Hindawi
Security and Communication Networks
Volume 2018, Article ID 6563089, 25 pages
https://doi.org/10.1155/2018/6563089

Research Article

Hybrid Secure Authentication and Key Exchange Scheme for M2M Home Networks

Uğur Coruh and Oğuz Bayat

Graduate School of Science and Engineering, Altınbaş University, İstanbul, Turkey

Correspondence should be addressed to Uğur Coruh; ugur.coruh@ogr.altinbas.edu.tr

Received 21 July 2018; Revised 28 September 2018; Accepted 9 October 2018; Published 1 November 2018

Guest Editor: Esther Palomar

In this paper, we analyzed Sun et al.'s scheme which proposes an M2M (Machine-to-Machine) secure communication scheme by using existing TD SCMA (Time Division-Synchronous Code Division Multiple Access) networks. They offer a password-based authentication and key establishment protocol for mutual authentication. Moreover, their proposed secure channel establishment protocol uses symmetric cryptography and one-way hash algorithms and they considered using their protected channel model for mobile users and smart home networks. In this paper, we propose to complete the missing part of Sun et al.'s scheme. This can occur by addressing privacy-preserving and message modification protection. Moreover, improvements can be made to MITM (Man-In-The-Middle) attack resistance, anomaly detection and DoS (Denial-of-Service) attacks with timing. ECDH (Elliptic Curve Diffie Hellman) cryptography based protected cipher-key exchange operation used on initial setup and key-injection operations to provide secure user registration, user password change and home gateway network join phases. We simulated both the proposed and Sun et al.'s schemes. We analyzed Sun et al.'s scheme for performance, network congestion and resource usage. Missing privacy-preserving was analyzed and compared with the GLARM scheme, and the storage cost of each phase was analyzed according to Ferrag et al.'s survey proposal. In Sun et al.'s scheme, future work for the security architecture of the home network is related to Li et al.'s protocol being implemented in our proposed design.

1. Introduction

Security trade-off and optimization are the most common problems in IoT (Internet-of-Things) devices which come with limited resources. The most common part of an IoT network is interconnection and interoperability between machines, which is called M2M communication.

M2M communication covers a wide area including home networks and telecommunication devices. Privacy protection and robust identification, authentication, and authorization requirements are provided with limited system resources.

The improvement of consumer products also improves smart home networks and this leads to an increase in wireless network connected devices per user. Moreover, this situation increases the deployment of private information to public networks and increases the security requirements for M2M communication according to the attacks essentially stated in [1].

The contributions of this paper focus on message modification, privacy-preserving considered in [2], state management, anomaly detection with the timing for reliable communication, and finally home gateway and user device revocation. Furthermore, we updated current scheme phases and provided security for user registration, password change, and home gateway join stages. In this paper, we aim to fill the security gaps in [3] to provide end-to-end security for practical usage in M2M networks.

In [3], the security protocol does not specify on privacy-preserving and message modification. We provide methods for solving these issues. Design [3] does not identify messages, and the scheme entity states are not managed well for reliable data communication. Moreover, mobile user device and home gateway device revocations are not mentioned in their scheme. Design [3] is not resistant to attacks such as the DoS attacks mentioned in [2, 4, 5] or the message modification mentioned in [6]. Our contributions are improving

Table 1: Summary of cryptosystems and countermeasures in M2M communication [1].

Cryptosystems and Countermeasures	[8]	[2]	[9]	[4]	[6]	[3]	[5]	[10]	[11]	Ours
Secure cryptographic hash function [12]		√	√	√	√	√		√	√	√
Original data acquisition								√		
Spatial-Domain transformation								√		
Time-domain transformation								√		
C-MA (Correlation coefficient-based matching algorithm)	√									
D-MA (Deviation ratio-based matching algorithm)	√									
AMACs (Aggregate message authentication codes) [13]		√							√	
Certificateless aggregate signature [14]			√							
ECDH (Elliptic Curve Diffie-Hellman) [15]				√						√
ID-based signature scheme [13]					√					
AES (Advanced encryption standard) [16]						√				√
Hybrid Linear Combination Encryption [17]							√			

Supported = √; empty = not supported.

current protection for a number of attacks and we propose an end-to-end enhanced secure authentication and communication scheme. Scheme [3] sends a clear password and ID (identification data) to the public network and assumes that a connection is secure during registration and password change operations. We analyzed privacy-preserving and compared it with the GLARM scheme [2]. In addition, the storage cost of the phases were examined according to survey guidelines [1, 3]. Future work for the security architecture of the home network which is related to [7] is performed in our proposed design.

This paper is organized as follows. Section 2 describes M2M network related authentication protocols, Section 3 reviews [3], and Section 4 shows its weaknesses. Section 5 presents a proposal of our plan. Section 6 explains the formal analysis and simulation of our proposed protocol. Section 7 presents a security analysis and Section 8 presents a performance comparison of the proposal and [3] security. Finally, Section 9 concludes the paper.

2. Related Works

Current scheme [3] is examined in the survey [1], and it is efficient in terms of performance and network congestion compared to the OTP based scheme in [18]; however, privacy-preserving is not analyzed and compared with the GLARM scheme [2], storage cost is not mentioned, and a comparison with the PBA scheme is not sufficient according to survey analysis in [1]. Current scheme [3], which is the composition of a home network model, is proposed in [18, 19]. Scheme [18] is proposed as an OTP based user authentication scheme and [19] is the biometric information based authentication scheme. Scheme [3] is efficient regarding the amount of calculation and network congestion volume compared with [18].

Table 1 shows M2M network authentication protocol cryptosystems and countermeasures. Additionally, [1] analyzed and classified several schemes to attacks in Table 2. We added our scheme to their results in Table 2. Scheme [1] organizes the designs as "fully supported" and "partially supported" for attacks in Table 2. If a scheme is classified as fully supported for an attack, this means that the authors of the scheme have proven formal verification techniques or simulations perform the reliability of their scheme against the selected attack and security analysis for all conditions. If a scheme claims an attack but there is insufficient formal verification, the simulation of the scheme is classified as partially supported to an attack. If the scheme does not provide security for an attack, then the scheme is classified as not supported. Additionally, this same notation is used in [20–23].

3. Review of Sun et al.'s [3] Scheme

In this section, we critique the security authentication scheme in an M2M home network service using existing the TD-SCMA network proposed by Sun et al. [3]. Their scheme has three main components: the user device, the M2M server, and the home gateway device. Scheme [3] consists of five phases: setup, user registration, user login and authentication, user password change, and home gateway join to the TD-SCMA network. The M2M server is a central server in this scheme, as shown in Figure 1.

M2M servers are defined by an individual identifier and contain a secret key for storage encryption. The user selects a password and user ID which are transferred to the M2M server for registration using mobile devices over an unsecured communication link. The M2M server encrypts and stores the password and the M2M server produces and sends random data to the mobile user device for mutual authentication input and reserves this arbitrary data in the M2M server database. This random value is stored in the mobile user device. During the login and authentication request, the user device generates a message which contains a user ID and an encrypted random number. This random number is encrypted with a user password. The user device then posts this message to the M2M server for registration. Each entity calculates a session key and they verify each other. The password is used as a primary key for mutual authentication. Authenticated users can access their home

Table 2: Summary of attacks in M2M networks and defense protocols [1].

Adversary model	[8]	**[2]**	[9]	**[4]**	**[6]**	**[3]**	**[5]**	[10]	[11]	**Ours**
Audio Replay attack	√	√		√	√	√			1	√
Changing distance attack	√									√
Same-type-device attack	√									√
Composition attack	√									√
Redirection attack	√	√	√	√			√		√	√
Man-in-the-middle attack	√	√	√	√	√	√			√	√
Substitution attack	√	√	√	√	√					√
DoS attack		√		√			√			√
Replay attack	√			√	√	√			√	√
Forging attack				√						√
Colluding attack	√			√			√			√
Flooding attack	√						√		√	√
Side-channel attack	√						√		√	√
False messages attack	√				√	√	√		√	√
Sybil attack					√	√			√	√
Movement tracking					√				√	√
Message modification					√					√
Impersonation attack					√	√	√			√
Guessing attack						√				√
Stolen-verifier attack						√				√
Wormhole attack	√	√		√		√			√	√
Blackhole attack	√	√		√	√	√			√	√
Attribute-trace attack					√					√
Eavesdropping attack					√	√			√	√
Chosen-plaintext attack					√				√	√
Spam attack	√				√	√			√	√
Identity theft attack	√					√				√
User manipulation attack	√					√	√		√	√
Routing attack	√					√				√
Linkability attack	√									√
Rejection attack										√
Successive-response attack										√
Packet analysis attack		√				√			√	√
Packet tracing attack		√				√			√	√
Brute-force attack	√	√		√	√		√	√		√

Fully supported = √, partially supported = ✓, and empty = not supported.

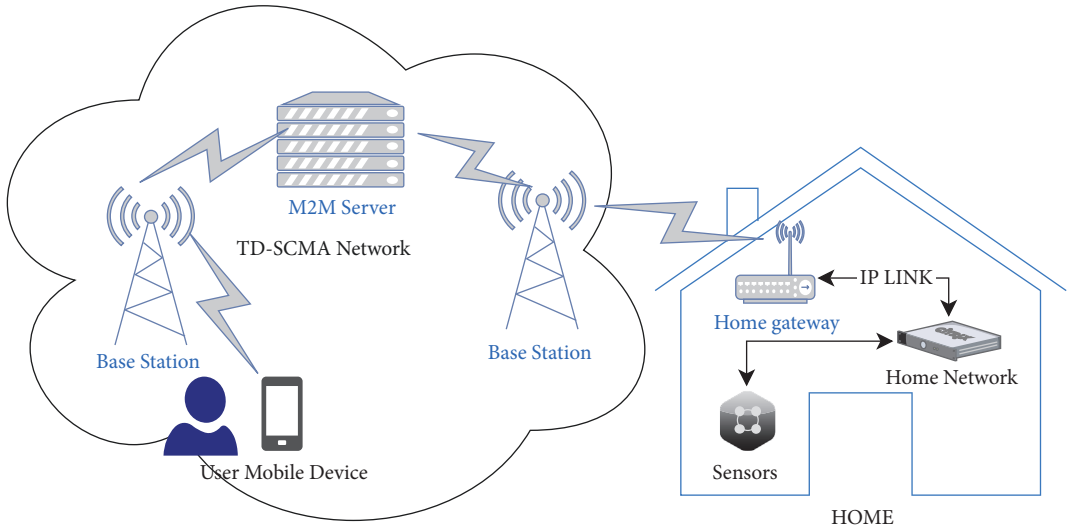

Figure 1: Structure of Sun et al.'s [3] M2M model.

Table 3: Notation of Sun et al.'s scheme [3].

Notation	Description
k	The secret key of the M2M server
uid	Mobile user's ID
mid	M2M server's ID
pw	User's password
umk	The session key between a user and M2M server
hid	Home gateway ID
sf	Status flag of the mobile user
$h(.)$	Cryptographic hash function
$h(m,n)$	Hashing of the concatenation of m and n
$[m]^n$	Most significant n bits of string m
$F_{e-AES}(a,b)$	Using AES function to encrypt a with b as the key
$F_{d-AES}(a,b)$	Using AES function to decrypt a with b as the key

gateway device information over the M2M server using their mobile devices. In this system, each home gateway device joins the network with their unique ID. When a home gateway boots up, it transmits its unique ID to the M2M server which searches for this ID related user and if it is a match, it then calculates the hash of the combined user ID, user arbitrary data, and M2M server ID data and transmits it to the home gateway for session key computation. The M2M server also calculates the session key for a home gateway for data exchange services. Protocol notations are listed in Tables 3 and 4 showing the parameter settings used in the current design.

The encryption and decryption procedures are based on AES (Advanced Encryption Standard), and SHA-1 (Secure Hash Algorithm 1) used as a one-way hash function in the protocol. Figures 2 and 3 show scheme [3] and the proposed scheme's performance metrics and comparisons for AES MATLAB implementation. Scheme [3] does not demonstrate that their measurements carried AES key schedule operation in encryption and decryption processes. According to our

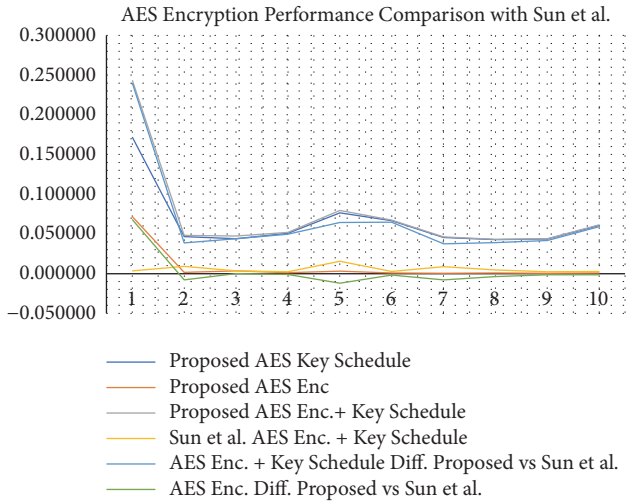

Figure 2: AES encryption performance comparison with Sun et al. [3].

