mathematics

MDPI

Article

A Provable Secure Cybersecurity Mechanism Based on Combination of Lightweight Cryptography and Authentication for Internet of Things

Adel A. Ahmed [1,*], **Sharaf J. Malebary** [1], **Waleed Ali** [1] **and Ahmed A. Alzahrani** [2]

[1] Information Technology Department, Faculty of Computing and Information Technology-Rabigh, King Abdulaziz University, Jeddah 25729, Saudi Arabia

[2] Information Technology Department, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia

* Correspondence: aaaabdullah1@kau.edu.sa; Tel.: +966-563-884-738

Abstract: Internet of Things devices, platform programs, and network applications are all vulnerable to cyberattacks (digital attacks), which can be prevented at different levels by using cybersecurity protocol. In the Internet of Things (IoT), cyberattacks are specifically intended to retrieve or change/destroy sensitive information that may exceed the IoT's advantages. Furthermore, the design of a lightweight cybersecurity mechanism experiences a critical challenge that would perfectly fit resource-constrained IoT devices. For instance, identifying the compromised devices and the users' data and services protection are the general challenges of cybersecurity on an IoT system that should be considered. This paper proposes a secure cybersecurity system based on the integration of cryptography with authentication (ELCA) that utilizes elliptic curve Diffie–Hellman (ECDH) to undertake key distribution while the weak bits problem in the shared secret key is resolved. In this paper, three systems of integration are investigated, while ELCA proposes secure integration between authentication and encryption to facilitate confidentiality and authenticity transfer messages between IoT devices over an insecure communication channel. Furthermore, the security of ELCA is proven mathematically using the random oracle model and IoT adversary model. The findings of the emulation results show the effectiveness of ELCA performance in terms of a reduced CPU execution time by 50%, reduced storage cost by 32–19.6%, and reduced energy consumption by 41% compared to the baseline cryptographic algorithms.

Keywords: IoT; ECDH; symmetric cryptographic; authentication

MSC: 68M25

check for updates

Citation: Ahmed, A.A.; Malebary, S.J.; Ali, W.; Alzahrani, A.A. A Provable Secure Cybersecurity Mechanism Based on Combination of Lightweight Cryptography and Authentication for Internet of Things. *Mathematics* **2023**, *11*, 220. https://doi.org/10.3390/math11010220

Academic Editor: Wei Fang

Received: 30 November 2022
Revised: 23 December 2022
Accepted: 28 December 2022
Published: 1 January 2023

1. Introduction

The Internet of Things (IoT) enables communication between various items and things that have internetworking devices as well as technological devices. An IoT device is configured with a unique IP address to perform various smart applications without human intervention. Moreover, IoT devices are extremely heterogeneous, differ in their capabilities, and have very limited resources in terms of storage capacity and processing complexity, input/output hardware features, and sources of energy [1]. The cybersecurity mechanism remains a significant challenge for IoT implementation and deployment due to the software and hardware vulnerability against cyberattacks. Moreover, cybersecurity has become a transversal discipline to guarantee the confidentiality, authenticity, and integrity of the generated data, transmitted and/or stored on IoT devices. Privacy and security must be ensured by the cybersecurity mechanism to generate trust in data, which is a decisive factor in making critical decisions for the development of all areas involved in this interconnected world. Generally, cyberattacks utilize the internet to gain unauthorized access to disable

IoT devices, and destroy and disrupt the critical information of the IoT [2–6]. Regardless of the network structure layers, the IoT is susceptible to numerous kinds of attacks at the application, network, and sensing layers. The access control mechanism can effectively monitor the access activities of resources by legitimate users [3]. For instance, cyberattacks cause dangerous compromises on the IoT the strengths of which include sensor imprisonment, known key security, stolen-verifier and controlled information, denial of service (DoS), link sniffing, man-in-the-middle, forced delay, session hijacking, brute force, and dictionary attacks [7–10]. Furthermore, key distribution is the predicament of the symmetric cryptography, and it represents the essential challenge task in a resource-constrained system such as the IoT. One of the practical solutions is using ECDH, which is considered an appropriate solution for secret key distribution among IoT devices. This is primarily due to ECDH having a smaller key size with higher security strength compared to an RSA cryptosystem [11]. Furthermore, ECDH requires fewer CPU resources, which causes less power consumption and processing delay compared to RSA.

Figure 1 illustrates the scenario of a cyberattack that can compromise the channel communication between the sensor devices and the IoT gateway or compromise the IoT cloud networks. The standard cryptosystem solutions (e.g., RSA, AES, DES) require the imperative computation overhead, long key size, high memory capacity, and long processing delay. As a result, they cannot be applied immediately to the technology or sensors with the lowest resource requirements, such as the IoT. Therefore, it is a difficult task to build effective, quick, small, and safe cryptographic techniques for the IoT. Additionally, the IoT networks should put in place a minimal cybersecurity system to guard against unauthorized attackers disclosing sensitive information and to confirm that users are permitted to use IoT services (e.g., authentication and access control) [12–19].

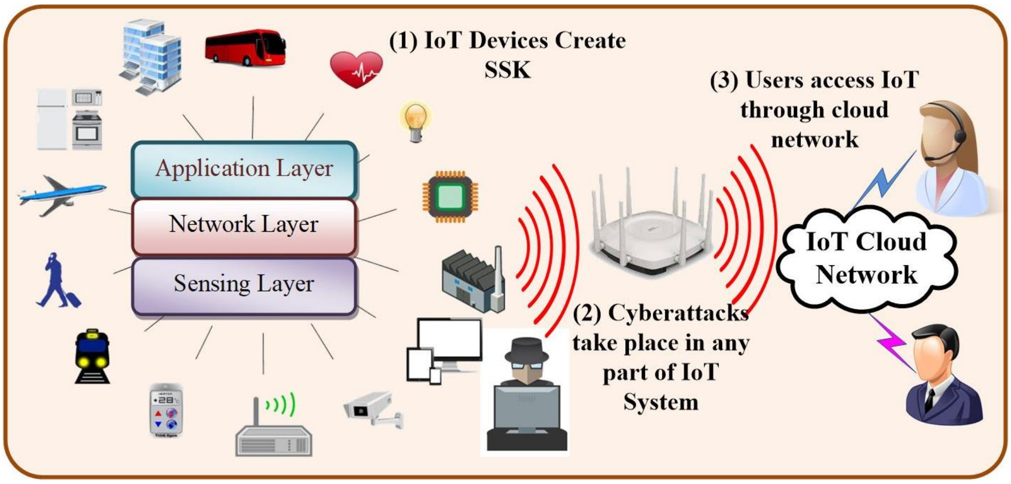

(1) IoT Devices Create SSK
(3) Users access IoT through cloud network
Application Layer
Network Layer
Sensing Layer
IoT Cloud Network
(2) Cyberattacks take place in any part of IoT System

Figure 1. Scenario of cyberattacks on the IoT network.

