralization, and consensus mechanism. The protocol instead focuses on chaining hash values together to create a tamper-evident record. This approach ensures data authenticity and prevents unauthorized modifications. Additionally, the secret key is updated in every session, further enhancing the security of the communication channel. By verifying the identities of communicating devices, ensuring the integrity of data transmissions, and preventing unauthorized access attempts through dynamic key updates, integration of the cumulative keyed-hash chain strengthens the security of the proposed protocol. The proposed solution, while effective, may not be suitable for deployment in current smart home environments due to the inherent resource limitations of many smart home devices, which often preclude the use of asymmetric cryptographic techniques.

In [23], the authors addressed the critical security challenge of replay attacks within IoT authentication schemes. Traditional defense mechanisms, which rely on timestamps or extensive nonce storage, are often impractical due to the resource constraints of IoT devices. Additionally, reliance on shared secrets such as session keys necessitates robust key management practices. These keys could be compromised if not securely stored or transmitted, undermining the entire authentication process. Thus, they introduced a novel lightweight user authentication scheme employing an improved challenge–response mechanism. The approach was designed to resist replay attacks without requiring time synchronization or large nonce lists, making it suitable for resource-limited IoT environments. By utilizing randomly generated nonces during authentication, the scheme ensures message freshness and mutual authentication between users and sensors, effectively mitigating the risk of replay attacks. A performance evaluation demonstrated that the scheme significantly outperformed existing methods in terms of storage and computational costs, making it efficient for IoT devices. Although communication overhead is reduced in certain interactions, it may increase slightly due to the absence of timestamps. However, the scheme's dependence on key management remains a critical consideration; any compromise in key storage or transmission could jeopardize the security of the entire system.

Overall, the above paper presents a secure and efficient authentication solution tailored to the unique challenges of the IoT environment while highlighting the essential need for robust key management to ensure the integrity of the authentication process.

While the primary focus of the current paper is on sybil flood defence, we acknowledge the importance of addressing other potential vulnerabilities such as replay attacks. To mitigate this risk, we have incorporated a timestamp mechanism into our proposed solution.

While a deeper discussion of replay attacks and their mitigation strategies is beyond the scope of this work, the included timestamp mechanism provides a practical and effective countermeasure. For additional context, readers may refer to the existing literature on replay attacks in IoT environments, such as [24,25].

4. Overview of the Proposed Solution

4.1. Bloom Filter Initialization

This section details the initial setup phase of the proposed mitigation method, focusing on the process of creating and distributing a Bloom filter for secure device enrollment within

the network. This Bloom filter plays a crucial role in efficiently authenticating devices and preventing unauthorized access.

- **Hash Chain Creation:** Each node generates a hash chain by repeatedly applying a cryptographic hash function to a chosen random number (x times). This process creates a chain of hash values, typically denoted as h^0 (root) to $h(k)$ (highest level). These hashed values are then stored locally on the device.
- **Root Hash Transmission:** Subsequently, each device transmits the root hash value (h^0) of its locally generated hash chain to the designated LBR. This root hash acts as a unique identifier for the device within the Bloom filter.
- **Bloom Filter Construction:** The LBR aggregates the root hash values from all connected devices and integrates them into the Bloom filter data structure. The method of integration varies depending on the specific Bloom filter implementation.
- **Bloom Filter Dissemination:** After the Bloom filter is constructed, the LBR broadcasts it to all devices within the network. Each device then stores the Bloom filter locally to enable future membership checks.

4.2. Device Authentication Process

The device authentication process is crucial for maintaining network security, particularly against blended attacks that combine sybil attacks (creating fake identities) with flood attacks (overwhelming the network). This process ensures the legitimacy of devices transmitting messages and prevents unauthorized or fabricated devices from disrupting the network. The verification algorithm employed in this process is outlined below:

Before transmitting a message, the device generates an Enhanced Message Authentication Tag (E-MAT) using a novel multistep cryptographic process, as follows:

- **Hash Chain Element Retrieval:** The device retrieves the next hash value from its precreated hash chain. This value is denoted as h^{y+1} , where y represents the index of the previous hash value used in a message (initially $y = 0$).
- **Timestamp Inclusion:** The device combines the retrieved hash value (h^{y+1}) with the current timestamp (t_s). This ensures the freshness of the message in an effort to prevent replay attacks.
- **Hash Function Application:** The device applies a cryptographic hash function

$$V = h(h^{y+1} || t_s) \quad (5)$$

to the combined data, creating a unique message authentication tag specific to the message and the current time.

- The final MAT and the chosen combination of elements form the message payload that is transmitted across the network; for instance, an example payload structure could be

$$h^0, V, (h^{y+1} || t_s).$$

4.3. Verification Process

The verification process, see Algorithm 1, begins with a check against a Bloom filter. The receiving node extracts the root hash h^0 from the incoming message and compares it to the entries within the Bloom filter. Because the Bloom filter is probabilistic, a positive result (presence of the root hash) suggests a high likelihood of the message originating from a legitimate device; however, this is not a definitive confirmation. If the Bloom filter indicates a potentially legitimate message, then the packet is allowed to proceed to further verification steps. Messages that do not pass the Bloom filter check are discarded to conserve resources.

Algorithm 1 Verification Process

```

1: procedure INITIALIZE:
2:   Set drop_counter = 0
3: procedure PACKET RECEPTION:
4:   Set drop_flag = 0
5:   Retrieve type from packet header.
6: procedure PACKET PROCESSING:
7:   if type ≠ 155 then
8:     Log Message arrived at tcpip_input
9:     Call Bloom filter
10: procedure RETURN FROM BLOOM FILTER:
11:  if drop_flag = 1 then
12:    Drop Packet
13:    drop_counter ++
14:    if drop_counter > 3 then
15:      Log Packet dropped more than 3 times.
16: procedure TIME CHECK:
17:  if drop_flag = 0 then
18:    Extract timestamp
19:    if timestamp ≠ current period then
20:      Drop Packet
21: procedure ROOT COMPARISON:
22:  Extract root
23:  Extract chain
24:  Set loop_counter = 0
25:  while loop_counter < x do ▷ x = # Iterations
26:    if hash(chain) == root then
27:      break
28:    else
29:      hash(chain) = chain
30:      loop_counter ++
31:  if loop_counter == x then
32:    Drop Packet

```

After the Bloom filter check, the receiving node extracts the timestamp t_s embedded within the message. This timestamp is compared against the current timestamp maintained by the node. If the comparison reveals that the message timestamp falls within an acceptable time window (not too old), the node can be confident that the message is fresh and not a replay attack attempting to resend an old message.

