Safeguarding IoT Systems: Novel Authentication Method to Counteract Sybil and Flood Attacks

Iain Baird
Iain.Baird@Napier.ac.uk
Edinburgh Napier University
Edinburgh, Midlothian, UK

Baraq Ghaleb
Edinburgh Napier University
Edinburgh, UK
b.ghaleb@napier.ac.uk

Isam Wadhaj
Edinburgh Napier University
Edinburgh, UK
I.Wadhaj@napier.ac.uk

Gordon Russell
Edinburgh Napier University
Edinburgh, UK
g.russell@napier.ac.uk

William J. Buchannan
Edinburgh Napier University
Edinburgh, UK
b.buchannan@napier.ac.uk

ABSTRACT

This paper introduces an innovative strategy for countering Sybil and DODAG Information Solicitation (DIS) flood attacks within lightweight Internet of Things (IoT) networks. The proposed method combines a one-way hash chain with a Bloom filter, leveraging the probabilistic and space-efficient attributes of Bloom filters for swift query responses and efficient identity verification across extensive data sets. Sybil attacks, characterized by the creation of multiple counterfeit identities to gain control over a network, and flood attacks, which overwhelm a network with excessive traffic, are significant threats addressed by this approach. Particularly in Routing Protocol for Low Power and Lossy Networks (RPL), Sybil attacks pose a challenge to threshold-based protection against flood attacks by frequently altering node identities. To mitigate these risks, the suggested solution advocates for each node in the network to generate a one-way series of hashes. The root of this hash chain is then employed for identifying and filtering legitimate traffic using the Bloom filter. This method aims to enhance security in lightweight IoT networks by thwarting Sybil and flood attacks through a combination of one-way hash chains and Bloom filters.

CCS CONCEPTS

• **Security and privacy → Denial-of-service attacks**.

KEYWORDS

Hash-chain, Bloom filter, Authentication.

ACM Reference Format:
Iain Baird, Baraq Ghaleb, Isam Wadhaj, Gordon Russell, and William J. Buchannan. 2024. Safeguarding IoT Systems: Novel Authentication Method to Counteract Sybil and Flood Attacks. In *2024 4th International Conference on Robotics and Control Engineering (RobCE 2024), June 27–29, 2024, Edinburgh, United Kingdom.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3674746.3674772

RobCE 2024, June 27–29, 2024, Edinburgh, United Kingdom

ACM ISBN 979-8-4007-1678-2/24/06
https://doi.org/10.1145/3674746.3674772

1 INTRODUCTION

The rapid growth of IoT has led to the widespread deployment of RPL, a routing protocol specifically designed for low-power and lossy networks (LLNs) [18] [19]. RPL's energy efficiency and scalability make it an ideal choice for IoT applications, where resource-constrained devices communicate over wireless networks. However, RPL's inherent trust-based model exposes it to vulnerabilities such as Sybil and flood attacks, which can severely disrupt network operations [17][5]. In a Sybil attack, an adversary creates multiple fake identities to manipulate the network's operations by presenting a false view of the network's topology [17][12]. This can be achieved by exploiting RPL's mechanism, allowing attackers to impersonate legitimate nodes and inject malicious routing messages . Once an attacker gains control of a significant portion of the network, they can manipulate routing information, leading to traffic disruption, denial-of-service attacks, and data manipulation.

Flood attacks, on the other hand, overwhelm the network with excessive traffic, rendering it unusable. Attackers can generate large volumes of routing messages or data packets, consuming network resources and preventing legitimate traffic from being delivered. Flood attacks can disrupt network communication, making the IoT infrastructure vulnerable to downtime and service outages[5]. The joining of Sybil and flood attacks poses a significant threat to RPL based networks. By employing multiple fake identities, attackers can launch coordinated flood attacks, amplifying the impact of the attack and making it more difficult to detect and mitigate. This combination of attacks can severely disrupt network operations, compromise data integrity, and hinder the functionality of IoT applications. Traditional security mechanisms, often designed with more powerful computing platforms in mind, are often ill-suited to address this concern, necessitating innovative solutions tailored to the specific needs of RPL-based IoT networks [17]. In response, this paper presents a novel approach leveraging Bloom filters combined with hash-chains to mitigate combined Sybil and flood attacks.

