iJRASET
International Journal For Research in
Applied Science and Engineering Technology

Volume: 12 Issue: VI Month of publication: June 2024

DOI: https://doi.org/10.22214/ijraset.2024.62939

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue VI June 2024- Available at www.ijraset.com

An ECC based Secure Authentication Protocol for M2M Communication in Industrial IOT Edge Device

Hirendra Singh Sengar
SOS in Computer Science and Applications, Jiwaji University, Gwalior, India

Abstract: Machine to machine communication in industrial IoT is becoming a common phenomenon now-a-days where all the machines like IoT edge devices communicate independently. It's always been a challenging task to create a protocol for authentication that is both secure and efficient with regard to resources for use in industrial IoT environments' M2M communication. Here we present a safe and effective authentication method that makes use of elliptic curve cryptography (ECC). We demonstrated these vulnerabilities by testing the current protocol with various assaults, including SSL strip attack and MITM attack. To further evaluate the protocol's security against various IoT threats, we employed both official and informal approaches, such as the Scyther tool and BAN logic. Secured from a variety of IoT Threats, such as Man-in-the-middle (MITM) attacks, replay attacks, and impersonation attacks, our suggested protocol, absence of device anonymity, weak mutual authentication etc. but also suitable for resource constraint IoT edge devices like RFID devices, sensors, routers and gateways.

Keywords: Authentication, Elliptic curve cryptography, Internet of Things (IoT), Security, Scyther tool, BAN logic

I. INTRODUCTION

Edge computing is the process where computing like's data analysis and handling operations are performed at the network edge rather than going through any centralized system. In edge computing the optimization of resources is also performed at the network edge by optimizing the IoT devices' operations and installed applications which produce the data (Cao et al. 2020). Now-a-days, implementation of edge computing in industrial area is becoming much popular. Industrial edge computing could be very much profitable for all the industries where the optimization is aimed to be achieved in the production and maintenance activities. Moreover, this technology can also be considered as an advantage for the IoT network, implemented in the industrial communication environment. Edge computing could be game changing for industrial IoT in many ways whether it is driving automation or the automation of maintenance procedures. Driving automation is completely achieved by the integration of edge computing in IoT devices and processes. Generally, in the industrial scenario, large datasets are produced by the IoT devices and applications. These data sets are then sent to the central server for further analysis. Afterwards, necessary actions take place based on the decision made by the analysis results generated by the central server. Using edge computing this processing time can be reduced through automation where decisions are taken on the basis of analysis performed on the network edge rather than center server. Edge computing also offers the predictive maintenance where devices have the capability to seek the recharge points whenever necessary (Cao et al. 2020). Edge computing uses IoT edge devices for collecting, processing and analyzing the data. The set of IoT edge devices has a vast range of devices including sensors, actuators, routers and gateways. These devices keep on communicating with each other to achieve all the operational goals of industry. This led to the emergence of communication between machines. Data exchange between machines.is a process where Industrial IoT edge

Collaborative data production and exchange occurs between devices completely autonomously. Its cheap cost and great efficiency made it the most popular and successful method in no time (Imani et al. 2018). Despite the many benefits, the main concern in machine-to-machine communication networks is the physical safety of people, industrial assets, and Internet of Things (IoT) edge devices (Sadeghi et al. 2015). Any time two edge devices in an industrial IoT network want to exchange data, they must first create a session key and start talking over the channel. Unfortunately, any malevolent party may conceivably jump in at any moment, listen in, and pose as a genuine user to hijack the conversation. By compromising other IoT devices, this malicious actor can acquire sensitive personal information. For secure connection, both devices intentionally generated the session key; however, this attacker can also hijack it. An adversary can achieve all of these aims using attacks like SSL strip, man-in-the-middle, replay, impersonation, etc.

Anyone can listen in on any M2M chat between two industrial IoT edge devices if they manage to get their hands on the session key. Researchers have occasionally proposed various authentication systems that can enable secure communication as solutions to these problems. Authentication ensures data integrity by screening incoming machine-to-machine communications. Unfortunately, due to inadequate security and their inherent complexity, current authentication systems do not meet modern industry standards. Industrial IoT devices are susceptible to replay attacks, impersonation attacks, and man-in-the-middle (MITM) attacks due to their limited memory and processing capabilities.This is an extremely important matter. Using elliptic curve cryptography, our novel, lightweight authentication approach surpasses the previous one in terms of security and usability. Public key elliptic curve cryptography is just as secure as RSA, but it doesn't require bigger keys. For low-powered devices, ECC might be the best encryption method because it safeguards data by computing points on an elliptic curve. It provides more robust security with less keys. In mathematics, a finite field is a structure with an infinite number of elements. Assuming that y2=x3+ax+b and that a and b are elements of Fp, an elliptic curve EFp is the set of all (x,y) pairs in Fp × Fp. The requirement that 4a3+27b ≠ 20 [4] also needs to be satisfied. Find the smallest positive integer n such that for all generators G and all cyclic subgroups EG, the product of n and EG yields the identity element O. Attacks like replay, impersonation, and man-in-the-middle (MITM) can exploit the limited memory and processing capability of industrial IoT devices. sequence of EG. More often than not, it is a prime number. According to Rostampour et al. (2020), it is mathematically difficult to calculate x for a specific set of parameters G, P, and EG. However, for any given location, it is easy to calculate P = x × G as long as n is big enough.We found several security holes in the existing M2M authentication process and fixed them by suggesting a more robust method based on ECC. Here are some important points that this paper brings up:

1) There is no guarantee of security when transmitting encrypted data across the secure channel. Thus, we have proven that the existing protocol is vulnerable to the SSL strip attack.
2) Furthermore, the cryptanalysis technique demonstrates that the existing mechanism is susceptible to man-in-the-middle attacks, enabling an assailant to effortlessly pilfer the session key and compromise the entire security.There is no guarantee of security when transmitting encrypted data across the secure channel. Thus, we have shown that the existing protocol is vulnerable to the SSL strip attack.

Our ECC-based protocol is more secure and efficient, and it can help you solve these security challenges. To do both official and informal evaluations Utilizing BAN logic and the Scyther tool, we evaluate the security robustness of the proposed protocol. The following elements make up the presentation of the document. In Section 2, the article provides a brief overview of the relevant literature. In Section 3, the study analyzes the current protocol's security. Section 4 explains the proposed protocol's authentication method; Section 5 evaluates the protocol's security; Section 6 calculates the protocol's performance in terms of overhead and costs; and Section 7 ends the study 7.

