Hindawi
Journal of Sensors
Volume 2022, Article ID 1309402, 16 pages
https://doi.org/10.1155/2022/1309402

Research Article

SF-LAP: Secure M2M Communication in IIoT with a Single-Factor Lightweight Authentication Protocol

Khurram Shahzad,[1] **Masoom Alam**,[1] **Nadeem Javaid**,[1] **Abdul Waheed**,[2,3] **Shehzad Ashraf Chaudhry**,[4,5] **Nafees Mansoor**,[6] **and Mahdi Zareei**[7]

[1]*Department of Computer Science, COMSATS University Islamabad, 44000, Pakistan*
[2]*Department of Computer Science, Women University Swabi, Swabi 23430, Pakistan*
[3]*School of Electrical and Computer Engineering, Seoul National University, Seoul 08826, Republic of Korea*
[4]*Department of Computer Science and Information Technology, College of Engineering, Abu Dhabi University, Abu Dhabi, UAE*
[5]*Department of Computer Engineering, Faculty of Engineering and Architecture, Nisantasi University, Istanbul 34398, Turkey*
[6]*Department of Computer Science and Engineering, University of Liberal Arts Bangladesh (ULAB), Dhaka, Bangladesh*
[7]*Tecnologico de Monterrey, School of Engineering and Sciences, Mexico*

Correspondence should be addressed to Nafees Mansoor; nafees@nafees.info

Received 20 June 2022; Revised 21 August 2022; Accepted 23 August 2022; Published 10 November 2022

Academic Editor: Akhilesh Pathak

Machine-to-machine communication allows smart devices like sensors, actuators, networks, gateways, and other controllers to communicate with one another. The industrial Internet of things (IIoT) has become a vital component. Many industrial devices are connected to perform a task automatically in machine-to-machine communication, but they are not properly secured, allowing an adversary to compromise them against a variety of attacks due to communication system vulnerabilities. Recently, a secure lightweight authentication protocol (SLAP) was proposed by Panda et al. They asserted that every known attack that could happen in the IIoT is deterred by their suggested protocol. In this study, we prove that the SLAP protocol is vulnerable to desynchronization, impersonation, replay, and eavesdropping attacks. To prevent these attacks and enhance that protocol, we need to implement a secure authentication mechanism that ensures the security of communication. This paper proposed a secure M2M Communication in IIoT with a single-factor lightweight authentication protocol (SF-LAP). Single-factor authentication is a simple and secure way to communicate. It uses less power and communication overhead while providing a secure mechanism for conversation. In the machine-to-machine (M2M) scenario, the proposed protocol uses an exclusive-OR operation and a hashing function to ensure secure communication between the sensor and the controller. The proposed mechanism uses a secure preshared key and timestamp technique to protect and safeguard this connection against desynchronization attacks and eavesdropping attacks. We used Burrows Abadi Needham (BAN) Gong, Needham, and Yahalom (GNY) logic, and the automated validation of Internet security protocols applications (AVISPA) tool for formal verification and perform a security analysis as an informal verification to make sure the suggested protocol is secure. Analysis that shows the SF-LAP consumes the least computing and communication overhead and is more secure because it prevents desynchronization and eavesdropping attacks to all of the known attacks that are modification attacks, tracing attacks, impersonation, man-in-the-middle, and replay attacks.

1. Introduction

Direct communication between devices, whether wired or wireless, is referred to as machine-to-machine and is used in industrial internet of things to increase productivity and efficiency while using less energy. Different devices such as actuators, controllers, and sensors are networked together with other devices in IIoTs. In order to increase productivity and efficiency, interconnect gathers data, analyzes it, and recommends effective measures. The main structure of our

paper is depicted in Figure 1. You should be familiar with the terms machine-to-machine communication, IIoT, and lightweight authentication protocol before reading our paper.

1.1. M2M Communication. With the proliferation of wireless devices, there is a growing demand for cellular network based M2M communication. The main feature of M2M is that it allows two or more devices to communicate without the need for human interaction, as well as perform and supervise certain functions automatically. The main goal of M2M in smart devices is to ensure secure communication, automate device interaction and communication, access high-speed internet, measure the costs of different devices, minimize computation and communication costs, ensure a cheap and easy way to connect, make lightweight and efficient software for better speed, address locomotive devices and treat on location, and make lightweight and efficient software for better speed. In terms of efficiency and security, M2M is more typically linked to medium-access smart home apps, mobile M2M standard services and platforms, and LTE [1]. Figure 2 depicts many features of M2M communication.

Wireless technologies such as WLAN, RFID, near field communication, Zigbee, Bluetooth, long-term evolution (LTE), and others are widely used in M2M communication. They differ in infrastructure, ranges, throughput, sizes, and efficiency [2]. The SD-M2M framework is proposed to minimize costs and enhance end-to-end quality for renewable energy management in the house, building, industrial, grid and microgrid, and field [3]. M2M introduces LTE-A to improve the physical layer. The cellular network for M2M communication improves the short access latency and high throughput, as well as minimizing the signaling overhead [4]. M2M communication is utilized in the e-health industry to improve patient care, data protection, tamperproof electronic report delivery, dynamic doctor assignments, lower costs, and increase system performance [5]. M2M communication is used in smart home appliances to monitor and control smart home devices. These appliances are connected to smart phones and can be operated and controlled remotely, making life easier [6].

1.2. Industrial IoT. Since many smart devices are connected, one of the biggest challenges in cybersecurity. For the purpose of preventing or reducing cybersecurity events and corporate data breaches, employees must be informed about these risks and, as a result, build the resilience of their organizations to cyberattacks [7]. The blockchain system uses Merkle trees to store data; however ,this article substitutes the incremental aggregator subvector commitment for the Merkle tree to reduce the size of the proof and the quantity of communication needed. In addition to lowering the storage pressure on nodes, the proof size has been decreased from its original data scheme to 15% [8]. The results show that the encryption and verification procedures are sped up by around 70 and 90%, respectively, using this technology. It might be utilized in cloud-assisted healthcare IIoT to offer comprehensive answers to the aforementioned issues. Additionally, this approach reduces communication expenses by more than 80% during the verification stage [9].

To protect IIoT infrastructure against deadly and complex multivariant botnet attacks, the study suggests a hybrid intelligent deep learning-enabled method. The results demonstrate speed efficiency, and the suggested procedures outperform in reliably recognizing multivariant sophisticated bot assaults [10]. The IIoT's intelligent sensors are used to collect the physical attributes of objects dispersed across a large area.The heuristic technique is recommended to assist the backbone node in balancing the incoming traffic for scheduling. Aggressive scheduling media access control is the name of the proposed effort. Other nodes will actively arrange the proper time slot if the owner node does not have data to broadcast at that moment. According to calculations and comparisons with the most recent proposals, this technique outperforms time division multiple access in terms of packet loss rate, energy consumption, packet delivery rate, and time required for various numbers of sensor nodes [11].

This paper [12] examines how to use new IIoT technologies in a cloud manufacturing system to address this problem. It provides a general system architecture for a plug and play service-oriented IIoT gateway solution for a cloud manufacturing system helped by the IIoT. In order to quickly store and query data in a cloud time series database while capturing precisely the correct quantity of data for field-level manufacturing equipment, service-oriented data schemas are developed. According to research, using purposefully designed service-oriented data schemas that use plug and play IIoT technologies to gather the essential data for high-level cloud manufacturing decision-making is an efficient way to connect field-level manufacturing equipment to a cloud manufacturing platform.

Due to the widespread usage of IIoT, there is an increasing demand for sensing technology to collect data and information. Industrial wireless sensor networks with industrial information perception capabilities have developed to fill this demand. The proposed multiobjective chaotic elite adaptive ant colony optimization has obvious advantages. The population will be initialized, increasing population diversity and the algorithm's ability to deviate from the local ideal. By constantly adjusting the algorithm trend, the adaptive optimization technique significantly accelerates algorithm convergence. In various scale scenarios, the suggested algorithm's performance is assessed. The simulation results demonstrate that this technique may significantly enhance routing when compared to other cutting-edge QoS routing systems. When compared to other cutting-edge QoS routing systems, the simulation results show that this technique can successfully reduce network energy consumption, improve routing security, and fulfill multi-QoS restrictions for the end-to-end latency and reliable service [13]. Figure 3 summarizes some of the IIoT features.