Hash-chains offer valuable cryptographic capabilities, particularly in resource-constrained environments. The inherent one-way property of hash functions renders them particularly appealing for various security applications, especially those emphasizing data integrity and streamlined verification processes. This characteristic ensures that it is computationally infeasible to reverse-engineer the original data from its hash value, providing a robust layer of protection. In the context of IoT security, incorporating the root of the

hash-chain into communication protocols serves as a pivotal strategy. The hash-chain root, essentially functioning as a cryptographic fingerprint, empowers nodes within the network to discern and filter out undesirable traffic through the utilization of a Bloom filter. This approach enhances the overall security posture by creating a reliable means of validating the legitimacy of information shared between entities. Whether applied to secure messaging, data transfer, or authentication processes, the utilization of hash functions and their integration into communication protocols represents a fundamental step in fortifying the assurance of data integrity and authenticity in security-sensitive applications.

The remainder of this paper is structured as follows: Section 2 furnishes an overview of the current literature and related work in the realm of IoT network security. It also introduces the fundamental concepts of Bloom filters and hash-chains, laying the groundwork for a deeper understanding of the proposed protocol. Section 3 delineates our proposed hash-chain based countermeasure, explaining its design principles and showcasing its seamless integration with the RPL protocol. In Section 4, the paper culminates by summarizing its key contributions and proposing potential avenues for future exploration.

2 BACKGROUND AND RELATED WORK

This section serves as a brief introduction to RPL, the routing protocol devised for Low Power Lossy Networks (LLNs). Additionally, outlining common network attacks against this protocol emphasizing Sybil and DIS flood attacks while also introducing the relevant existing research to counter these attacks

2.1 The RPL Protocol Overview

RPL [19] is an IPv6 proactive distance-vector routing protocol designed by the IETF community specifically to fulfill the unique requirements of a wide range of low-power and LLN applications. It organizes its physical network into a form of Destination Oriented Directed Acyclic Graph (DODAG) where each DODAG is rooted at a single destination, referred to as the LLNs Border Router (LBR) [18] [19]. The term “upward routes” is used to refer to routes that carry the traffic from normal nodes to the LBR whereas routes that carry the traffic from the DODAG LBR to other nodes are called the downward routes [19]. See Figure 1.

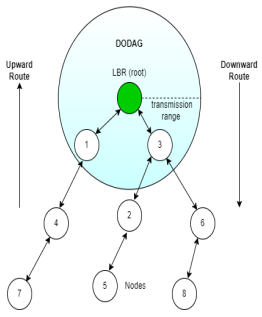

Figure 1: RPL DODAG visualisation

To facilitate the construction of these routes, RPL relies on a series of control messages that play a pivotal role in its operation which include:

- DODAG Information Object (DIO): These messages are used by the protocol to disseminate routing parameters required to establish the DODAG topology, and maintain upward routes.
- Destination Advertisement Object (DAO): These messages are used by nodes to advertise their reachability to the DODAG LBR. They are sent by the nodes to establish downward routes towards the root.
- DODAG Information Solicitation (DIS): These messages are used for the initiation of DODAG formation, allowing a node to actively solicit DIO messages in order to join a specific DODAG.

The construction of the DODAG topology, aka upward routes, starts with the root multi-casting control messages called DODAG Information Objects (DIOs) to its RPL neighbors which carry the necessary routing information and configuration parameters required to build the DODAG. The nodes then follow a pre-defined objective function to join the DODAG based on the information included in the received DIOs. Subsequently, DAO messages are used to establish downwards carrying traffic from the nodes to the DODAG root. A new node seeking to join an established DODAG can broadcast a DIS control message to neighboring nodes within the DODAGs, requesting topology information. A detailed discussion of RPL implementation details is beyond the scope of this paper. For a more thorough examination, readers are referred to the extensive literature available on RPL [6].