II. RELATED WORKS

Delivering and routing messages among several IoT edge devices with limited resources is what machine-to-machine communication is all about in an industrial IoT context. They are ill-suited to the internet's existing standard protocols. Some form of lightweight protocols is usually required. In industrial IoT, 6LoWPAN, MQTT, AMQP, CoAP, XMPP, and DDS are among the most popular lightweight protocols for machine-to-machine communication. To keep tabs on production, 6LoWPAN protocol communication links internet-connected intelligent sensors, controllers, and actuators that communicate with other machines (M2M devices) (Shelby et al. 2014) Researchers Granjal et al. There are certain security concerns, but overall, the protocol is useful for easing communication. Many new research proposals have surfaced in an effort to address its shortcomings. In their 2013 study, Hussen et al. introduced This paper presents SAKES, a key establishment system that prioritizes security, and describes how the researchers used BAN logic and the Scyther tool to evaluate the protocol's security. The paper starts with a brief literature review in Section 2, then moves on to analyze the existing protocol's security in Section 3. Section 4 explains the proposed protocol's authentication mechanism, and Section 5 evaluates the protocol's security. Section 6 compares the protocol's performance in terms of costs and overhead, and Section 7 concludes the paper 7. that utilizes two distinct cryptographic techniques, to achieve their security objectives. Utilization of symmetric keys occurs during the authentication process. encryption, whereas the key setup phase generates a session key between devices and the server using an asymmetric encryption key technique based on ECC.

Qiu et al. (2016) proposed a three-step procedure for 6LoWPAN networks, which were pre-deployment, handover, and key establishment and mutual authentication (AKE). Their proposed protocol may be resilient to attacks including replay, impersonation, hacking, and man-in-the-middle, according to the security research. One more lightweight protocol that follows 6LoWPAN is Message Queue Telemetry Transport, also known as MQTT.

MQTT's recent meteoric rise in popularity is due, in no small part, to the fact that it has almost no impact on the environment. As part of the Internet of Things paradigm in industry, it facilitates communication between machines. Internet of Things devices can communicate with one another via the MQTT protocol. by utilizing multiple routing techniques, hence facilitating machine-to-machine (M2M) communication. Everyone taking part in this protocol. is either a publisher, a subscriber, or a broker. Login credentials, such as a username and password, are required by MQTT for authentication and data security purposes. Additionally, it employs SSL/TSL encryption (TLS/SSL-MQTT security principles online: HiveMQ, 4 October 2023). Several security issues are resolved when MQTT is used in the IoT. With the addition of SSL/TLS and session key management, MQTT became more secure (Wells et al. 2016). Their suggested Secure MQTT protocol relies on lightweight attribute encryption (ABE) and is stable, efficient, and extensible (Goyal and associates, 2006; Bethencourt and associates, 2007). Not only that, but AMQP is another example of a point-to-point protocol that follows the publish-subscribe model. A few of the many features offered by the AMQP include message storage and routing. According to Roebuck (2012), AMQP enhances security by utilizing SASL authentication and TLS encryption. The request-response protocol CoAP is another one that makes it to the queue. Datagram Transport Layer Security (DTLS) handles packet loss, reorders messages, and decreases their size to guarantee data integrity. Due to the high number of message exchanges required to establish a secure session, the communication cost increases. As a workaround for CoAP's DTLS issue, Raza et al. (2013) proposed a lightweight approach. This method reduces overheads by compressing the DTLS. Through their performance analysis, Dunkels et al. (2004) proved that this suggested method was advantageous in terms of processing speeds, energy usage, and network-wide reaction times. In their 2015 paper, Kalra et al. proposed an HTTP cookie-based two-way authentication mechanism for usage with embedded devices and cloud servers. To ensure the safety of the suggested protocol, we ran it using the AVISA tool. From kalra et al. [16], Kumari et al. (2018) derived a method that still has many flaws, such as offline password guessing attacks, insider assaults, device anonymity, and robust mutual authentication. Also, someone proposed adopting an ECC based authentication mechanism to further increase security. Next, Rostampour et al. (2020) compared the two approaches' protections against man-in-the-middle attackss. conducted a traceability and attack analysis, exposing their vulnerabilities. Rostampour et al. published a unique ECC-based authentication system for Bluetooth sensors and RFID devices in 2020. We have evaluated the robustness of the suggested protocol against a number of Internet of Things (IoT) threats using both formal and informal security evaluations.

.

III. SECURITY ANALYSIS OF PREVIOUS PROTOCOL

A. SSL strip attack

In Esfahani et al. (2019) protocol, without masking, all messages are transmitted in encrypted form across a secure channel during the registration step. The SSL layer, in conjunction with SSL certificates, often establishes a secure connection, which safeguards data while it is in transit. The SSL certificate converts the connection to HTTPS. In contrast to https, which encrypts all data in transit, HTTP is insecure since it transfers data in an unencrypted format. Although SSL striping gets around HTTPS's man-in-the-middle protection, which makes data unreadable to the attacker, it is still there. By using SSL striping, an HTTPS connection becomes an unprotected http connection. An assault known as man-in-the-middle occurs during the striping process. These exploits strip a connection's SSL encryption and replace it with the weaker HTTP. The perpetrator accomplishes this by assuming a central position within the network. All of the striping takes place during the unencrypted TCP hello. A malicious actor can prevent a legitimate user from accessing https://examplewebsite.com by intercepting the request before it reaches the server. As soon as the malicious actor gets the SYN/ACK message, they send the same message back to the server. As soon as the server gets the request from the website, it will transmit this message to the client. This bad guy has forgone the safer https connection in favor of the more vulnerable http one, such as http://examplewebsite.com. Each and every one of the The change from https to http in the connection could put data at risk of a man-in-the-middle attack. According to Sendong Zaho et al. (2012), this type of data comprises sensitive information such as login credentials, bank account details, and any other personal details of clients. Esfahani et al. (2019) found that an attacker can easily eavesdrop on the communicated smart sensor ID and parametric values of functions f2=h(f1), f3=PSK f1 during the registration step by utilizing a man-in-the-middle attack that takes use of an SSL strip attack.

The attacker can launch further attacks, like man-in-the-middle or password guessing, on top of the initial one.on the basis of the sensor's unique identifier and other information etc.