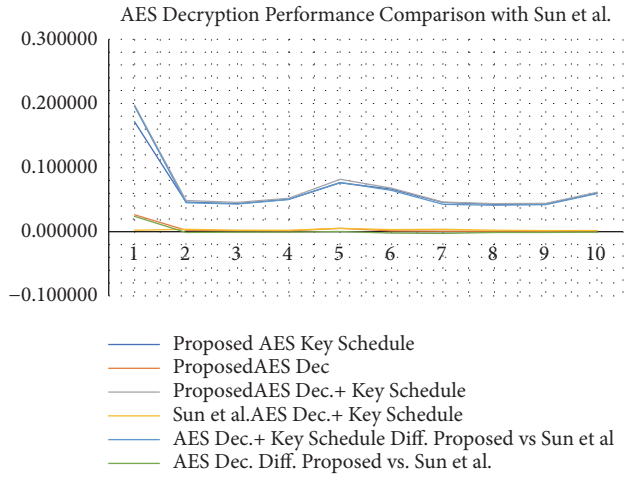

Figure 3: AES decryption performance comparison with Sun et al. [3].

Table 4: Protocol parameter settings at Sun et al.'s scheme [3].

Parameters	Settings and Description
s	Salt, a 16-bit random number
k	The secret key of M2M server, 64 bits
uid	The equipped SIM card number, 80 bits
mid	The equipped SIM card number, 80 bits
hid	The equipped SIM card number, 80 bits
umk	A session key, 128 bits
pw	User's password, 64 bits
sf	User's statue flag, 1 bit
x_1	128 bits
x_2	128 bits
$h(.)$	SHA-1 Hash, 160 bits output
m	128 bits
n	128 bits
t	128 bits
$F_{e-AES}(a, b)$	AES encryption function with 128 bits input, 128 bits key and 128 bits output
$F_{d-AES}(a, b)$	AES decryption function with 128 bits input, 128 bits key and 128 bits output

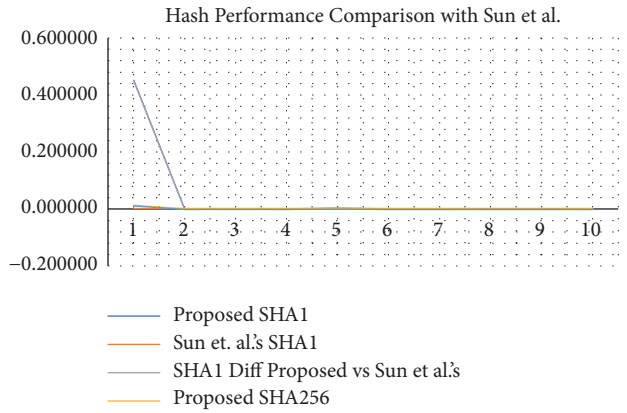

Figure 4: Hash performance comparison with Sun et al. [3].

measurement and comparison results, we noticed that their analysis results were close to the AES encryption and decryption without a cipher-key schedule operation's measurement results. Overall measurements for the proposed scheme are close to the measurement results of [3]. Current measurement results show that AES encryption is faster than decryption, which is the same as the result reported in scheme [3]. These results are presented in Figures 2 and 3. Figure 4 shows scheme [3]'s and the proposed scheme's performance metrics and comparisons for the SHA-1 MATLAB implementation.

The following subsections cover the complete phase examination of the design [3].

3.1. Sun et al.'s [3] Setup Phase. During the setup operation, the M2M server selects the 64-bit server secret key k. This key is not directly used in cryptographic procedures; it is used as an input to compute the operation key. ID personalization is not specified in the setup phase for each entity; it is specified during the setup operation. There must be a single 80-bit M2M server ID personalized to the smart card to define the M2M server. The server can interact with smartcards. The home gateway device is uniquely identified with an 80-bit ID and customized to a smartcard which is deployed with the home gateway device. Additionally, users are uniquely identified with an 80-bit ID and stored on the smart card.

3.2. Sun et al.'s [3] User Registration Phase. A new user submits a concatenated 80-bit user ID, a 64-bit clear password, and an 80-bit home gateway ID as follows: $< uid, pw, hid >$ for registration. Scheme [3] assumes that the communication channel is protected. For this reason, the password is sent in the clear form. Additionally, the current scheme assumes that communication is performed reliably. For this reason, there is no message identification. The M2M server selects a random number s for salt and concatenates with both the M2M server secret key and user ID, followed by calculating the one-way hash of the buffer with SHA-1 and received the first 128 bits of the 160-bit SHA-1 output for the AES encryption key, as follows:

$$d = h(k, uid, s) \tag{1}$$

$$key_{AES} = [d]^n \tag{2}$$

The M2M server encrypts the user password for storage as follows:

$$p = F_{e-AES}(pw, key_{AES}) \tag{3}$$

User ID, home gateway ID, salt, encrypted password, and statue flag $< uid, hid, s, p, sf >$ are stored in the database and salt s is transmitted to the user device. Finally, the user device stores salt for login and the authentication operation.

3.3. Sun et al.'s [3] User Login and Authentication Phase. This phase ensures secure user communication by performing mutual authentication between the M2M server and the

user device over the TD-SCMA network. Messages are not distinguished and reliable data communication is not considered. The user device selects a random number x_1 and calculates the following parameters for an authentication request:

$$g = F_{e-AES}(x_1, pw) \tag{4}$$

$$h_1 = h(x_1) \tag{5}$$

The user device sends the request parameters $< uid, g, h, s >$ to the M2M server, which searches *uid* and validates *s* in the M2M server database. If *uid* does not exist, the request is rejected. Otherwise, the M2M server retrieves the key from the message and decrypts the stored password and processes identical computations to verify the received hash value to guarantee that the user password is correct.

$$key_{AES} = [h(k, uid, s)]^n \tag{6}$$

$$pw' = F_{d-AES}(p, key_{AES}) \tag{7}$$

$$x_1' = F_{d-AES}(g, pw') \tag{8}$$

$$h_1' = h(x_1') \tag{9}$$

If the received h_1' value is equal to the calculated h_1 value, it guarantees that the user password and ID are correct, and the user is confirmed by the M2M server. Otherwise, the M2M server declines the operation. If the user is verified, then the M2M server selects another arbitrary number x_2 to calculate *umk* for the communication session key and sends the required values to the user device for the same session key creation; thus,

$$umk = [h(x_1', x_2)]^n \tag{10}$$

$$h_2 = h(umk, mid) \tag{11}$$

The M2M server stores *umk* and sends $< mid, x_2, h_2 >$ to the user device. The user device uses a created x_1 and received mid, x_2, and h_2 for session key *umk* generation and verifies the M2M server h_2 value as follows:

$$umk' = [h(x_1, x_2)]^n \tag{12}$$

$$h'_2 = h(umk', mid) \tag{13}$$

If the calculated h'_2 value is equal to the approved h_2 value, then the session key is stored in the mobile user device for secure communication, and, finally, the user device calculates the hash h_3, as follows:

$$h_3 = h(umk', uid) \tag{14}$$

The user device sends h_3 to the M2M server, and, finally, the M2M server verifies the h_3 hash; thus,

$$h'_3 = h(umk, uid) \tag{15}$$

If the h_3 value is equal to the h'_3 value, then a mutual authentication is established between the M2M server and the user device and *umk* is used for secure data transmission.

3.4. Sun et al.'s [3] User Password Change Phase. A user password provides time-based security, and, for this reason, it must be periodically updated. In this phase, the user updates an old password with a new password, thereby receiving a new arbitrary salt value *s* for the next login and authentication operation. Scheme [3] assumes that the data transmission channel is secure and reliable for a password change operation. The password change operation starts from the user device, which sends the user ID, clears the old password, and clears the new password and the current random salt value is formatted thus $< uid, pw, pw_{new}, s >$ to the M2M server. The M2M server searches for the user ID and obtains an encrypted password *p* from the database, after which it recovers the password encryption key and encrypts a clear password *pw* for verification with some subsequent calculations:

$$key_{AES} = [h(k, uid, s)]^n \tag{16}$$

$$p' = F_{e-AES}(pw, key_{AES}) \tag{17}$$

If the calculated password p' is equal to the stored *p*, then the M2M server selects a new random salt s_{new} and encrypts the new password to store in the M2M server database; thus,

$$key'_{AES} = [h(k, uid, s_{new})]^n \tag{18}$$

$$p_{new} = F_{e-AES}(pw_{new}, key'_{AES}) \tag{19}$$

The M2M server stores the encrypted password p_{new} and sends new salt s_{new} to the user device. The user device stores s_{new} for a new login and authentication process.

3.5. Sun et al.'s [3] Home Gateway Join TD-SCMA Network. Home gateway devices send network join requests to M2M servers when powered. This request includes a prepersonalized unique identifier $< hid >$ that is stored on a smart card. Each home gateway must be associated with a user device on the M2M server. For this reason, the M2M server checks related user devices with this home gateway device ID. If a user device exists, then the M2M server calculates the session key H_{key} for data transmission followed by key recover data H_p for the home gateway device; thus,

$$H_p = [h(uid, mid, s)]^m \tag{20}$$

$$H_{key} = [h(hid, H_p)]^t \tag{21}$$

H_{key} is saved on the M2M server and H_p is posted to the home gateway device which calculates H_{key} for data transmission. The home gateway device does not need to know the user ID for the session key production:

$$H_{key} = [h(hid, H_p)]^t \tag{22}$$

4. Weaknesses of Sun et al.'s [3] Scheme

Sun et al.'s [3] scheme provides less calculation cost compared to the scheme in [18]. Scheme [3] can resist a stolen-verifier

attack, replay attack, guessing attack, and impersonation attack. However, there are several security vulnerabilities in overall security.

Scheme [3] does not ensure anonymity. The user ID is sent plain during the authentication phase. Moreover, during the authentication phase, privacy-preserving is not analyzed, and this argument is indicated in [1], which offers to compare scheme [3] privacy-preserving with GLARM [2]. GLARM uses temporary user ID that is obtained from the original user ID and is transferred in messages for privacy-preserving. The real user ID is never transferred in messages. Storage cost is not analyzed in scheme [3], but, in our review, we simulated the scheme and analyzed every storage and resource change for each phase.

A performance analysis is inadequate for comparison with other designs. Moreover, performance measurement initial configurations are not indicated in their work. In the review section, we simulated and analyzed every scheme phase and we used algorithm performance metrics and memory usage.

Scheme [3] does not provide secure registration. The password is sent in an open format with an original user ID. They assume that this communication channel would be shielded so if we have a secure channel that is already protected, then we do not need further authentication.

Design [3] also does not provide a safe password change operation like the registration operation. The user ID and password are sent in the unciphered format during this operation.

An open formatted password sending during registration and a password change operation with the original user ID has a significant problem with privacy security in scheme [3]. Scheme [3] password change and user registration phase are explained in Section 3.

Transferred messages are modifiable during communication. There is no MAC (Message Authentication Code) verification for each phase.

There is no reliability during operations. Messages are not distinguished with a sequence number or message tag and there is no acknowledgment of messages between communicating entities.

Scheme [3] does not have a message timeout and anomaly detection method. Moreover, there is no communication recovery progress in the protocol phases for dropped or corrupted messages.

The home gateway joins the network phase unprotected from DoS attacks. A fraudulent device can attempt to join the network since there is no restriction on this operation. If the home gateway ID is retrieved from the home gateway device communication, then the attacker can process the M2M server session key generation for the fraudulent device or device simulators.

User and device revocation are not placed at [3]. Salt is used in the home gateway join network phase for session key agreement, and the salt parameter is updated in the password change phase. After the password change request, the new salt value is stored in the user device and the M2M server, but the home gateway device uses the session key that was created with the old salt, and there is no reboot procedure to update the salt and session key on the home gateway side. If user updates owned the password periodically for security reasons, the home gateway would not be affected by this change and risk would increase.

5. Proposed Scheme

In this section, we propose an enhanced scheme to fill the missing parts of Sun et al.'s [3] scheme. A proposed scheme notation is listed in Table 5 and our contributions are arranged as shown in Table 5 .

5.1. Message Modification Protection with Shortened MAC. Our first contribution to the design [3] is providing message modification protection. Scheme [3] messages are modifiable during transmission. There is no MAC verification security. Moreover, the MAC calculator source must be verified by the receiver. For this reason, we opted to use keyed MAC algorithms such as C-MAC (Cipher-based MAC) [24] or H-MAC (the Keyed-Hash Message Authentication Code) [25] based on DES (Data Encryption Standard) or AES. This protection method was performed in [6]. According to performance and resource restrictions, we have decided to use a shortened MAC as described in the PBA scheme [26] to confirm messages. Moreover, the GLARM scheme [2] uses the MAC algorithm specified in [27]. We used an H-MAC based truncated MAC algorithm that has a similar offset selection to HOTP (HMAC-Based One-Time-Password), as explained in [28].