2.2 The Sybil-Flood Attack

The combined Sybil [13] and flood attack poses [2] a significant threat to RPL-based networks, compromising their integrity and functionality. In this form of attack, the attacker first creates a large number of fake nodes known as Sybil identities, thereby deceiving the network into accepting these false entities as legitimate nodes [7]. Subsequently, the attacker floods the network control messages, such as DIS and DAO messages, in an attempt to disrupt the routing protocol and prevent legitimate nodes from communicating. This combination of attacks not only undermines the network’s ability to determine routes accurately but also severely impacts its overall performance, leading to increased energy consumption, packet loss, and potential network paralysis. The combined Sybil and flood attack is particularly challenging to defend against because it is difficult to distinguish between Sybil nodes and legitimate nodes. Additionally, the flood of control messages can make it difficult to identify the source of the attack. As a result, this attack can be highly effective at disrupting RPL-based networks.

For instance, the RPL DIS dissemination mechanism is vulnerable to exploitation by a malicious node seeking to disrupt the network. An attacker can create the illusion of numerous new nodes attempting to join the network by generating and multicasting a large number of DIS messages, each associated with a different fabricated identity. Upon receiving these DIS messages with diverse identities, the neighboring node assumes that numerous new nodes intend to join the network. As per the RPL protocol, the nodes

repeatedly restart their trickle algorithms and proceed to broadcast DIO messages at their fastest configured rate. This barrage of control messages overwhelms the respective nodes, draining their energy resources and saturating their communication bandwidth and ultimately resulting in the denial of service.

The trickle timer in RPL networks is a double-edged sword. On the one hand, it optimizes network performance and battery life for resource-constrained IoT devices. It achieves this by employing an exponential back off strategy for sending control messages such as DIO and DAO. When the network appears stable, the trickle timer increases the interval between transmissions, reducing unnecessary control traffic and saving battery power [11].

On the other hand, the effectiveness of the trickle timer hinges on proper configuration. Setting excessively long intervals can lead to sluggish network behavior and delayed responses, while overly aggressive settings might generate unnecessary control traffic, negating the battery saving benefits.

2.3 Bloom Filter

Bloom filters are recognized as probabilistic data structures that excel in facilitating high-speed access decisions across a diverse range of applications [1]. In the context of the Internet of Things (IoT), their value extends beyond rapid decision-making, encompassing a noteworthy contribution to space efficiency. Beyond their ability to quickly determine the probable presence or absence of an element, Bloom filters stand out for their compact representation, making them particularly advantageous in resource-constrained IoT environments where efficient use of storage space is crucial. While Bloom filters excel in providing a 0% false negative response, it is essential to note that they are susceptible to false positives. A false positive occurs when the Bloom filter indicates that a searched element is in the array, however, it is not. The probability of false positives can be managed by adjusting parameters such as the size of the array, the number of hash functions, and the number of entries. Striking a balance between minimizing false positives and optimizing resource memory is a crucial consideration in the effective deployment of Bloom filters. The design and application of Bloom filters necessitate careful consideration to manage the false positive rate (FPR). To minimize the likelihood of false positives, key parameters such as the size of the array (N), the number of entries in the array (M), and the count of hash functions employed (K) must be thoughtfully configured. The probability of encountering a false positive during a search operation can be calculated using the formula [10]:

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

The formula calculates the FPR by considering the interplay between these three variables. The inner term, $\left(1 - \frac{1}{M}\right)$, represents the probability that a single bit position remains unset after hashing a single element. Raising this term to the power of KN accounts for the probability that all bits remain unset after hashing all of the elements with different hash functions. Finally, raising this result to the power of (K) represents the probability that this scenario (all bits unset) occurs for each of the elements being hashed. By adjusting (M), (K), and (N), developers can achieve a desired balance between the false positive rate and space efficiency requirements of a specific application.