B. MAN-in-the-middle attack

This type of assault occurs when an adversary intercepts a communication channel and then sits in the middle of it, learning all the secret information. In order to execute a man-in-the-middle assault, the attacker follows these steps:

1) An adversary intercepted the communication using SSL strip attack. He eavesdropped the smart sensor’s ID and other information sent by the server during the registration phase.
2) The smart sensor creates a random number R1 and computes the values of M1= (h (f2i) ⊕ R1), AIDi= (h (R1) ⊕ IDi), and M2=h (R1 || M1 || AIDi) before sending message 3—the authentication requirement—to the router. This process shares a new secret key, SK, with the router
3) The enemy eavesdrops on every passing value by intercepting the public channel. He keeps track of M1, AIDi, M2, and f3i values. By changing the formula M1=(h(f2i) □ R1) to R1=(h(f2i) □ M1) (because if x=y □ z, then z=y □ x), and entering the values of M1 and f2i, he can now extract the random number R1. The attacker then passes message 3 to the router.
4) After receiving message 3, the router calculates f1i=f3i□PSK, R1=(h(f2i)□ M1), IDi=AIDj□h(R1), and verifies that h(R1 || M1 || AIDi) = M2 is true.
5) After generating a random number, R2, the router computes AIDj=R2 □ h(IDi), M1| = f1i ⊕ h(IDi), M2| = h(M1| || SKij = h(R1 || R2) and AIDj || R2). After that, it sends message 4 to the smart sensor along with M1|, M2|, and AIDj. 6. An adversary eavesdrops on all of the passing values from message 4 by intercepting the public channel. By changing the formula AIDj=R2 □ h(IDi) to R2= AIDj □ h(IDi), he can now extract the random number R2. The adversary already knows ID thanks to the SSL strip attack. The adversary then sends message 4 to the smart sensor.
6) After receiving message 4, the smart sensor computes R2= AIDj   h(IDi) and verifies that h(R2 || M1| || AIDi) = M2| is true. It computes M1|| = SKij ⊕ h(R2), creates the session key SKij = h(R1 || R2), and transmits M1|| as message 5 to the router.
7) The router receives the message 5 and checks $M_1^{\|} \oplus SK_{ij}$ = h(R2) which holds true. It means that the legitimate key is held my smart sensor. Any further message sent between the smart sensor and router is encrypted by the session key SK_{ij}.
8) The adversary is now able to get the secrets as random numbers by implementing the SSL strip and MITM attack successfully. He is now able to generate the session key by using the formula SK_{ij} =h(R1 || R2). Any encrypted message sent between the smart sensor and router can easily be decrypted by this session key and attacker can read it properly. Both the smart sensor and router believes that they have the strong connection and they communicate freely without knowing the fact that the man sits in between is able to read all the confidential messages.

C. *Replay attack*

which includes The channels over which real smart sensors communicate with routers include M1, M2, f3i, and AIDi. The attacker can ascertain the value of the smart sensor's due to its computation using AIDi, a random integer R1, and a hash algorithm. The enemy may be able to listen in on your conversation. on communication 3 and relay it to the router. As soon as the router gives him permission, his replay attack will begin. When the real router communicates with the smart sensor by message 4 (M1|, M2|, AIDi), the same thing happens. The adversary is able to ascertain the AIDj value by knowing the smart sensor ID in addition to the random integer R2. The adversary can prevent message 4 from reaching the sensor by sending it back. After the sensor verifies the authentication request, he can easily initiate the replay attack.

D. *Impersonation attack*

An attacker pulls off a successful impersonation of a legitimate protocol participant in this attack. By utilizing the replay attack, an attacker is able to effectively mimic a valid sensor after sending three messages (f3i, AIDi, M1, M2, and M2) to the router. Following the smart sensor message 4 (M1|, M2|, AIDi) transmission, an adversary can successfully impersonate the legitimate router once again using the same replay attack.

E. *Absence of device anonymity*

Sending the smart sensor device's ID in plain text without masking compromises device anonymity in the current protocol.
Even though Esfahani et al. (2019) used a secure channel to transmit information, an adversary could still simply eavesdrop on the connection and get the smart sensor's ID if they used an MITM attack or SSL strip.

F. *Mutual authentication*

The current protocol's level of mutual authentication between routers and smart sensors is insufficient. Both the sensor and router sides conduct validation tests to achieve mutual authentication. In response to message 3, the router verifies whether h(R1 || M1 || AIDi) = M2 is true, and in response to message 4, the smart sensor verifies if h(R2 || M1| || AIDi) = M2 is true.

In this case, mutual authentication is relian. on the values of random numbers R_1 and R_2. The adversary can easily bypass this mutual authentication security by knowing the random numbers R_1 and R_2 after implementing the MITM attack so the existing protocol does not ensures the strong mutual authentication.

IV. PROPOSED METHODOLOGY

As we have seen in predecessor protocol that a hash function does not give guarantee of security thus we have eliminated the use of hash function in our proposed protocol. Elliptic curve cryptography is enough here to provide the higher level of security. Our protocol comprises two stages: registration is the first, and authentication is the second. For the sake of argument, let's pretend that everyone is in agreement over the parameters of the elliptic curve.

A. *Registration phase*

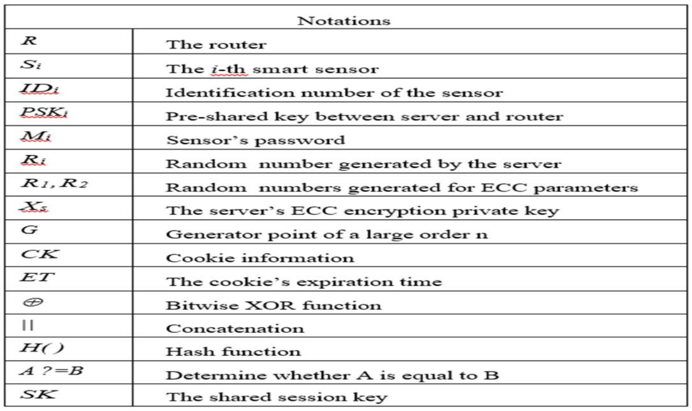
Our protocol works in two phases, registration to initiate a communication smart sensor you need to sign up for the server. All of this signup business takes place through an encrypted connection. Numerous earlier protocols have made use of this technique, including (Esafani et al. 2019), (Wang et al. 2017), and (Rostampour et al. 2020).but the registration process over secure channel alone does not give guarantee of the security. As we have seen in the security analysis of existing proposals that the smart sensor ID can easily be traced by the attacker by implementing the SSL strip attack so we need something more than the simply sending ID over secure channel. Any device ID sent in the plain text without masking is easily traceable by the attacker. In Kalra et al. (2015) protocol unique identifier of the device is sent in plain text without masking. This ID can be easily traced by many attackers so ID must be sent with masking (Rostampour et al. 2020). Registration process of our protocol is as follows: Smart sensor generates random number X_i, computes masked ID as $SID_i=X_i \oplus ID_i$ and then sends this masked ID to authentication server. Authentication server then generates a random number R_i, computes CK_i as $CK_i=(R_i \oplus X_s \oplus SID_i \oplus ET)$ and $CK^{|}_i$ as $CK^{|}_i=CK_i * G$. Now, the server sends $CK^{|}i$ to the smart sensor. Server stores SID_i, CK_i and smart sensor stores SID_i and $CK^{|}_i$.