1.3. Lightweight Authentication Protocols. Authentication protocols are specifically designed to transfer the data between two or more users through a proper and secure communication channel. These channels sometimes use a

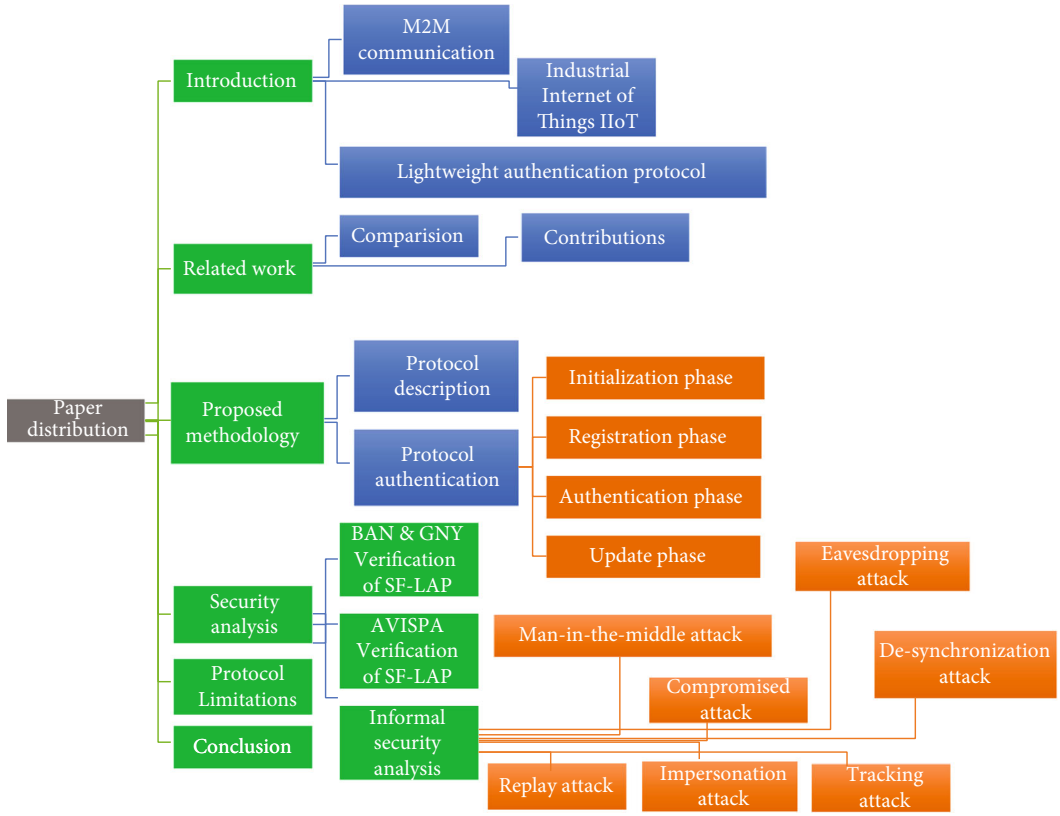

Figure 1: SF-LAP paper structure.

different cryptography algorithm for different devices and different security purposes. For instance, to achieve authority, integrity, and authenticity rely on different layers to make a system in secure manners. A bulk of authentication techniques are widely used, but we target M2M communication and propose a new SF-LAP secure protocol.

A new notion for an Internet of things micropayment framework in terms of a wearable device is presented in the paper [14]. This payment mechanism encrypts and decrypts communication messages between numerous organizations using an elliptic curve integrated encryption technology. Customers may buy things using a wearable gadget and communicate sensitive payment information via a mobile app, according to protocol. Customers, banks, and merchants are all securely connected through the program. The implementation of wearable device micropayments makes use of a secure end-to-end solution. To accomplish security features, NFC combines lightweight cryptography and elliptic curve cryptography's device pairing technology. The proposed protocol uses BAN logic to conduct mutual authentication among the many parties engaged in the protocol, enabling real-time transactions between banks and IoT devices.

The protocol's viability and safety are demonstrated using formal BAN logic. Based on the security analysis and testing findings, the proposed authentication protocol is more suitable for scaling to enormous authentication. It may reach higher security standards at a reduced cost. It also protects against desynchronization and denial-of-service

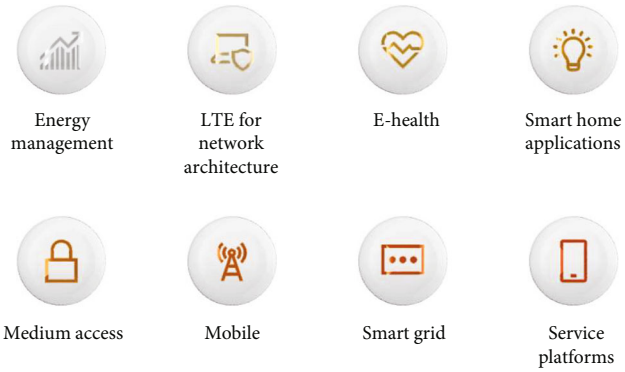

Figure 2: Different aspects of M2M Communication.

assaults. Furthermore, this approach does not handle less tag storage and time. The paper [15] proposes an anonymous authentication method for resource-constrained IoT (RC-IoT) devices on page no. 1822, based on elliptic curve cryptography and signcryption. This protocol lowers the cost of communication and computation. This work uses a random oracle model to formalize theoretical security and simulated studies to show that this protocol effectively minimizes resource use.

The study [16] proposes lightweight 3FA for WSN for healthcare remote user IoT application, based on hash, and XOR includes features of biometrics, smart devices and user password, session key, and mutual authentication. Protocol

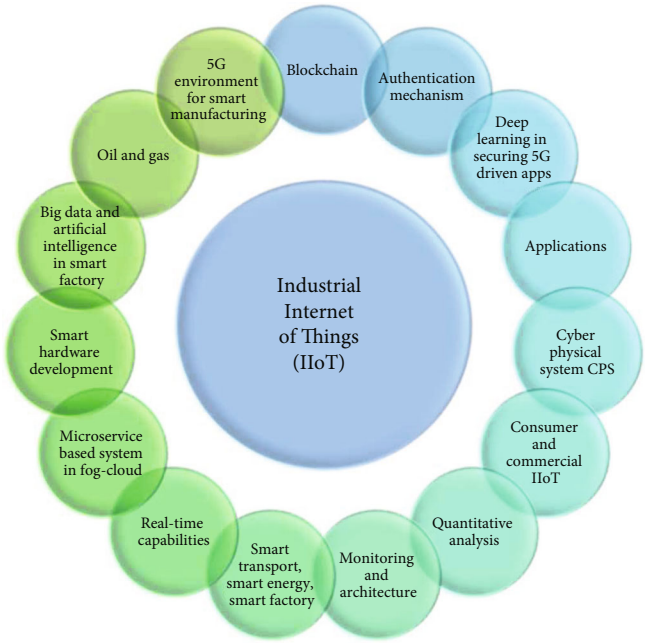

Figure 3: Different features of IIoT.

is based on BAN logic and formal verified on AVISPA tool, reduces the communication and computation overhead. In [17], study proposes secure authentication mechanism provides authentication, integration, confidentiality, and access control to monitor healthcare devices and to meet complex situation, remote surgery, real time applications, mental and physical health, blood pressure ECG, etc. This scheme controls computation cost, IoT verification method, authentication mechanism in application layer, network, and perception. Also, IoT devices verification in real time application.

The fundamental challenges in the [18] M2M study are mutual authentication, authentication key agreement, and network. To overcome these challenges, several initiatives are being implemented, such as LTE and 5G for better networking speeds and protocols. The study finds that group-based secure lightweight authentication and key agreement reduce communication and computational cost. In addition, 6G is being researched in a number of industries, including e-health, military, e-education, and e-commerce, all of which depend on IoT and blockchain technologies and billions of devices [19].

ISAG is proposed in this [20] study to ensure authentication and key agreement in LTE networks to avoid MITM and DOS attacks. The lightweight authentication protocol, according to the [21] VANET study, provides a secure way for emergency vehicles to verify illegal legitimacy while also protecting them from reputation, device theft, and impersonation attacks. This study [22] proposed a secure authentication protocol for LTE networks. REPS-AKA3 improves network performance by reducing bandwidth consumption, storage, and communication overhead, as well as using the AVISPA tool for formal verification. Lightweight authentication mechanism post quantum FLAT is proposed in [23] for service and certificate providers, as well as RC-IoT devices in M2M communication devices that can withstand post quantum changes.

The rapid adoption of wireless wearable Internet of things devices in textiles where data is transmitted increases the possibility of an eavesdropping attack. The mutual authentication protocol between IoT devices and gateway is strengthened by this [24] document, which supports cutting-edge encryption methods. The study provides the necessary conclusions and also verifies the authenticity mechanism's security in comparison to other mutual authentication protocols using the AVISPA and Scyther tools, ensuring resilience against eavesdropping attacks. The blockchain technology is used in M2M communication IoT devices to avoid a centralized system. The [25] study uses blockchain technology, as well as the AES, SHA-256 algorithm, and HMAC, to improve the processing time and speed of hardware chips in smart home devices.

Light-AHAKA, a novel [26] lightweight protocol with a key agreement to encrypts sensitive data, is proposed as a secure mechanism for cloud RC-IoT systems. The study proposes [27] key agreement, text message, and registration phase for the authentication process between devices and server in order to reduce RC-IoT device, communication, and computation overhead and improve battery life. For M2M communication, [28] proposes a key exchange and mutual authentication protocol. It is also safe, light, efficient, and productive. MITM and replay attacks over the network are not possible with this protocol. AVISPA and Scyther provided formal verification.

This [29] study proposes a mutual and transitive authentication mechanism for intermediate devices and gateways. The session key is verified using BAN logic, and the resilience against critical attacks is verified using the Scyther tool. The simulation results demonstrate performance efficiency while also reducing handshake time, memory, and energy consumption. This proposed technique is small and can be used with RC-IoT devices. The study of telerobotic surgery in M2M communication devices introduces a new efficient device scheme, as well as its detection and classification against malicious nodes. This system is more efficient, secure, and capable of detecting and classifying rogue devices. This protocol is resistant to replay attacks, DDOS attacks, and eavesdropping. Its results are superior to those of the ECFU and LEACH protocols [30].

As described in the paper [31], the ultra-lightweight RFID systems security authentication protocol uses bit-crossing XOR restructuring operations to successfully deflect denial of service threats, replay, resist forgery, and desynchronization while authenticating the data center. The research [32] compares and contrasts two mutual authentication protocols: hash and Rabin public key. The reader, server, and tag are all desynchronized as part of the impersonation attack. This method secures the connection between the tag and the reader. The verification process uses the Scyther and random oracle models to ensure that this scheme is secure and efficient for RFID systems.

The remaining portion of the paper is structured as section 2 examines related research, compares the proposed protocol to all previous proposals, and highlights this paper's

Table 1: Comparison of SF-LAP with other protocols.