Bloom filters facilitate both the addition of entries and search operations through the utilization of hash functions. In the illustrated Figure 2, the process of adding a new entry is evident as it transforms the allocated memory space from 0 to a 1. Utilizing the same hash function, the search operation involves comparing the search result with the data stored in the entries within the array. This dynamic mechanism allows for efficient and rapid membership tests, contributing to the versatility and effectiveness of Bloom filters in various applications.

In Figure 2, during the search operation, labeled as Search 2, the attempt to ascertain whether an item is part of the Bloom filter is made. As the item is not present in the Bloom filter, this determination is not confirmed until the return from hash 3. Conversely, in the case of the third item to be searched for, a negative entry is detected at the first hash check, rendering further hash 2 and hash 3 searches unnecessary. For Search 1, where all hash functions return positive results, it is important to highlight that this positive outcome does not correspond to entry 1, 2, or 3. Such a scenario would be considered a false positive, emphasizing the inherent trade-off in Bloom filters where false positives can occur, albeit at a controlled rate, as part of their probabilistic nature.

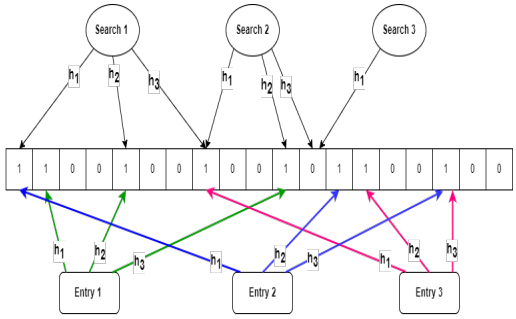

Figure 2: Standard Bloom filter visualisation

One of the fundamental limitations of standard Bloom filters lies in their inherent immutability: entries cannot be deleted after insertion. This poses a significant challenge in scenarios where data is dynamic and requires updates. Recognizing these limitations, research efforts have focused on two key areas: enabling deletions and enhancing space efficiency. Techniques like counting Bloom filters [3] and scalable Bloom filters with 2D or 3D arrays address these challenges [14][15], offering greater flexibility and improved space utilization compared to traditional Bloom filters.

2.4 Recent Related Work

Pu et al. [16] propose a novel lightweight authentication scheme for RPL networks. By utilizing both node identifiers (MAC addresses) and Physical Unclonable Functions (PUFs) in conjunction with a Bloom filter, the scheme aims to effectively mitigate Sybil attacks, a common threat in these networks. During node registration, the MAC address and PUF value are sent to the DODAG root, which

then applies a hashing function and adds them to a Bloom filter. This filter is subsequently disseminated throughout the network, enabling nodes to verify the legitimacy of their peers. The combination of PUFs and Bloom filters offers a robust and efficient solution for authentication in resource constrained RPL environments. This scheme leverages a Bloom filter for its efficient memory footprint and fast verification times but acknowledges the inherent trade-off with a small probability of false positives. While the paper deems this acceptable within the specific context of RPL networks, it is important to note the potential for unauthorized communication to slip through due to these false positives.

Gebretsadik et al. [4] proposed the use of 2D enhanced Bloom filters for intrusion detection, employing two Bloom filters in series to alleviate the load on the machine learning black box and enhance intrusion detection performance. In this innovative approach, the first Bloom filter stores information related to intruder data packet uniform resource locator(URLs), while the second one retains details about legitimate traffic. Notably, the Bloom filters are not fixed but are continuously updated with information from the machine learning algorithm. This dynamic update mechanism allows the addition of new entries to either the Bloom filter storing intruder URLs or the legitimate traffic Bloom filter, enabling effective filtering without the necessity for constant usage of the intruder detection machine learning black box.