Cryptography, digital signature, and authentication are the essential solutions to defend against cyberattacks on the IoT. One of the two widely used encryption techniques symmetric (private key) or asymmetric (public key) encryption can be used with IoT cryptography. The same key is used for the cryptographic operation in symmetric encryptions at both the source and the destination. The distribution of the private key among IoT devices determines how strong the symmetric encryption is. As opposed to symmetric encryptions, asymmetric encryptions use two distinct keys: the public key and the private key. The public key can be communicated across a secure channel to the authorized devices, while the private key is kept hidden and never shared.

While encryption can guarantee privacy, message authentication can guarantee authenticity/integrity of the received data. Nevertheless, IoT systems need both authentication and confidentiality. It may be attractive to integrate encryption and authentication; however, not all combinations will provide both privacy and authentication. Certainly, it is a very difficult task to combine cryptographic tools securely, which means that, sometimes, outstanding cryptographic tools can be integrated in a way that produces an insecure combination. Consequently, without proven security of a specific combination, it is risky to use it. The popular methods to merge message authentication and encryption can be described as follows [11]:

- **Method 1**: Encrypt-and-authenticate (EAT), which means the original data should be encrypted using K_1 as $C = E_{k1}(M)$ and the message authentication code should be calculated using K_2 as $T = MAC_{k2}(M)$. The sending message is the pair (C, T), which should be sent separately as shown in Figure 2a.
- **Method 2:** Authenticate-then-encrypt (ATE), which means the tag T is first calculated, and then the original data and T are encrypted together. The sending message is $C = E_{k1}(M{+}T)$ where $T = MAC_{k2}(M)$ as illustrated in Figure 2b.
- **Method 3:** Encrypt-then-authenticate (ETA), which means the original data M is first encrypted using K_1 as $C = E_{k1}(M)$, and then the tag T is calculated over C. The sending message is the pair (C, T) where $T = MAC_{k2}(C)$ as illustrated in Figure 2c.

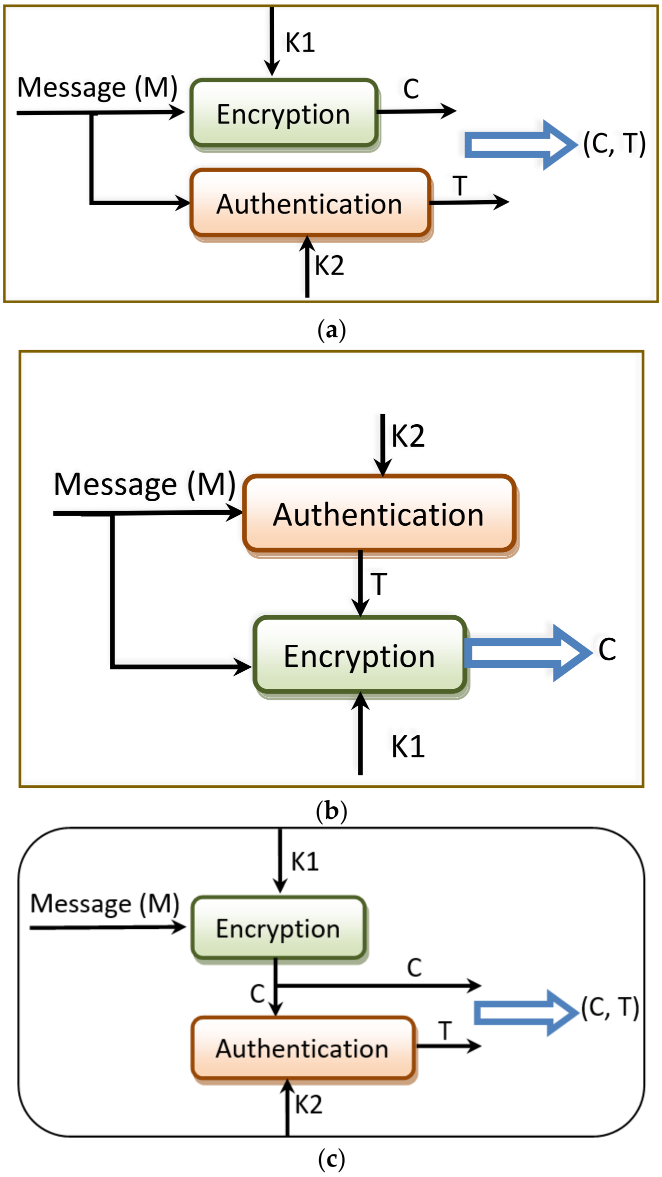


Figure 2. Integration methods between encryption and authentication: (**a**) Encrypt-and-authenticate; (**b**) authenticate-then-encrypt; (**c**) encrypt-then-authenticate.

1.1. Adversary Model on IoT

The main goal of an adversary cyberattack against the IoT is to disrupt its control function by taking advantage of one or more weaknesses that a malicious adversary could use to penetrate the IoT environment's security system [20–22]. The adversary is presumptively capable of reading, transmitting, and faking IoT network traffic, which could raise concerns about sensed data, IoT device privacy, and IoT gateway control management. The most crucial adversary attacks on ELCA are described as follows:

- **Spoofing attack.** To obtain the IoT device credential needed to access the sensed data, the attacker intercepts or eavesdrops on the IoT network traffic.
- **A man-in-the-middle.** In this attack, the malicious adversary has the ability to connect to any IoT device and listen to any network data. Additionally, the adversary can alter the captured messages before they are transmitted to the receiver if it engages in active man-in-the-middle behavior [8].
- **A replay attacks.** A replay attack creates a replica of the message to be used later, as opposed to transmitting it directly to the recipient. An opponent does this by intercepting the data and delaying, replaying, or retransmitting it.
- **A brute force.** Even though the domain parameters that both parties use for ECDH are adequately robust, the malicious adversary in this attack tries every possible combination of letters, digits, and characters to crack the shared secret key.
- **A sensor capture attack.** In this attack, the impostor adversary seizes a sensor node and takes the shared secret key and shared domain parameters in order to carry out unethical activities on the Internet of Things network.
- **A stolen-verifier attack.** If the imposter attacker has obtained the shared secret key from an IoT device, they can pretend to be an authorized device to launch attacks against other IoT devices, steal data, or get around access controls.

1.2. Research Motivation

The motivation of the proposed method is to develop a cybersecurity mechanism that securely combines a lightweight cryptography with authentication to prevent a cyberattack and fit the resource-constrained IoT system. In addition, the proposed solution protects IoT messages from modification, and spoofing attacks.