Resilience against attacks	[33]	[34]	[35]	[36]	[37]	[38]	SF-LAP
Replay attack resilience	✓	✓	✓	✓	✓	✗	✓
MITM attack resilience	✓	✓	✗	✓	✓	✓	✓
Impersonation attack resilience	✓	✓	✓	✓	✓	✗	✓
Modification attack resilience	✗	✗	✓	✓	✓	✓	✓
Compromise attack resilience	✗	✗	✗	✓	✗	✓	✓
Tracing attack resilience	✗	✗	✗	✓	✗	✓	✓
Eavesdropping attack resilience	✗	✓	✓	✗	✗	✗	✓
Session-key breach	✗	✓	✓	✓	✓	✓	✓
Identity confidentiality resilience	✗	✗	✓	✓	✓	✓	✓
Formal verification by AVISPA	✓	✗	✓	✓	✗	✓	✓
Desynchronization attack resilience	✗	✗	✗	✗	✗	✗	✓

contribution. Section 3 proposes methodology and provides a description and steps for authentication. Section 4 uses the BAN and GNY logic and the AVISPA tool, respectively, to conduct formal and informal security analyses. The final section 5 of our paper includes the references and the conclusion.

2. Related Work

This study proposes a three-factor mutual authentication mechanism based on elliptic curve cryptography enabling patient privacy protection in the Telecare Healthcare Information System. The communication expenses are lower using this protocol. Uses the real-or-random model, BAN logic, and the AVISPA to perform formal security evaluations. This protocol outperforms the Ryu et al. approach, which exposes itself to privileged suspicious assaults, patient anonymity, and password update errors. This protocol can protect against many security threats, such as privileged suspicious assaults, stolen mobile phones, and insider attacks. Using the ROR model and the BAN logic demonstrated, believing their approach could assure mutual session and authentication key security. Compared to similar current protocols, this protocol is more secure and has reduced communication costs [33].

SAKE is a symmetric authentication key exchange protocol is proposed in this [34] article for IIoT. It relies on hash function operations, concatenation, and XOR to ensure message integrity, key exchange, and lightweight authentication. Regarding security and performance on page no. 8, analyzes the SAKE protocol to seven modern and IoT-related authentication schemes. According to the comparison results, the SAKE protocol, page no. 9, uses the least number of computational resources and has the 3rd communication overhead of both eight AKE protocols providing twelve security measures. To show that the proposed protocol offers complete forward secrecy and secure session key agreement, it is compared to AKE-related protocols. The [35] data confidentiality, message privacy, and authenticity are among the security services provided by the suggested paradigm. According to the protocol's security verification and performance findings, the proposed system is suitable for use in real-world IoT applications. However, desynchronization assaults are not protected by this article.

The author [36] proposes an industrial RC-IoT authentication protocol. For M2M communication in the IIoT, the LAKD authentication protocol has been presented. This project aims to achieve a low computational cost that is appropriate for IIoT devices with limited resources. To do this, the proposal employs lightweight operations such as addition, subtraction, XOR, and hash function. Although the AVISPA tool and BAN logic confirmed mutual authentication and resistant to MITM and replay attacks, this suggested technique does not solve many of the previously described threats, such as forgery, session key disclosure, impersonation, and desynchronization attacks.

For storage overhead and M2M communication, the document [37] presents a based authentication system that relies on XOR operations and hashes for session key agreement, mutual authentication, device identity confidentiality, and protection against impersonation, modification, and man-in-the-middle and replay attacks. This article, however, does not provide protection against desynchronization assaults. As a result, the system we propose is resistant to desynchronization assaults. [38] SLAP is lightweight authentication and secure protocol that uses two techniques, hash and XOR operations. Modification attacks, tracing, man-in-the-middle, impersonation, and replay attacks are all safe with the proposed protocol. The proposed SLAP's correctness is checked using BAN logic. The protocol is safe, according to the AVISPA tool. SLAP protocol is suitable for various attacks, but it does not protect against desynchronization attacks. The proposed authentication framework covers forgery attacks, session key disclosure, impersonation, MITM, and replay attacks.

2.1. Comparison. It should be noted that the protocols in question are vulnerable to some well-known attacks. The proposed protocols do not provide sufficient security for session keys. As a result, an adversary can quickly obtain and exploit the session key. When we compare the security features of the SF-LAP protocol to those of other M2M communication protocols, we can see how well it defends against various attacks. [33] TMIS proposed 3FA based on

Elliptic curve cryptography, securing joint sessions and authentication keys but not against all other attacks. [34] Concatenation, XOR, and hash functions are used in the SAKE protocol. Of the eight AKE protocols, it consumes the fewest computing resources and has the third-lowest communication overhead, with twelve security features but no protection against further attacks. This [35] protocol protects against forgery, session key, impersonation, MITM, and replay attacks but not against desynchronization attacks. [36] LAKD is a RC-IoT authentication protocol for M2M communication that uses four different techniques to achieve low computational cost for IIoT devices, including addition, subtraction, XOR, and hash. Many of the previously discussed attacks, such as forgery, session key disclosure, impersonation, and desynchronization attacks, are not addressed by this proposed method. So this protocol is vulnerable.

This [37] protocol achieved session key agreement, device identity confidentiality, mutual authentication, protection against impersonation, modification, replay, and MITM attack for storage overhead and M2M communication. Still, it was unable to protect against desynchronization attacks. As a result, our proposed scheme, SF-LAP, is providing protection against all of these attacks as well as desynchronization attacks. Table 1 shows a comparison of all of these procedures to SF-LAP. [38] SLAP is lightweight authentication uses two techniques, hash and XOR operation. The author claimed that his protocol is safe and provide resilience against modification attacks, tracing, man-in-the-middle, impersonation, and replay attacks, but we have some proof that the proposed protocol is vulnerable against eavesdropping, replay, impersonation, and desynchronization attack. The proposed protocol is checked by BAN logic with the extension of GNY logic includes more rules for verification purpose.

2.1.1. Eavesdropping Attack on SLAP [38]. An attacker can identify and track the messages of a controller among the communications exchanged between a controller and a sensor. To track the victim, the attacker must eavesdrop on two of their sessions. The attacker successfully eavesdrops on the session between the controller and the sensor to get $b_i = h($ $r\|PSK), D_1 = b_i \oplus R_1, D_2 = h(R_1) \oplus \mathrm{ID}_s$, and $D_{3'} = h(\mathrm{ID}_s \|$ $R_1\|r\|TS_0)$. Since the value of r is a simple random number that repeats and can be known by the attacker by guessing it, an adversary may simply learn the value of b_i. By eavesdropping in on the starting session, the attacker was able to successfully retrieve the value of D_1 and the controller ID is now available for use in the next authentication session.

Now that the attacker is aware of the message the controller will send the sensor as ID_r, they may watch the controller in the subsequent session. The attacker has a variety of options for completing this level. In this case, the attacker acts as a sensor and sends a message of request to the controller. With knowledge of the controller's responses to this message, the attacker obtains the messages that were sent by it and compares the values with D_1. When a value matches D_1, the attacker knows that the message is about the controller, and the controller is monitored. The adversary is aware of the value of M_1 sent from the sensor to the controller. The attacker can find and follow a specific controller with this kind of attack. The attacker employs this strategy because it is aware of the value of ID_r and has learned this from previously eavesdropping in on a session.

2.1.2. Replay and Impersonation Attack on SLAP [38]. In this attack, the adversary sends the controller the messages it got from the previous session to mimic a real sensor. The adversary eavesdrops on the session between controller and sensor. An opponent who had previously known the values of D_1 and D_2 now gets after the session as a consequence of preventing $D_{3'} == D_3, r == h(ID\|rr)$ and $D_{3'} == D_{3'}$. The attacker then pretends to be a sensor and sends the controller a request message. $D_{3'}$ is sent to the adversary by the controller. After receiving $D_{3'}$, the adversary provides M_1, which was obtained from the previous session and forwarded to the controller.

The controller receives M_1, which verifies that the opponent is the actual sensor. The opponent then receives M_2 from the controller. The opponent makes sure the controller has validated the validity of the controller after getting M_2. The adversary has therefore effectively completed its offensive task. Finally, by leveraging the stolen messages from the previous session, the attacker may have fooled the controller into thinking it was a legitimate sensor. The controller in this message decided to utilize random numbers rather than the Nonce technique because of the circumstances that led to this approach. The controller's identification is accepted once the sensor's identity has been verified.

2.1.3. Desynchronization Attack on SLAP [38]. SLAP was unable to stop the desynchronization attack because the values of b_i are stored as $b_i = h(r \| PSK)$. The b_i joins the opponent and can disrupt the synchronization between the controller and sensor when the b_i cannot be effectively received by the controller. The attacker passes messages from the controller to the sensor. In other words, the attacker uses this method to perform a man-in-the-middle attack by intercepting a request message from the sensor and sending it to the controller.

The attacker then sends the controller the M_1 message that was sent by the sensor. The controller, rather than the sensor, intercepts the communication and resends it to the adversary. The $D_{3'}$ message is compared to the D_3 and r that the adversary was already known by this controller as a result. The attacker executes the attack and is known that utilize M_2 to deliver the values of D_4, D_5, D_6, and D_7 to the sensor through $b_{i'} = h(r' \| PSK)$ makes it to get for him to do so. The attack is effective since the attacker may now compute the shared key because he knows the value of $D_7 == D_{7'}$.

2.1.4. SF-LAP VS SLAP [38] Protocol. The two key distinctions between these two are the involvement of the server in SF-LAP as compared to S and C, and the server clock timestamp function which is an added feature of SF-LAP. These modifications in SF-LAP mitigate the preceding attacks that were addressed in SLAP as giving resilience and providing resistance against eavesdropping, replay,

impersonation, and desynchronization attacks. The following are some further variations between these two protocols are shown in Table 2.