B. *Authentication phase*

During the authentication stage, intelligent. $sensor_i$ generates a random number R_1 and computes some security parameters as $M_1 = R_1 * G$, $M_2 = R_2 * CK^{|}_i$ and $AID_i = SID_i * G$. Afterwards, smart $sensor_i$ sends AID_i, M_1, M_2 to $router_i$ for authentication. $Router_i$ fetches CK_i and ET from server based on pre-shared key PSK_i and validates ET. To authenticate smart sensor, $router_i$ computes $M_2^{|} = CK_i * M_1$ and checks whether $M_2^{|} = M_2$, if it is true router authenticates smart sensor as legitimate device. Once the verification is done, the $router_i$ generates a random number R_2 and computes $M_3 = R_2 * G$ and $M_4 = R_2 * AID_i$. Afterwards, the $router_i$ sends M_3 and M_4 to $sensor_i$.

$Sensor_i$ receives M_3 and M_4 and computes $M^{|}_4= SID_i * M_3$ and verifies if M|4 = M4 is true. If it is, the sensor verifies the router's identity, computes Vi = SIDi and CK|i, and transmits this information to the router. Following receipt of the Vi, the routeri computes M5 = CKi * G and V|i = SIDi * M5. The router then verifies that V|i = Vi is valid and authenticates the smart sensor as a genuine device. The router's and smart sensor's mutually agreed upon secret key is SK=R1*M3=R2*M1. There has been representation of the registration and authentication process in figure 1 and 2.

TABLE 1 – NOTATIONS USED DURING THE IMPLEMENTATION OF PROTOCOL



Notations	
R	The router
S_i	The *i*-th smart sensor
ID_i	Identification number of the sensor
PSK_i	Pre-shared key between server and router
M_i	Sensor's password
R_i	Random number generated by the server
R_1, R_2	Random numbers generated for ECC parameters
X_s	The server's ECC encryption private key
G	Generator point of a large order n
CK	Cookie information
ET	The cookie's expiration time
$\oplus$	Bitwise XOR function
||	Concatenation
$H()$	Hash function
$A ?=B$	Determine whether A is equal to B
SK	The shared session key

TABLE 1 – NOTATIONS USED DURING THE IMPLEMENTATION OF PROTOCOL

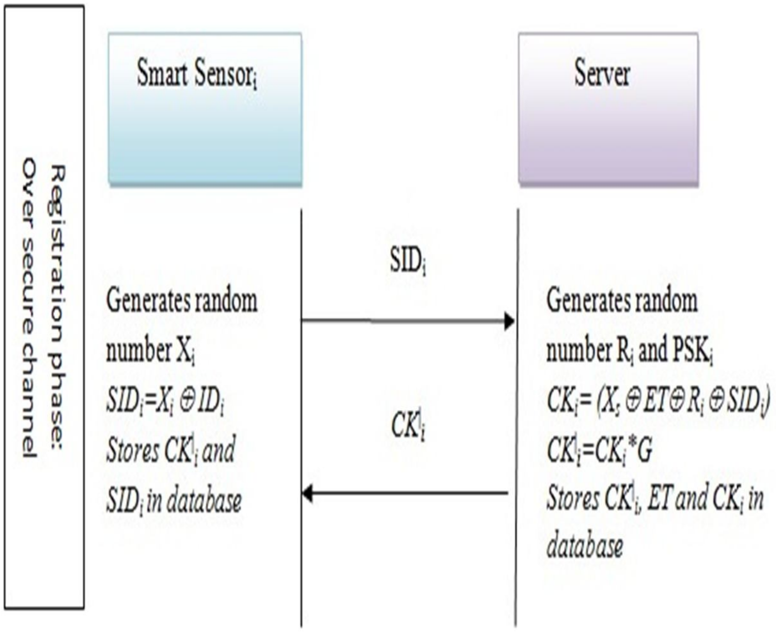

Registration phase:
Over secure channel
Smart Sensor$_i$
Server
SID$_i$
CK$^|_i$
Generates random number X$_i$
$SID_i=X_i \oplus ID_i$
Stores CK$^|_i$ and SID$_i$ in database
Generates random number R$_i$ and PSK$_i$
$CK_i=(X_i \oplus ET \oplus R_i \oplus SID_i)$
$CK^|_i=CK_i*G$
Stores CK$^|_i$, ET and CK$_i$ in database

Fig. 1 registration phase of the proposed ECC protocol

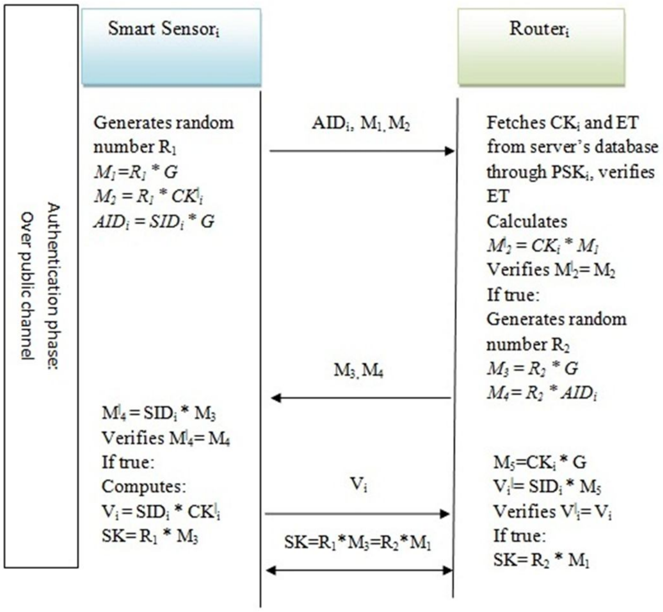


Fig. 2 authentication phase of the proposed ECC protocol

V. SECURITY ANALYSIS OF THE PROPOSED PROTOCOL

Here, we assess the confidentiality of the security parameters and confirm that the suggested protocol is resistant to replay, impersonation, SSL strip, man-in-the-middle, and other attacks. Additional security measures include device anonymity and mutual authentication etc.

A. *Informal security analysis*

The prior protocol's security analysis showed that an attacker could still listen in on the conversation and decipher the encrypted communications. This part ensures that all parameters (SID, CKi, CK|i) transferred between the smart sensor and router are secure by analyzing the level of security.