To ensure that the timestamp within the message has not been tampered with, the receiving node performs an additional verification. This involves combining the next hash value h^{y+1} from the message's chain with the included timestamp t_s . A cryptographic hash function is then applied to the combined data $h^{y+1}||t_s$ and the resulting hash value is compared to the message's V value. Any alteration to the timestamp would change the combined data, and consequently the resulting hash value.

As the final verification post-Bloom filter, the receiving node performs a final hash comparison check. This involves extracting the hash chain value h^{y+1} from the message's MAT. The node then iteratively hashes this value (no more than x) used by the sending device to create its original hash chain. After each hashing iteration, the resulting hash value is compared to the message's root hash h^0 . Because the root hash is created by iteratively hashing an initial value $h(k)$ a specific number of times x , any mismatch during this verification process indicates that the message has been tampered with.

The Bloom filter acts as the first line of defense, filtering out potentially illegitimate messages. Following this initial screening, a robust three-step verification process safe-

guards network integrity. This process effectively filters out messages that appear legitimate but are not (false positives) and thwarts attempts to reuse old messages (replay attacks). Any message failing even one step of this verification is deemed unreliable and potentially compromised. Consequently, such messages are dropped before reaching the LBR.

The network's integration of new nodes and removal of inactive ones are facilitated by leveraging the global repair mechanism in the RPL protocol at periodic intervals. This approach additionally reduces the number of stored hashes required by each node, leading to lower memory usage and potentially improved performance. However, the simplicity and time-efficiency of the global repair mechanism for node management come at a cost; because it is implemented by LBR and affects all network nodes, this method can be resource-intensive due to the increased control traffic it generates.

4.4. Rapid Attack Detection via Mitigation Failure Monitoring

A simple counter mechanism can be implemented to monitor the number of packets failing mitigation. This approach enables rapid attack detection by identifying nodes experiencing a significant increase in mitigation failures. As illustrated in Figure 3, a threshold can be set (e.g., three dropped packets in the target node, in this case Node 2) to trigger an alert, indicating a potential attack.

```
00:10.442 ID:3 DATA send to 1 'Hello 10'
00:10.584 ID:2 Message arrived in tcpip_input!
00:10.587 ID:2 Not in the bloom filter, dropping packet
00:10.589 ID:2 Packet dropped more than 3 times
```

Figure 3. Mote output showing that more than three packets have failed mitigation.

5. Performance Evaluation

To evaluate the effectiveness of our proposed mechanism against flooding attacks, a multistage simulation scenario replicating a Denial-of-Service (DoS) attack was designed. In the first stage, a single controlled attacker node launches a flooding attack against a designated target node. In the second stage, the attack is further escalated by introducing two attacker nodes. In both scenarios, the message transmission frequency is progressively increased.

Our defence mechanism leverages the hash chain root to populate a Bloom filter implemented on the target node. The root hash is then included within the legitimate message payload (see Figure 4). This approach enables efficient message filtering at the target node, allowing only messages with a valid hash (indicating pre-authenticated sources) or potential false positives to pass through the filter.

```
00:42.740 ID:2 Message sent to root number: 4
00:42.776 ID:1 DATA recv in message: 4
00:42.781 ID:1 Root: 4dc158c364378314f3bd609cfe:
00:42.786 ID:1 Chain: efc7f973aba3c82d2f0c8910:
```

Figure 4. Example of mote output showing legitimate traffic.

To model the possibility of false positives in the Bloom filter, random numbers were injected into the attacker node's message payload prior to implementation of post-filter verification, as shown in Figure 5.

This simulation serves two primary objectives. First, it assesses the target node's resilience against a DoS attack under increasing message loads. Second, it demonstrates the efficacy of the Bloom filter in blocking messages that do not originate from authenticated sources, thereby mitigating the impact of flood attacks. The Contiki operating system, a popular choice for resource-constrained wireless sensor networks (WSNs) programmed in C, served as the development platform for our proposed mechanism [26]. A 32-bit Bloom

filter was implemented and integrated with the SHA-256 hashing library directly on the target node. This approach facilitated efficient execution of the proposed mechanism within the resource-constrained environment of the sensor network. To simulate the resource limitations and communication characteristics of a low-power sensor network, the Cooja simulator was employed. The Cooja simulator was chosen due to its extensive support for various hardware platforms commonly used in WSN deployments, including TelosB, Z1, and Tmote Sky nodes. Additionally, Cooja offers a visual representation of the simulated network, which aids in understanding network behavior. Furthermore, it allows node output to be displayed, facilitating analysis of sensor data and communication patterns.

```
02:01.039 ID:3 DATA send to 1 'Hello 120'
02:01.103 ID:2 Message arrived in tcpip_input!
02:01.122 ID:1 DATA recv '39,99,120 ' from 3
02:01.124 ID:1 Root : 39, Chain : 99
```

Figure 5. Example of mote output showing false positive.

Figures 6 and 7 depict the simulated network topology; Node 1 represents the Link Border Router (LBR), which is responsible for routing traffic between the sensor network and the external network, Node 2 serves as the target node under attack, and Nodes 3 and 4 represent malicious attacker nodes attempting to disrupt network operations through a flood attack.

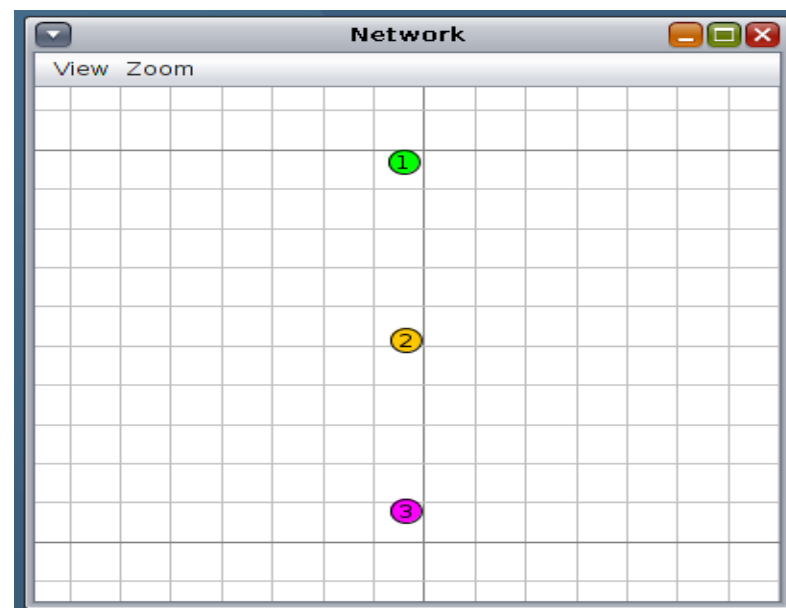


Figure 6. Scenario 1 DODAG visualisation: One attacker.