2.1.5. SLAP [38] Limitations

(i) The SLAP provides a simple random number that may be reused and can cause of pseudo-random number generator attack. While the threshold time safety factor is missing when not synchronized properly

(ii) Both the devices are synchronized separately can cause of desynchronization attack. The comparison between SLAP and SF-LAP is shown in Figure 4

(iii) Attack resilience is depicted on the left side. SLAP is represented with orange bars and SF-LAP with blue bars. The absence of bars indicates that the paper lacks resistance to the attack. For instance, the fact that a desynchronization attack represents a blue bar but not an orange one indicates that SF-LAP can provide resilience against that attack while SLAP cannot

2.2. Contribution. To enhance SLAP, we add two to three new techniques called SF-LAP, compare them, and then explain why SF-LAP is more secure than SLAP. SF-LAP identifies some limitations of using the SLAP protocol as a base paper. SF-LAP makes it better by including some techniques like adding a nonce technique to a secure authentication channel and including a server and server clock between these two devices. This server clock's job is to simultaneously provide a timestamp for both devices. This timestamp has the advantage of being resistant to eavesdropping and desynchronization attacks and includes the safety chain threshold time for the authentication process. To obtain the appropriate results for SF-LAP in comparison to SLAP, we analyze it using the BAN with an extension of GNY logic [39] and run it on AVISPA and prove the SF-LAP protocol. SF-LAP provides resilience against eavesdropping, replay, impersonation, and desynchronization attack while taking into account all previous attacks. SF-LAP protocol is more secure than previous lightweight authentication protocol.

3. Proposed Methodology

The previously proposed authentication protocol was good and secure in terms of modification, tracing, man-in-the-middle, impersonation, and replay attacks, but it was unable to offer resilience against desynchronization and eavesdropping attacks. We proposed a new sensor-to-controller authentication technique SF-LAP that enables secure communication between the sensor and the controller while also offering resistance against desynchronization and eavesdropping threats while accounting for all prior assaults. It has a different process, but we must declare and highlight the similarities in SF-LAP, which we derived from SLAP rather than the timestamp technique, server involvement, resilience against desynchronization attack, and eavesdropping attack.

Moreover, to prevent a pseudo-random number generation attack from the repetition of random numbers, we used the nonce technique in this paper inspired by the paper of Rostampour et al. [39]. SF-LAP provides pseudocode for the authentication process while BAN with extension of GNY logic is used for formal verification and security analysis for informal verification. It was necessary to use the same criteria of overcomes and to provide a more secure protocol than the previous one. These logics are believes, possesses, once said, jurisdiction, fresh, session, and encryption used to prove the authentication protocol. The rules of these logics are the message-meaning rule, nonce verification rule, possession rule, freshness rule, recognizability rule, jurisdiction rule, and message interpretation rule. Table 3 shows the notation we use for this authentication mechanism.

3.1. Protocol Description. In SF-LAP protocol PSK has already been shared between S and A as shown in Figure 5. S, C, and A make up SF-LAP protocol. S sends information to C with ID_s and timestamp TS, which verifies the message and sends it back to S. By using the server clock to generate timestamps, the server ensures that the S and C communicate with a single timestamp TS. Desynchronization and eavesdropping attacks are prevented by the server timestamp for both the S and C.

3.2. Protocol Authentication Process. Simple timestamp can be use but due to obtain several timestamps by an adversary. Adversary can guess the pattern which can be reason for the several attacks like guessing and replay attack. Nonce is an arbitrary number, random number, or pseudo-random number that used for once where the values are dynamically changed. We use nonce along with the timestamp, the purpose of N along with timestamp is to secure the authentication process. The authentication mechanism is depicted in Figure 6, shows how the S and C exchange data and share the secret message.

3.2.1. Initialization Phase. Authentication server (A) believes that sensor (S), and controller (C) are already known PSK then, A always believe that session key is shared between S and C. S requests to C for communication and shares the data, then C request to A for TS. A generates TS with its clock that shares the TS_1 to both S and C, simultaneously. To make theoretically understandable, we says $\mathrm{TS}_1 = \mathrm{TS}_{S1}$ for S and TS_{C1} for C to avoid further confusion because one timestamp is shared on both sides, but practically its T S_1 with same hash on both sides. So now we can say that, A generates TS_{S1} and TS_{C1} with its clock and shares to both S and C, simultaneously.

3.2.2. Registration Phase. S generates N_s and hash the value of $a = (\mathrm{ID}_s \,\|\, \mathrm{PSK})$ and XOR with ID_s that is $D_1 = h(a) \oplus \mathrm{I}$ D_s and concatenate the value of TS_1 with $b = (N_a \,\|\, \mathrm{PSK})$ and XOR with N_s. S encrypts these values and sends message $M_1 = \{D_1, D_2, N_a, \mathrm{TS}_{S1}\}$ to C. Then, C selects the timestamp TS_{S1} and compares it with TS_{C1} that C already taken from

Table 2: SF-LAP VS. SLAP Protocol.

SF-LAP	SLAP
It provides a clear use-case in 3 steps	Does not provide the use-case diagram
Provides security against eavesdropping attacks	Does not provide resilience against eavesdropping attacks
Provides security against replay attacks	Does not resist replay attack
Provides security against impersonation attacks	Does not resist impersonation attack
Provides security against desynchronization attacks	Does not resist desynchronization attack
Timestamps both the S and C simultaneously	Does not provide this phenomenon
Communicates between S and C via server	There is no medium other than the sensor and controller
Provides the facility of timestamp through the server clock	Not provide the facility of this type of a timestamp
Provides pseudocode of our procedure	Does not provide any pseudocode
A special nonce function is used which provides the random number only for one time and then it never uses the same random number again in its communication	There is a simple random number is used in communication that is vulnerable for replay attack

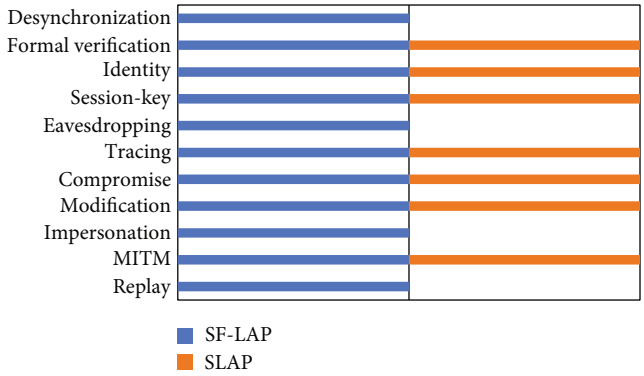

Figure 4: Comparison between SF-LAP and SLAP.

the A and verifies for the same values of both TS_{S1} and TS_{C1}. If false, then terminate the whole authentication process but if true, then computes the values of $ID_s = D_1 \oplus h(a)$ and $N_s = D_2 \oplus h(b \| TS_{S1})$. Then calculate the value of D_3 while take TS_2 from A and concatenate with N_s such as $D_3 = h(N_s \| TS_{C2}) \oplus N_c$, while C share its ID as ID_c, XORed with both the nonce values such that $D_4 = IDc \oplus h(b \| N_s \| N_c)$. Concatenate ID_c, a and value of nonce that is $D_5 = h(IDc \| a \| N_s \| N_c)$ and returns the message to S that is $M_2 = \{D_3, D_4, D_5, N_a, TS_{C2}\}$ while proceeding for authentication phase.

3.2.3. Authentication Phase. Sensor matches TS_{C2} and TS_{S2}, both the timestamp values and terminate the authentication process if both timestamp are mismatch while compute the values of $N_c = D_3 \oplus h(N_s \| TS_{S2})$ and $ID_c = D_4 \oplus h(b \| N_s \| N_c)$. Then check the match case of D_5 with $D_{5'}$, if equals, then take a timestamp TS_3 from A and calculates $D_6 = h(b \| N_s \| TS_{S3})$ and send the encrypted message to C as $M_3 = \{D_6, TS_{S3}\}$. C selects the timestamp TS_{C3} shared by A and compares with TS_{S3} and terminate the authentication process if both timestamp are not matches. In match case, C verifies the value of D_6 with $D_{6'}$. If equal, computes $D_7 = h(a \| N_s \| N_c \| TS_{C4})$, the key $\kappa = h(N_s \| N_c \| ID_s)$ and send the message $M_4 = \{D_7, TS_{C4}\}$ to the sensor S.

3.2.4. Update Phase. The update phase is an additional feature in which S sends OK after decrypts the message and make some calculation here. When S decrypts the message $M_4 = \{D_7, TS_{C4}\}$ and make some calculation here, sends OK for the confirmation of authentication phase to C that the authentication process is successfully secured. Moreover, the sensor selects TS_4 and verifies TS_{C4} with TS_{S4}, if true then verifies $D_7 = h(a \| N_s \| N_c \| TS_{C4})$ with the fresh value. If not verify, then terminates. If true, then computes $\kappa = h(N_s \| N_c \| ID_s)$. While after completing the authentication process, sensor fresh the values and sends the OK message that means to update all the values such that $ID_s = h(a' \| b' \| N_{s'} \| N_{c'})$, to avoid from the guessing attack. So, S and C updates there values to avoid from guessing attack on the previous values and now they can share secrets in a secure manner, and the process of all the authentication mechanism is completed. For more illustration, the whole authentication process taken from Figure 6 of SF-LAP protocol is given in pseudocode 1.

4. Security Analysis

To prove that our suggested SF-LAP protocol is more secure than others, we performed formal and informal security research. The three methods of verification that we used in our paper are BAN and GNY logic [39] verification of SF-LAP, AVISPA verification of SF-LAP, and informal security analysis.