1) *SSL strip attack:* Registration in the technique of Esfahani et al. (2019) involves the maskless transfer of the smart sensor's ID and functional parameters f2, f3. A malicious actor executing an SSL strip attack might readily intercept these values. Using the intercepted data to launch a man-in-the-middle exploit. is one possible application. ID and other parameters to get access to even more sensitive information. By using a combination of a random number and ECC points, our suggested protocol conceals the smart sensor's and router's ID and other parameters from direct transmission. Protecting sensitive information by preventing an attacker from seeing them directly is what masking is all about (rostampur et. al. 2020). The suggested protocol is immune to SSL attacks since, even if an attacker were to succeed in decrypting an HTTPS connection and re-establishing it as an HTTP connection, the ECC's masking and stringent security measures would prevent the attacker from deciphering the parameters in plain text.
2) *Man-in-the-middle attack:* Based on what we learned from the prior protocol's security study, a third party can guess the session key by mining the sent messages for random numbers. In order to create the authentication security parameters, our suggested smart sensor and router both utilize scalar multiplications over an elliptic curve, with M1 = R1 * G and M3 = R2 * G, respectively. For the purpose of mutual authentication, the smart sensor and router communicate these parameters via messages. In order to determine the worth of these messages' random integers, the adversary would have to tackle the infeasible task of obtaining x from xG along an elliptic curve. The suggested protocol is safe from man-in-the-middle attacks since an attacker cannot determine the session key due to the lack of random numbers.
3) *Replay attack:* To circumvent security measures, an attacker can launch a replay attack, which entails storing the transferred messages and then sending them back to the communication parties. Since the smart sensor and router randomly participate in each session, our proposed protocol renders every eavesdropped session meaningless. It is possible for the attacker to get the messages M3| and M4|, based on a newly created value R2|, by sending the cached messages AIDi, M1, and M2 to the router in place of a legitimate sensor. An adversary can bypass protocol security measures by successfully implementing a replay attack by replaying Vi = SIDi * CK|i to the router. This happens when the adversary knows the session key, which in turn requires knowing the values of passwords M1 and M3, even though it has other values, such as M3| and M4|. Even if it were to intercept the channel directly and learn the values of M1 and M3, it would still be unable to calculate the session key SK (R1*M3 or R2*M1) without knowing the values of R1 and R2. This disproves the possibility of a replay attack on our suggested protocol.
4) *Impersonation attack:* Since the session key is important, the objective of this kind of attack is to impersonate a valid device. for smart sensork is unavailable in our proposed protocol, another smart sensor cannot impersonate the smart sensori. Due to its lack of knowledge regarding the random numbers utilized in session key construction, smart sensori cannot access session keys issued by routeri. When routerk tries to pose as a legitimate routeri, the same thing occurs.
5) *Absence of device anonymity:* Our suggested protocol protects the device's anonymity by masking the smart sensor's ID before transmission, rather than sending it in plain text. No channel interceptor using an SSL strip attack and an MITM assault will ever be able to deduce the sender's identity from the data packets.
6) *Mutual authentication:* Our proposed approach increases the degree of smart sensor mutual authentication. and the router is sufficient. Validation tests conducted at the sensor and router ends allow for mutual authentication. In the same way that the router verifies the truth of CKi * M1 = M2 upon receipt of message 3, the smart sensor verifies the truth of SIDi * M3=M4 upon receipt of message 4. In this case, the values of SID and CKi directly affect the likelihood of mutual authentication. The proposed protocol guarantees strong mutual authentication by making it so that an opponent, even via a man-in-the-middle attack, cannot circumvent it without knowing the security parameters SIDi and CKi.

B. Formal security analysis

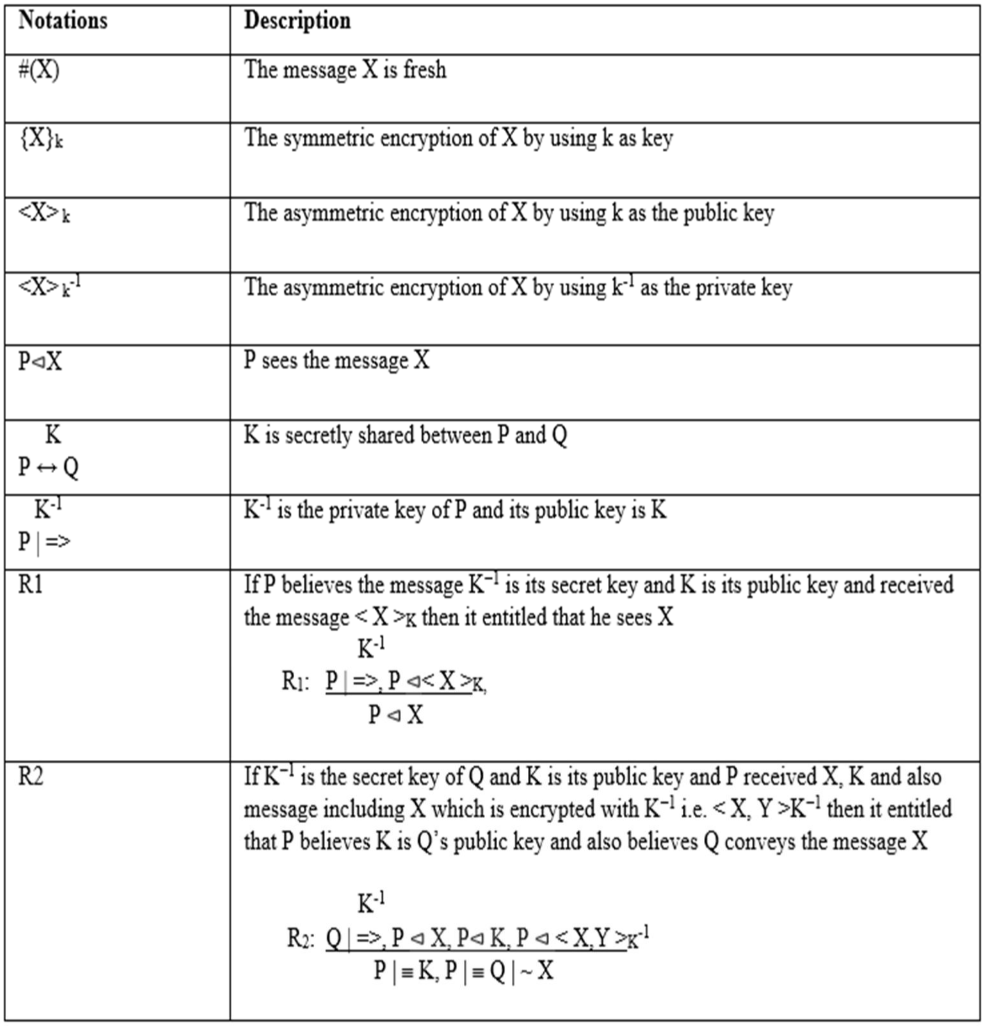
Research has yielded a plethora of security analyzers, some of which are available in manual form (e.g., GNY logic; Burrows et al., 1989; Syverson and Cervesato, 2000) and others in automatic tool form (e.g., Scyther; Cremers, 2008; Blanchet, 2007). Here, we 1) assess the suggested protocol's security utilizing manual and automated methods, namely BAN logic and the Scyther tool.