1.3. Research Contribution

The following contributions are reported in this research:

- It proposes a lightweight symmetric encryption based on the scalar multiplication of the hash function and the base point of the elliptic curve. The modular multiplicative based on order of base point has been used to create the final ciphertext. Additionally, the proposed ELCA confidentially distributes a shared secret key between IoT parties over an insecure communication channel using the ECDH method. Indeed, the secure shared key is an ephemeral that resolves the weak bits problem and is recommended by RFC8442 to provide perfect forward secrecy.
- It proposes an efficacious secure combination between authentication and encryption to facilitate confidentiality and authenticity transfer messages between IoT devices over an insecure communication channel.
- A comprehensive cryptanalysis based on the random oracle model mathematically proves the security of the proposed combination between authentication and encryption on the IoT.
- The well-known IoT adversary model is also exploited to verify the security strength and to prove the security of the proposed scheme.
- Finally, the performance of the suggested ELCA is also evaluated in terms of CPU execution time, power consumption, and storage cost through a number of emulation experiments.

The rest of this paper is organized as follows: the related works on authentication and encryption over an IoT platform is presented in Section 2. The algorithm of the proposed ELCA is explained in Section 3. Additionally, Section 4 describes the cybersecurity analysis for the ELCA mechanisms. The implementation and evaluation of ELCA on the IoT is presented in Section 5. Finally, Section 6 presents the conclusion and future work. All notations used in ELCA are summarized in Table 1.

Table 1. Frequently used notation.

Notation	Meaning	Notation	Meaning
C	Ciphertext	m	Converting M to the integer number
CCA	Chosen-ciphertext attack	MAC	Message authentication code
CPA	Chosen-plaintext attack	n	Order of G
CMA	Chosen-message attack	O	An extra point at infinity of the curve
d	Private key	P	Modular prime
D	Destination node	Pb	Random point in the curve
ECC	Elliptic curve cryptography	Pb.X1	X coordinate of Pb
ECDH	Elliptic curve Diffie–Hellman	PPT	Probabilistic polynomial time
ELCA	Effective, lightweight cryptographic and authentication	PRF	pseudorandom function
EU-CMA	Existentially unforgeable under chosen-message attack	Q	Public key
G	Base point generator	ROM	Random oracle model
h	Subgroup cofactor	S	Source node
IND-CPA	Indistinguishability chosen-plaintext attack	SSK/X_K	Shared secret key
M	Plaintext message	T	Authentication tag

2. Related Works on Cryptographic and Authentication Algorithms

A small number of studies have previously been established to fit resource-constrained devices, particularly for sensors and actuators on IoT networks, despite the fact that many academics have investigated the security algorithms on the IoT. In our earlier work [23], the digital certificate authority was used to link a public key to its owner using a digital certificate, thereby authenticating the sender's genuine identity. Therefore, the related efforts in this research focus on creating simple cryptographic algorithms and lightweight authentication across IoT networks.

Elliptic curve integrated encryption (ECIES), which is combined with advanced standard encryption and is known as ECIES AES, was proposed by V. Shoup. Additionally, ECIES includes rabbit encryption, known as ECIES Ra, in accordance with the specifications in RFC4503. NIST proposed a lightweight authenticated encryption with associated data (AEAD) that can operate with a device that has limited resources, such as an Internet of Things system [24]. The encryption and tag provided by AEAD can be used as a message authentication code (MAC). AEAD provides data authentication, confidentiality, and integrity as a result. To match an IoT resource-constrained system, Byoungjin Seok et al. [25] created secure device-to-device communication using the concepts of AEAD and ECC.

A secure data sharing mechanism for device-to-device communication on the 5G mobile system was presented by Atefeh et al. [26]. The virtual check concept was used in this study as a system of encouragement to encourage manipulators' involvement in the development of data sharing. In the study suggested by Adeel et al. [27], a public key infrastructure (PKI)-based lightweight authentication method was combined with elliptic ElGamal encryption. Additionally, Yasir et al. [28] created a small cryptographic system that relies on ECC and ElGamal over public key infrastructure (EEoP). Additionally, Adel et al. [29] proposed a powerful multifactor authentication (CMA) system that makes use of the concept of combining various hash functions with geolocation authentication over the IoT. In order to verify the key generation, Sciancalepore et al. [30] integrated ECDH exchange with a digital certificate. In order to enhance user authentication, Mohammad Ayoub et al. [31] created a secure ECC-based authentication and encryption system that makes use of user credentials and biometric parameters. Secure IoT (SIT), which makes use of a 64-bit key of Feistel and a consistent substitution–permutation, was proposed by Muhammad U. et al. [32]. Shah et al. [33] presented the integration of Diffie–Hellman-based cryptography and authentication. To share a secret key through the Internet of Things, multifactor authentication is used. One-time passwords (OTPs) that rely on ECC and isogeny to ensure IoT security were proposed by Badis Hammi et al. [34]. The OTP based on ECC's unpredictability is not guaranteed though. A safe system with privacy and authentication based on three factors was proposed by Rangwani, D. et al. [35].

The limitations of the previous literature studies [23–35] are summarized in Table 2. In this table, the main limitations can be specified in four facts: First, the integration between authentication and encryption has not been proven to be secure. Second, the outstanding construction of the IoT and the resource constraints have not been considered. Third, the vulnerabilities of ECDH (i.e., weak bits and chosen-ciphertext attack) have not been resolved and recovered. Finally, the cryptanalysis under a random oracle model has not been investigated.

Table 2. Summary of Related Works.

Approaches	Date Published	Methodology and Features	Limitations
AEAD [24]	2020	It provided the cipher and the tag that offers data confidentiality, integrity, and authentication.	It does not provide secure integration.
B. Seok et al. [25]	2020	In order to accommodate an IoT system with limited resources, it developed a secure device-to-device communication using the concepts of AEAD and ECC.	The cryptanalysis was not studied.
Adeel et al. [27]	2019	In order to manage the public key infrastructure (PKI), it combined the two algorithms ElGamal and ECC.	It lacks the adversary mode analysis.
Yasir et al. [28]	2017	It created a small-scale cryptography system that utilizes ECC and ElGamal.	The cryptanalysis was not studied.
Adel et al. [29]	2019	It proposed a secure multifactor authentication (CMA) that uses robust combiners of the hash functions and geolocation authentication over IoT.	The time processing complexity is high.
KMP [30]	2017	To verify the key generation, ECDH exchange and a digital certificate were included.	Due to the implicit certificate's power consumption, it does not fit IoT resource constraints.