In Yuan et al. [21] the author explores a distinctive hash chain approach for authentication within resource constrained IoT networks. The methodology involves nodes sharing a secret key, enabling them to create a unique hash by concatenating the original challenge sent from the root, the pre-shared secret key, and the node's individual unique ID. When transmitting this hash to another node, the receiving node checks for any prior communication by examining the hash. If the hash has not been sent previously, the receiving node recomputes the hash using sender node ID, secret key, and the received challenge. If the recomputed hash matches the received hash, it is added to the receiving node's database. Subsequently, the receiving node appends its own hashed concatenation, combining challenge, secret key, and unique ID, to the received hash from the original sender. This process results in the sender hash being chained with the receiver hash. Eventually, every node contributes its own hash to the chain, creating a shared database for comparison among all nodes in the network.

2.5 Hash-chain

A hash-chain is a cryptographically secure sequence of values created by applying a hash function to an initial input, known as the seed, and subsequently iterating the hash function on the resulting hash value [8]. The inherent property of hash functions being one-way makes it computationally infeasible to reverse them, preventing the derivation of the original key or any preceding hash values from a given hash in the chain.
To define a hash-chain, the following elements need to be considered:

- Number of hashes in the chain (x): This specifies the total number of times the hash function will be applied.
- Hash function (h): This is the specific cryptographic hash function that will be used to generate the chain, such as SHA-256 or MD5.
- Key or seed (k): This is the initial input value to which the hash function will be applied. It serves as the starting point for the chain.

The hash-chain is constructed as follows:

- First hash: The hash function is applied to the key, producing the first hash value in the chain:
$$h^x = h(k)$$
- Subsequent hashes: The hash function is then applied successively to the previous hash value, generating a chain of hashes:
$$h^{x-1} = h(h^x), h^{x-2} = h(h^{x-1}), ..., h^1 = h(h^2)$$
- Root hash: The final hash value in the chain is referred to as the root hash. It is the result of applying the hash function to the second-to-last hash value:
$$h^0 = h(h^1)$$

3 PROPOSED SECURE TECHNIQUE

3.1 Pre-deployment Bloom Filter initialization

- Step 1: Each individual device selects a unique random number.
- Step 2: Each node hashes a random number and then iteratively hashes the result, resulting in a chain of hash values stored by the device from h^x to h^0.
- Step 3: Subsequently, each device transmits the root of its hash chain (h^0) to the LLN border router (LBR).
- Step 4: The LBR aggregates the roots of hash chains received from the various networked devices and incorporates them into a Bloom filter.
- Step 5: Upon finalization of the Bloom filter, the LBR disseminates it to all devices within the network.

3.2 Device Authentication

The Device Authentication stage is crucial, especially when there are suspicions of a blended attack within the network. Its primary objective is to authenticate devices when transmitting DIS messages, thereby ensuring that each message originates from a genuine and legitimate source, rather than being part of a blended Sybil flood attack. The verification algorithm for this process runs as follows:

- Step 1: When the device needs to send a message, it will concatenate the next stored hash from the created hash-chain (h^{y+1}) where y is previous message hash-chain entry, with the timestamp (t_s) hashing the result :
$$V = h(h^{y+1} \mid\mid t_s)$$
- Step 2: The device will add the root of the hash, V and the next hash from its chain with the timestamp.
$$h^0, V, (h^{y+1} \mid\mid t_s)$$
- Step 3: During the initial verification phase, the root hash of the arriving message is extracted and compared with the contents of a Bloom filter. If the Bloom filter indicates

the presence of the presented root hash, suggesting a high probability of message legitimacy, the packet is allowed to proceed.

- Step 4: Following this verification at the Bloom filter, the node extracts the timestamp (t_s), comparing to the current timestamp. If the comparison falls within an acceptable time frame , the node can confidently conclude that there is no attempt at a replay attack.
- step 5: By hashing ($h^{y+1} \mid\mid t_s$) and comparing to V the node can verify the integrity of the timestamp. The node can then extract h^{y+1}, hashing this value no more than x times, comparing the result after each hashing operation to that of the root hash h^0.

The system's integrity hinges on a robust three-step verification process, which follows the initial filtering conducted by the Bloom filter. This process effectively eliminates false positives and mitigates replay attacks. Messages that fail to pass all three checks are deemed unreliable and potentially compromised, and are consequently dropped without being forwarded to the LBR.