1) *Security evaluation by BAN logic:* An evaluation of security utilizing BAN logic has begun, with multiple steps now underway. As illustrated in Table 3, the initial step is to represent all the messages using the BAN logic form. The following stage involves idealizing a subset of the proposed protocol's messages. This step involves removing plaintext messages that do not contribute to enhancing secrecy. In Table 4 we can see the idealized message format. In addition, Table 5 details all of the security goals and assumptions.

From these idealized messages, we can infer security goals, and Table 6 shows the assumptions made by the protocol. Table 2 displays all of the BAN logic rules utilized in this paper. The value of CKi is known to the router, denoted as D1: R ⊋ CKi, according to Table 6, which is based on assumption A5. From R2 and A2, D1, and IM1, we can infer that D2: R }≡ M1 is true. This signifies that the router now considers M1 to be the genuine smart sensor public key, confirming the achievement of security goal G1. In elliptic curve cryptography, the router generates a unique identifier (R2) that serves as its private key. When combined with the smart sensor's unique identifier (M1), this forms the shared session key (SK=R2*M1). In light of assumption A6, it is evident that the smart sensor knows the value of SIDi, denoted as D3: Si ⊋ SIDi. From R2, we may infer that D3 is D4: Si } ≡ M3 by utilizing A3, D3, and IM2. Having accomplished security target G2, the smart sensor now accepts M3 as the router's genuine public key. The formula for the smart sensor's shared session key with the router is SK=R1*M3. Here, R1 is the smart sensor's produced private key in ECC.

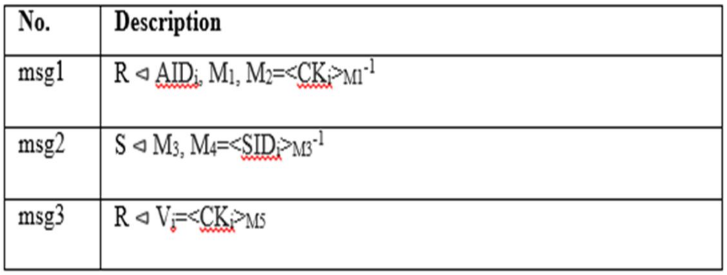
2) *Security evaluation by scyther tool:* All the security claims of the protocol have been proved using Scyther tool (Cremers, 2008). It is an automated tool used for the verification of all claims generated in the security protocols. If any attack is found during the verification, the whole scenario is depicted by a picture graphically. Moreover, it is used to verify the authentication and secrecy in the different IoT protocols. Our proposed protocol analyzed using Scyther tool and described in the figure 3 with SPDL representation. It can be easily seen in the figure that this protocol is attack free within all given bounds.

TABLE 2 BAN LOGIC RULES AND NOTATIONS



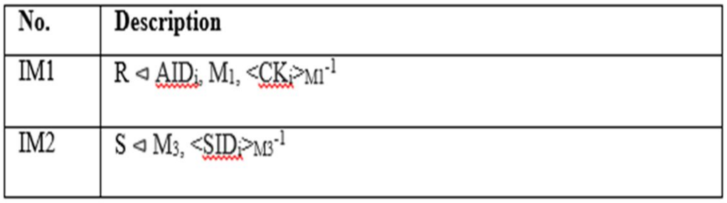
Notations	Description
#(X)	The message X is fresh
$\{X\}_k$	The symmetric encryption of X by using k as key
$<X>_k$	The asymmetric encryption of X by using k as the public key
$<X>_{k^{-1}}$	The asymmetric encryption of X by using k^{-1} as the private key
$P \lhd X$	P sees the message X
$P \overset{K}{\leftrightarrow} Q$	K is secretly shared between P and Q
$\overset{K^{-1}}{P \mid =>}$	K^{-1} is the private key of P and its public key is K
R1	If P believes the message K^{-1} is its secret key and K is its public key and received the message $< X >_K$ then it entitled that he sees X $R_1: \dfrac{\overset{K^{-1}}{P \mid =>}, P \lhd < X >_K,}{P \lhd X}$
R2	If K^{-1} is the secret key of Q and K is its public key and P received X, K and also message including X which is encrypted with K^{-1} i.e. $< X, Y >K^{-1}$ then it entitled that P believes K is Q's public key and also believes Q conveys the message X $R_2: \dfrac{\overset{K^{-1}}{Q \mid =>}, P \lhd X, P \lhd K, P \lhd < X,Y >_{K^{-1}}}{P \mid \equiv K, P \mid \equiv Q \mid \sim X}$

TABLE 3 EXPRESSING THE MESSAGES OF THE PROTOCOL



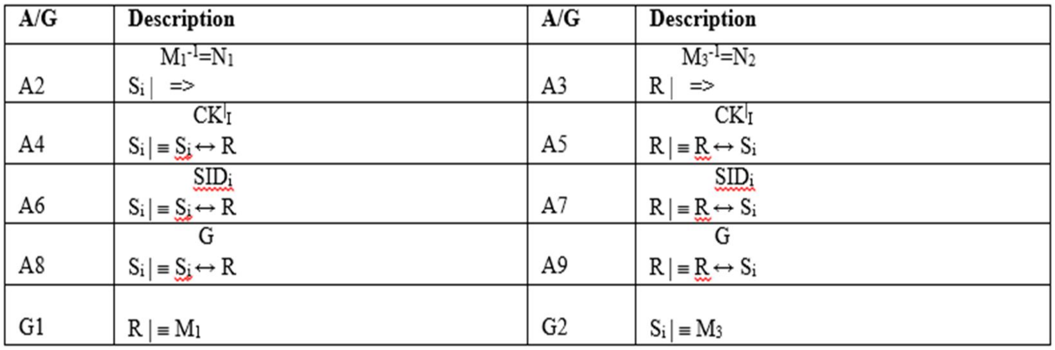
No.	Description
msg1	$R \lhd AID_i, M_1, M_2=<CK_i>_{M1}{}^{-1}$
msg2	$S \lhd M_3, M_4=<SID_i>_{M3}{}^{-1}$
msg3	$R \lhd V_i=<CK_i>_{M5}$