Table 2. *Cont.*

Approaches	Date Published	Methodology and Features	Limitations
M. Ayoub et al. [31]	2020	It created a secure ECC-based authentication and encryption system that strengthens user authentication by using personal information and biometrics.	Due to the vulnerability of biometric parameter mistake, it does not fit IoT resource constraints.
SIT [32]	2017	It used the idea of combination 64-bit key of Feistel and a uniform substitution–permutation.	Due to power consumption, it does not fit the IoT resource limitations.
Shah et al. [33]	2017	To share a secret key via an IoT network, it integrated authentication and cryptography based on Diffie–Hellman.	It does not prove the security for integration.
B. Hammi et al. [34]	2020	It proposed OTP that relies on ECC and isogeny to guarantee IoT security.	The randomness of the OTP based on ECC is not ensured.
Rangwani, D [35]	2021	It suggested a safe, private, and three-factor authentication mechanism for the Internet of Things.	It does not study the effect of three-factor authentication on the operating system.

3. System Design of ELCA Algorithm

The system design of the proposed ELCA algorithm mainly consists of key management based on ECDH, symmetric encryption algorithm with a random padding system, and message authentication based on multifactor hash function. This research proposes secure integration between symmetric cryptography and authentication based on method 3 (e.g., encrypt-then-authenticate). The three algorithms are organized to guarantee cyberattack protections on the IoT. The three proposed functions in this study were created under the following presumptions:

- The IoT gateway has a robust security mechanism and hence cannot be compromised.
- The shared secret key (SSK) is calculated based on ECDH and it is considered as the private key of the ELCA cryptography.
- SSK in all IoT devices uses the preinstalled two secure keys: the public key, which is calculated at all involved IoT devices, and the private key, which is not known publicly.
- All keys in the proposed system are ephemeral (dynamic), which means they must be changed in each new session.
- The domain parameters of the ECDH are inserted and programmed into all IoT devices during the initialization session.
- The detail of ELCA is explained in the following sections.

3.1. Key Management Algorithm Based on ECDH

The exchange of the common secret key between the IoT devices is the essential concern in traditional symmetric cryptography. This is primarily due to the insecure communication channel that makes IoT devices susceptible to many cyberattacks. Consequently, the proposed encryption mechanism utilizes the ECDH to securely calculate rather than distribute a new SSK for each transmission session between IoT devices (i.e., forward secrecy). The elliptic curve is a set of points identified by solving the following equation:

$$\begin{aligned} &E = \{(x,y)|y^2 = x^3 + ax + b\} \cup \{O\}, \\ &\quad \textit{where } a, b \in K(\mathbb{Z}/P\mathbb{Z}) \textit{ satisfy } (4a^3 + 27b^2) \neq 0 \end{aligned} \tag{1}$$

where K presents an integer finite field over a modular prime P. An extra point at infinity (e.g., O) has been added to the equation to add any point to itself. Let us assume that S and D are the IoT source and the IoT destination, respectively. The domain parameters of elliptic curve consist of p, G, n, h which are the prime number, the base point generator, the order of G, and the subgroup cofactor that is usually 1. These parameters demonstrate the agreed information between S and D to utilize the ECDH key exchange protocol. In each new session, the private key at S and D is generated using the random function, which is selected between 1 and n-1. The public key is a point in the curve, namely Q, which is produced using scalar multiplication of d and G (e.g., $Q = d \times G$) as shown in Figure 3. In this figure, S has a key pair (d_S, Q_S) and D (d_D, Q_D), which represent the private and public keys at each node. Each S and D should receive the public key from the other party prior to implementing the ECDH protocol. Later, S computes its SSK point as $K(X_K, Y_K) = d_S \times Q_D$ and D computes its SSK point as $K(X_K, Y_K) = d_D \times Q_S$. As a result, the agreed SSK is the x coordinate of the point K, which is $k_1 = \mathrm{X}_K$. Moreover, $k_2 = \mathrm{Y}_K$ represents the agreed SSK for authentication. It is interesting to note that the SSK that is calculated by both parties is equal because $d_S \times Q_D = d_S \times d_D \times G = d_D \times d_S \times G = d_D \times Q_S$, where " × " denotes elliptic curve scalar multiplication.

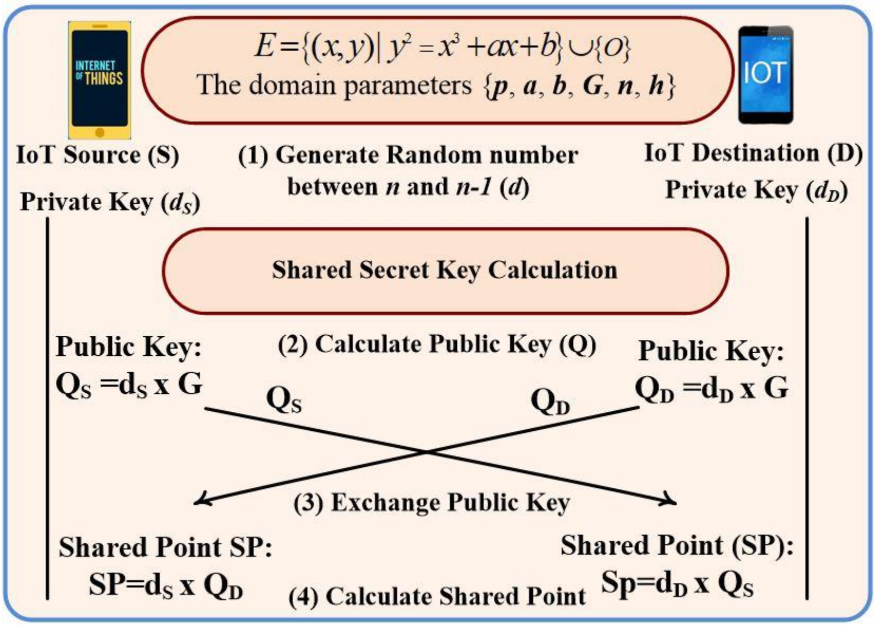

$E=\{(x,y)|\, y^2 = x^3 + ax + b\} \cup \{O\}$
The domain parameters $\{p, a, b, G, n, h\}$
INTERNET OF THINGS
IOT
IoT Source (S)
Private Key (d_S)
(1) Generate Random number between n and n-1 (d)
IoT Destination (D)
Private Key (d_D)
Shared Secret Key Calculation
Public Key: $Q_S = d_S$ x G
(2) Calculate Public Key (Q)
Public Key: $Q_D = d_D$ x G
Q_S
Q_D
(3) Exchange Public Key
Shared Point SP: SP=d_S x Q_D
(4) Calculate Shared Point
Shared Point (SP): Sp=d_D x Q_S

Figure 3. ECDH key management.