From time to time, newly configured nodes may need to join the network. To accommodate this, the global repair mechanism within RPL can be used to reset the protocol periodically [9]. This allows for:

- The integration of new nodes into the network.
- The removal of any non essential or inactive nodes.

Additionally, by periodically resetting the protocol with the global repair mechanism, we achieve a reduction in the number of stored hashes needed in each node. This translates to lower memory usage and potentially improved performance.

While the global repair mechanism offers a simple and time-efficient solution for integrating new nodes and removing inactive ones, it comes at a cost. Implemented by the LBR and affecting all nodes in the network, this method can be resource-intensive due to the increased control traffic it generates.

Therefore, the frequency of global repairs needs careful consideration. A balance must be struck between maintaining an optimal number of stored hashes (reducing memory usage) and ensuring the network can accommodate new nodes and remove inactive ones efficiently.

4 CONCLUSION AND FUTURE WORK

This paper introduces a novel authentication method that integrates Bloom filters with hash-chains, establishing a robust defense against Sybil and DIS flooding attacks. Specifically tailored for lightweight generic IoT devices, the focus lies in delivering enhanced security solutions for these devices. The proposed authentication scheme demonstrates strong resistance against Sybil and DIS flood attacks. However, to further enhance security, it's important to consider potential replay attack scenarios.

One approach to address this could be to incorporate sequence numbers into messages. This would prevent a malicious user from simply replaying a captured authentication message. Another approach is to employ challenge-response mechanisms [20]. This adds an extra layer of security by requiring the node to respond to a unique challenge before being granted access.

It's important to note that even if a replay attack were successful, the integrity of the system remains strong. The Bloom filter itself does not store the complete message or any hash values, preventing attackers from forging a new message that would pass verification.

To assess the effectiveness of the proposed approach, the authors recommend a comprehensive evaluation within a simulated environment, such as Contiki. The methodology involves gathering data and fine-tuning the Bloom filter parameters to minimize the false positive rate. Additionally, the simulation would be subjected to various attack scenarios, including scenarios with multiple attackers and varying flood rates.