Each request and response in a public channel is protected with MAC and if MAC confirmation fails; then the request is terminated by the receiver entity.

We simulated and measured the performance of the H-MAC configuration for SHA-256 (Secure Hash Algorithm 256) and SHA-1 with 16-byte and 8-byte keys, as shown in Table 6.

5.2. Privacy-Preserving with Temporal ID. The second contribution to the scheme is replacing the original user ID with a temporary ID to provide privacy-preserving as stated in the GLARM scheme [2]. The GLARM scheme uses a preshared key to derive the ID and temporal ID used in all procedures. Scheme [3] focuses on privacy security, but, during registration and password change phases, they transferred the password and user ID in an open format. Additionally, the original user ID is always posted in the open format for all stages. Therefore, privacy security has a significant problem in scheme [3] and it does not guarantee anonymity.

In our scheme, we have a robust method to solve this problem, namely, ECDH key pairing during registration. Key pairing requests are distinguished by the user ECDH key pairing identifier (*ecdh_id*). This *ecdh_id* value is calculated from the user ID (*uid*) with a trimmed SHA-1 during the user setup phase; thus,

$$[ecdh\_id] = F_{SHA-1}(uid)^n \tag{23}$$

The user sends the public key (pub_{user}) to the M2M server during the registration stage with *ecdh_id* and *uid* is never revealed, as follows:

$$< ECD\_SYN, ecdh\_id, pub_{user} > \tag{24}$$

Table 5: Notation of the proposed scheme.

Notation	Description
k	The secret key of the M2M server
uid	Mobile user's ID
t_uid	Temporal Mobile User's ID
mid	M2M server's ID
pw	User's password
umk	The session key between User and M2M server
hid	Home Gateway ID
t_hid	Temporal Home Gateway ID
u_st	User State
h_st	Home Gateway State
$a \parallel b$	concatenation of a and b
$[m]^n$	Most significant n bits of string m
$F_{SHA256}(.)$	Cryptographic SHA256 hash function
$F_{SHA1}(.)$	Cryptographic SHA1 hash function
$F_{e-AES}(a, b)$	Using AES function to encrypt a with b as the key
$F_{d-AES}(a, b)$	Using AES function to decrypt a with b as the key
$F_{H-MAC}(a, key)$	Keyed Cryptographic Mac and Hash Function
$ecdh\_id$	User ECDH Key Pairing Identifier
key_{ECDH}	ECDH Secret Key
pub_{user}	ECDH User Public Key
$priv_{user}$	ECDH User Private Key
pub_{m2m}	ECDH M2M Server Public Key
$priv_{m2m}$	ECDH M2M Server Private Key
$[pub_x, priv_x] = F_{i-ECDH}()$	ECDH Key Pair Generation for Entity x
$key_{ECDH} = F_{s-ECDH}(pub_{user}, priv_{m2m})$	M2M Server ECDH Secret Key Derivation
$F_{e-ECDH}(a, key_{ECDH})$	ECDH Encryption with input a
$F_{d-ECDH}(a, key_{ECDH})$	ECDH Decryption with input a
key_{enc}	Encryption Key
key_{mac}	MAC (Message Authentication Code) Key

Table 6: Truncated MAC calculation performances.

Algorithm	Measurement Mean (sn)	Measurement Std. (sn)
H-MAC-SHA256 16 bytes key	0.000273	0.000094
H-MAC-SHA1 16 bytes key	0.000268	0.000189
H-MAC-SHA256 8 bytes key	0.000267	0.000109
H-MAC-SHA1 8 bytes key	0.000283	0.000134
ISO9797-ALGO-3 16 bytes key	0.000392	0.000226

M2M server uses its private key ($priv_{m2m}$) and receives the public key (pub_{user}) to derive encryption key (key_{ECDH}). For ECDH secure key establishment is as follows:

$$key_{ECDH} = F_{s-ECDH}\left(pub_{user}, priv_{m2m}\right) \tag{25}$$

The M2M server encrypts integrity and confidentiality keys with key_{ECDH}. These encrypted keys and the M2M server public key (pub_{m2m}) are sent to the user device in the same message to decrease the request size in the proposed scheme; thus,

$$\begin{aligned} &Enc\left(key_{enc} \parallel key_{mac}\right) \\ &\quad = F_{e-ECDH}\left(key_{enc} \parallel key_{mac}, key_{ECDH}\right) \end{aligned} \tag{26}$$

$$\begin{aligned} &< ECDH\_ACK, ecdh\_id, \text{pub}_{\text{m2m}}, \\ &\quad Enc\left(key_{enc} \parallel key_{mac}\right) > \end{aligned} \tag{27}$$

The user receives the public key and derives the ECDH encryption key to decrypt the encrypted keys. After revealing the keys, the user device prepares the registration message ($REGISTER\_SYN$) by calculating the registration message MAC and encrypting the user ID (uid), password (pw), and home gateway ID (hid) with a confidentiality key to be sent with $ecdh\_id$, as follows:

$$\begin{aligned} mac_{msg} = \text{F}_{H-MAC}\left(REGISTER\_SYN \parallel ecdh_{id} \parallel uid \parallel \right. \\ \left. pw \parallel hid, key_{mac}\right) \end{aligned} \tag{28}$$

$$enc_{msg} = \mathrm{F}_{e-AES}\left(uid \parallel pw \parallel hid \parallel mac_{msg}, key_{enc}\right) \quad (29)$$

$$< REGISTER\_SYN, ecdh\_id, enc_{msg} > \quad (30)$$

The problem in [3] is that *REGISTER_SYN* is a similar request message containing sensitive assets in open format. However, we protect these sensitive assets with encryption and MAC. When the M2M server receives encrypted values with the *REGISTER_SYN* message, then it is decrypted with key_{ECDH} and acquires the clear user ID, after which it calculates a temporary ID with a H-MAC based truncated MAC algorithm by using an integrity key. The output length of this algorithm is the same as the original ID length. The temporary ID generation operation is also is also identically performed in the user device and the next requests are performed using the temporary ID to protect user privacy in the network. An H-MAC based truncated MAC algorithm is typically resistant to brute-force attacks. In addition, ECDH key pairing provides a defense against MITM attacks, and MAC protects messages from modification. These listed protection methods are not provided by scheme [3].

5.3. Forward and Backward Secrecy Improvement. The third participation pertains to the improvement of secrecy. MAC and temporal ID operations are enhanced with H-MAC methods. We used protection keys encapsulated with ECDH keys to provide forward and backward secrecy improvement.

Scheme [3] registration and password change phases are not protected; however, we protect every step that we proposed. Our scheme follows general standard algorithms such as ECDH, Trimmed H-MAC, and AES. These algorithms, which are used to protect sensitive assets during communication, are resistant to brute-force attacks.

5.4. Reliable Operation Processing and State Management. The fourth contribution pertains to service reliability. Scheme [3] does not include the operation plus message timeout, message acknowledgment segment, and message identifier segment. Moreover, the user and home gateway state management are poor. There is only one bit to keep user state online or offline.

We have provided a timeout measurement with an internal device clock without a timestamp. This kind of usage does not require an internal RTC (real-time clock) module for devices. If a timeout occurs during transmission, then entities can terminate or resend requests. Additionally, this timeout measurement can be used to detect eavesdropping attacks by checking operation processing times that are used by the timeout control. Each message is identified with a one-byte tag. Entities can evaluate requests with this identifier. The proposed scheme message identifiers are listed in Table 7.

Additionally, we have updated the state of the machines for the current design. Enhanced states in the proposed protocol are shown in Figures 5 and 6.

5.5. Proposed Scheme Phases. Our proposed scheme consists of ten phases. We designed a practically good system in the real network and we assumed that there was no secure communication network over the public channel. Moreover, in our scheme, the home gateway devices and user devices have WIFI connection capability via the private home network. We added ECDH key sharing between the user device and the M2M server. This operation was consolidated with user registration. Additionally, we added one more phase according to the current scheme for key injection to the home gateway device, and the user logout operation was also defined in the proposed protocol.

5.5.1. Proposed Scheme M2M Server Setup. During the setup operation, the M2M server generates ECDH pub_{m2m} and $priv_{m2m}$ key pairs, as follows:

$$\left[pub_{m2m}, priv_{m2m}\right] = F_{i-ECDH}\,() \quad (31)$$

The generated public key length is 576 bits, and the private key length is 832 bits. Moreover, the M2M server selects a 64-bit server secret key (k), which is mentioned in scheme [3]. During the setup phase, there must be single 80-bit M2M server ID (*mid*) personalized to the smart card to define the M2M server in the network. The server can communicate with the smart cards.

5.5.2. Proposed Scheme Home Gateway Setup. During the setup operation, the home gateway device is uniquely identified with an 80-bit ID (*hid*) and is personalized to a smartcard which is deployed with the home gateway device.

5.5.3. Proposed Scheme User Setup. During the setup operation, users uniquely identified with an 80-bit ID (*uid*), which is stored on the smart card. In addition, for registration, the key pair's user device generates ECDH pub_{m2m} and $priv_{m2m}$ key pair, as follows:

$$\left[pub_{user}, priv_{user}\right] = F_{i-ECDH}\,() \quad (32)$$

A temporal 80-bit *ecdh_id* is used for only key pairing that is generated in this phase with the following operation:

$$[ecdh\_id] = F_{SHA-1}\,(uid)^{n} \quad (33)$$

The generated public key length is 576 bits, and the private key length is 832 bits. Additionally, the user should define the 64-bit password (pw) and 80-bis home gateway id (*hid*) for the registrations.

5.5.4. Proposed Scheme User Registration and Key Pairing. The user resets the operation timer for the *ECDH_SYN* message sending operation and sends the *ECDH_SYN* message to the M2M server for registration, as follows:

$$< ECD\_SYN, ecdh\_id, pub_{user} > \quad (34)$$

The M2M server derives key_{ECDH} with pub_{user} and its own $priv_{m2m}$ to use as a transport key; thus,

$$key_{ECDH} = F_{s-ECDH}\left(pub_{user}, priv_{m2m}\right) \quad (35)$$

Table 7: Proposed scheme message identifiers.

Message Name	Message Phase	Message Sender	Message Receiver
ECDH_SYN	User Registration and Key Pair	User	M2M Server
ECDH_ACK	User Registration and Key Pair	M2M Server	User
REGISTER_SYN	User Registration and Key Pair	User	M2M Server
REGISTER_ACK	User Registration and Key Pair	M2M Server	User
KEY_SYN	User Home Gateway Key Injection	User	Home Gateway
KEY_ACK	User Home Gateway Key Injection	Home Gateway	User
JOIN_SYN	Home gateway Join Network	Home Gateway	M2M Server
JOIN_ACK	Home gateway Join Network	M2M Server	Home Gateway
JOIN_OK	Home gateway Join Network	Home Gateway	M2M Server
LOGIN_SYN	User Login and Authentication	User	M2M Server
LOGIN_ACK	User Login and Authentication	M2M Server	User
LOGIN_OK	User Login and Authentication	User	M2M Server
PWD_SYN	User Change Password	User	M2M Server
RE_JOIN	User Change Password	M2M Server	Home Gateway
PWD_ACK	User Change Password	M2M Server	User
PWD_NACK	User Change Password	User	M2M Server
PWD_OK	User Change Password	User	M2M Server
LOGOUT_SYN	M2MServer User Logout	User	M2M Server
LOGOUT_ACK	M2MServer User Logout	M2M Server	User

The M2M server generates a 16-byte key_{enc} and key_{mac} for secure transmission. These keys are securely transported under key_{ECDH} between the entities, as follows:

$$Enc\left(key_{enc} \parallel key_{mac}\right) = F_{e-ECDH}\left(key_{enc} \parallel key_{mac}, key_{ECDH}\right) \tag{36}$$

The M2M server resets the operation timer for the *ECDH_ACK* message sending operation and sends the *ECDH_ACK* message to the user for the key pair, as follows:

$$< ECDH\_ACK, ecdh\_id, \mathrm{pub}_{\mathrm{m2m}}, Enc\left(key_{enc} \parallel key_{mac}\right) > \tag{37}$$

The user device checks whether the *ECDH_SYN* operation time is within an acceptable time range; if not, then it destroys sensitive assets and terminates the operation; otherwise, it derives key_{ECDH} as follows:

$$key_{ECDH} = F_{s-ECDH}\left(pub_{m2m}, priv_{user}\right) \tag{38}$$

The user device decrypts $Enc(key_{enc} \parallel key_{mac})$ with key_{ECDH} and keeps $key_{enc} \parallel key_{mac}$ in temporary memory. At the end of this operation, the M2M server completes user key pairing, and the MAC keys are securely shared for sensitive operations, as follows:

$$\left[key_{enc} \parallel key_{mac}\right] = F_{d-ECDH}\left(Enc\left(key_{enc} \parallel key_{mac}\right), key_{ECDH}\right) \tag{39}$$