3.2. Secure Integration between Encryption and Authentication

The combination between encryption and authentication should be carefully designed because it is very hard to combine cryptographic tools correctly to provide both privacy and authenticity. This means that excellent cryptographic tools can sometimes be applied in a way so that the result is not secure. This research proposes secure integration between symmetric cryptography and authentication based on the encrypt-then-authenticate method called ELCA. In order to fit the maximum transmission unit in the IoT network, the message M is parsed into several chunks based on Secp192r1 elliptic curve domain parameters [36]. Hence, the maximum size of each chunk is 127 bytes, and the minimum size is 24 bytes. The cryptographic steps of ELCA at the source node are implemented as follows:

- Calculate E = StrToInt(Hash(X_K)); the Hash is a secure cryptographic hash function such as CMA [29] or SHA-256 [37].
- Calculate the curve point $Pb(X_1,Y_1) = E \times G$; the ECC scalar multiplication has a one-way function property, which means it is hard to reverse.
- Calculate the ciphertext $C_i = (m_i \times \mathrm{X}_1) \bmod n$; where i represents the chunk number. The padding scheme is used to convert the chunk (M_i) to the integer number m_i, which should be agreed upon in reversible protocol.

- Calculate a hash function for C_i as Z = StrToInt(Hash(C)) mod n.
- Calculate the authentication code as $T_s = (Y_K \times Z)$ mod n;
- The transmitted message is the pair (C_i,T_s).

The cryptographic steps of ELCA at the destination node upon receiving the pair (C_i,T_s) are performed as follows:

- Calculate a hash function of the integer number m as Z = StrToInt(Hash(C_i)) mod n where Hash() represents the similar cryptographic hash function that is used in the encryption process.
- Calculate $T_d = (Y_K \times Z)$ mod n.
- If $T_d = T_s$, the message is accepted (e.g., message is authentic, and integrity checked). Otherwise, the message is rejected.
- If the message is accepted, calculate E = StrToInt(Hash(X_K)).
- Calculate the curve point $Pb(X_1,Y_1) = E \times G$.
- Calculate $m_i = (C_i \times X_1{}^{-1})$ mod n where $X_1{}^{-1}$ mod n can be resolved using a modular multiplicative inverse.
- Convert the m_i to string M_i and recover the plaintext M = where L is the number of chunks.

Figure 4 shows the flow phases and Algorithm 1 presents the pseudo code of the ELCA algorithm. In these figures, the source node and the destination must use the same domain parameters of the ECDH equation. Upon the public key being calculated at the two parties, it is sent to the other party, which can calculate the shared secret key. Finally, the combination of encrypt-then-authenticate in ELCA is utilized as explained above.

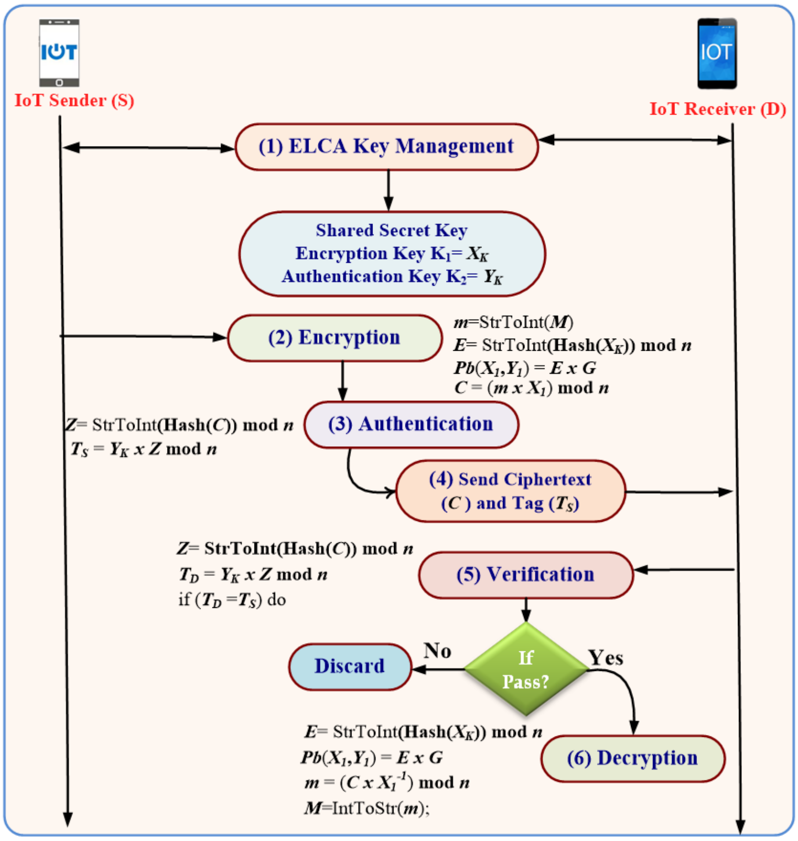


Figure 4. Flow diagram of ELCA algorithm.

Algorithm 1 Pseudo code of ELCA algorithm

```
                         ELCA at IoT Sender (S)
Input: Secp192r1 domain parameters $p, a, b, G, n, h;$
Output: $Q_S, T, C;$ // $Q_S$: Public key of S, $T$: authentication tag $C$: Ciphertext
Start Algorithm (ELCA)
| While (new session start) do
|     Determine the private key ($d_S$); // $1 \leq d_S \leq n$
|       $Q_S$ = ($d_S \times G$); // $Q_S$: the public key of S
|       Send_Public_key ($Q_S$); // Send the public key to destination
|         Receive_Public_key($Q_D$); // Receive the public key of D
|     K($X_K$,$Y_K$) = $d_S \times Q_D$; // calculate the shared key
|     For ($i$ = 0; $i$<$L$; $i$++) // L: number of chunks
|         $m_i$ = StrToInt($M_i$); // convert the plaintext to an integer.
|       $E$ = StrToInt(Hash($X_K$)) mod $n$; // $E$: the hash fun. of key $X_K$
|       $Pb(X_1,Y_1) = E \times G$;
|       $C_i$ = ($m_i \times X_1$) mod $n$; // $C_i$: the ciphertext of message $m_i$
|       $Z$ = StrToInt(Hash($C_i$)) mod $n$; // hash fun. for integer $m$.
|       $T_S = Y_K \times Z$ mod $n$; // $T_S$: Authentication code at the sender
|       Send("$C_i$"+" $T_S$"); // The source sends "$C_i$"+" $T_S$" to D
|     End; // For Loop *Statement*
|   End; // While loop
End; // Algorithm
                         ELCA at IoT Receiver (D)
Input: the domain parameters $p, a, b, G, n, h;$
Output: $Q_D, T_S, C;$ // $Q_D$: Public key of D
Start Algorithm (ELCA)
| While (new session start) do
|   Determine the private key ($d_D$); // $1 \leq d_D \leq n$
|   $Q_D$ = ($d_D \times G$); // $Q_D$: the public key of D
|   Send_Public_key ($Q_D$); // Send the public key to source node
|   Receive_Public_key($Q_S$); // Receive the public key from source
|   $K(X_K,Y_K) = d_D \times Q_S$; // *if* $Q_S$ is a valid curve point, the shared key will be
calculated
|   Foreach (msg received; i++) do
|       Get($T_S$, $C_i$); // Receive the message pair ($T_S$, $C_i$)
|       $Z$ = StrToInt(Hash($C_i$)) mod $n$; // hash fun. for $C$
|       $T_D = Y_K \times Z$ mod $n$; // $T_D$: Authentication code at the destination
|       If $T_d = T_s$, the message is accepted. Otherwise, the message is rejected.
|       $E$ = StrToInt(Hash($X_K$)) mod $n$;
|       $Pb(X_1,Y_1) = E \times G$;
|       $m_i$ = ($C_i \times X_1{}^{-1}$) mod $n$; // Recover the padded message
|     For ($i$ = 0; $i$<$L$; $i$++) // $L$: number of chunks
|         $M_i$ = Convert_IntToStr($m_i$); // convert integer to plaintext.
|         $M = M + M_i$ // concertante all chunks.
|     End; // *for loop*
|  End; // While loop
End; // Algorithm
```