ACKNOWLEDGMENTS

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

REFERENCES

[1] Burton H Bloom. 1970. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM* 13, 7 (1970), 422–426.
[2] Akshaya Dhingra, Vikas Sindhu, et al. 2022. A Review of DIS-flooding Attacks in RPL Based IoT Network. In *2022 International Conference on Communication, Computing and Internet of Things (IC3IoT)*. IEEE, 1–6.
[3] Bin Fan, Dave G. Andersen, Michael Kaminsky, and Michael D. Mitzenmacher. 2014. Cuckoo Filter: Practically Better Than Bloom *(CoNEXT '14)*. Association for Computing Machinery, New York, NY, USA, 75–88. https://doi.org/10.1145/2674005.2674994
[4] Fitsum Gebreegziabher Gebretsadik, Sabuzima Nayak, and Ripon Patgiri. 2023. eBF: an enhanced Bloom Filter for intrusion detection in IoT. *Journal of Big Data* 10, 1 (2023), 102.
[5] Baraq Ghaleb, Ahmed Al-Dubai, Elias Ekonomou, Mamoun Qasem, Imed Romdhani, and Lewis Mackenzie. 2018. Addressing the DAO Insider Attack in RPL's Internet of Things Networks. *IEEE Communications Letters* 23, 1 (2018), 68–71.
[6] Baraq Ghaleb, Ahmed Y Al-Dubai, Elias Ekonomou, Ayoub Alsarhan, Youssef Nasser, Lewis M Mackenzie, and Azzedine Boukerche. 2018. A Survey of Limitations and Enhancements of the ipv6 Routing Protocol for Low-power and Lossy Networks: A Focus on Core Operations. *IEEE Communications Surveys & Tutorials* 21, 2 (2018), 1607–1635.
[7] Muhammad Ali Khan, Rao Naveed Bin Rais, and Osman Khalid. 2023. Collaborative Detection and Prevention of Sybil Attacks against RPL-Based Internet of Things. *Computers, Materials & Continua* 77, 1 (2023).
[8] DaeYoub Kim and Jihoon Lee. 2020. A reverse hash chain path-based access control scheme for a connected smart home system. *IEEE Consumer Electronics Magazine* 10, 1 (2020), 93–100.
[9] Kevin Dominik Korte, Anuj Sehgal, and Jürgen Schönwälder. 2012. A study of the RPL repair process using ContikiRPL. *Autonomous Infrastructure, Management and Security* (2012). https://doi.org/10.1007/978-3-642-30633-4_8
[10] Gang Liu, Wei Quan, Nan Cheng, Bohao Feng, Hongke Zhang, and Xuemin Sherman Shen. 2018. BLAM: Lightweight Bloom-filter based DDoS mitigation for information-centric IoT. In *2018 IEEE global communications conference (GLOBECOM)*. IEEE, 1–7.
[11] Arslan Musaddiq, Yousaf Bin Zikria, and Sung Won Kim. 2018. Energy-Aware Adaptive Trickle Timer Algorithm for RPL-based Routing in the Internet of Things. In *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*. 1–6. https://doi.org/10.1109/ATNAC.2018.8615408
[12] James Newsome, Elaine Shi, Dawn Song, and Adrian Perrig. 2004. The sybil attack in sensor networks: analysis & defenses. In *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks* (Berkeley, California, USA) *(IPSN '04)*. Association for Computing Machinery, New York, NY, USA, 259–268. https://doi.org/10.1145/984622.984660
[13] Abdullah Orman, Yunus Üstün, and MURAT Dener. 2023. Detailed Analysis of Sybil Attack in Wireless Sensor Networks. *Uluslararası Sürdürülebilir Mühendislik ve Teknoloji Dergisi* 7, 1 (2023), 41–54.
[14] Ripon Patgiri. 2021. robustBF: A High Accuracy and Memory Efficient 2D Bloom Filter. *CoRR* abs/2106.04365 (2021). arXiv:2106.04365 https://arxiv.org/abs/2106.04365
[15] Ripon Patgiri, Sabuzima Nayak, and Samir Kumar Borgohain. 2019. PassDB: A Password Database Using 3D Bloom Filter. *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)* (2019). https://doi.org/10.1109/hpcc/smartcity/dss.2019.00162

[16] Cong Pu and Kim-Kwang Raymond Choo. 2022. Lightweight Sybil attack detection in IoT based on bloom filter and physical unclonable function. *computers & security* 113 (2022), 102541.

[17] Cong Pu, Cong Pu, and Cong Pu. 2020. Sybil Attack in RPL-Based Internet of Things: Analysis and Defenses. *IEEE Internet of Things Journal* (2020). https://doi.org/10.1109/jiot.2020.2971463

[18] A. Vasseur. 2011. RPL : The IP Routing Protocol Designed for Low Power and Lossy Networks Internet Protocol for Smart Objects (IPSO). *null* (2011). https://doi.org/null

[19] Tim Winter, Pascal Thubert, Anders Brandt, Jonathan Hui, Richard Kelsey, Philip Levis, Kris Pister, Rene Struik, Jean-Philippe Vasseur, and Roger Alexander. 2012. *RPL: IPv6 Routing Protocol for Low-power and Lossy Networks.* Technical Report.

[20] Aruna Yadav, Sanjeev Kumar, and Jagendra Singh. 2022. A review of physical unclonable functions (PUFs) and its applications in IoT environment. *Ambient Communications and Computer Systems: Proceedings of RACCCS 2021* (2022), 1–13.

[21] Shengli Yuan and Randy Phan-Huynh. 2022. A Lightweight Hash-Chain-Based Multi-Node Mutual Authentication Algorithm for IoT Networks. In *2022 IEEE Future Networks World Forum (FNWF)*. IEEE, 72–74.