After key pairing registration data sending, the user registration operation starts from the user side. The user device

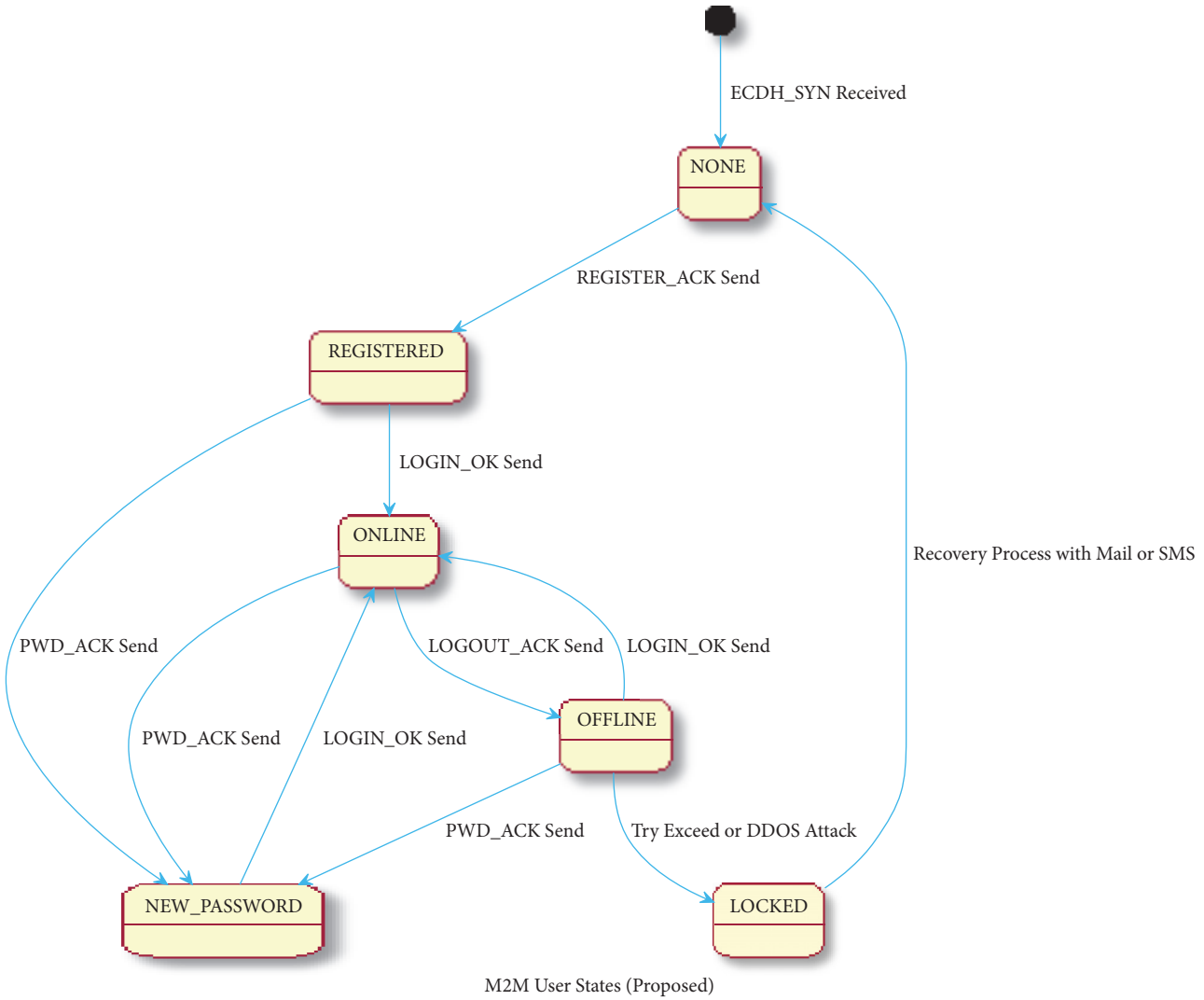

Figure 5: User state processing.

prepares the MAC of the *REGISTER_SYN* message and the encrypted message payload with the following operations:

$$mac_{msg} = \mathrm{F}_{H-MAC}\left(REGISTER\_SYN \parallel ecdh_{id} \parallel uid \parallel pw \parallel hid, key_{mac}\right) \quad (40)$$

$$enc_{msg} = \mathrm{F}_{e-AES}\left(uid \parallel pw \parallel hid \parallel mac_{msg}, key_{enc}\right) \quad (41)$$

The user resets the operation timer for *REGISTER_SYN* and sends the message to the M2M server; thus,

$$< REGISTER\_SYN, ecdh\_id, enc_{msg} > \quad (42)$$

The M2M server parses the message and finds a related decryption key with *ecdh_id* and checks whether the *ECDH_ACK* operation time is within an acceptable time range; if not and if *ecdh_id* is not found, then it destroys sensitive assets and terminates the operation. Otherwise, it decrypts the encrypted payload and verifies the MAC; thus,

$$\left[uid \parallel pw \parallel hid \parallel mac_{msg}\right] = \mathrm{F}_{d-AES}\left(enc_{msg}, key_{enc}\right) \quad (43)$$

$$mac'_{msg} = \mathrm{F}_{H-MAC}\left(REGISTER\_SYN \parallel ecdh_{id} \parallel uid \parallel pw \parallel hid, key_{mac}\right) \quad (44)$$

If mac_{msg} and mac'_{msg} do not match, then the M2M server destroys sensitive assets and terminates the operation. Otherwise, it chooses a salt s. It also calculates 10 bytes of temporary ID (t_uid) for operations as follows:

$$t\_uid = \mathrm{F}_{H-MAC}\left(uid, key_{enc}\right) \quad (45)$$

The M2M server calculates the password encryption key (key_{AES}) with SHA-1, as follows:

$$d = \mathrm{F}_{SHA-1}\left(k \parallel t\_uid \parallel s\right) \quad (46)$$

$$key_{AES} = [d]^{n} \quad (47)$$

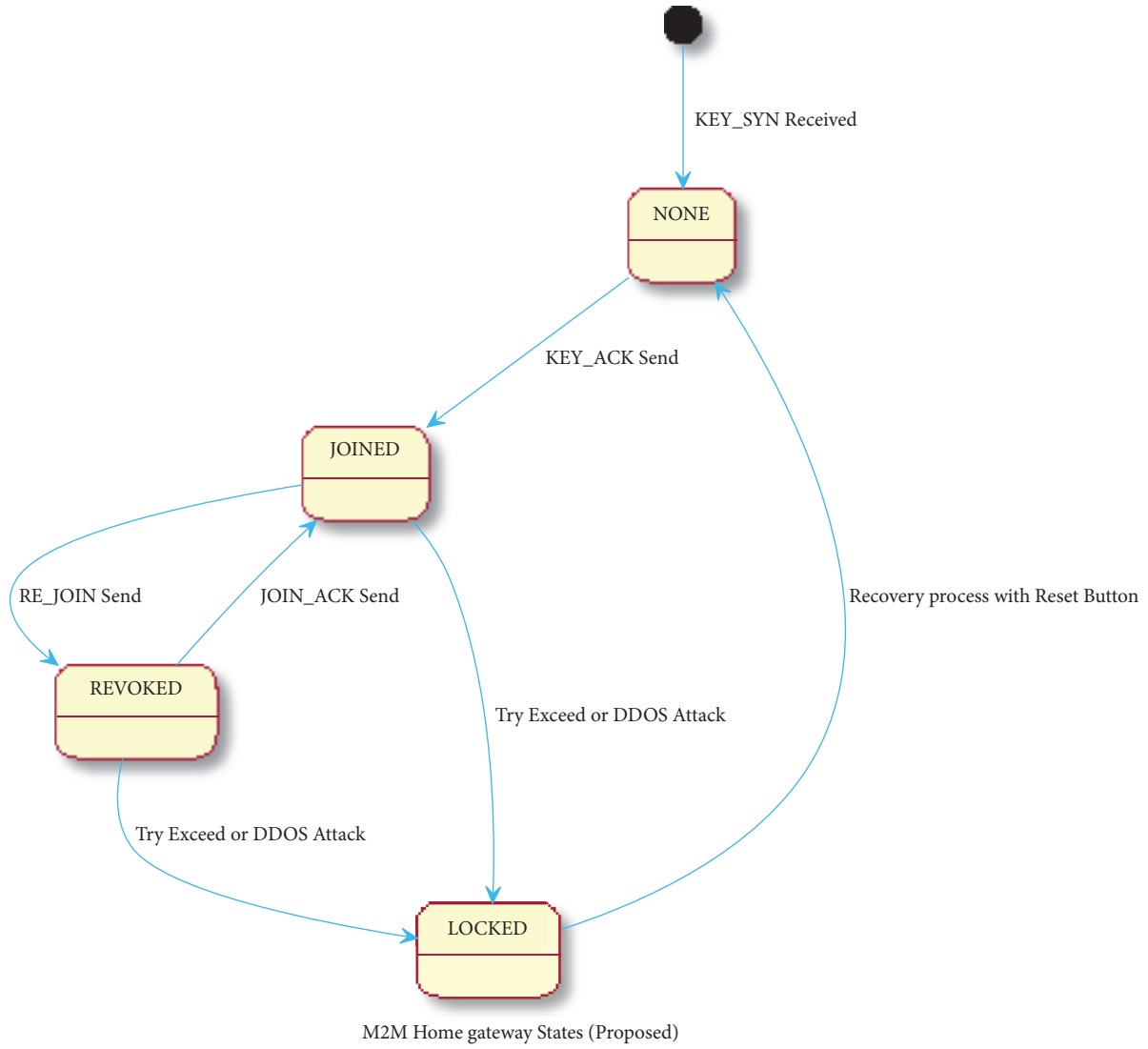

Figure 6: Home gateway state processing.

The M2M server encrypts the user password (pw) with key_{AES}, as follows:

$$p = \mathrm{F}_{e-AES}\left(pw, key_{AES}\right) \tag{48}$$

The M2M server calculates the temporary 10-byte home gateway ID (t_hid) for storage as follows:

$$t\_hid = \mathrm{F}_{e-AES}\left(hid, key_{enc}\right) \tag{49}$$

The M2M server sets the user state (u_st) to *REGISTERED* and the home gateway state (h_st) to *NONE*.

$$u\_st = \mathrm{REGISTERED} \tag{50}$$

$$h\_st = \mathrm{NONE} \tag{51}$$

At the end of the operation, the M2M server stores the following parameters:

$$< t\_uid, t\_hid, s, p, sf, key_{enc}, key_{mac} > \tag{52}$$

After this operation, the M2M server prepares a *REGISTER_ACK* message MAC and an encrypted payload as follows:

$$mac_{msg} = \mathrm{F}_{H-MAC}\left(REGISTER\_ACK \parallel ecdh\_id \parallel s, key_{mac}\right) \tag{53}$$

$$enc_{msg} = \mathrm{F}_{e-AES}\left(s \parallel mac_{msg}, key_{enc}\right) \tag{54}$$

The M2M resets the operation timer for *REGISTER_ACK* and sends it to the user; thus,

$$< REGISTER\_ACK, ecdh\_id, enc_{msg} > \tag{55}$$

The user device checks whether the *REGISTER_SYN* operation time is within an acceptable range; if not, then it destroys sensitive assets and terminates the operation.

Otherwise, it decrypts the encrypted payload and verifies the MAC as follows:

$$\left[s \parallel mac_{msg}\right] = \mathrm{F}_{d-AES}\left(enc_{msg}, key_{enc}\right) \tag{56}$$

$$mac'_{msg} = \mathrm{F}_{H-MAC}(\mathrm{msg} = REGISTER\_ACK \parallel ecdh\_id \parallel s, key_{mac}) \tag{57}$$

If mac'_{msg} and mac_{msg} are not equal, then the user device destroys sensitive assets and terminates the operation. Otherwise, it stores s and commits key_{mac} and key_{enc} storage for the next operations.