The impact of the proposed solution was evaluated using the Contiki operating system's energest module [27]. This module provides detailed power consumption data by tracking the active running time (in Contiki ticks) of the four key components through hardware timers: CPU active time (CPU), CPU idle time (LPM), radio transmission time (TX), and radio listening time (RX). By monitoring the duration of these states throughout the simulation, the power consumption for each component can be calculated using the formula presented in Equation (6). This allowed us to assess the overall power consumption of the target node under varying network conditions, such as when the attack intensifies in frequency both with and without the proposed mitigation techniques in place.

$$EnergyConsumed = \frac{Runtime \times Amps \times Volts}{Ticksperssecond} \quad (6)$$

Table 1 presents the current and voltage readings for the Tmote Sky nodes, along with the Rtimer counter values and various other simulation parameters. Rtimer, a real-time timer module in Contiki, provides the total number of timer ticks elapsed per second. In the Cooja simulator, individual nodes can periodically print their running tick count. In all the simulations, the target node was coded to print this value every 20 s, providing a snapshot of the ongoing component usage.

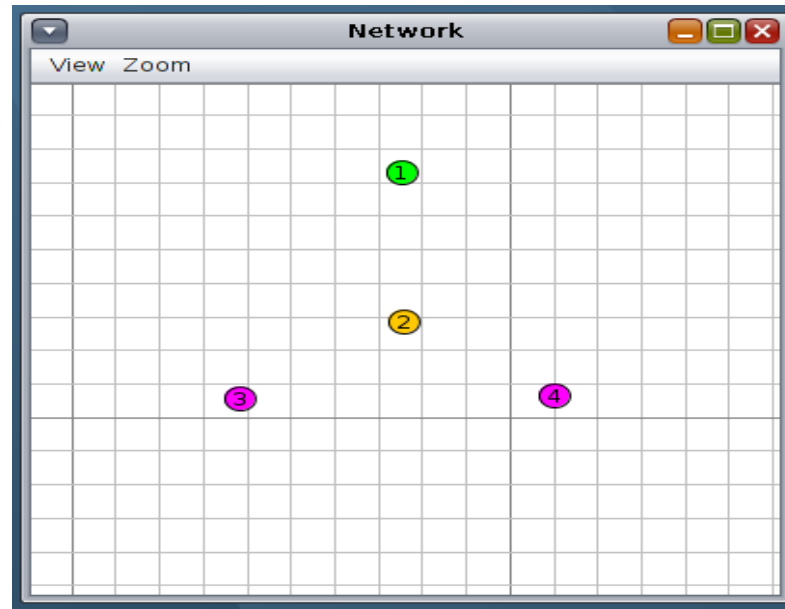


Figure 7. Scenario 2 DODAG visualisation: Two attackers.

Table 1. Simulation parameters.

Operating System	Contiki-OS	
Simulation Environment	Cooja	
Simulated Area	100 m × 100 m	
Power Analysis	Energest Time Function	
Sky Mote	RAM	10 k, 48 k Flash
	Voltage	3 V
	TX Current	17.4 mA
	RX Current	18.8 mA
	LPM Current	0.023 mA
	CPU Current	1.8 mA
	Transceiver	CC2420 2.4 Ghz (802.15.4)
RTimer	32,768 ticks per second	
Simulation Time	240 s	
Measurement Interval	20 s	
Node Transmission range	50 m	
Objective Function	MrHof	

The proposed solution was implemented on a target node within the sensor network. The evaluation process focused on monitoring the power consumption of three key node components within that node: CPU, radio transmission, and radio reception. Two additional key metrics were measured to assess the mitigation strategy’s effectiveness alongside the main metrics: the number of messages dropped due to the defence mechanism, and the number of false positive messages that passed through the Bloom filter.

5.1. False Positives vs. Dropped Messages

As previously described, the simulation incorporated a random number generator within the attacker node code to introduce false positives. This generator produced random numbers for both the root and chain elements, which were then added to the message payload. This approach stresses the Bloom filter's ability to differentiate between legitimate messages and unexpected data, beyond simply dropping packets with no payload. Table 2 presents the results.

Table 2. Dropped messages vs. false positives.

One Attacking Node Mitigated				
Message frequency every	8 s	4 s	2 s	1 s
Avg filtered messages dropped	28	56.8	116.8	232.8
Avg false positives	1	2.8	2.4	6
Two Attacking Nodes Mitigated				
Message frequency every	8 s	4 s	2 s	1 s
Avg filtered messages dropped	57	116	235	465
Avg false positives	1.6	2	4	13

Despite the use of a random seed to generate these attached numbers, the number of false positives across all simulation runs roughly doubled with two attacker nodes compared to one, with only one exception. However, the overall ratio between dropped messages and false positives remained below 5%. This difference, seen at the 4 s frequency, can be attributed to the combination of the random number generator and the random seed used for multiple simulation iterations.

5.2. Power Consumption

Power consumption is a paramount concern for LLNs due to their resource-constrained nature and reliance on battery power. Any attack that elevates the network's overall power demand directly translates to a reduced network lifetime.

5.2.1. Active CPU Power Consumption

The power consumption of the active CPU can be calculated using Equation (6) and the data presented in Table 1:

$$ActiveCPUpower = \frac{Runtime \times 1.8 \text{ mA} \times 3 \text{ V}}{32,768}. \quad (7)$$

Equation (7) was used to calculate the active CPU power consumption for both the single-node and two-node attack scenarios. This calculation was performed for three cases: the baseline, an unmitigated attack, and a mitigated attack. The results of this comparison are presented in Figures 8 and 9.

The proposed mitigation strategy demonstrates significant energy savings in terms of CPU activity compared to the unmitigated attack scenario with a single attacker node (Figure 8). On average, a 59% reduction in active CPU energy consumption was observed across the various message frequency increments. The effectiveness of the mitigation strategy diminished when the number of attacking nodes increased to two, as is clear from Figure 9. In this scenario, the average power savings in terms of active CPU energy were only 6.4%. This observation suggests that as the number of attacker nodes continues to rise, the mitigated scenario's active CPU power consumption might eventually surpass that of the unmitigated scenario.