4.1. BAN and GNY Logic Verification of SF-LAP. We used these [39] logics to verify the authentication mechanism, prove that SF-LAP provides acceptable authentication between sensor and controller and synchronizes both of them with the server clock. In these logics, there are a variety of notations that are employed. In our scenario, S and C are the agents instead of P and Q. M is used for the message, κ

for a secret key, K^{-1} for the private key, S for the sensor, and C for the controller. The following constructions are employed to demonstrate the use and connections of these logic.

(a) $S \equiv M$: means that the S believes M, and M are true

(b) $S \triangleleft M$: indicates that the S received/sees the message M.

(c) $S \ni M$: indicates that the S possesses the message M.

(d) $S \sim M$: indicates that S sends messages at the sending time of message M. The S believed M, or S once said M.

(e) $S \Rightarrow M$: if other principals believe S believes M, then M means the other principals believe S believes M. M is under jurisdiction of S.

(f) $P\#(M)$: indicates that M is fresh M had never been sent before the current run of the protocol. Nonces are expressions created to demonstrate freshness, and they frequently include a timestamp. Without using nonces, we can get that not-so-fresh message feeling

(g) $S \leftrightarrow^{K} C$:S and C share the key K and may use it to communicate. No principal other than S or C or a principal trusted by S or C can find K.

(h) $\longrightarrow^{K} S$:S has the session key K.

(i) $\{M\}_K$:K is the key used to encrypts M. This represents a signature of M when K is a private key. For example, K^1

To get from protocol steps to logical inferences, we use idealization. This tries to convert the sent message into the intended semantics. For example: in the given protocol $C \leftrightarrow^{K} S$ means that the key is shared between C and S. One goal of idealization is to leave out parts of the message that do not add to the recipients' beliefs. The plaintext is not used in these logics because it can be forged. The protocol's idealization is not clearly defined. It is contingent on how specific steps are interpreted.

4.1.1. BAN and GNY Logic Rules. These rules are taken from a paper [40] that provides the formal analysis required to verify the authentication protocol. S and C are used for sensor and controller, M and N are used for statement. Message meaning rules govern how messages are interpreted. Rather than write the English equivalents, we used the predefined symbols. When using a shared key set:

$$\frac{S \mid \equiv C \xrightarrow{K} S, S \triangleleft \{M\}K}{S \mid \equiv C \mid \sim M}. \tag{1}$$

Table 3: Paper's notations and description.

Notation	Description
PSK	Preshared key
S	Sensor
C	Controller
A	Authentication server
ID_s	Identity of S
ID_c	Identity of C
M	Message
N_s	Nonce of S
N_c	Nonce of C
N_a	Nonce of A
N_K	Nonce revoke
h	Hash function
$\|\|$	Concatenation
$\oplus$	XOR operation
$\{M\}_K$	Encrypted message with session key
TS	Timestamp
K^{-1}	Private key
κ	Session key

When using session keys,

$$\frac{S \mid \equiv \xrightarrow{K} C, S \triangleleft \{M\}_K^{-1}}{S \mid \equiv C \mid \sim M}. \tag{2}$$

A principal S must believe in $C^{'}s$ beliefs according to the Jurisdiction rule:

$$\frac{S \mid \equiv C \Rightarrow M, S \mid \equiv C \mid \equiv M}{S \mid \equiv M}. \tag{3}$$

The rules for nonce-verification show how to ensure that communication is both new and fresh in the senders believe:

$$\frac{S \mid \equiv (\#(M)), S \mid \equiv C \mid \sim M}{S \mid \equiv C \mid \equiv M}. \tag{4}$$

If any component of the formula is fresh, it cannot be changed; therefore, the whole formula has to be fresh:

$$\frac{S \mid \equiv (\#(M))(M)}{S \mid \equiv (\#(M))(M, N)}. \tag{5}$$

When both communication devices have believe on each other, then Belief rule is:

$$\frac{S \mid \equiv C \mid \equiv (M, N)}{S \mid \equiv C \mid \equiv (M)}. \tag{6}$$

SF-LAP is a secured way to share a key between the S

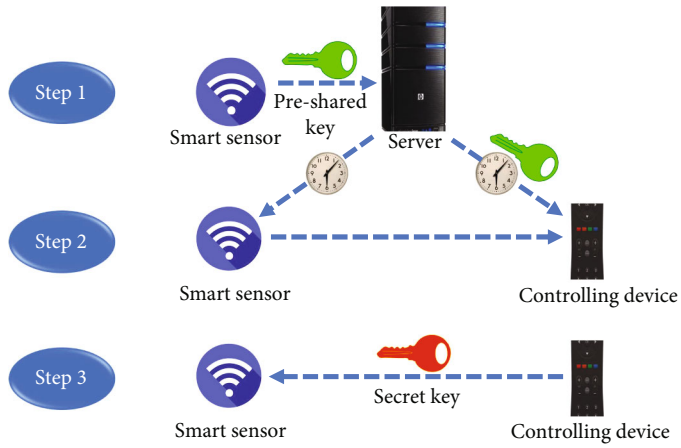

Figure 5: A use case diagram for the authentication process.

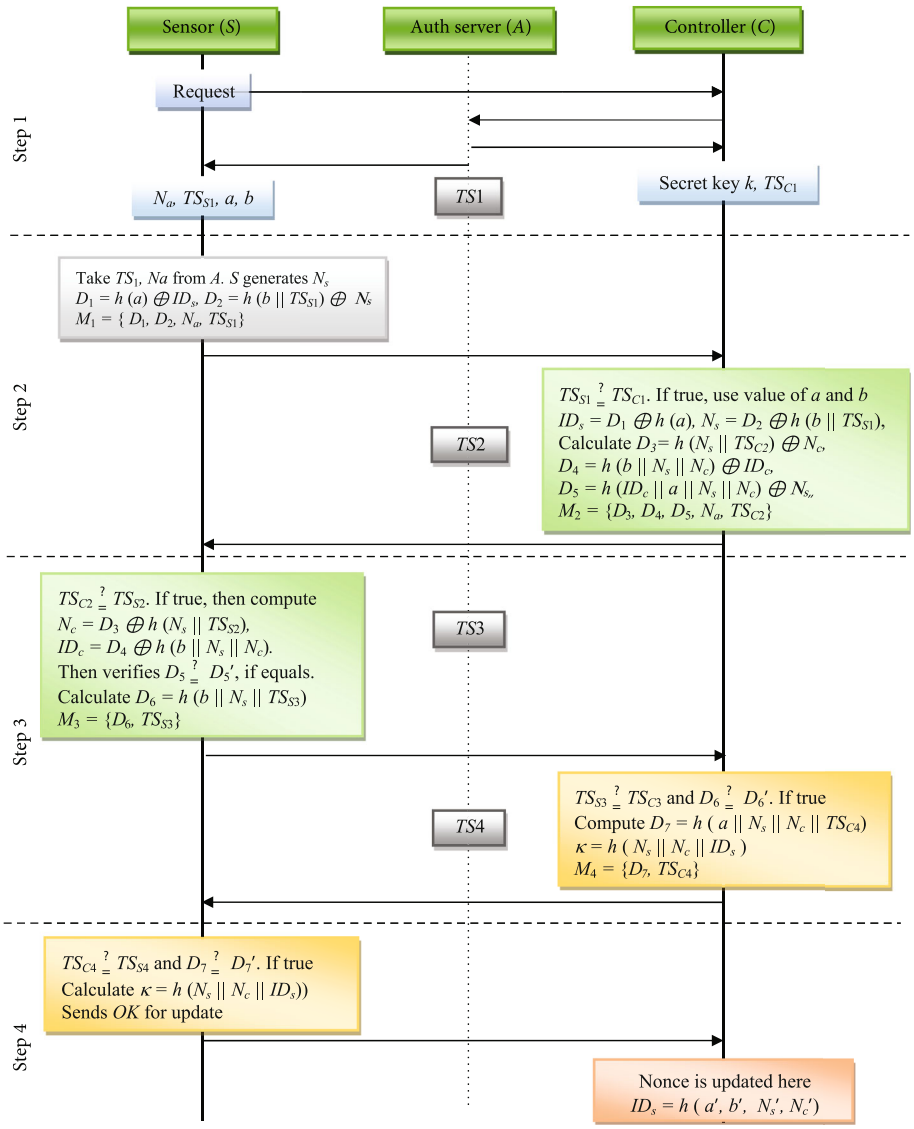

Figure 6: Sensor and controller authentication mechanism via server.

```
1: (//) double slash are used for comments
2: Procedure SF-SLAP AUTHENTICATION PROTOCOL
3://we assume that A already known PSK
4://S takes TS1 and Na from A and generates Ns
5://S takes the value of a and x or with IDs
6://S takes value of b, concatenate with TS1 and x or with Ns
7: S| ~ M1 to C //sensor sends message M1 to controller
8:   if (C ◁ M1, and TS1 match) then //controller receives message M1 and match the timestamp TS1
9:      return true;
10:        if (C | ~ D3, D4, D5, Na, TS2 to S) then //controller sends D3, D4, D5, Na, and TS2 to sensor
11:           return true;
12:              if (S ◁ M2, and TS2 match) then //sensor receives message M2 and match the timestamp TS2
13:                 return true:
14:                    if (S | ~ D6, TS3 to C) then //sensor sends D6 with timestamp TS3 to controller
15:                       return true;
16:                       if (C ◁ M3, and TS3 match) then //controller receives M3 and matches the timestamp TS3
17:                             return true;
18:                                if (C | ~ D7 and TS4 to S) then //controller sends D7 and timestamp TS4 to sensor
19:                                   return true;
20:                                      if (S ◁ M4, and TS4 match) then //sensor receives message M4 and match timestamp TS4
21:                                         return true;
22:                                         while (S sends OK to C for update the values of a', b', Ns', and Nc')
23:                                      else
24:                                      return false; //TS4 does not match
25:                                else
26:                                return false;
27:                          else
28:                          return false; //TS3 does not match
29:                    else
30:                    return false;
31:              else
32:              return false; //TS2 does not match
33:        else
34:        return false;
35:   else
36:   return false; //TS1does not match
37: end procedure
```

Pseudocode 1: SF-SLAP authentication protocol.

and C. So, the goal of our proposed technique is to firstly ensure that the shared key is secured between the S and C.