5.5.5. Proposed Scheme for the User Login and Authentication Phase. The user device selects a random x_1 number and authentication parameters g and h_1, as follows:

$$g = \mathrm{F}_{e-AES}(x_1, pw) \tag{58}$$

$$h_1 = \mathrm{F}_{SHA-1}(\mathrm{x}_1) \tag{59}$$

According to user identifier privacy, the user calculates t_uid as follows:

$$t\_uid = F_{H-MAC}(uid, key_{enc}) \tag{60}$$

The user device prepares a *LOGIN_SYN* message MAC, as follows:

$$mac_{msg} = \mathrm{F}_{H-MAC}(\mathrm{msg} = LOGIN\_SYN \parallel t\_uid \parallel g \parallel h_1 \parallel s, key_{mac}) \tag{61}$$

The user device resets the execution timer for the *LOGIN_SYN* message and transmits the message to the M2M server; thus,

$$< LOGIN\_SYN, t\_uid, g, h_1, s, mac_{msg} > \tag{62}$$

The M2M server verifies t_uid and s, which are present in the database. If the record is not found, then the M2M server terminates the process. If the user is found but the state is *LOCKED* or not equal to *REGISTERED*, *OFFLINE*, and *NEW_PASSWORD*, then it ceases the operation. If all conditions are valid, then it verifies the *LOGIN_SYN* message MAC, as follows:

$$mac'_{msg} = \mathrm{F}_{H-MAC}(LOGIN\_SYN \parallel t\_uid \parallel g \parallel h_1 \parallel s, key_{mac}) \tag{63}$$

If the message mac_{msg} and calculated mac'_{msg} are equal, the computation of key_{AES} continues in order to verify h_1. The M2M server calculates the password encryption key (key_{AES}) with SHA-1, as follows:

$$d = \mathrm{F}_{SHA-1}(k \parallel t\_uid \parallel s) \tag{64}$$

$$key_{AES} = [d]^n \tag{65}$$

$$pw' = \mathrm{F}_{d-AES}(p, key_{AES}) \tag{66}$$

$$x'_1 = \mathrm{F}_{d-AES}(g, pw') \tag{67}$$

$$h'_1 = \mathrm{F}_{SHA-1}(\mathrm{x}'_1) \tag{68}$$

If the received h_1 and the calculated h'_1 do not match, then the stored password is wrong and the M2M server terminates the operation; otherwise, the M2M server selects another random number x_2 and calculates umk and h_2, as follows:

$$umk = \left[\mathrm{F}_{SHA-1}\left(\mathrm{x}'_1 \parallel x_2\right)\right]^n \tag{69}$$

$$h_2 = F_{SHA1}(umk \parallel mid) \tag{70}$$

M2M server calculates *LOGIN_ACK* message MAC as follows:

$$mac_{msg} = \mathrm{F}_{H-MAC}(LOGIN\_ACK \parallel mid \parallel x_2 \parallel h_2 \parallel s, key_{mac}) \tag{71}$$

After this step, the M2M server resets the operation timer for *LOGIN_ACK* and sends the message to the user, as follows:

$$< LOGIN\_ACK, mid, x_2, h_2, mac_{msg} > \tag{72}$$

The user checks whether the *LOGIN_SYN* operation time is an acceptable range. If not the operation is terminated; otherwise, it verifies the *LOGIN_ACK* message MAC, as follows:

$$mac'_{msg} = \mathrm{F}_{H-MAC}(\mathrm{msg} = LOGIN\_ACK \parallel mid \parallel x_2 \parallel h_2 \parallel s, key_{mac}) \tag{73}$$

If the calculated mac'_{msg} and received mac_{msg} are not equal, then the action is terminated; otherwise, it verifies that h_2 is received from the M2M server, as follows:

$$umk' = \mathrm{F}_{SHA1}(x_1 \parallel x_2)^n \tag{74}$$

$$h'_2 = \mathrm{F}_{SHA1}\left(umk' \parallel mid\right) \tag{75}$$

If the received h_2 value and calculated h'_2 value are equal, then authentication is completed. After this point, the user device keeps umk' for secure data transmission. Otherwise, the user device terminates the process. Until this step, the user verifies the M2M server and then calculates h_3 for the M2M server mutual authentication, as follows:

$$h_3 = \mathrm{F}_{SHA1}\left(umk' \parallel t\_uid\right) \tag{76}$$

The user device calculates the *LOGIN_OK* message MAC and sends the *LOGIN_OK* message to the M2M server, as follows:

$$mac_{msg} = \mathrm{F}_{H-MAC}(LOGIN\_OK \parallel h_3, key_{mac}) \tag{77}$$

$$< LOGIN\_OK, h_3, mac_{msg} > \tag{78}$$

The M2M server checks whether the *LOGIN_ACK* operation time is in an acceptable time range; if not, then it terminates the operation. Otherwise, it verifies the *LOGIN_OK* message MAC, as follows:

$$mac'_{msg} = \mathrm{F}_{H-MAC}(\mathrm{msg} = LOGIN\_OK \parallel h_3, key_{mac}) \tag{79}$$

If the calculated mac'_{msg} is equal to the received mac_{msg}, then the M2M server verifies h_3, as follows:

$$h'_3 = \mathrm{F}_{SHA1}\,(umk \parallel t\_uid) \tag{80}$$

If the received h_3 value and calculated h'_3 value match, then authentication is completed. The M2M server sets the user state to *ONLINE* and keeps *umk* for secure data transmission.

5.5.6. Proposed Scheme User Home Gateway Key Injection. A successfully registered user has key_{enc} and key_{mac} values. These keys are used to protect sensitive data transmissions between the home gateway device, M2M server, and mobile user device. In the current design [3], the home gateway device can connect a home network and sensors, which means these devices are capable of IP (Internet Protocol) network connections over WIFI or Ethernet. We assume that this device is connected to the home IP network and can be found by a mobile user device that is connected to the same network by searching over the home network. Without a home network, the home gateway device can set up a personal-private-network by using access point mode and allowing the user device to connect via prefabricated and marked information on the device. Additionally, this connection can be authenticated over Bluetooth capable home gateway devices.

This personal-private-network provides a secure data transmission channel. The user sends keys to the home gateway device over this secure channel with a *KEY_SYN* message, as follows:

$$< KEY\_SYN, key_{enc} \parallel key_{mac} > \tag{81}$$

The home gateway device stores the key_{enc} and key_{mac} values for the network join operation and returns the *KEY_ACK* message to the user device, as follows:

$$< KEY\_ACK > \tag{82}$$

After this message, the key injection is complete.

5.5.7. Proposed Scheme Home Gateway Join Network. After the key injection operation, the home gateway device can securely join the network. The first home gateway device calculates the temporal home gateway ID (t_hid) from *hid*, as follows:

$$t\_hid = F_{H-MAC}\,(hid, key_{enc}) \tag{83}$$

The home gateway device prepares the *JOIN_SYN* message with MAC and resets the operation timer for this message and sends it to the M2M server, as follows:

$$mac_{msg} = F_{H-MAC}\,(JOIN\_SYN \parallel t\_hid, key_{mac}) \tag{84}$$

$$< JOIN\_SYN, t\_hid, mac_{msg} > \tag{85}$$

The M2M server searches t_hid in the database and attempts to obtain the detailed record. If the user-related device is not found, then the server terminates the operation. Otherwise, the M2M server obtains $\_uid$ u_st, h_st, s, key_{enc}, and key_{mac} for the related operation. If the home gateway state (h_st) or user state (u_st) is *LOCKED*, or the home gateway state is already *JOINED*, then the server terminates the operation. Otherwise, the M2M server verifies the *JOIN_SYN* message MAC, as follows:

$$mac'_{msg} = F_{H-MAC}\,(JOIN\_SYN \parallel t\_hid, key_{mac}) \tag{86}$$

If the received mac_{msg} and the calculated mac'_{msg} do not match, then the server terminates the operation. Otherwise, the server proceeds with the session key encryption. The M2M server calculates the session key, H_{key} and the session key parameter H_p for the home gateway device, as follows:

$$H_p = \left[F_{SHA1}\,(t\_uid \parallel mid \parallel s)\right]^m \tag{87}$$

$$H_{key} = \left[F_{SHA1}\left(t\_hid \parallel H_p\right)\right]^t \tag{88}$$

The M2M server stores H_{key} and prepares the *JOIN_ACK* message with MAC and the encrypted payload, as follows:

$$mac_{msg} = F_{H-MAC}\left(JOIN\_ACK \parallel H_p, key_{mac}\right) \tag{89}$$

$$enc_{msg} = F_{e-AES}\left(H_p \parallel mac_{msg}, key_{enc}\right) \tag{90}$$

$$< JOIN\_ACK, enc_{msg} > \tag{91}$$

The home gateway device checks whether the *JOIN_SYN* operation time is valid; if not, then the device terminates the process. Otherwise, the device decrypts the *JOIN_SYN* message and verifies MAC, as follows:

$$\left[H_p \parallel mac_{msg}\right] = F_{d-AES}\left(enc_{msg}, key_{enc}\right) \tag{92}$$

$$mac'_{msg} = F_{H-MAC}\left(JOIN\_ACK \parallel H_p, key_{mac}\right) \tag{93}$$

If the received mac_{msg} and calculated mac'_{msg} do not match, then the home gateway device terminates the operation. Otherwise, the device calculates H'_{key} and keeps it in memory for secure transmission, as follows:

$$H'_{key} = \left[F_{SHA1}\left(t\_hid \parallel H_p\right)\right]^t \tag{94}$$

The home gateway device prepares the *JOIN_OK* message for the M2M server with MAC, as follows:

$$mac_{msg} = F_{H-MAC}\,(JOIN\_OK \parallel t\_hid, key_{mac}) \tag{95}$$

$$< JOIN\_OK, t\_hid, mac_{msg} > \tag{96}$$

The M2M server checks whether the *JOIN_ACK* operation time is valid; if not, then it terminates the operation. Otherwise, it verifies *JOIN_OK* MAC and sets the home gateway state (h_st) to *JOINED*, as follows:

$$mac'_{msg} = F_{H-MAC}\,(JOIN\_OK \parallel t\_hid, key_{mac}) \tag{97}$$

If MAC verification has failed or a timeout occurs, and *JOIN_OK* is not received, then the M2M server terminates the operation.

5.5.8. Proposed Scheme for the User Password Change. The user device calculates t_uid for the operation. It also prepares a *PWD_SYN* message with MAC and an encrypted payload, as follows:

$$t\_uid = F_{H-MAC}(uid, key_{enc}) \tag{98}$$

$$mac_{msg} = F_{H-MAC}(PWD\_SYN \parallel t\_uid \parallel pw \parallel pw\_new \parallel s, key_{mac}) \tag{99}$$

$$enc_{msg} = F_{e-AES}(pw \parallel pw\_new \parallel s \parallel mac_{msg}, key_{enc}) \tag{100}$$

The user resets the operation timer for *PWD_SYN* and sends the message to the M2M server, as follows:

$$< PWD\_SYN, t\_uid, enc_{msg} > \tag{101}$$

The M2M server acquires information from t_uid and checks the status of the user. If the user state is *LOCKED* or not equal to *ONLINE*, then the M2M server terminates the operation. Otherwise, the M2M server acquires keys from the record and decrypts *PWD_SYN* and verifies the message MAC, as follows:

$$[pw \parallel pw\_new \parallel s \parallel mac_{msg}] = F_{d-AES}(enc_{msg}, key_{enc}) \tag{102}$$

$$mac'_{msg} = F_{H-MAC}(PWD\_SYN \parallel t\_uid \parallel pw \parallel pw\_new \parallel s, key_{mac}) \tag{103}$$

If the received mac_{msg} and calculated mac'_{msg} do not match, then the M2M server terminates the operation. Otherwise, the M2M server calculates key_{AES} to verify whether the password is correct, as follows:

$$key_{AES} = [F_{SHA1}(k \parallel t\_uid \parallel s)]^n \tag{104}$$

$$p' = F_{e-AES}(pw, key_{AES}) \tag{105}$$

If the stored p is not equal to p', then the password verification fails and the M2M server terminates the operation. Otherwise, the M2M server selects a new salt s_{new} and calculates a new key'_{AES}. Finally, the M2M server encrypts the password for storage as follows:

$$key'_{AES} = [F_{SHA1}(k \parallel t\_uid \parallel s_{new})]^n \tag{106}$$

$$p'_{new} = F_{e-AES}(pw\_new, key'_{AES}) \tag{107}$$

The M2M server stores p'_{new} and prepares a *PWD_ACK* message with MAC and the encrypted payload. The prepared message is sent to the user device, as follows:

$$mac_{msg} = F_{H-MAC}(PWD\_ACK \parallel s_{new}, key_{mac}) \tag{108}$$

$$enc_{msg} = F_{e-AES}(s_{new} \parallel mac_{msg}, key_{enc}) \tag{109}$$

$$< PWD\_ACK, enc_{msg} > \tag{110}$$

The user device checks whether the *PWD_SYN* operation time is inside the valid range. If not, it then sets $fail\_flag$ to true. If the execution time check fails, then the device skips MAC verification and sends *PWD_NACK*. Otherwise, the user device verifies *PWD_ACK* MAC, as follows:

$$[s_{new} \parallel mac_{msg}] = F_{d-AES}(enc_{msg}, key_{enc}) \tag{111}$$

$$mac'_{msg} = F_{H-MAC}(PWD\_ACK \parallel s_{new}, key_{mac}) \tag{112}$$

If the received mac_{msg} and calculated mac'_{msg} do not match, then the user device sets $fail\_flag$ to true and sends *PWD_NACK* to the M2M server, as follows:

$$mac_{msg} = F_{H-MAC}(PWD\_NACK \parallel t\_uid, key_{mac}) \tag{113}$$

$$< PWD\_NACK, t\_uid, mac_{msg} > \tag{114}$$

The M2M server verifies *PWD_NACK* MAC. If MAC fails, then the server rejects the request. Otherwise, the server restores p_{new} to p and sets the user device state to the previous state. If $fail\_flag$ is not true, then the user stores s_new and prepares*PWD_OK* for the M2M server with MAC, as follows:

$$mac_{msg} = F_{H-MAC}(PWD\_OK \parallel t\_uid, key_{mac}) \tag{115}$$

$$< PWD\_OK, t\_uid, mac_{msg} > \tag{116}$$

The M2M server verifies the *PWD_OK* message MAC, as follows:

$$mac'_{msg} = F_{H-MAC}(PWD\_OK \parallel t\_uid, key_{mac}) \tag{117}$$

If the received mac_{msg} is equal to the calculated mac'_{msg}, then the M2M server sets the user device state to *NEW_PASSWORD* and the home gateway device state to *REVOKED*. Otherwise, the server rejects the requests. The home gateway device key's revocation is reported to the device with a *RE_JOIN* message by the M2M server, which calculates the *RE_JOIN* message with MAC and sends it to the home gateway device. When the home gateway device verifies the *RE_JOIN* message MAC, the home gateway device starts joining the flow again to refresh its session key; thus,

$$mac_{msg} = F_{H-MAC}(RE\_JOIN \parallel t\_hid, key_{mac}) \tag{118}$$

$$< RE\_JOIN, mac_{msg} > \tag{119}$$

5.5.9. Proposed Scheme User Logout. User device safe logout for reliable operations is the most critical part. The user device calculates a temporary user ID (t_uid) and *LOGOUT_SYN* message with MAC for the M2M server, as follows:

$$mac_{msg} = F_{H-MAC}(LOGOUT\_SYN \parallel t\_uid, key_{mac}) \tag{120}$$

$$< LOGOUT\_SYN, t\_uid, mac_{msg} > \tag{121}$$

The user *LOGOUT_SYN* message is sent to the M2M server, as follows:

$$mac'_{msg} = F_{H-MAC}(LOGOUT\_SYN \parallel t\_uid, key_{mac}) \quad (122)$$

The M2M server finds the keys from *t_uid* and calculates *LOGOUT_SYN* mac'_{msg} and compares it with the received mac_{msg}. If MAC does not match, then the M2M server rejects the operation. Otherwise, it sets the user device state to *OFFLINE* and prepares a *LOGOUT_ACK* message for the user device, as follows:

$$mac_{msg} = F_{H-MAC}(LOGOUT\_ACK \parallel t\_uid, key_{mac}) \quad (123)$$

$$< LOGOUT\_ACK, t\_uid, mac_{msg} > \quad (124)$$

When the user receives *LOGOUT_ACK*, it verifies MAC, as follows:

$$mac'_{msg} = F_{H-MAC}(LOGOUT\_ACK \parallel t\_uid, key_{mac}) \quad (125)$$

If the received mac_{msg} and calculated mac'_{msg} match, then the user device updates its state to the *OFFLINE* case. Otherwise, it rejects the request.

5.5.10. Home Gateway Rejoin Network. This operation is processed after the password change operation. The M2M server sets the home gateway device state to *REVOKED* and prepares the *RE_JOIN* message for the home gateway device, as follows:

$$mac_{msg} = F_{H-MAC}(RE\_JOIN \parallel t\_hid, key_{mac}) \quad (126)$$

$$< RE\_JOIN, mac_{msg} > \quad (127)$$

The home gateway device calculates *t_hid* and verifies *RE_JOIN* MAC. If MAC verifies it, then the home gateway device processes the home gateway join network flow.

6. Protocol Analysis

The formal protocol analysis is not processed. We plan to process BAN-LOGIC or ProVerif for future work. We created a MATLAB simulation and tested valid and invalid case behaviors for our proposal and [3] designs.

7. Security Analysis

The proposed scheme guarantees protection against replay attacks, changing distance attacks, same-type-device attacks, composition attacks, redirection attacks, MITM attacks, substitution attacks, DoS attacks, forging attacks, colluding attacks, flooding attacks, false message attacks, Sybil attacks, message modifications, wormhole attacks, blackhole attacks, attribute-trace attacks, eavesdropping attacks, chosen-plaintext attacks, spam attacks, identity theft attacks, user manipulation attacks, routing attacks, linkability attacks, rejection attacks, successive response attacks, packet analysis attacks, packet tracing attacks, and brute-force attacks.

We used a hybrid cryptographic scheme (ECDH, AES, SHA-1, and H-MAC) to share confidentiality and integrity keys for required entities securely. Identity information for home gateway and user devices were transformed to temporary identities for anonymity with H-MAC, which uses a securely shared MAC key. In addition, user password and home gateway key revocations were developed in the proposed scheme.

7.1. Replay Attack. Scheme [3] is classified as partially supported for this attack in [1, 3] claims protection for this attack. They assume that if an intruder intercepts a valid login message and replays the same message that contains $< uid, g, h_1, s >$ after logout detection, then the login operation terminated according to the random number input x_1 verification. However, for the home gateway device, there is no replay attack protection. The attacker can intercept a home gateway device request and send *hid* to the M2M server to obtain encryption key information. A registered attacker can send bogus data to the M2M server over this communication channel. We added an additional home gateway key's injection phase to protect the device registration from this kind of attack. Additionally, state machine flags detect invalid replay requests. Invalid status detection aborts the operation on the M2M server side. Another issue pertains to [3]. They assume that registration and password change channels are secure and that these operations are being performed by the user device. However, if there is a secure channel, then there is no more authentication required. Our scheme assumes that there is no secure channel furthermore to protect each phase for replay attacks. For these reasons, we classified our protocol as fully supported.

7.2. Changing Distance Attack. Scheme [3] is listed as not being supported for this attack in [1]. There is no time measurement during the operations for each phase in [3]. An intruder can intercept a message and then send one after its own analysis to the receiver. There is no detection of this attack. The changing distance attack has effects on request timing and the proposed scheme has a minimum and maximum operation time range for the detection of suspicious operations. This additionally provides timeout detection for each request. For these reasons, we classified our scheme as being fully supported.

7.3. Same-Type-Device Attack. Scheme [3] is classified as not being supported for this attack in [1]. Scheme [3] The home gateway join request can intercept an intruder, and the original *hid* can collect from the public channel. The attacker uses *hid* to send multiple register messages to the M2M server to acquire the encryption key. If more than one registration request sending is available, then the encryption key can be revealed via sniffing and decoding the home gateway and M2M server traffic. Moreover, transferred messages can be

modified by an attacker, and fraudulent information can be sent to the M2M server to block services.

Our proposed scheme only accepts MAC verified messages. A join message is sent by a home gateway device after user key injection. Moreover, each device is uniquely identified with an ID, and this ID protects the device from sniffing with H-MAC diversification on the public network. These protection mechanisms defend against same-type-device attacks. If an intruder virtually registers the device to the M2M server, it should have a user integrity key which is securely injected through the home gateway with the personal network as described in the home gateway join phase section. For these reasons, we classified our scheme as being fully supported.

7.4. Composition Attack. Scheme [3] is classified as being not supported for this attack in [1]. Composition attacks are based on collecting several attributes. The composition of this attribute reveals sensitive assets. Scheme [3] user registration and password change operation are not as secure as we mentioned previously. Moreover, the user ID can be sniffed and used to collect or request user-sensitive information over bogus systems. In our scheme, all sensitive assets are protected by the encryption key and MAC. These encryption keys are transported under ECDH keys. This mechanism protects against revealing sensitive assets. For these reasons, we labeled our scheme as being fully supported.

7.5. Redirection Attack. Scheme [3] is classified as being not supported for this attack in [1]. An attacker initiates a redirection attack by simulating the M2M server to obtain the user and home gateway device's network information and sensitive assets. Scheme [3] targets M2M server information injection and protection not mentioned and requests to redirect via a fraudulent base station to the attacker server, and messages can be analyzed and modified for the real server.

Our design protects redirection attacks with time measurements and redirected messages are encrypted. MAC protects useful information that cannot be obtained from messages. The attacker cannot modify messages. Furthermore, we have the secure key injection to provide secure registration for home gateway devices as mentioned previously. Sensitive assets are never transmitted over a public network and messages are protected with MAC. For this reason, the attacker must know the encryption and MAC keys to sniff or modify messages. Therefore, we ordered our scheme as being fully supported.

7.6. MITM Attack. [3] is labeled as being partially supported for this attack in [1]. However, there is no MAC verification for transmitted messages in scheme [3]. Messages can be modified and operations ceased. Additionally, terminated activities force the user to retry and this causes it to collect more data about the attacked user to recover data. Another issue is that the attacker can cause the transmission of wrong data with modification.

Our scheme MAC verification provides message modification attack protection. Sensitive assets are protected with an encryption key for confidentiality and a MAC key for integrity. Sensitive assets are never transmitted clear over a public network and this provides resistance to an MITM attack. For these reasons, we classified our scheme as being fully supported.

7.7. Substitution Attack. Scheme [3] is marked as not supported for this attack in [1]. The substitution attack is a particular type of MITM attack to replace original implementation with derived implementation to leak information from the transmission. Scheme [3] user registrations, user password changes, and home gateway devices join phases that are not protected. An attacker can quickly retrieve a sensitive password and user ID from messages.

The proposed scheme original user ID is never transmitted in a public network. This privacy protection is provided with the ECDH-key-pair between the user device and the M2M server and a secure key injection between the user device and the home gateway device. This method offers secure key sharing between the home gateway device, the user device, and the M2M server to protect the ID on the public network in the event of this kind of attack. Therefore, a substitution attack is not suitable for this proposed scheme, and we organized our scheme as being fully supported.

7.8. DoS Attack. [3] is assigned as not being supported for this attack in [1]. A DoS attacker sends a bulk message system to use more system resources to block servers. Scheme [3] does not have any protection or detection of this kind of attack. In our scheme, on the other hand, DoS attacks are avoided with an operation time measurement and state machine control mechanism. Moreover, messages are protected with MAC and encryption keys. Attackers cannot send valid messages to modify state machine variables on the M2M server. If an attacker sends fewer messages than expected within the minimum operation period, then this is marked as a suspicious operation and it is terminated. Moreover, there is a try count mechanism if an attacker sends continuous messages with invalid MACs; then the user or device is stated as *BLOCKED* and requests are rejected. This prevents the use of more system resources, such as cryptographic operations on the server. We classified our scheme as fully supported for this attack.

7.9. Forging Attack. Scheme [3] is marked as not being supported for this attack in [1]. Scheme [3] user registration, user password changes, and home gateway device join phases are not protected. The attacker can quickly retrieve a sensitive password and user ID from messages. However, in the proposed scheme, forging attack modifications are prevented with MAC protection, state machine control, and operation time measurements. The attacker cannot extract keys from our scheme to join the network and standard algorithms immune to brute-force attacks are used. We deemed our protocol as being fully supported for this attack.

7.10. Colluding Attack. Design [3] is classified as not being supported for this attack in [1]. The colluding attack is a

guessing attack. The attacker can use old information for new requests. Scheme [3] uses salt (s) to derive a home gateway join key. This salt is updated with a user password change on the server side; however, there is no revocation and the home gateway key is updated after a password change operation.

Our scheme password change request also has a key revocation flow to update any old keys. In addition, every message is protected with MAC protection and IDs are never transmitted over a public network. Therefore, an attacker should know the MAC key for the colluding attack, and this is not suitable for this proposed scheme. For these reasons, our plan is deemed by us to be fully supported for this attack.

7.11. Flooding Attack. [3] is labeled as not being supported for this attack in [1]. Flooding attacks such as SYN-FLOODING attacks occur during communication channel initiation. An attacker can sniff the first message and replay it to block services. Scheme [3] does not have a try count mechanism for requests to avoid this kind of attack. Our protocol avoids the operation time measurement and state machine control mechanism. Additionally, messages are protected with MAC verification. The attacker should know the MAC key in order to send the first request to the M2M server, so this is not feasible for the attacker and our scheme is fully supported against this attack.

7.12. False Message Attack. Scheme [3] is classified as partially supported for this attack in [1]. A malicious message attacker can inject false messages to block the system. Scheme [3] does not have MAC verification or try count for requests.

In the proposed scheme, an attacker cannot send any messages without a MAC key. MAC protection and bulk requests trigger the timing control mechanism to detect the attacker. Moreover, rejections never reveal usable information for attackers. We organized our scheme as being fully supported for this attack.

7.13. Sybil Attack. [3] is labeled as partially supported for this attack in [1]. Sybil attacks are based on copying the identity of the user device or home gateway device in the current network. Scheme [3] user ID is sent in an open format and it can be sniffed by an attacker.

Sybil attacks are avoided with a temporary ID, MAC protection, and sensitive message encryption in this proposed scheme. We identified our project as being fully supported for this attack.

7.14. Message Modification Attack. [3] is classified as being not supported for this attack in [1]. Scheme [3] messages are not protected with MAC and they can be modified during transmission.