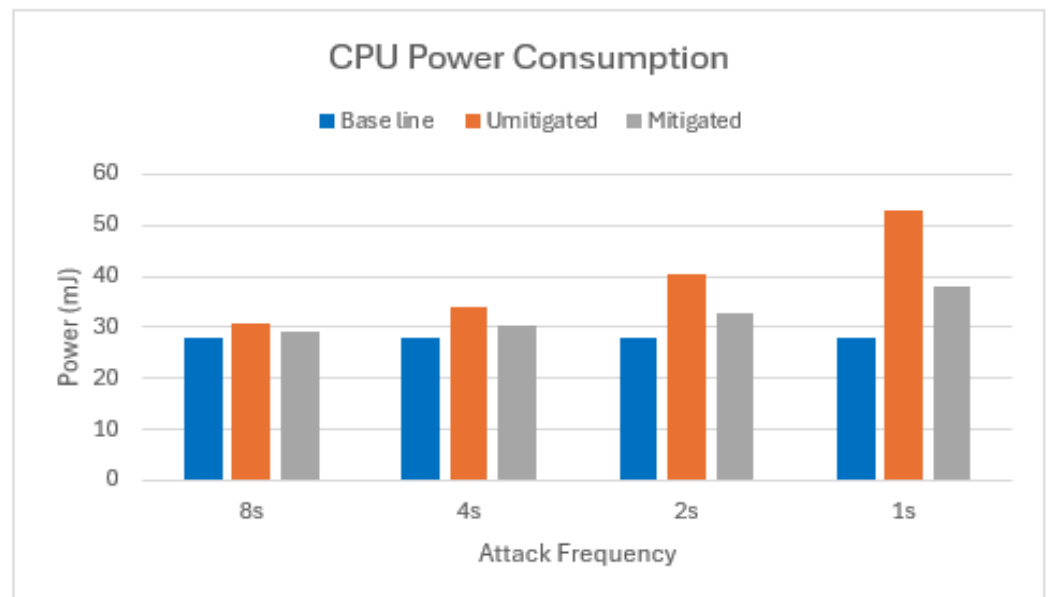


Figure 8. CPU power consumption of target node in scenario with one attacker.

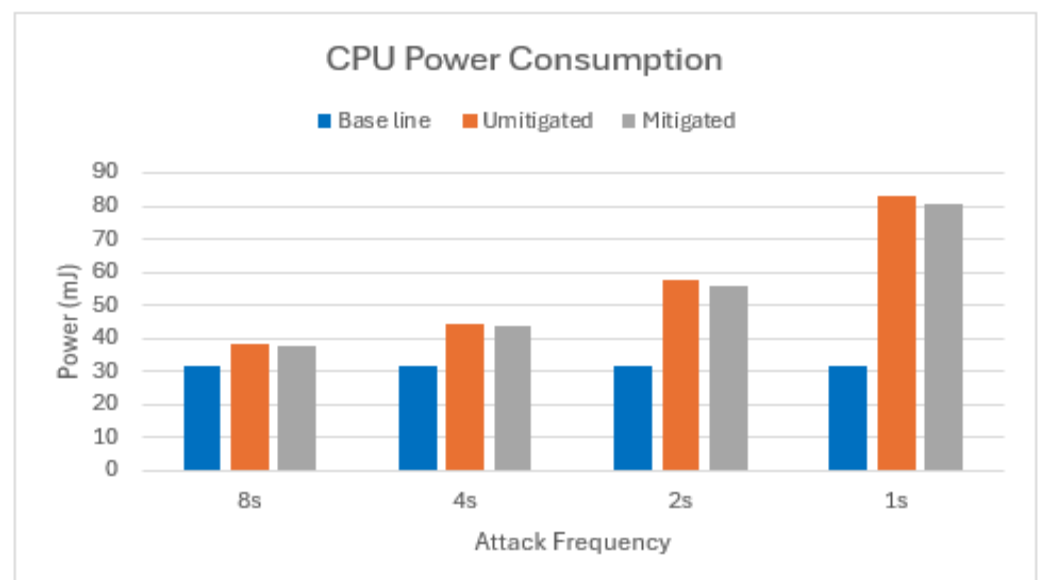


Figure 9. CPU power consumption of target node in scenario with two attackers.

5.2.2. TX Power Consumption

In energy-constrained IoT networks, minimizing radio transmission costs (TX costs) is crucial for maximizing node lifetime. Data transmission, especially frequent transmissions in large-scale deployments, significantly contributes to TX energy consumption. This is because intermediate nodes expend energy forwarding received packets. Therefore, mitigating the transmission of illegitimate packets can substantially extend network longevity.

As evidenced in Figures 10 and 11, our mitigation strategy effectively reduces the transmission of illegitimate packets by node 2. This is reflected in the near-identical TX energy consumption observed between the mitigated scenario and the baseline. This finding aligns with the observation that the Bloom filter of node 2 drops approximately 95% of the packets received from the attacker nodes, with the remaining 5% constituting false positives.

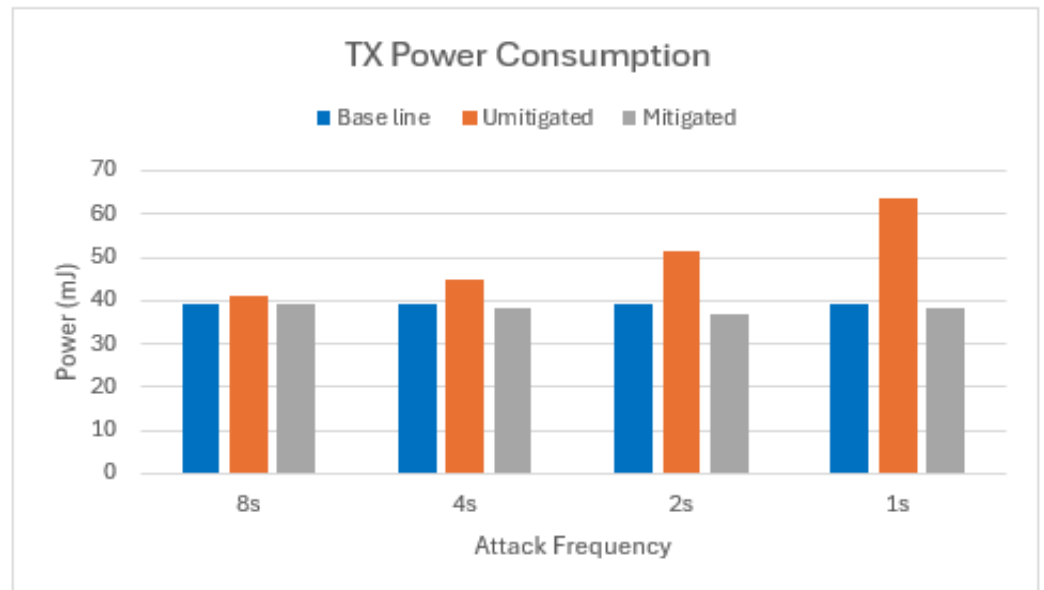


Figure 10. TX power consumption of target node in scenario with one attacker.