4.1.2. Goals. *Goal* 1: $S| \equiv (S \overset{K}{\longrightarrow} C)$

Goal 2: $S| \equiv C| \equiv (S \overset{K}{\longrightarrow} C)$

Goal 3: $C| \equiv (S \overset{K}{\longrightarrow} C)$

Goal 4: $C| \equiv S| \equiv (S \overset{K}{\longrightarrow} C)$

4.1.3. Idealized Form. SF-LAP protocol according to idealized Needham-Schroeder Shared-Key [BAN89a] is as follows:

M1: S → A: S, C, N_A
M2: A → S: $\{N_A, C, k_{SC}, \{k_{SC}, S\}k_{CA}\}k_{SA}$
M3: S → C: $\{k_{CA}, S\}k_{CA}$
M4: C → S: $\{N_C\}k_{SC}$
M5: S → C: $\{N_C - 1\}k_{SC}$

4.1.4. Assumptions. The following assumptions we have been made are based on the SF-LAP authentication protocol from Figure 6:

(i) A believes PSK is shared to S and C S $\overset{Ksc}{\leftrightarrow}$ C

(ii) S and C believes that A fresh TS (S $\overset{TS1}{\leftrightarrow}$ C)

(iii) A believes S, and C receive TS1

(iv) S believes that C receives M1

(v) C believes that M1 is send by S

(vi) C and S believe that A fresh TS (S $\overset{TS2}{\leftrightarrow}$ C)

(vii) A believes S, and C receives TS2

(viii) C believes that S receives M2

(ix) S believes that M2 is send by C

(x) S and C believe that A fresh TS ($S \overset{TS3}{\leftrightarrow} C$)

(xi) A believes S and C receive TS3

(xii) S believes that C receives M3

(xiii) C believes that M3 is send by S

(xiv) C and S believe that A fresh TS ($S \overset{TS4}{\leftrightarrow} C$)

(xv) A believes S, and C receives TS4

(xvi) C believes that S receives M4

(xvii) S believes that M4 is send by C

4.1.5. Some [BAN89a] Rules for SF-LAP. R1: S |≡ ($S \overset{Ksa}{\leftrightarrow} A$): S believes that the key is shared between S and A

R2: C |≡ ($C \overset{Kca}{\leftrightarrow} A$): C believes that the key is shared between C and A

R3: S |≡ A ⇒$S \overset{K}{\leftrightarrow} C$: S believes that A controls key between S and C

R4: C |≡ A ⇒$S \overset{K}{\leftrightarrow} C$: C believes that A controls key between S and C

R5: S |≡ A ⇒#($S \overset{K}{\leftrightarrow} C$): S believes that A controls and fresh the value of key, shared between S and C

R6: S |≡ # (Ns): S believes that the value of nonce is fresh by S

R7: C |≡ # (Nc): C believes that the value of nonce is fresh by S

R8: S |~ {Ns, $S \overset{Ksc}{\leftrightarrow} C$, # (Ksc), { $S \overset{Ksc}{\leftrightarrow} C$ }Kca }Ksa : S received the fresh encrypted values

R9: C |~ {$S \overset{Ksc}{\leftrightarrow} C$}Kca from A

R10: S |~ {Nc, $S \overset{Ksc}{\leftrightarrow} C$}Ksc from C

R11: C |~ {Nc-1, $S \overset{Ksc}{\leftrightarrow} C$}Ksc from S

R12: C |≡ # ($S \overset{Ksc}{\leftrightarrow} C$): C believes that value of key is fresh, shared between S and C

4.1.6. Proof from BAN and GNY Logic. By applying BAN and GNY logic [39] on SF-LAP protocol:

(i) S | ≡ A | ~ (Ns, $S \overset{Ksc}{\leftrightarrow} C$, #($S \overset{Ksc}{\leftrightarrow} C$), {$S \overset{Ksc}{\leftrightarrow} C$}Kca)

Message meaning rule using R1 and R8

(ii) S | ≡ #(Ns, $S \overset{Ksc}{\leftrightarrow} C$, #($S \overset{Ksc}{\leftrightarrow} C$), {$S \overset{Ksc}{\leftrightarrow} C$}Kca)

Freshness conjuncatenation rule using 1 and R6

(iii) S | ≡ A | ≡ (Ns, $S \overset{Ksc}{\leftrightarrow} C$, #($S \overset{Ksc}{\leftrightarrow} C$), {$S \overset{Ksc}{\leftrightarrow} C$}Kca)

Nonce verification rule using 2 and 1

(iv) S | ≡ A | ≡ ($S \overset{Ksc}{\leftrightarrow} C$)

Belief Conjuncatenation rule using 3

(v) S | ≡ A | ≡ (# $S \overset{Ksc}{\leftrightarrow} C$)

Belief conjuncatenation rule using 3

(vi) S | ≡ ($S \overset{Ksc}{\leftrightarrow} C$)

Jurisdiction rule using 4 and R3

(vii) S | ≡ #($S \overset{Ksc}{\leftrightarrow} C$)

Jurisdiction rule using 4 and R5 (Goal 1 achieved)

(viii) C | ≡ A | ~ $S \overset{Ksc}{\leftrightarrow} C$

Message meaning rule using R2 and R9

(ix) C | ≡ S | ≡ $S \overset{Ksc}{\leftrightarrow} C$

Nonce verification rule using R12 and 8 (Goal 4 achieved)

(x) C | ≡ $S \overset{Ksc}{\leftrightarrow} C$

Jurisdiction rule using R4 and 9 (Goal 3 achieved)

(xi) S | ≡ C | ~ (Nc, $S \overset{Ksc}{\leftrightarrow} C$)

Message meaning rule using 6 and R10

(xii) S | ≡ # (Nc, $S \overset{Ksc}{\leftrightarrow} C$)

Freshness conjuncatenation rule using 7

(xiii) S | ≡ C | ≡ (Nc, $S \overset{Ksc}{\leftrightarrow} C$)

Nonce verification rule using 12 and 11

(xiv) S | ≡ C | ≡ $S \overset{Ksc}{\leftrightarrow} C$

Belief conjuncatenation rule using 13 (Goal 2 achieved)

4.2. AVISPA Verification of SF-LAP. A powerful tool AVISPA determined that SF-LAP provides sufficient authentication against desynchronization attacks. In the AVISPA tool, four backends entities are responsible for a specified function at execution. HLPSL is used for a high-level language, and HLPS2IF is used for the translator.

The HLPSL representations of the proposed protocol, including the sensor and controller, are shown in Figure 7. AVISPA automatically validates security protocols and determines whether they are classified as either SAFE or UNSAFE depending on predetermined criteria. AVISPA validates that SF-LAP is SAFE protocol. Figure 8 shows the AVISPA simulation results using the on the fly model checker (OMFC) and constraint logic-based attack searcher (CT-ATSE) backends, demonstrating that SF-LAP is secure.

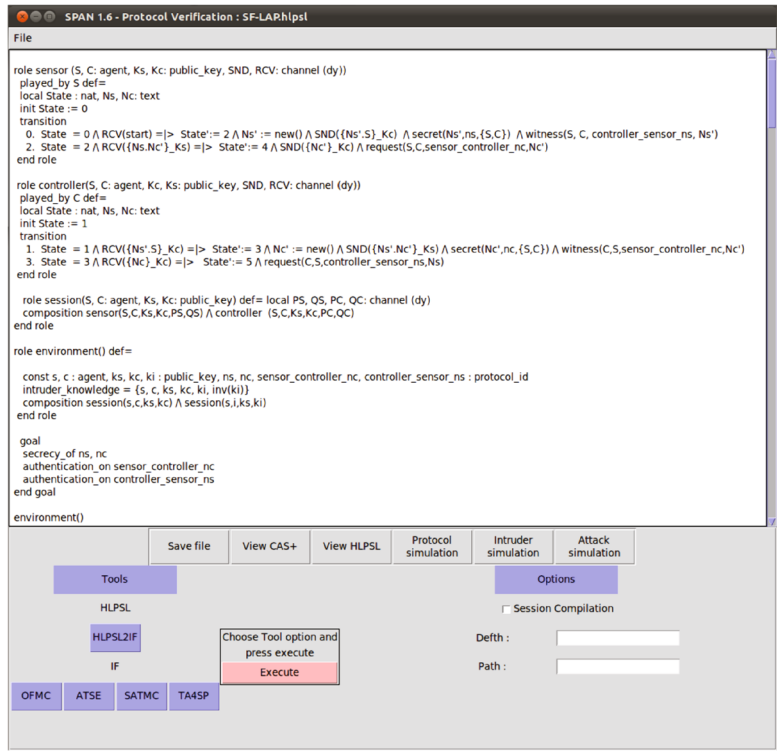

Figure 7: HLPSL representation of SF-LAP.