TABLE 4 IDEALIZATION OF THE SELECTED MESSAGES



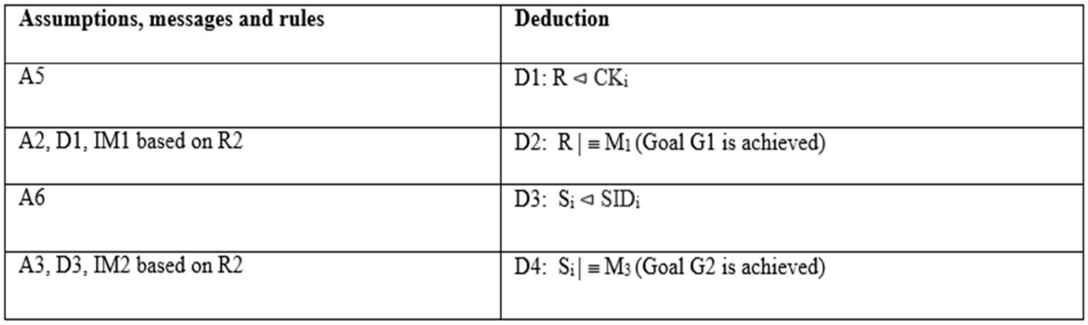
No.	Description
IM1	$R \lhd AID_i, M_1, <CK_i>_{M1}{}^{-1}$
IM2	$S \lhd M_3, <SID_i>_{M3}{}^{-1}$

TABLE 5 ASSUMPTIONS AND SECURITY GOALS

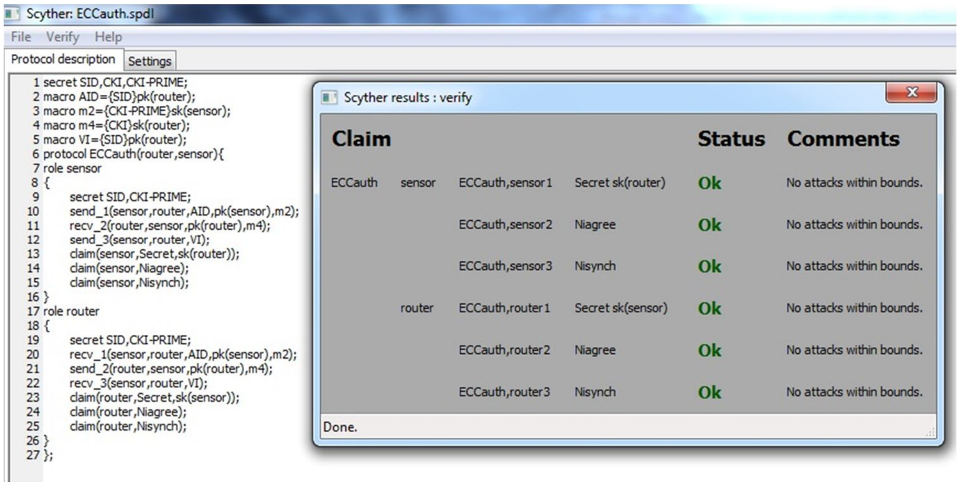


A/G	Description	A/G	Description
A2	$S_i \mid \overset{M_1{}^{-1}=N_1}{=>}$	A3	$R \mid \overset{M_3{}^{-1}=N_2}{=>}$
A4	$S_i \mid\equiv S_i \overset{CK\vert_I}{\leftrightarrow} R$	A5	$R \mid\equiv R \overset{CK\vert_I}{\leftrightarrow} S_i$
A6	$S_i \mid\equiv S_i \overset{SID_i}{\leftrightarrow} R$	A7	$R \mid\equiv R \overset{SID_i}{\leftrightarrow} S_i$
A8	$S_i \mid\equiv S_i \overset{G}{\leftrightarrow} R$	A9	$R \mid\equiv R \overset{G}{\leftrightarrow} S_i$
G1	$R \mid\equiv M_1$	G2	$S_i \mid\equiv M_3$

TABLE 6 SECURITY GOALS DEDUCTION OF PROPOSED PROTOCOL



Assumptions, messages and rules	Deduction
A5	D1: $R \lhd CK_i$
A2, D1, IM1 based on R2	D2: $R \mid\equiv M_1$ (Goal G1 is achieved)
A6	D3: $S_i \lhd SID_i$
A3, D3, IM2 based on R2	D4: $S_i \mid\equiv M_3$ (Goal G2 is achieved)



```
secret SID,CKI,CKI-PRIME;
macro AID={SID}pk(router);
macro m2={CKI-PRIME}sk(sensor);
macro m4={CKI}sk(router);
macro VI={SID}pk(router);
protocol ECCauth(router,sensor){
role sensor
{
	secret SID,CKI-PRIME;
	send_1(sensor,router,AID,pk(sensor),m2);
	recv_2(router,sensor,pk(router),m4);
	send_3(sensor,router,VI);
	claim(sensor,Secret,sk(router));
	claim(sensor,Niagree);
	claim(sensor,Nisynch);
}
role router
{
	secret SID,CKI-PRIME;
	recv_1(sensor,router,AID,pk(sensor),m2);
	send_2(router,sensor,pk(router),m4);
	recv_3(sensor,router,VI);
	claim(router,Secret,sk(sensor));
	claim(router,Niagree);
	claim(router,Nisynch);
}
};
```

```
Claim                                                              Status   Comments

ECCauth   sensor   ECCauth,sensor1   Secret sk(router)             Ok       No attacks within bounds.
                   ECCauth,sensor2   Niagree                       Ok       No attacks within bounds.
                   ECCauth,sensor3   Nisynch                       Ok       No attacks within bounds.
          router   ECCauth,router1   Secret sk(sensor)             Ok       No attacks within bounds.
                   ECCauth,router2   Niagree                       Ok       No attacks within bounds.
                   ECCauth,router3   Nisynch                       Ok       No attacks within bounds.
Done.
```

Fig. 3 Verification results with SPDL representation

VI. PERFORMANCE EVALUATIONS

The performance of the proposed work can be evaluated in terms of communication, computation and storage overheads.

A. *Communication overhead*

Here we'll pretend that the five messages sent out during authentication and registration are Message1, Message2, Message3, Message 4, and Message 5. Message 2 contains the security parameter CK|i, and Message 1 contains the masked id SIDi of the smart sensor. Included in Message 3 is AIDi, with |M1|=|R1 * G|, |M2|=| R1 * CK|i|, and |AIDi|= |SIDi * G|. Also included in Message 4 are M3 and M4, with M3 being equal to |R2 * G| and M4 being equal to |R2 * AIDi|. One last thing: Vi, which is calculated as |Vi |=|SIDi * CK|i|, is included in message 5. The suggested ECC-based protocol uses generated ECC points to convey messages. We have set the parameters in the following ways for the purpose of evaluation: There are 224 bits for an ECC point and 128 bits for an ID.