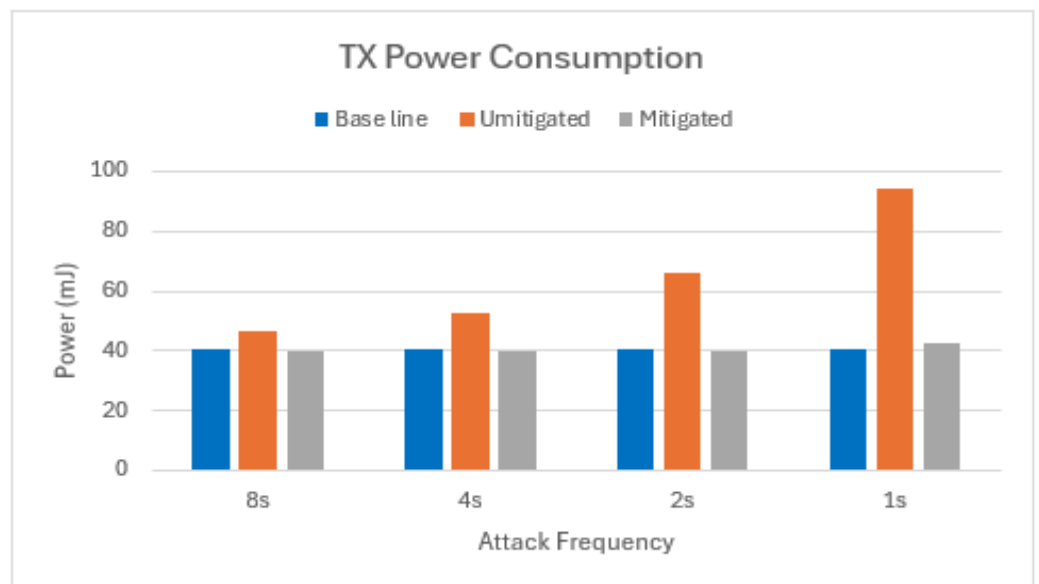


Figure 11. TX power consumption of target node in scenario with two attackers.

5.2.3. RX Power Consumption

While transmitting data (TX) involves short bursts, receiving data (RX) requires continuous listening, making it a major power drain compared to the energy saving sleep mode. This problem is amplified in nodes with high power consumption during listening, such as Tmote Sky. Their radio hardware consumes significantly more power while receiving, meaning that even brief listening periods significantly impact battery life. Because RX operations are often more frequent than TX ones, especially in protocols with frequent communication, high power consumption during listening dramatically reduces overall network lifetime. Nodes deplete their batteries faster, requiring more frequent maintenance or replacement. Therefore, reducing RX power consumption is crucial for extending network lifespan, particularly for nodes with high listening currents.

Our mitigation strategy successfully reduced the RX duty cycle (the time the radio spends in listen mode). In the worst-case scenario, an unmitigated two-node attack resulted in an average duty cycle of 2.75%. By implementing our mitigation technique, the duty

cycle dropped to 1.93%, translating to a significant energy savings of 49% on average in the scenario with one attacker, as shown in Figure 12.

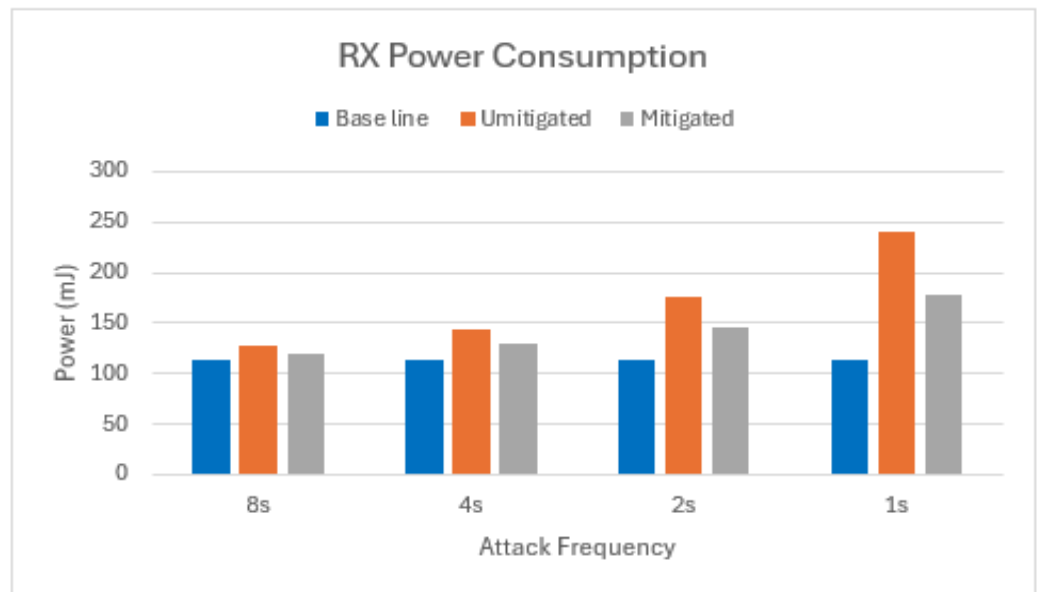


Figure 12. RX power consumption of target node in scenario with one attacker.

Figure 13 reveals a similar trend in the scenario with two attackers. Our mitigation strategy achieves an average of 46% energy savings in terms of RX energy consumption.

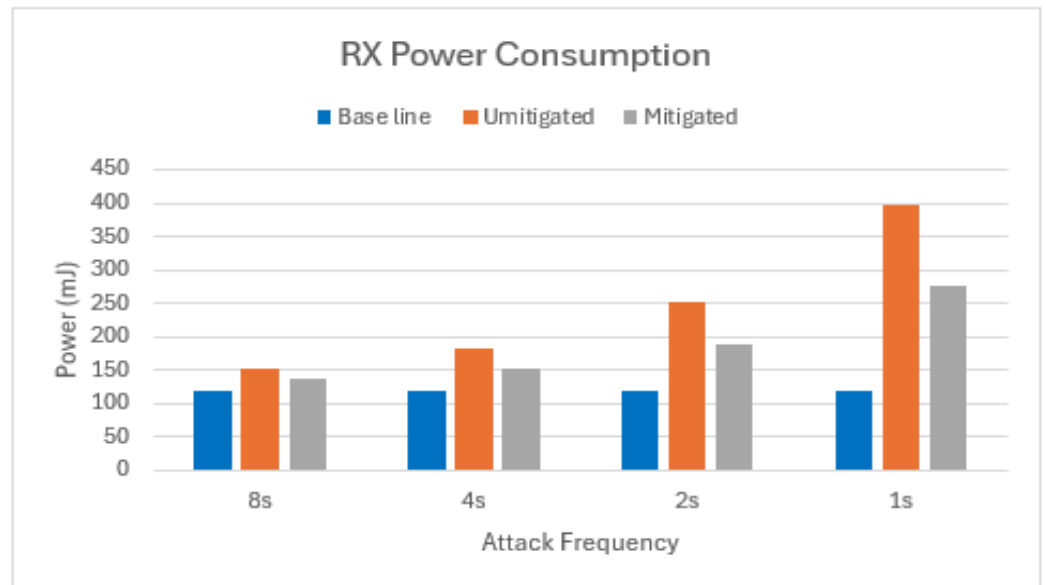


Figure 13. RX power consumption of target node in scenario with two attackers.