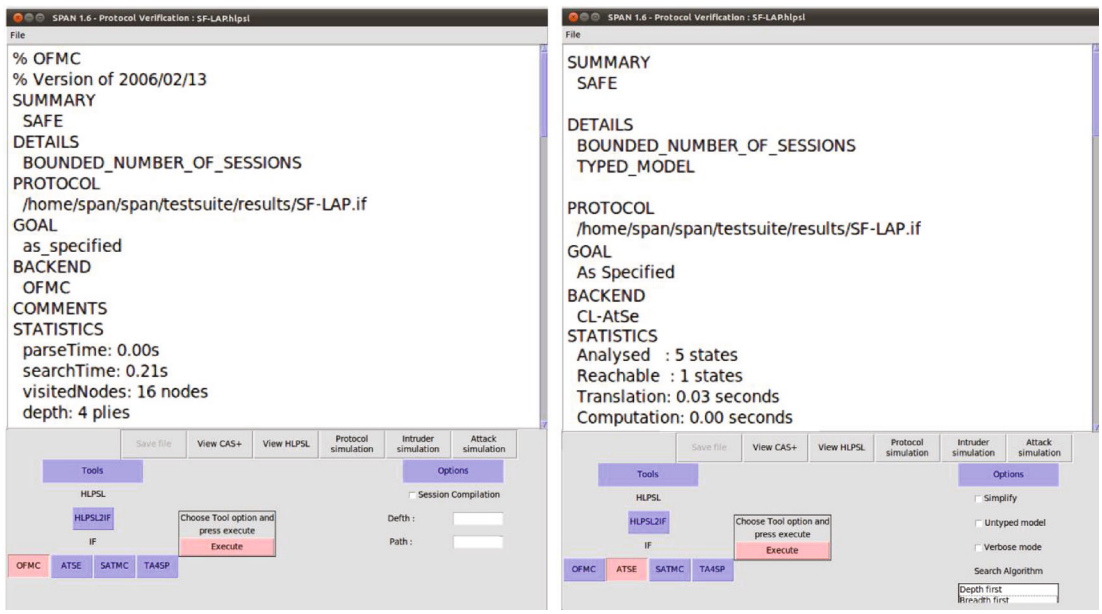

Figure 8: AVISPA result using OMFC and ATSE back-end.

4.3. Informal Security Analysis. SLAP claims that the shared key between sensor and controller was not secured, so it is vulnerable as we proved in section 2.1 on how SLAP is compromised against eavesdropping, impersonation, replay, and desynchronization attack, and hence compromised the shared key and was unable to secure the communication. Our approach uses Nonce and timestamp to overcome this vulnerability.

4.3.1. SF-LAP Resists Replay Attacks. A replay attack repeats data and allows the attacker to delay it. For timestamp purposes, SF-LAP uses server time to synchronize the time for the S and the C. The server clock records the time of the S and sets a limited time threshold for the message; if there is a delay, the message is automatically terminated. Thus, the issue of message delay is avoided by this protocol. The

authentication process's hash, all messages, and the attack are all altered if the attacker changes the timestamp. Replay attacks are therefore impossible with SF-LAP.

4.3.2. SF-LAP Resists Impersonation Attacks. By locating identity or initial handshake vulnerabilities, an adversary must launch an attack in place of the server. As we add the server between S and C, we hide the sensor IDs. S already gave A his IDs, so A has no chance to show his ID during this procedure. There is no way for an adversary to interfere with or compromise the system at the sensor location because the controller restricts new sensors here in place of verified sensors. Besides this, an adversary who is unable to discover the values of D1 and D2 is unable to overhear C and S talking. To secure this process, the request message is sent from the actual sensor to the controller using the value of Nonce from authentication and the match case value of D3. From the controller, the sensor gets D'3. The controller receives M1, which was forwarded and gleaned from the previous session after the sensor has sent D'3.

After the controller receives M1, the sensor receives M2, which verifies the validity of the message from the sensor. The sensor makes sure that the controller has confirmed the controller legitimacy after receiving M2. As a result, the adversary's offensive mission has been unsuccessful. The attacker must then prevent the controller and reliable sensor from communicating with each other using the previous session's stolen message. The controller in this message chose to use Nonce rather than a simple random number to prevent an adversary's plan. The identification of the controller is recognized once the sensor's identity has been established. Therefore, an attacker cannot guess the IDs or IDc, and the protocol will be secure from impersonation attacks.

4.3.3. SF-LAP Resists Eavesdropping Attacks. The messages of C cannot be found and followed by an attacker among the correspondence between C and S. This communication between S and C cannot be deleted or modified. An adversary cannot intercept this data. The server also knows PSK, which has previously communicated it to the C. As a result, the network is secure when $b = h$ (Na ‖ *PSK*), D_1 = h(a) ⊕ IDs and D_2= h (b ||TS_{S1}), an authentication protocol successfully offers resilience against eavesdropping on the session between C and S. Since the value of Ns is a nonce technique, it is impossible for an adversary to guess because it is an arbitrary number along with the random number generated by S, which changes its value dynamically. As a result, an attacker is prevented from attacking at the beginning of the session, and the values of D_1 and IDs are protected.

In step two, the attacker can no longer read the message on the controller side, and the C will now send the ID_c to the S. At this level, the attacker has no chance of succeeding, so the attacker cannot serve as S, and hence actual S sends M_1 to the C. Response C compares the values with D_1 after receiving the M_1 and reads it that was sent from the S to the C and C does not know the value that comes from the previous session. So, SF-LAP protects against eavesdropping attacks.

4.3.4. SF-LAP Resists MITM Attacks. An adversary successfully interrupts communication between S and C resulting in a man-in-the-middle attack. IDs are not shared during the authentication process because they are declared to the C in SF-LAP ahead of time as the sensor ID. Our approach secures against MITIM attacks during communication by making sure the shared secret key is safe before the authentication procedure. When b is successfully received by C, the value of b is saved as b = h (Na || PSK), and the message from the controller to the sensor cannot be passed by the adversary or prevented from being used to carry out a MITM attack by intercepting a request message from the S and sending it to the C.

4.3.5. SF-LAP Resists Desynchronization Attacks. The sensor compared the D'3 message to the D3 and Na, which were therefore unknown to the adversary, before sending the message M1 to the controller. The attacker in this instance is ineffective and unable to compute the shared key because they are unable to carry out the attack, and are unaware that M2 is being used to deliver the sensor's values for D4, D5, D6, and D7 through b'= h (Na' || PSK), and do not know the value of D7 == D7'. On the A is where the PSK is kept, and the server clock is used to time stamp the S and C. The S and C are timestamped using the server clock, and the PSK is stored on the A.

An adversary cannot interrupt on the back-end side because Ns, Nc, and Na prevent desynchronization and the TS of the server is used simultaneously on both sides. Because special Nonce Ns on the S side and Nc on the C side, along with TS1, TS2, TS3, or TS4, are concatenated with each other and TS is fresh in each phase and does not store the old value in it, if an adversary wants to modify something, hashes will change. Additionally, because Nonce is tied to the server clock, it changes dynamically every second. Therefore, SF-LAP resists desynchronization attacks since there is no chance that IDs and IDc will be changed.

4.3.6. SF-LAP Resists Modification Attacks. The values are one-way hashes in step 2 of the SF-LAP authentication protocol; specifically, D1 hashes the value of a, D2 hashes the value of b, D3 hashes Ns concatenated with TSC2, D4 hashes the concatenated values of b, Ns, and Nc, and D5 hashes the concatenated value of IDc, a, Ns, and Nc. A modification attack happens when an adversary modifies one of the messages M1, M2, M3, or M4. S and C exchange information and timestamp each other using the server clock. The values of b, Ns, and D7 are hashed in step 3, along with the values of a, Ns, and Nc. The server timestamp and the value of each of these Ds, such as TSS1 with D2, TSC2 with D3, TSS3 with D6, and TSC4 with D7, make the hash values unique at each stage. The hash of all D's values will change if an adversary attempts to modify any of them because they are all concatenated and XORed together, making it impossible for the adversary to modify any step. SF-LAP resists modification attacks as a result.

4.3.7. SF-LAP Resists Tracing Attacks. When the IDs or IDc remain the same throughout the process, a tracing attack takes place. As the authentication mechanism is synchronized with the server clock, which an adversary cannot alter, the nonce dynamically changes the values here. The timestamp produced by the server clock ensures that an adversary cannot break it. Additionally, we use a nonce technique in conjunction with a timestamp to make sure that every number differs from the one before it in a random manner, preventing the adversary from determining the true key. Additionally, SF-LAP added an update phase in which S validates TSS4 and D7 before updating all the values, including a', b', Ns', and Nc', which are sent to C and updated upon the OK request. The adversary will be unable to guess or track these values once all the previous values have been deleted from this authentication system.

4.3.8. SF-LAP Resists Compromised Attacks. If the attacker has access to the key and can compute the message M1, M2, M3, or M4, it constitutes a compromised attack. Before this communication, we have a secret key that is shared over a secure channel and contains a nonce technique. A compromised attack cannot succeed against this protocol because an adversary cannot guess it.

5. Conclusion

A single-factor lightweight authentication protocol for machine-to-machine communication in the industrial Internet of things is proposed in this paper. The proposed protocol uses an exclusive-OR operation and a hashing function to provide a secure mechanism for conversation, ensuring secure communication between the sensor and the controller. It protects and safeguards this connection against desynchronization and eavesdropping attacks using a secure preshared key and timestamp technique. To ensure that SF-LAP is secure, we used BAN and GNY logic and the formal verification by using AVISPA tool and security analysis, verify that SF-LAP is secure. Finally, we conclude that SF-LAP is more secure because it prevents desynchronization, eavesdropping, and all other types of attacks such as modification attacks, tracing attacks, man-in-the-middle, impersonation, and replay attacks. The secure method of authentication provided by the SF-LAP protocol is available when a sensor is already connected to a controller. What steps will be taken if we want to connect a new sensor to the same controller? We will explain this authentication mechanism in more detail later on, along with how a new sensor can connect to the same controller.