4. Cybersecurity Analysis

In order to measure the security level of ELCA, the cryptanalysis for ELCA on the IoT was developed and analyzed.

4.1. Cryptanalysis of ELCA

Let us imagine that, even if the shared secret key is unknown, the adversary may decrypt encrypted messages and bypass the authentication and encryption of the ELCA mechanism. The following are some examples of the most typical cryptanalysis attacks that have been studied using the random oracle model:

- **Chosen-plaintext attack (CPA).** It is expected that the adversary will obtain the ciphertexts for any plaintexts of its choosing. Additionally, the adaptive CPA (CPA2) allows the adversary to select a fresh input for ELCA (*ELCA$_E$*) encryption based on an analysis of the plaintext queries he previously selected and the accompanying ciphertexts [38]. By assuming that an advertiser A has access to an encryption oracle with any pair of equal-length messages (m1, m2) as input, we can describe the definition of CPA mathematically [20–22].

Definition 1. *Let ELCA$_E$ = (K, E, D) be an encryption mechanism in ELCA, E is encryption, D is decryption, and K is the space of all keys. The advantage of indistinguishability of chosen-plaintext attack (IND-CPA) of **A** is defined as:*

$$\begin{aligned} Adv^{in-cPa}_{ELCA_E}(A) &= P_r[k \leftarrow K; C \leftarrow E_k(m_1) : A(C) = 1] \\ &- P_r[k \leftarrow K; C \leftarrow E_k(m_2) : A(C) = 1] \end{aligned} \tag{2}$$

- If the advantage of IND-CPA is negligible, which indicates that A is struggling, the aforementioned equation demonstrates that ELCA is secure. Contrarily, ELCAE is not stable if the IND-advantage of CPA is non-negligible, indicating that A is performing well.
- **Chosen-ciphertext attack (CCA).** It is expected that the adversary will obtain the decryption of any ciphertext(s) of its choosing. A further benefit of the adaptive CCA (CCA2) is that the adversary can select a fresh input for the decryption of ELCA (*ELCA$_D$*) based on the analysis of his previously chosen queries [39].

Definition 2. *Let ELCA$_E$ = (K, E, D) be an encryption mechanism in ELCA, and **A** is an adversary who can access the encryption (E) and decryption (D) oracle. The advantage of IND-CCA of **A** is defined as:*

$$\begin{aligned} Adv^{in-cca}_{ELCA_E}(A) = P_r[k \leftarrow K; C \leftarrow E_k(m_b); b \leftarrow \{0,1\}; \\ b' \leftarrow A(E_k(.), D_k(.)) : b' = b] \end{aligned} \tag{3}$$

According to the aforementioned definition, the adversary is free to access the decryption oracle at any time and with any ciphertext C, with the exception of the previously answered queries from its encryption oracle. Therefore, if the adversary who was provided access to the oracles may find little benefit in differentiating the two occurrences of b (0/1), then ELCAE can be regarded secure against IND-CCA.

4.1.1. Cryptanalysis of Combination between Cryptographic Tools

The combination cryptanalysis will use an all or nothing approach to validate both message confidentiality and authentication for every possible combination between them. This does not mean that the combination is not always secure for every encryption and authentication; however, it means there exists even one case where the combination is not secure. The security level that should be considered in the analysis is IND-CPA for encryption and existentially unforgeable under chosen-message attack (EU-CMA) for authentication. The two attacks (e.g., IND-CPA and EU-CMA) meet the requirement for gaining chosen-ciphertext security together with existential unforgeability. Generally, the proposed cryptanalysis approach to prove the security for the combination is to prove that a given combination meets the definition of the secure communication channel [11]. Let tuple of algorithms ($\overline{K}$, $\overline{ET}$, $\overline{D}$, $\overline{V}$) be a combination of (K, E, D) and (K, T, V), where $\overline{K}$ represents the ECDH key-generation algorithm and produces shared secret keys (k_1 = X_K, k_2 = Y_K). The combination algorithm in ELCA is represented by $\overline{ET}$, which receives a pair of keys (k_1, k_2) and a message m as input and outputs C and authentication tag T. Furthermore, $\overline{V}$ represents the verification procedure in ELCA, which applies a combination of $E(X_K)$ and

$T(Y_K)$ upon receiving a pair of keys (k_1, k_2) and a value C and/or T. Latterly, $\overline{V}$ outputs 1 or 0. The $\overline{D}$ represents the decryption algorithm in ELCA, which applies a combination of $E(X_K)$ and $T(Y_K)$ upon receiving a pair of keys (k_1, k_2) and a value C. Finally, $\overline{D}$ recovers the original message m.

The satisfactory requirement is that for every $k_1 = X_K$, $k_2 = Y_K$, and for every value m, $\overline{D}_{k1,k2}$ ($\overline{ET}_{k1,k2}$ (m)) = m and $\overline{V}_{k1,k2}$ ($\overline{ET}_{k1,k2}$ (m)) = 1. The combination ($\overline{K}$, $\overline{ET}$, $\overline{D}$, $\overline{V}$) is required to satisfy both a CCA-security and authentication security for $\overline{ET}_{k1,k2}$ as defined in the following:

Definition 3. *ELCA = ($\overline{K}$, $\overline{ET}$, $\overline{D}$, $\overline{V}$) is considered as a secure combination of encryption and authentication if (K, E, D) has IND-CPA and the scheme (K, T, V) is EU-CMA.*

Next we analyze the three combination approaches that are illustrated in Figure 2.