In our scheme, message integrity is protected with a MAC key. Attackers should know the MAC key in order to modify messages. We ordered our design as being fully supported for this attack.

7.15. Wormhole Attack. Scheme [3] is marked as being partially supported for this attack in [1]. Wormhole attacker tunnels packets from one point to another point in the network for blocking or for customized analysis. This attack affects transmission and execution time. Scheme [3] does not have an operation time analysis for requests. However, in our scheme, operation time measurement can detect this kind of attack. Moreover, attackers should have MAC and encryption keys to join the M2M secure network. Our scheme is deemed by us to be fully supported for this attack.

7.16. Blackhole Attack. Scheme [3] is assigned as partially supported for this attack in [1]. A blackhole attacker returns a false response to broadcast messages to block communications between entities. Scheme [3] does not have message verification and it can be affected by this attack.

Our proposed design requests and responses are protected with MAC verification. The attacker cannot send a valid response to the requesting entity without an integrity key. Without such keys, the attacker cannot affect the communication channel. Moreover, this kind of attack affects request timing and the proposed scheme time analysis feature detects this attack. We claim that our scheme is fully supported for this attack.

7.17. Attribute-Trace Attack. Scheme [3] is labeled as not being supported for this attack in [1]. An attacker traces attributes and analyses data from all collected information. For this reason, we protect [3] registration and password change phase. Sensitive items, such as user ID and password, can be sniffed during transmission, as mentioned in Section 3.

The proposed scheme encrypts sensitive assets with the encryption key and an asymmetric key pair is used for cipher-key transportation. Therefore, attribute tracing is hardened for usable information revealing. We have deemed our scheme as being fully supported for this attack.

7.18. Eavesdropping Attack. Design [3] is assigned as being partially supported for this attack in [1]. Eavesdropping attacks are based on listening to the communication channel and obtaining useful information about entities. Scheme [3] user registration and password change phases can be easily affected by this attack and sensitive assets such as password and user ID are readily revealed.

In the current scheme, the eavesdropping attack is prevented with the ECDH-key-pair provided for key transportation and message MAC protection. Moreover, sensitive assets are encrypted with the encryption key and IDs are diversified to a temporal ID on the public network for anonymity. We marked our scheme as being fully supported for this attack.

7.19. Chosen-Plaintext Attack. Scheme [3] is listed as being not supported for this attack in [1]. Scheme [3] home gateway key revocation that is missing has been considered in Sections 1 and 3.

In our scheme, the selected plaintext attack is not proper. There are keys and user revocation mechanisms to refresh key values. Additionally, the algorithms use 128-bit AES encryption and ECDH encryption which are not ideal for brute-force attacks. Our scheme is fully supported for this attack.

7.20. Spam Attack. Scheme [3] is assigned as being partially supported for this attack in [1]. The attacker can join the network and send invalid messages to the system to break the communication channel and states during operations or they modify original message data [3]. Messages can change and invalid messages can be sent over [3] the channel to suspend communication.

The proposed scheme uses message MAC protection, operation time measurement, and state machine management to avoid spam attacks. For these reasons, our scheme is labeled as being fully supported for this attack.

7.21. Identity Theft Attack. Design [3] is classified as being partially supported for this attack in [1]. An attacker sniffs the entity ID and simulates messages [3]. User identifiers are stored on a smartcard, but, during data transmission, the user ID is sent in a clear format. The attacker can easily sniff the user ID and simulate messages.

The proposed scheme uses a temporary ID and hides the identity with H-MAC. Real IDs are never transmitted over public networks. We classified our scheme as being fully supported for this attack.

7.22. User Manipulation Attack. Scheme [3] is assigned as being partially supported for this attack in [1]. In this attack, an adversary attempts to appear as an M2M server. In [3], the home gateway device can route to the bogus M2M server and then the home gateway device's message encryption key can be retrieved from the messages. Additionally, the attacker can sniff or modify sensitive assets carried in messages between the home gateway device and the M2M server.

The attacker can acquire the user ID (*uid*) and registered device ID (*hid*) from an unprotected user registration phase. Additionally, the attacker sniffs traffic between the M2M server and the home gateway device to obtain H_p for a key calculation. After this step, the attacker can provide a fraudulent network for the attacked home gateway device to route a request to the fake M2M server and return H_p to the home gateway device for the session key. Home gateway devices use H_p and calculate H_{key}. This procedure is identical to that performed by the attacker and sensitive information can be collected from user home gateway devices or commands can be sent to the home gateway to control devices, as follows:

$$H_p = [h(uid, mid, s)]^m \tag{128}$$

$$H_{key} = \left[h\left(hid, H_p\right)\right]^t \tag{129}$$

In our scheme user manipulation attack is not applicable. Because we encrypt all sensitive assets H_p are encrypted and MAC protected during transmission as follows:

$$mac_{msg} = F_{H-MAC}\left(JOIN\_ACK \parallel H_p, key_{mac}\right) \tag{130}$$

$$enc_{msg} = F_{e-AES}\left(H_p \parallel mac_{msg}, key_{enc}\right) \tag{131}$$

$$< JOIN\_ACK, enc_{msg} > \tag{132}$$

Other requests also ciphered. Our scheme fully supported for this attack.

7.23. Routing Attack. Scheme [3] is listed as being partially supported for this attack in [1]. In this attack, the attacker first uses user manipulation attack to acquire the home gateway device's communication key and routes the user request to intercept user and home gateway communication. During this phase, the attacker can collect sensitive information from the home gateway device. Scheme [3] explained the operation as being suitable. However, in the proposed scheme, routing attacks are avoided with MAC and encrypted messages and operation time measurements can detect routings. For these reasons, our scheme is fully supported for this attack.

7.24. Linkability Attack. Scheme [3] is marked as not being supported for this attack in [1]. In this attack, the attacker can obtain the required inputs from different messages to calculate or decode sensitive assets. Scheme [3] user registration and password change operations are not protected and the user ID and password is sent in a clear format. In addition, the home gateway join phase is not sufficiently well protected, and the attacker can reveal a secure channel key. This is explained in the user manipulation attack.

Sensitive and usable assets are never transmitted over a public network. Therefore, the linkability attack is not applicable to the proposed scheme.

Our scheme is fully supported for this attack.

7.25. Rejection Attack. Scheme [3] is labeled as not being supported for this attack in [1]. This is a type of eavesdropping attack. An attacker simulates an M2M server and refuses home gateway or user requests to block service. Scheme [3] user and home gateway devices can be routed to the bogus M2M server, as explained in the routing attack.

The proposed design avoids eavesdropping attacks, and this protects our system from rejection attacks. A home gateway can be online after key injection and response protection with MAC verification. The attacker should know the integrity key to send a valid rejection message for the requested entity. The proposed scheme is not suitable for this attack. For these reasons, our project is fully supported for this attack.

7.26. Successive Response Attack. Scheme [3] is classified as being not recommended for this attack in [1]. In this attack, an adversary server simulates the M2M server and accepts wrong authorization requests. After this operation, the attacker can send bogus home gateway information to the user device. Scheme [3] user registration and password change phases are considered to work on the secure channel, so there is no protection in these phases. Also according to design [3], the home gateway join phase is not protected well. The home gateway secure channel key can be sniffed as explained by the user manipulation attack.

Our scheme messages are protected with MAC verification. Therefore, wrong messages are discarded and successive response attacks are avoided. If an attacker wants to send a

Table 8: Analyzed attacks.

SC1	Replay Attack	**SC16**	Blackhole Attack
SC2	Changing Distance Attack	**SC17**	Attribute-Trace Attack
SC3	Same-Type-Device Attack	**SC18**	Eavesdropping Attack
SC4	Composition Attack	**SC19**	Chosen-Plaintext Attack
SC5	Redirection Attack	**SC20**	Spam Attack
SC6	Man-In-The-Middle Attack	**SC21**	Identity Theft Attack
SC7	Substitution Attack	**SC22**	User Manipulation Attack
SC8	DOS Attack	**SC23**	Routing Attack
SC9	Forging Attack	**SC24**	Linkability Attack
SC10	Colluding Attack	**SC25**	Rejection Attack
SC11	Flooding Attack	**SC26**	Successive Response Attack
SC12	False Message Attack	**SC27**	Packet Analysis Attack
SC13	Sybil Attack	**SC28**	Packet Tracing Attack
SC14	Message Modification	**SC29**	Brute-Force Attack
SC15	Wormhole Attack		

successive response, then he should know the integrity key, and this is not suitable in the proposed scheme. For these reasons, our method is fully supported for this attack.

7.27. Packet Analysis Attack. Method [3] is classified as partially endorsed for this attack in [1]. In this attack, the attacker analyzes traffic to obtain the user ID, password, and sensitive assets during transmission.

Scheme [3] login operation does not reveal the password, but, during the registration and password change phase, the user ID and password are sent in a clear format and are open to sniffing. Other requests are encrypted.

The proposed scheme packet analysis is hardened and not suitable for the attacker. It uses the ECDH-key-pair provided for key transportation and message MAC protection. Moreover, sensitive assets are encrypted with the encryption key and IDs are diversified into a temporal ID on the public network for anonymity. For these reasons, our scheme is fully supported for this attack

7.28. Packet Tracing Attack. Scheme [3] is classified as being partially supported for this attack in [1]. In this attack, packets are traced with attributes and requests and responses are matched with identifiers to collect or decode sensitive information. Scheme [3] data packages can be determined with the user ID. However, after login, all traffic is encrypted with the session key, so packet tracing is not an issue. However, registration and password change requests have a significant problem with packet tracing attacks. The user ID and password are sent in a clear format and packets are modified according to attacker requirements.

In the proposed scheme, packets can be traced but they cannot be modified. Sensitive assets are revealed during transmission. Packet analysis is hardened. The proposed scheme uses the ECDH-key-pair provided for key transportation and message MAC protection. Moreover, sensitive assets are encrypted with the encryption key and IDs are diversified into a temporal ID on the public network for anonymity. For these reasons, our scheme is fully supported for this attack.

7.29. Brute-Force Attack. Scheme [3] is classified as not being recommended for this attack in [1]. There are two kinds of brute-force attack.

The first is about decoding sniffed traffic. If we do not think about nonsecure user registration and password change phases in [3], then the used standard AES algorithm is immune to the brute-force attack. However, design [3] user registration and password change phases send clear sensitive data (user ID, home gateway ID, and user password), which is a significant problem.

In our scheme, we have secure communication for all stages and we used standard AES and trimmed H-MAC algorithms immune to brute-force attacks. Moreover, [3] does not have the revocation mechanism. In the proposed scheme, brute-force is avoided with a user password, key revocation, and the home gateway key's revocation mechanism.

The second one is about repeatedly attempting to log in. Scheme [3] does not have try counter for this kind of attack. The proposed scheme provides the try count to avoid invalid attempts. We classified our system as fully supported for this kind of attack.

8. Performance and Security Comparisons

In this section, we compare the security of our design with the previous designs [2–6, 8–11] and we compare the performance of our schemes with the previous design [3]. Previous scheme results were taken from the survey [1] and the proposed scheme analysis was added to the results. The security analysis attacks are shown in Table 8. Table 9 shows the security comparison of each scheme and how our proposed system provides more security and

Table 9: Security comparison [1].

	SC1	SC2	SC3	SC4	SC5	SC6	SC7	SC8	SC9	SC10	SC11	SC12	SC13	SC14	SC15	SC16	SC17	SC18	SC19	SC20	SC21	SC22	SC23	SC24	SC25	SC26	SC27	SC28	SC29
Our.	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]	[√]
[3]	√					√						√	√		√	√		√		√	√	√	√				√	√	
[2]	√				[√]	[√]	√	√							√	√											√	√	√
[4]	√				[√]	[√]	√	√	√	√					√	√													√
[5]					√			√		√	√	√				√						√							√
[6]	√					√	√					√	√	√		√	√	√	√	√									√
[8]	[√]	[√]	[√]	[√]	√	√	√			√	√	√			√	√				√	√	√	√	√					√
[9]					√	√	√																						
[10]																													√
[11]	√				[√]	[√]					√	√			√	√		√	√	√	√	√					√	√	

Fully supported = [√], partially supported = √, and empty = not supported.

Table 10: Operation notations and approximately single operations.