Below is a breakdown of how much each message costs: Assembled in Message 1 are 128 bits of SIDi, CKi, M1+M2+AIDi, M3+M4, and Message 4 5 of Vi, all of which are 224 bits long. Following is the formula for determining the total bandwidth of the suggested protocol:

$$\text{bw} = \sum_{i=1}^{5} \text{Message i} \qquad (1)$$

We save bandwidth and provide better efficiency in terms of communication overheads with our suggested technique, which has a calculated bandwidth of 1696 bits, which is shorter than the esfahani approach (Esfahani et al. 2019).

B. *Computation overhead*

For simplicity's sake, let's call the time it takes for the authentication server to compute C1 during registration, the smart sensor and router to compute C2 and C3, and TECM the time it takes to multiply a number by itself over an elliptic curve. The computed costs in the suggested procedure are as follows: C1=1TECM, C2=6TECM, and C3=6TECM.

C. *Storage overhead*

Consider C4 and C5 as the smart sensor's and router's respective memory costs for storing the security parameters. For the sake of cost calculation, let's assume that CKi and SIDi are 128 bits long and that the elliptic curve cryptosystem is ECC-224 bits. With C4=224 bits, our ECC-based suggested protocol allows the smart sensor to save the cookie CK|i as an ECC point. C5 is 256 bits since the router keeps track of CKi and SIDi. Due to its massive storage capacity, the server cost is negligible in this case.

VII. CONCLUSION

In this study, we aim to create an industrial Internet of Things (IoT) authentication protocol that uses ECC for safe machine-to-machine communication.

Our proposed technique is more suited for limited devices since we removed the hash function's utility and added stringent security via the ECC module. Despite the authors' claims to the contrary, we demonstrated that the current system is susceptible to assaults such as the SSL strip attack and others. Our suggested innovation uses the ECC module to increase the current protocol's security level. To back up all of the security claims, we used the Scyther tool and BAN logic.

REFERENCES

[1] Cao K, Liu Y, Meng G and Sun Q (2020) An overview on edge computing research, IEEE Access, Vol. 8, pp. 85714–85728

[2] Imani A, Keshavarz-Haddad A, Haghighat J and Eslami M (2018) Security challenges and attacks in M2M communication, International proceedings of 9th Symposium on Telecommunications, Tehran, Iran, 2018, pp. 264-269.

[3] Sadeghi A-R, Wachsmann C and Waidner M (2015) Security and privacy challenges in industrial Internet of Things, International proceedings of ACM/EDAC/IEEE Design Automation Conference (DAC), San Francisco, CA, USA, pp. 1-6

[4] Rostampour S, Safkhani M, Bendavid Y and Bagheri N (2020) ECCbAP: A secure ECC-based authentication protocol for IoT devies, Pervasive and Mobile Computing, Vol. 67

[5] Shelby Z, Hartke K and Bormann C (2014) 'The constrained application protocol (coap), Internet engineering task force

[6] Granjal J, Monteiro E and Silva J. S. (2013) Application-layer security for the wot: extending coap to support end-to-end message security for internet-integrated sensing applications, International proceedings of conference on Wired/Wireless Internet Communication, Springer, pp. 140–153.

[7] Hussen H R, Tizazu G A, Ting M, Lee T, Choi Y and Kim K H (2013) Sakes: Secure authentication and key establishment scheme for m2m communication in the ip-based wireless sensor network (6l0wpan), International proceedings of Ubiquitous and Future Networks (ICUFN), Fifth International Conference on IEEE, pp. 246–251

[8] Qiu Y and Ma M (2016) A mutual authentication and key establishment scheme for m2m communication in 6lowpan networks, IEEE Transactions on Industrial Informatics, vol. 12, no. 6, pp. 2074–2085

[9] HiveMQ team (2015) https://www.hivemq.com/blog/mqtt-security-fundamentals-tls-ssl/ Wells L J, Camelio J A, Williams C B and White J (2014) Cyberphysical security challenges in manufacturing systems, Manufacturing Letters, vol. 2, no. 2, pp. 74 – 77

[10] Goyal V, Pandey O, Sahai A and Waters B (2006) Attribute-based encryption for fine-grained access control of encrypted data, International proceedings of 13th ACM conference on Computer and communications security, pp. 89–98.

[11] Bethencourt J, Sahai A and Waters B (2007) Ciphertext-policy attribute based encryption, International proceedings of Security and Privacy, IEEE Symposium on, pp. 321–334.

[12] Roebuck K (2012) Advanced Message Queuing Protocol (AMQP): High-impact Strategies-What You Need to Know: Definitions, Adoptions, Impact, Benefits, Maturity, Vendors, Emereo Publishing

[13] Raza S, Shafagh H, Hewage K, Hummen R and Voigt T (2013) Lithe: Lightweight secure coap for the internet of things, IEEE Sensors Journal, vol. 13, no. 10, pp. 3711–3720.

[14] Dunkels A, Gronvall B and Voigt T (2004) Contiki- a lightweight and flexible operating system for tiny networked sensors, International proceedings 29th Annual IEEE International Conference on local computer networks, pp. 455–462

[15] Kalra S and Sood S K (2015) Secure authentication scheme for IoT and cloud servers, Pervasive and Mobile Computing, pp. 210–223

[16] Kumari S, Karuppiah M, Das A K, Li X, Wu F and Kumar N (2018) A secure authentication scheme based on elliptic curve cryptography for iot and cloud servers, The Journal of Supercomputing, pp. 6428–6453

[17] Esfahani et al. (2019) A Lightweight authentication mechanism for M2M communications in industrial IoT environment, IEEE Internet of Things Journal, vol. 6, no. 1, pp. 288-296.

[18] S Zhao, D Wang, D Zhao, S Yang, C Ma (2012) A scheme to prevent SSLStrip attack. In Information and communications security (ICICS) Lecture Notes in Computer Science, vol 7618. Springer, Berlin, Heidelberg

[19] Wang K H, Chen C M, Fang W, Wu T Y (2017), A secure authentication scheme for internet of things, Pervasive and Mobile Computing,42, 15–26

[20] L Gong, R Needham, R Yahalom, Reasoning about belief in cryptographic protocols, International proceedings of symposium on Research in Security and Privacy, (IEEE) 1990, pp. 234-248

[21] Burrows M, Abadi M and Needham R (1989) BAN a logic of authentication, Technical report 39, Digital Equipment Systems Research center, Palo Alto, California

[22] Syverson P and Cervesato I (2000) The logic of authentication protocols, International School on Foundations of Security Analysis and Design, Springer, pp. 63–137

[23] Cremers C J F (2008) The Scyther tool: Verification, falsification, and analysis of security protocols, Computer Aided Verification, Springer Berlin Heidelberg, 2008, pp. 414–418

[24] Blanchet B (2007) CryptoVerif: Computationally sound mechanized prover for cryptographic protocols, Dagstuhl seminar Formal Protocol Verification Applied, p. 117