- **Encrypt-and-authenticate** (EAT). This combination can reveal the original message m for any encryption mechanism. For instance, if *(K, T, V)* provides a secure message authentication code and $\overline{T}_k(m)$ = (m, $T_k(m)$), it does not necessarily imply privacy. Hence, the combination ($E_{k1}(m)$, $\overline{T}_{k2}(m)$) completely reveals m and is therefore not *IND-CPA*. As a result, the EAT does not yield a secure combination of encryption and message authentication.
- **Authenticate-then-encrypt** (ATE). Let us discuss the contrived encryption example that suffices to show that the ATE method is not always secure.
  - ➢ Let us assume that there exists an encryption ($E_k(m)$) mechanism that works as follows: any 0 in m is changed to 00, and any 1 in m is changed randomly to 01 or 10. The decryption of C ($D_k(C)$) in this scheme works as follows: change 00 back to 0, and 01 and 10 back to 1. Nevertheless, a pair of bits 11 will result in $\perp$.
  - ➢ Define $\overline{E}_k(m)$ = $PRF \oplus E_k(m)$ and PRF is a pseudorandom function that creates a new number for each message to encrypt.
  - ➢ Let us study the cryptanalysis of the ATE combination based on $\overline{E}_k(m)$ with any message authentication in the presence of a CCA attack. Let A be an adversary who implements the CCA attack as follows. Given a challenge $C = \overline{E}_{k1}$ ((m, $T_{k2}(m)$), A basically complements the first two bits of C and verifies if the resulting ciphertext is valid. If the new C is valid, then A decides that the first bit of m was 1. This is primarily due to the fact that if the first bit of m equals 1, then the first two bits of $\overline{E}_{k1}$ (m) can be 01 or 10. Therefore, the complement of these two bits still yields the same bit 1. However, if the new C is not valid, then A decides that the first bit of m equals 0. This is mainly due to the fact that 0 is mapped to 00 and so flipping these bits yields 11, which means an incorrect C. Accordingly, m is null ($\perp$), which contradicts with the assumption that T_{k2} is still computed over m.

4.1.2. Proven Security of ETA Combination in ELCA Using ROM

The ETA combination in the proposed ELCA is proven secure based on the following security analysis.

Theorem 1. *Let $ELCA_E$ = (K, E, D) be the encryption of ELCA that is secure under IND-CPA, and let $ELCA_M$ = (K, T, V) be the authentication of ELCA that is EU-CMA. Then, ELCA = ($\overline{K}$, $\overline{ET}$, $\overline{D}$, $\overline{V}$) created by the encrypt-then-authenticate is a secure combination of $ELCA_E$ and $ELCA_M$.*

Methodology of Proof. The contradiction methodology is used to prove Theorem 1. Since $ELCA_M$ is *EU-CMA*, all queries (except that obtained from encryption oracle) to the decryption oracle can be assumed to be invalid. Thus, the cryptanalysis of *ELCA* can be reduced to *IND-CPA* of $ELCA_E$ because the decryption oracle is effectually useless. At the beginning, this paper proves that, except with negligible probability, the only valid queries made by A were C that were previously obtained from the encryption oracle. Therefore, if

ELCA is proven as not secure under CCA, then it should be that $ELCA_E$ is not secure under *IND-CPA*, which contradicts the assumption in Theorem 1.

Proof. Let A be any PPT adversary that implements CCA attack on *ELCA*, which can be denoted as $PrivK^{CCA}_{A,ELCA}$ (n). Additionally, let us define $VQuery_{A,ELCA}$ (n) to be the event that A inputs a valid query (C,T) to its decryption oracle, which does not reply $\perp$. Generally, if we prove that the $P_r[VQuery_{A,ELCA}$ (n)] is at most negligible, then that will be sufficient to prove Theorem 1. This is because if the decryption oracle does not reply $\perp$, then T is a valid tag for C. Consequently, if (C,T) is a valid input for the decryption oracle, this means that A essential forged a message authentication. If the probability that $VQuery$ occurs is non-negligible, A_{mac} can be constructed to break the message $ELCA_M$ as follows: Let us define $q(\cdot)$ to be a polynomial that represents the upper bounds of queries that are issued from A. The $\text{Mac} - \text{forge}_{A_{mac},ELCA_M}$(n) is interacted by A_{mac}, which calls the A with chosen random k_i for encryption where $i \leftarrow${1, ... q(n)}. Moreover, A_{mac} uses k_1 and its MAC oracle to simulate the encryption and decryption oracle for A. Let us assume that all queries to the decryption oracle are invalid except the i^{th} query, which is hoped to be valid. This means if A queries the encryption oracle with M, A_{mac} computes $C = E_{k1}(M)$ and calls its MAC oracle to obtain a hope forged T for C. Finally, A_{mac} returns the pair (C,T) to A as its oracle reply. On the other hand, if A sends any decryption oracle query (C,T) except i^{th}, A_{mac} will review if (C,T) has been created before, then A_{mac} returns M. Otherwise, A_{mac} returns $\perp$. However, A_{mac} returns (C,T) as its message authentication forgery and halts upon receiving i^{th} decryption oracle query from A. We remark that since $ELCA_M$ provides a unique tag, this means that the query C was never requested by A_{mac} to its MAC-tag oracle. This is primarily due to (C,T) not being gained from an encryption query, which means there is only a single likelihood that T is a valid tag for C. The probability that the i^{th} query is the first valid query by A is at least $1/q$(n) since A makes at most q(n). Consequently, the probability that A_{mac} does well in $\text{Mac} - \text{forge}_{A_{mac},ELCA_M}$(n) is at least $1/q$(n) times the probability that the $VQuery$ event occurs. Subsequently, the probability of A_{mac} to do well in $\text{Mac} - \text{forge}_{A_{mac},ELCA_M}$(n) is at most negligible probability; this means $VQuery$ occurs with at most negligible probability, which proves the first part of Theorem 1. As a result, for some negligible function *negl(n)*, the probability of $VQuery$ can be written as:

$$P_r[VQuery_{A,ELCA}(n)] < negl(n)$$

Given that the probability of $VQuery$ happens at most negligible probability, the combination of encrypt-then-authenticate in *ELCA* will be proven to be CCA-secure. For simplicity, if we prove the security of $ELCA_E$ against *IND-CPA* attack, then ELCA is proven secure. Let an adversary A_{enc} be created using A for the CPA experiment with $ELCA_E$. A_{enc} selects a key k_2 and calls A. Each time A requests an encryption query for M, A_{enc} calls its encryption oracle with M and receives back C. After that, A_{enc} calculates $T = T_{k2}$(C) and returns the pair (C,T) to A. In contrast, when A requests a decryption query for the pair (C,T), A_{enc} will search about the pair (C,T) in its history table, which was previously generated from its encryption query, and returns M to A if it is available. Otherwise, A_{enc} returns $\perp$. It is clear to conclude that if A_{enc} succeeds in $PrivK^{CPA}$ when $VQuery$ does not happen, then this equals the success of A in $PrivK^{CCA}$ when $VQuery$ does not happen, which can be defined as follows [11]:

$$\begin{aligned} &\boldsymbol{P}_r[PrivK^{CPA}_{A_{enc},ELCA_E}(n) = 1 \cap \neg VQuery^{CPA}_{A,ELCA}(n)] \\ &\quad = \boldsymbol{P}_r[PrivK^{CCA}_{A,ELCA}(n) = 1 \cap \neg VQuery^{CPA}_{A,ELCA}(n)] \end{aligned} \tag{4}$$