Operation Name	Operation Notation	Approximately Time (sn)
ECDH Generate Key Pair	*EI*	0.009486
ECDH Key Derive	*EG*	0.004638
ECDH Encryption	*EE*	0.001605
ECDH Decryption	*ED*	0.000293
AES Encryption	*SE*	0.073256
AES Decryption	*SD*	0.068992
H-MAC – SHA256	*HM*	0.000267
SHA-1	*HH*	0.046198

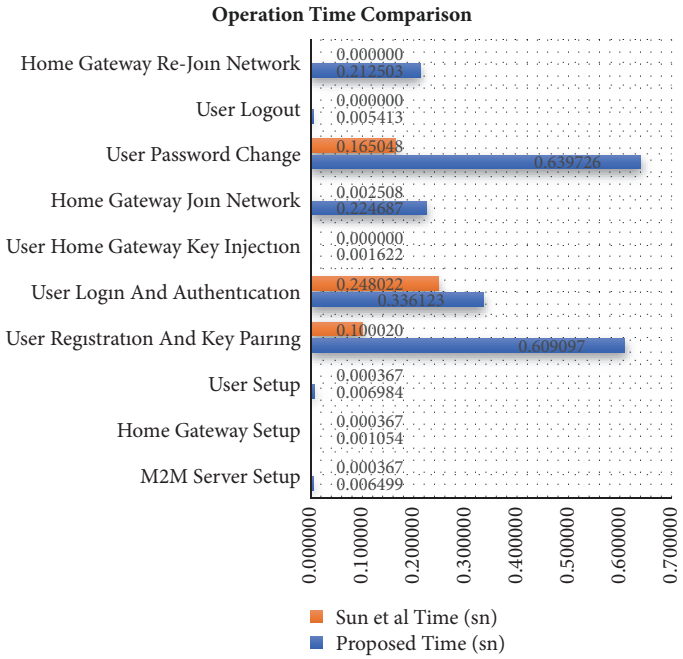

Figure 7: Operation time comparison between Sun et al. [3] and the proposed scheme.

countermeasures for assaults than previous designs. We also compared the proposed method with [3] for performance, network congestion, and resource usage. Table 10 shows the operations notations, and Table 11 shows the operation costs of scheme [3] and the proposed scheme. The calculated times are based on approximately single operation times, and the measured times were based on MATLAB simulated results. Performance comparisons are also shown in Figure 7. The most frequently used login and authentication operation has seven more H-MAC operations in our scheme compared to scheme [3]. The H-MAC is used for anonymity and message modification protection. H-MAC operation usage increases current login and authentication operation times by only 0.088101 seconds. Operation resource usages and network congestion comparisons are shown in Figures 8 and 9.

9. Conclusions

In this paper, we offer an end-to-end secure communication scheme for M2M networks. All required analyses of the proposed scheme have been provided, but formal verification is not complete in this paper. Future work is planned to complete formal verification, increase performance, and decrease resource usage for the proposed scheme based on our measurements.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Table 11: Operation comparison between Sun et al.'s [3] and the proposed scheme.

Scheme Operations	User		M2M Server		Home Gateway		Total	
	Proposed	[3]	Proposed	[3]	Proposed	[3]	Proposed	[3]
M2M Server Setup	0	0	$1EI$	0	0	0	$1EI$	0
Home Gateway Setup	0	0	0	0	0	0	0	0
User Setup	$1EG + 1HH$	0	0	0	0	0	$1EG + 1HH$	0
User Registration and Key Pairing	$1SE + 1SD + 2HM + 1EG + 1ED$	0	$2SE + 1SD + 4HM + 1EG + 1EE + 1HH$	$1SE + 1HH$	0	0	$3SE + 2SD + 6HM + 2EG + 1EE + 1ED + 1HH$	$1SE + 1HH$
User Login And Authentication	$1SE + 3HM + 4HH$	$1SE + 5HH$	$2SD + 4HM + 5HH$	$2SD + 4HH$	0	0	$2SD + 1SE + 7HM + 9HH$	$2SD + 1SE + 9HH$
User Home Gateway Key Injection	0	N/A	0	N/A	0	N/A	0	N/A
Home Gateway Join Network	0	0	$1SD + 2SE + 10HM + 5HH$	$2HH$	$1SD + 4HM + 1HH$	$1HH$	$2SD + 2SE + 14HM + 6HH$	$3HH$
User Password Change	$1SE + 1DE + 3HM$	0	$3SE + 1SD + 4HM + 2HH$	$2SE + 2HH$	0	0	$4SE + 2SD + 7HM + 2HH$	$2SE + 2HH$
User Logout	$2HM$	N/A	$3HM$	N/A	0	N/A	$5HM$	N/A
Home Gateway Re-Join Network	0	N/A	$1SD + 2SE + 11HM + 5HH$	N/A	$1SD + 5HM + 1HH$	N/A	$2SD + 2SE + 16HM + 6HH$	N/A

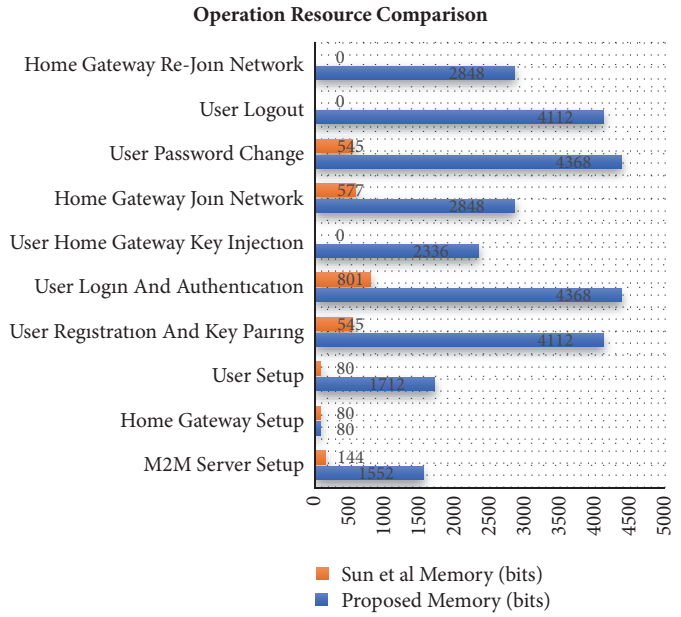

Figure 8: Operation resource comparison between Sun et al. [3] and the proposed scheme.

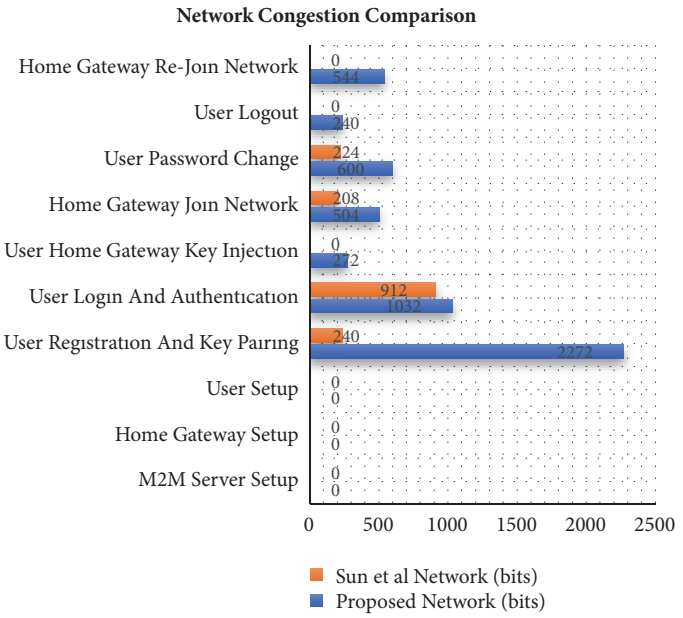

Figure 9: Network congestion comparison between Sun et al. [3] and the proposed scheme.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] M. A. Ferrag, L. A. Maglaras, H. Janicke, J. Jiang, and L. Shu, "Authentication protocols for internet of things: a comprehensive survey," *Security and Communication Networks*, vol. 2017, Article ID 6562953, 41 pages, 2017.

[2] C. Lai, R. Lu, D. Zheng, H. Li, and X. Sherman, "GLARM: group-based lightweight authentication scheme for resource-constrained machine to machine communications," *Computer Networks*, vol. 99, pp. 66–81, 2016.

[3] X. Sun, S. Men, C. Zhao, and Z. Zhou, "A security authentication scheme in machine-to-machine home network service," *Security and Communication Networks*, vol. 8, no. 16, pp. 2678–2686, 2015.

[4] C. Lai, H. Li, R. Lu, and X. Shen, "SE-AKA: a secure and efficient group authentication and key agreement protocol for

LTE networks," *Computer Networks*, vol. 57, no. 17, pp. 3492–3510, 2013.

[5] C. Lai, H. Li, X. Liang, R. Lu, K. Zhang, and X. Shen, "CPAL: A conditional privacy-preserving authentication with access linkability for roaming service," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 46–57, 2014.

[6] A. Fu, S. Lan, B. Huang, Z. Zhu, and Y. Zhang, "A novel group-based handover authentication scheme with privacy preservation for mobile WiMAX networks," *IEEE Communications Letters*, vol. 16, no. 11, pp. 1744–1747, 2012.

[7] X. Li, Y.-P. Xiong, J. Ma, and W.-D. Wang, "An efficient and security dynamic identity based authentication protocol for multi-server architecture using smart cards," *Journal of Network and Computer Applications*, vol. 35, no. 2, pp. 763–769, 2012.

[8] D. Chen et al., "S2M: a lightweight acoustic fingerprints based wireless device authentication protocol," *IEEE Internet of Things Journal*, vol. 4, no. 1, p. 1, 2016.

[9] C. Lai, H. Li, R. Lu, R. Jiang, and X. Shen, "SEGR: A secure and efficient group roaming scheme for machine to machine communications between 3GPP and WiMAX networks," in *Proceedings of the 2014 1st IEEE International Conference on Communications, ICC 2014*, pp. 1011–1016, Australia, June 2014.

[10] H. Zhu, X. Lin, Y. Zhang, and R. Lu, "Duth: A user-friendly dual-factor authentication for Android smartphone devices," *Security and Communication Networks*, vol. 8, no. 7, pp. 1213–1222, 2015.

[11] C. Lai, H. Li, R. Lu, R. Jiang, and X. Shen, "LGTH: a lightweight group authentication protocol for machine-type communication in LTE networks," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM '13)*, pp. 832–837, December 2013.

[12] W. H. Murray, "Modern cryptography," *The Journal of Information System Security*, vol. 3, no. 4, pp. 30–33, 1995.

[13] M. Bellare and P. Rogaway, "The exact security of digital signatures-how to sign with RSA and rabin," *Lecture Notes in Computer Science*, pp. 399–416, 1996.

[14] D. He, M. Tian, and J. Chen, "Insecurity of an efficient certificateless aggregate signature with constant pairing computations," *Information Sciences*, vol. 268, pp. 458–462, 2014.

[15] E. Barker, L. Chen, A. Roginsky, and M. Smid, *Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography*, Gaithersburg, 2013.

[16] P. Chown, "Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)," RFC Editor RFC3268, 2002.

[17] J. Y. Hwang, S. Lee, B.-H. Chung, H. S. Cho, and D. Nyang, "Group signatures with controllable linkability for dynamic membership," *Information Sciences*, vol. 222, pp. 761–778, 2013.

[18] J. Jeong, Y. C. Min, and H. Choo, "Integrated OTP-Based User Authentication Scheme Using Smart Cards in Home Networks," in *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS '08)*, pp. 294-294, USA, January 2008.

[19] Y.-K. Lee, H.-I. Ju, D.-W. Kim, and J.-W. Han, "Home network modelling and home network user authentication mechanism using biometric information," in *Proceedings of the IEEE International Symposium on Consumer Electronics*, pp. 1–5, 2006.

[20] M. A. Ferrag, L. A. Maglaras, H. Janicke, J. Jiang, and L. Shu, "A systematic review of data protection and privacy preservation schemes for smart grid communications," *Sustainable Cities and Society*, vol. 38, pp. 806–835, 2018.

[21] M. A. Ferrag, L. Maglaras, A. Argyriou, D. Kosmanos, and H. Janicke, "Security for 4G and 5G cellular networks: A survey of existing authentication and privacy-preserving schemes," *Journal of Network and Computer Applications*, vol. 101, pp. 55–82, 2018.

[22] M. A. Ferrag, L. Maglaras, A. Derhab, A. V. Vasilakos, S. Rallis, and H. Janicke, "Authentication schemes for Smart Mobile Devices: Threat Models, Countermeasures, and Open Research Issues," pp. 1–22, 2018.

[23] W. Han and Y. Xiao, "Privacy preservation for V2G networks in smart grid: a survey," *Computer Communications*, vol. 91-92, pp. 17–28, 2016.

[24] M. J. Dworkin, "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC," 2007.

[25] J. M. Turner, The keyed-hash message authentication code (hmac), Fed. Inf. Process. Stand. Publ., 2008.

[26] C. Lyu, D. Gu, Y. Zeng, and P. Mohapatra, "PBA: Prediction-Based Authentication for Vehicle-to-Vehicle Communications," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 1, pp. 71–83, 2016.

[27] J. Katz and A. Y. Lindell, "Aggregate message authentication codes," in *Topics in Cryptology–CT-RSA*, vol. 4964, pp. 155–169, Springer, Berlin, Germany, 2008.

[28] D. M'Raihi, M. Bellare, F. Hoornaert, D. Naccache, and O. Ranen, "HOTP: An HMAC-Based One-Time Password Algorithm," RFC Editor RFC4226, 2005.