5.2.4. Total Power Consumption

Our mitigation strategy demonstrates significant energy savings when a single attacker node is present. On average, the total energy consumption across all scenarios is reduced by 58% compared to an unmitigated attack, as shown in Figure 14.

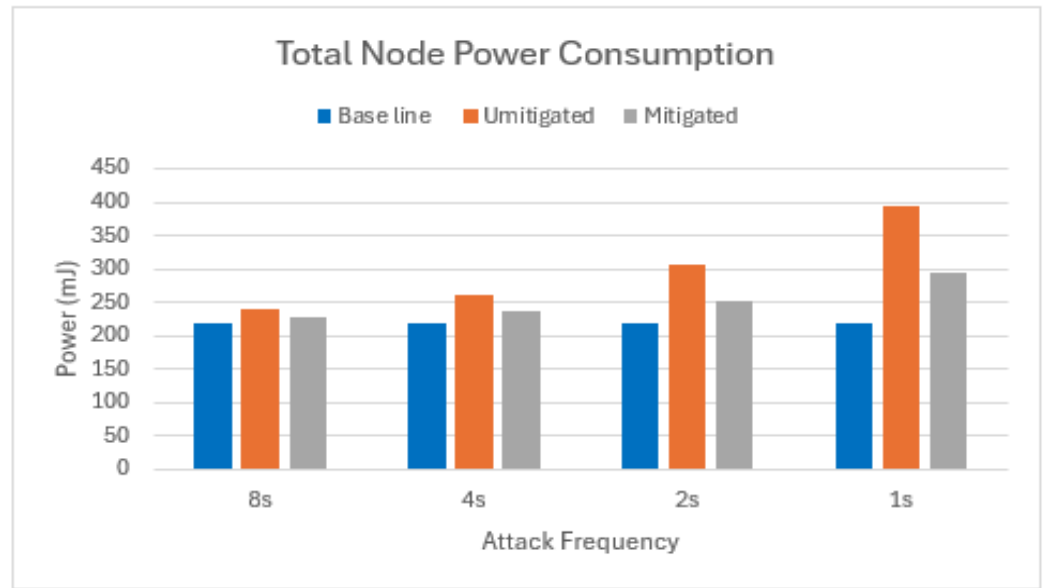


Figure 14. Total power consumption of target node in scenario with one attacker.

However, the effectiveness diminishes when the number of attackers increases. With two attacker nodes, the average total energy savings drop to 49%, as shown in Figure 15.

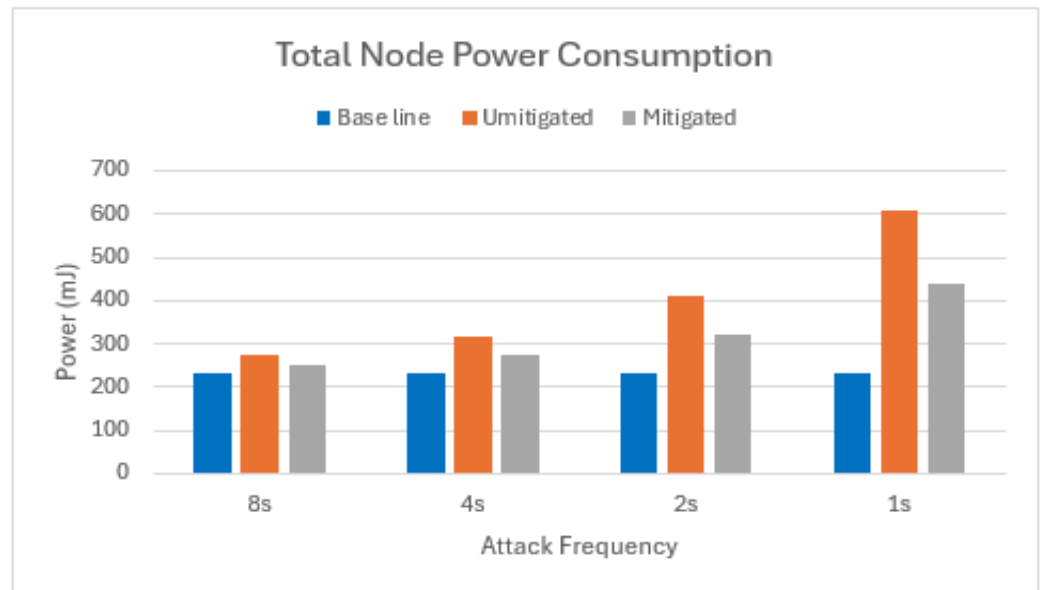


Figure 15. Total power consumption of target node in scenario with two attackers.

This decrease can be primarily attributed to the increased cost of active CPU power consumption required to process and filter out a larger volume of malicious messages from the additional attackers, as already discussed in Section 5.2.1 regarding the active CPU power consumption.

5.3. Informal Security Analysis

The following informal discussion is provided to assess the resilience of the proposed mitigation considering potential additional attack vectors and their potential impact on the system’s security.

5.3.1. First Pre-Image Attack

One potential attack vector against the proposed system involves capturing a hash from the hash chain and attempting to reverse-engineer the seed used to generate the chain. This attack aims to compromise the security of the system by gaining access to all messages associated with the node.

To mitigate this threat, the system employs several defence mechanisms. First, the use of a strong cryptographic hashing algorithm such as SHA-256 significantly increases the computational complexity of reversing the hash function. Second, ensuring that the seed generation process produces high-entropy random values reduces the likelihood of an attacker successfully guessing or brute-forcing the seed. The combination of these defence mechanisms provides a robust level of security, making it difficult for an attacker to reverse-engineer the seed and compromise the system.

5.3.2. Replay Attack

A replay attack occurs when an attacker intercepts a legitimate message, re-sends it at a later time, and attempts to deceive the recipient into processing it as if it were a new message. To mitigate replay attacks, the proposed system employs a timestamp mechanism. Each message is assigned a unique timestamp, which is verified by the recipient upon receipt. If the timestamp is older than a predefined threshold, the message is rejected as a potential replay attack.