Data Availability

All the relevant data are available in the paper.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This project has been sponsored by the University of Liberal Arts Bangladesh (ULAB), Dhaka Bangladesh.

References

[1] O. A. Amodu and M. Othman, "Machine-to-machine communication: an overview of opportunities," *Computer Networks*, vol. 145, pp. 255–276, 2018.

[2] M. Weyrich, J.-P. Schmidt, and C. Ebert, "Machine-to-machine communication," *IEEE Software*, vol. 31, no. 4, pp. 19–23, 2014.

[3] Z. Zhou, J. Gong, Y. He, and Y. Zhang, "Software defined machine-to-machine communication for smart energy management," *IEEE Communications Magazine*, vol. 55, no. 10, pp. 52–60, 2017.

[4] M. Hasan, E. Hossain, and D. Niyato, "Random access for machine-to-machine communication in lte-advanced networks: issues and approaches," *IEEE Communications Magazine*, vol. 51, no. 6, pp. 86–93, 2013.

[5] K. Saleem, A. Derhab, J. Al-Muhtadi, and B. Shahzad, "Human-oriented design of secure machine-to-machine communication system for e-healthcare society," *Computers in Human Behavior*, vol. 51, pp. 977–985, 2015.

[6] R. Daș and G. Tuna, "Machine-to-machine communications for smart homes," *International Journal of Computer Networks and Applications*, vol. 2, no. 4, pp. 196–202, 2015.

[7] A. Corallo, M. Lazoi, M. Lezzi, and A. Luperto, "Cybersecurity awareness in the context of the industrial Internet of things: a systematic literature review," *Computers in Industry*, vol. 137, article 103614, 2022.

[8] J. Wang, J. Chen, Y. Ren, P. K. Sharma, O. Alfarraj, and A. Tolba, "Data security storage mechanism based on blockchain industrial Internet of things," *Computers & Industrial Engineering*, vol. 164, article 107903, 2022.

[9] M. Ali, M.-R. Sadeghi, X. Liu, Y. Miao, and A. V. Vasilakos, "Verifiable online/offline multi-keyword search for cloud-assisted industrial Internet of things," *Journal of Information Security and Applications*, vol. 65, article 103101, 2022.

[10] T. Hasan, J. Malik, I. Bibi et al., "Securing industrial internet of things against botnet attacks using hybrid deep learning approach," *IEEE Transactions on Network Science and Engineering*, 2022.

[11] D. K. Sah, T. N. Nguyen, K. Cengiz, B. Dumba, and V. Kumar, "Load-balance scheduling for intelligent sensors deployment in industrial internet of things," *Cluster Computing*, vol. 25, no. 3, pp. 1715–1727, 2022.

[12] C. Liu, S. Ziwei, X. Xun, and L. Yuqian, "Service-oriented industrial internet of things gateway for cloud manufacturing," *Robotics and Computer-Integrated Manufacturing*, vol. 73, article 102217, 2022.

[13] C. Li, Y. Liu, J. Xiao, and J. Zhou, "Mceaaco-qsrp: a novel qos secure routing protocol for industrial internet of things," *IEEE Internet of Things Journal*, 2022.

[14] B. Sriramulu, P. V. Rao, D. R. Vemula, B. R. Reddy, and T. J. Lakshmi, "A secure iot-based micro-payment protocol for wearable devices," *Peer-to-Peer Networking and Applications*, vol. 15, no. 2, pp. 1163–1188, 2022.

[15] X. Ding, X. Wang, Y. Xie, and F. Li, "A lightweight anonymous authentication protocol for resource-constrained devices in

internet of things," *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 1818–1829, 2022.

[16] B. H. Taher, H. Liu, F. Abedi, H. Lu, A. A. Yassin, and A. J. Mohammed, "A secure and lightweight three-factor remote user authentication protocol for future IoT applications," *Journal of Sensors*, Article ID 8871204, 2018 pages, 2021.

[17] K. Dewangan, M. Mishra, and N. K. Dewangan, "A review: a new authentication protocol for real-time healthcare monitoring system," *Irish Journal of Medical Science*, vol. 190, no. 3, pp. 927–932, 2021.

[18] M. M. Modiri, J. Mohajeri, and M. Salmasizadeh, "A novel group-based secure lightweight authentication and key agreement protocol formachine-type communication," *Scientia Iranica*, 2021.

[19] K. Shahzad, A. O. Aseeri, and M. A. Shah, "A blockchain-based authentication solution for 6g communication security in tactile networks," *Electronics*, vol. 11, no. 9, p. 1374, 2022.

[20] S. Gupta, B. L. Parne, and N. S. Chaudhari, "ISAG: IoT-enabled and cecrecy aware group-based handover scheme for e-health services in M2M communication network," *Future Generation Computer Systems*, vol. 125, pp. 168–187, 2021.

[21] C. Wang, R. Huang, J. Shen, J. Liu, P. Vijayakumar, and N. Kumar, "A novel lightweight authentication protocol for emergency vehicle avoidance in VANETs," *IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 14248–14257, 2021.

[22] M. A. Mobarhan and M. Salamah, "REPS-AKA3: a secure authentication and re- authentication protocol for LTE networks," *Journal of Network and Computer Applications*, vol. 201, article 103345, 2022.

[23] E. Karacan, S. Akleylek, and A. Karakaya, "Pq-flat: a new quantum-resistant and lightweight authentication approach for m2m devices," in *2021 9th International Symposium on Digital Forensics and Security (ISDFS)*, pp. 1–5, Elazig, Turkey, 2021.

[24] H. Dalkilic and M. H. Ozcanhan, "A strong mutual authentication protocol for securing wearable smart textile applications," *Advances in Electrical and Computer Engineering*, vol. 22, no. 1, pp. 31–38, 2022.

[25] V. Joanne Marie, J. E. Santos, V. Pascua, and N. M. C. Tiglao, "Hardware-accelerated blockchain-based authentication for the Internet of things," in *Cognitive Radio Oriented Wireless Networks and Wireless Internet. CROWNCOM WiCON 2021 2021*, H. Jin, C. Liu, A. S. K. Pathan, Z. M. Fadlullah, and S. Choudhury, Eds., vol. 427 of Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, pp. 283–295, Springer, Cham, 2022.

[26] A. H. Aly, A. Ghalwash, M. Nasr, and A. A. El-Hafez, "A new lightweight authenticated key agreement protocol for Iot in cloud computing," *Journal of Engineering Science and Technology*, vol. 16, no. 5, pp. 3987–4005, 2021.

[27] M. M. Samy, W. R. Anis, A. A. Abdel-Hafez, and H. D. Eldemerdash, "An optimized protocol of m2m authentication for internet of things (Iot)," *International Journal of Computer Network & Information Security*, vol. 13, no. 2, pp. 29–38, 2021.

[28] C. Thammarat and C. Techapanupreeda, "A secure authentication and key exchange protocol for m2m communication," in *2021 9th International Electrical Engineering Congress (iEECON)*, pp. 456–459, Pattaya, Thailand, 2021.

[29] R. Krishnasrija, A. K. Mandal, and A. Cortesi, *A lightweight mutual and transitive authentication mechanism for iot network*.

[30] M. P. Lokhande, D. D. Patil, L. V. Patil, and M. Shabaz, "Machine-to-machine communication for device identification and classification in secure telerobotics surgery," *Security and Communication Networks*, vol. 2021, Article ID 5287514, 16 pages, 2021.

[31] X. Wang, K. Fan, K. Yang et al., "A new RFID ultra-lightweight authentication protocol for medical privacy protection in smart living," *Computer Communications*, vol. 186, pp. 121–132, 2022.

[32] M. Hosseinzadeh, O. H. Ahmed, S. H. Ahmed et al., "An enhanced authentication protocol for RFID systems," *IEEE Access*, vol. 8, pp. 126977–126987, 2020.

[33] J. Ryu, O. Jihyeon, D. Kwon et al., "Secure ecc-based three-factor mutual authentication protocol for telecare medical information system," *IEEE Access*, vol. 10, pp. 11511–11526, 2022.

[34] Q. Fan, J. Chen, M. Shojafar, S. Kumari, and D. He, "SAKE∗: a symmetric authenticated key exchange protocol with perfect forward secrecy for industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6424–6434, 2022.

[35] A. Adeel, M. Ali, A. N. Khan et al., "A multi-attack resilient lightweight IoT authentication scheme," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 3, article e3676, 2022.

[36] E. Lara, L. Aguilar, M. A. Sanchez, and J. A. Garcia, "Lightweight authentication protocol for m2m communications of resource-constrained devices in industrial internet of things," *Sensors*, vol. 20, no. 2, p. 501, 2020.

[37] A. Esfahani, G. Mantas, R. Matischek et al., "A lightweight authentication mechanism for m2m communications in industrial iot environment," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 288–296, 2019.

[38] S. Panda, S. Mondal, and N. Kumar, "SLAP: a secure and lightweight authentication protocol for machine-to-machine communication in industry 4.0," *Computers & Electrical Engineering*, vol. 98, article 107669, 2022.

[39] P. Syverson and I. Cervesato, "The logic of authentication protocols," in *Foundations of Security Analysis and Design. FOSAD 2000*, R. Focardi and R. Gorrieri, Eds., vol. 2171 of Lecture Notes in Computer Science, pp. 63–137, Springer, Berlin, Heidelberg, 2000.

[40] S. Rostampour, N. Bagheri, Y. Bendavid, M. Safkhani, S. Kumari, and J. J. P. C. Rodrigues, "An authentication protocol for next generation of constrained Iot systems," *IEEE Internet of Things Journal*, 2022.