One potential limitation of the proposed solution is the possibility of replay attacks being launched against other nodes within the network if the attack is executed with sufficient speed. To mitigate this risk, the time window for accepting messages can be adjusted and fine-tuned to reduce the likelihood of successful replay attacks.

5.3.3. Eavesdropping Attack

While the proposed solution effectively addresses sybil flood attacks, it is essential to consider the potential threat of eavesdropping attacks. Eavesdroppers can intercept network traffic to gain unauthorized access to sensitive data, compromising the privacy and security of IoT devices. However, even if an attacker were to intercept a message and obtain its contents, they would be unable to fabricate a valid response from the same node. The hash chain mechanism ensures that each message includes a unique one-time password, making it infeasible for an attacker to generate a legitimate message without possessing the corresponding secret key.

5.3.4. Routing Table Falsification Attack

A common attack vector against RPL-based IoT networks involves routing table falsification. In this type of attack, a malicious node intentionally misleads other nodes by advertising routes to nonexistent or unreachable destinations. By disseminating fake DAO control messages to the parent node, the attacker can induce the parent to store incorrect routing information, potentially disrupting network communication and leading to security vulnerabilities.

To enhance the resilience of RPL-based networks against routing table falsification attacks, nodes can implement a verification mechanism that requires child nodes to provide the root hash of their sub-DODAGs. This root hash serves as a cryptographic fingerprint of the sub-DODAG's topology and membership. By comparing the received root hash against existing Bloom filter entries, parent nodes can validate the legitimacy of the advertised route before storing it in their routing tables. This additional layer of security can significantly reduce the effectiveness of falsification attacks, as malicious nodes would need to compromise multiple nodes and manipulate their sub-DODAGs in order to create a convincing fake route.

6. Conclusions

This paper introduces a novel lightweight security framework designed to address the prevalent challenges posed by sybil and flooding attacks in resource-constrained IoT networks. By leveraging the strengths of hash chains and Bloom filters, we propose a multilayered defense mechanism that effectively filters malicious traffic while ensuring both device authenticity and message integrity. Our approach is meticulously tailored to the unique constraints of Low-power and Lossy Networks (LLNs), making it highly suitable for a broad spectrum of IoT applications where efficiency and security are paramount.

Simulation results within the Contiki operating system underscore the efficacy of our proposed solution in mitigating these critical security threats, demonstrating its potential to significantly enhance the resilience of IoT deployments. The lightweight nature of our framework ensures that it can be seamlessly integrated into existing IoT infrastructures without imposing substantial computational or energy overhead, thereby preserving the operational longevity of resource-limited devices.

Overall, the proposed framework offers a robust and adaptable foundation for improving the security and reliability of IoT networks. By addressing key vulnerabilities through a combination of innovative techniques, this research lays the groundwork for further exploration and refinement of security solutions tailored to the evolving needs of IoT ecosystems. Future research building on this work could lead to even more comprehensive and scalable security strategies, ultimately contributing to the widespread adoption of secure and efficient IoT technologies.

Future Work

Future research should prioritize the complementary integration of the proposed system with existing flood mitigation approaches, such as those employed by FloodDefender [28]. Combining the strengths of multiple techniques could potentially achieve more robust and efficient flood prevention. Additionally, optimizing the Bloom filter parameters through adaptive mechanisms and exploring LLN-specific Bloom filter variants offers promising avenues for further enhancing system performance and accuracy.

Another crucial area for future research is to investigate the performance of Bloom filters as the network size increases. As IoT networks scale, the number of nodes and volume of traffic can significantly impact the effectiveness of Bloom filters, potentially leading to higher false positive rates and increased computational demands. Understanding these scalability challenges is essential for optimizing Bloom filter-based security mechanisms in large-scale IoT deployments. By analyzing how Bloom filters perform under varying network sizes and conditions, future work can contribute to the development of more robust and scalable security solutions that maintain efficiency and accuracy even as IoT networks expand.

Another promising avenue for future research involves developing a comprehensive defense strategy against DIS flooding attacks by combining Bloom filtering with other robust countermeasures, such as rate limiting, anomaly detection algorithms, and machine learning-based approaches. A rigorous evaluation of this integrated strategy is essential in order to assess its performance under diverse conditions. Key metrics such as packet delivery ratio, network latency, energy consumption, and computational overhead should be employed to quantify the solution's impact. Moreover, the evaluation should encompass a variety of network topologies to accurately reflect real-world challenges, including both indoor and outdoor environments. Crucially, hardware constraints and environmental factors must be considered in order to assess the solution's practical feasibility and robustness. By conducting comprehensive testing under real-world conditions, we aim to demonstrate the applicability and effectiveness of the proposed solution in practical IoT deployments.

Author Contributions: Conceptualization, I.B.; Methodology, I.B.; Software, I.B.; Validation, I.B.; Investigation, I.B.; Resources, B.G.; Data curation, I.B.; Writing—original draft, I.B.; Writing—review

& editing, B.G., I.W., G.R. and W.J.B.; Supervision, B.G. and I.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Edinburgh Napier University Research Starter Grants.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

2FA	Two-Factor Authentication
BF	Bloom Filter
bBF	Benign Bloom Filter
CPU	CPU Active Time
DoS	Denial-of-Service
DIO	DODAG Information Object
DAO	Destination Advertisement Object
DIS	DODAG Information Solicitation
DODAG	Destination-Oriented Directed Acyclic Graph
eBF	Enhanced Bloom Filter
h^0	Root Hash
IBF	Intrusion Bloom Filter
IoT	Internet of Things
LBR	Local Border Router
LLN	Low-Power and Lossy Networks
LPM	Low Power Mode
mJ	Millijoules
PUF	Physical Unclonable Function
RPL	Routing Protocol for Low-Power and Lossy Networks
RX	Radio Listening Time
t_s	Timestamp
TX	Radio Transmission time
UDP	User Datagram Protocol
WSN	Wireless Sensor Network

References

- Mamdouh, M.; Awad, A.I.; Khalaf, A.A.; Hamed, H.F. Authentication and identity management of iot devices: Achievements, challenges, and future directions. *Comput. Secur.* **2021**, *111*, 102491. [CrossRef]
- Nižetić, S.; Šolić, P.; Gonzalez-De, D.L.; Patrono, L. Internet of things (IoT): Opportunities, issues and challenges towards a smart and sustainable future. *J. Clean. Prod.* **2020**, *274*, 122877. [CrossRef] [PubMed]
- Azrou, M.; Mabrouki, J.; Guezzaz, A.; Kanwal, A. Internet of things security: Challenges and key issues. *Secur. Commun. Netw.* **2021**, *2021*, 5533843. [CrossRef]
- Vasseur, J.; Agarwal, N.; Hui, J.; Shelby, Z.; Bertrand, P.; Chauvenet, C. Rpl: The ip routing protocol designed for low power and lossy networks. *Internet Protoc. Smart Objects (IPSO) Alliance* **2011**, *36*, 1–20.
- Winter, T.; Thubert, P.; Brandt, A.; Hui, J.; Kelsey, R.; Levis, P.; Pister, K.; Struik, R.; Vasseur, J.P.; Alexander, R. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. *RFC 6550*; RFC Editor, March 2012. Available online: <https://www.rfc-editor.org/info/rfc6550> (accessed on 9 January 2024)
- Newsome, J.; Shi, E.; Song, D.; Perrig, A. The sybil attack in sensor networks: Analysis & defenses. In Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks 2004, Berkeley, CA, USA, 26–27 April 2004; pp. 259–268. [CrossRef]
- Pu, C. Sybil attack in rpl-based internet of things: Analysis and defenses. *IEEE Internet Things J.* **2020**, *7*, 4937–4949. [CrossRef]
- Ghaleb, B.; Al-Dubai, A.; Ekonomou, E.; Qasem, M.; Romdhani, I.; Mackenzie, L. Addressing the dao insider attack in RPL's internet of things networks. *IEEE Commun. Lett.* **2018**, *23*, 68–71. [CrossRef]

9. Rajasekar, V.; Rajkumar, S. A study on impact of dis flooding attack on rpl-based 6lowpan network. *Microprocess. Microsyst.* **2022**, *94*, 104675. Available online: <https://www.sciencedirect.com/science/article/pii/S0141933122002058> (accessed on 12 February 2024). [[CrossRef](#)]
10. Pongle, P.; Chavan, G. A survey: Attacks on rpl and 6lowpan in iot. In Proceedings of the 2015 International Conference on Pervasive Computing (ICPC), Pune, India, 8–10 January 2015; pp. 1–6.
11. Bang, A.O.; Rao, U.P.; Kaliyar, P.; Conti, M. Assessment of routing attacks and mitigation techniques with RPL control messages: A survey. *ACM Comput. Surv. (CSUR)* **2022**, *55*, 1–36. [[CrossRef](#)]
12. Dhingra, A.; Sindhu, V. A review of dis-flooding attacks in RPL based iot network. In Proceedings of the 2022 International Conference on Communication, Computing and Internet of Things (IC3IoT), Chennai, India, 10–11 March 2022; pp. 1–8.
13. Bloom, B.H. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM* **1970**, *13*, 422–426. [[CrossRef](#)]
14. Kiss, S.Z.; Hosszu, É.; Tapolcai, J.; Rónyai, L.; Rottenstreich, O. Bloom filter with a false positive free zone. *IEEE Trans. Netw. Serv. Manag.* **2021**, *18*, 2334–2349. [[CrossRef](#)]
15. Gebretsadik, F.G.; Nayak, S.; Patgiri, R. eBF: An enhanced bloom filter for intrusion detection in IoT. *J. Big Data* **2023**, *10*, 102. [[CrossRef](#)]
16. Fan, B.; Andersen, D.G.; Kaminsky, M.; Mitzenmacher, M.D. Cuckoo filter: Practically better than Bloom. In Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies, Sydney, Australia, 2–5 December 2014; pp. 75–88.
17. Patgiri, R.; Nayak, S.; Borgohain, S.K. Passdb: A password database with strict privacy protocol using 3d bloom filter. *Inf. Sci.* **2020**, *539*, 157–176. [[CrossRef](#)]
18. Huang, Q.; Huang, H.; Wen-ming, W.; Li, Q.; Wu, Y. An authentication scheme based on novel construction of hash chains for smart mobile devices. *Wirel. Commun. Mob. Comput.* **2020**, *2020*, 8888679. [[CrossRef](#)]
19. Pinto, A.; Costa, R. Hash-chain based authentication for IoT devices and REST web-services. In Proceedings of the International Symposium on Ambient Intelligence, Seville, Spain, 1–3 June 2016; pp. 189–196.
20. Saldamli, G.; Ertaul, L.; Kodirangaiah, B. Post-Quantum Cryptography on IoT: Merkle’s Tree Authentication. In Proceedings of the International Conference on Wireless Networks (ICWN), Las Vegas, NV, USA, 30 July–2 August 2018; pp. 35–41.
21. Pu, C.; Choo, K.-K.R. Lightweight sybil attack detection in iot based on bloom filter and physical unclonable function. *Comput. Secur.* **2022**, *113*, 102541. [[CrossRef](#)]
22. Alshahrani, M.; Traore, I. Secure mutual authentication and automated access control for iot smart home using cumulative keyed-hash chain. *J. Inf. Secur. Appl.* **2019**, *45*, 156–175. [[CrossRef](#)]
23. Feng, Y.; Wang, W.; Weng, Y.; Zhang, H. A replay-attack resistant authentication scheme for the Internet of Things. In Proceedings of the 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), Guangzhou, China, 21–24 July 2017; Volume 1, pp. 541–547.
24. Lazzaro, S.; Angelis, V.D.; Mandalari, A.M.; Buccafurri, F. Is your kettle smarter than a hacker? A scalable tool for assessing replay attack vulnerabilities on consumer IoT devices. In Proceedings of the 2024 IEEE International Conference on Pervasive Computing and Communications (PerCom), Biarritz, France, 11–15 March 2024; pp. 114–124.
25. Rango, F.D.; Potrino, G.; Tropea, M.; Fazio, P. Energy-aware dynamic Internet of Things security system based on Elliptic Curve Cryptography and Message Queue Telemetry Transport protocol for mitigating replay attacks. *Pervasive Mob. Comput.* **2020**, *61*, 101105. [[CrossRef](#)]
26. Simha, S.; Mathew, R.; Sahoo, S.; Biradar, R.C. A review of rpl protocol using contiki operating system. In Proceedings of the 2020 4th International Conference Trends Electronics and Informatics (ICOEI), Tirunelveli, India, 15–17 June 2020; pp. 259–264.
27. Sabovic, A.; Delgado, C.; Bauwens, J.; Poorter, E.D.; Famaey, J. Accurate online energy consumption estimation of iot devices using energest. In Proceedings of the Advances on Broad-Band Wireless Computing, Communication and Applications: Proceedings of the 14th International Conference on Broad-Band Wireless Computing, Communication and Applications (BWCCA-2019), Antwerp, Belgium, 7–9 November 2019; pp. 363–373.
28. Shang, G.; Zhe, P.; Bin, X.; Aiqun, H.; Kui, R. FloodDefender: Protecting data and control plane resources under SDN-aimed DoS attacks. In Proceedings of the IEEE INFOCOM 2017—IEEE Conference on Computer Communications, Atlanta, GA, USA, 1–4 May 2017; pp. 1–9.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.