Implying that:

$$\begin{aligned} &\boldsymbol{P}_r[PrivK^{CPA}_{A_{enc},ELCA_E}(n) = 1] \\ &\qquad \geq \boldsymbol{P}_r[PrivK^{CPA}_{A_{enc},ELCA_E}(n) = 1 \cap \neg VQuery_{A,ELCA}(n)] \\ &\qquad = \boldsymbol{P}_r[PrivK^{CCA}_{A,ELCA}(n) = 1 \cap \neg VQuery_{A,ELCA}(n)] \end{aligned} \tag{5}$$

Let us use the contradiction by assuming a non-negligible function ε exists such that:

$$\boldsymbol{P}_r[PrivK^{CCA}_{A,ELCA}(n) = 1] = \frac{1}{2} + \varepsilon(n) \tag{6}$$

Using the fact that $P_r[VQuery_{\boldsymbol{A,ELCA}}\ (n)]$ is negligible, this means it is smaller than $\varepsilon(n)/2$. As a result, we can conclude the following:

$$\boldsymbol{P}_r[PrivK^{CCA}_{A,ELCA}(n) = 1 \cap VQuery_{A,ELCA}(n)] < \frac{\varepsilon(n)}{2} \tag{7}$$

This means:

$$\begin{array}{l} \boldsymbol{P}_r[PrivK^{CCA}_{A,ELCA}(n) = 1] = \\ \left(\begin{array}{l} \boldsymbol{P}_r[PrivK^{CCA}_{A,ELCA}(n) = 1 \cap VQuery_{A,ELCA}(n)] \\ \qquad + \boldsymbol{P}_r[PrivK^{CCA}_{A,ELCA}(n) = 1 \cap \neg VQuery_{A,ELCA}(n)] \end{array} \right) \\ < \left(\boldsymbol{P}_r[PrivK^{CCA}_{A,ELCA}(n) = 1 \cap \neg VQuery_{A,ELCA}(n)] + \frac{\varepsilon(n)}{2} \right) \end{array} \tag{8}$$

By means that $\boldsymbol{A}$ succeeds in $PrivK^{CCA}$ with probability 1/2 + ε(n), then Equation (8) can be expressed as:

$$\begin{array}{r} \boldsymbol{P}_r[PrivK^{CCA}_{A,ELCA}(n) = 1 \cap \neg VQuery_{A,ELCA}(n)] > \\ \boldsymbol{P}_r[PrivK^{CCA}_{A,ELCA}(n) = 1] - \frac{\varepsilon(n)}{2} \\ = \frac{1}{2} + \varepsilon(n) - \frac{\varepsilon(n)}{2} = \frac{1}{2} + \frac{\varepsilon(n)}{2} \end{array} \tag{9}$$

Equations (5) and (9) can be combined as:

$$\boldsymbol{P}_r[PrivK^{CPA}_{A_{enc},ELCA_E}(n) = 1] > \frac{1}{2} + \frac{\varepsilon(n)}{2} \tag{10}$$

Equation (10) shows that the advantage of A_{enc} to succeed in $PrivK^{CPA}$ is non-negligible over 1/2. As a result, this contradicts *IND-CPA* of $ELCA_E$ and we conclude that the combination of encrypt-then-authenticate in *ELCA* is *CCA*-secure. □

4.2. ELCA Cybersecurity Analysis

ELCA contains important security features such as impersonation resilience against key compromise and perfect forward secrecy (PFS). ELCA employs a hash function to produce a pseudorandom function (PRF) since it may be thought of as a random oracle function. As stated in Section 3, the ELCA's (i.e., CMA's) hash function uses the shared secret key (X_K) as an input to create the secure random parameter (H(X_K)), which is then multiplied by the base point (G) in a scalar manner to obtain the random point *Pb()*. To protect against IND-CPA and replay attacks, *Pb.X1* (i.e., the x coordinate of *Pb*) is a random value that is periodically modified.

Proven Security of ELCA in ROM

The length of the shared secret key $X_K \in \{0,1\}^L$ can be represented as $L = |X_K| = |n| = |p|$, which is equals the length of the used elliptic curve Secp192r1 (e.g., 192 bits). The hash function is instantiated in ROM using the established security in ELCA as $H(.) :\ \{0,1\}^* \rightarrow \{0,1\}^L$.

Theorem 2. *If* ***Pb*** *is a* ***(t,ϵ)****-pseudorandom function (PRF), then the ELCA cryptographic is secure against IND-CPA.*

Methodology of Proof. The second theorem is proven using the contradiction methodology. Let us assume that *A* runs in *PPT* exist and that they compromise $ELCA_E$'s security. With non-negligible cost, algorithm *A* creates a *PPT* distinguisher *B* that separates the output of *Pb* from a random number. Since *Pb* is a *PRF*, the prior conclusion that *Pb* is a random function is incorrect. As a result, the initial hypothesis is incorrect, and the $ELCA_E$ needs to be secure.

Proof . Let us assume *A* attacks $ELCA_E$ in the sense of IND-CPA and two messages M_0, M_1 are used as follows:

$$\left| \begin{array}{c} \boldsymbol{P}_r[H(X_K) \leftarrow \mathbb{Z}_n^*; Pb \leftarrow H(X_K) \times G; C \leftarrow M_0 \times Pb.X1 : \boldsymbol{A}(C) = 0] \\ -\boldsymbol{P}_r[H(X_K) \leftarrow \mathbb{Z}_n^*; Pb \leftarrow H(X_K) \times G; C \leftarrow M_1 \times Pb.X1 : \boldsymbol{A}(C) = 0] \end{array} \right| = \gamma(L) \tag{11}$$

where *γ(L)* is non-negligible. The algorithm *B* was constructed to distinguish *Pb* from the random function. This can be accomplished by determining if *Pb* is a *PRF* or a totally random function utilizing *B*'s ability to call *Pb*. *B* functions as follows: (1) Pick a random *b* between 0 and 1, (2) B computes *C* = *Pb.X1* × M_b *mod n*, (3) Run the experiment *A(C)* to obtain *A*'s guess as to the encrypted message. *A* correctly predicted if *b*=, $\overline{b}$ then *B* estimates the *PRF* and the result is "1" as indicated by *B*. However, *A* guessed incorrectly if $b \neq \overline{b}$ if *B* guesses random function and this can be represented by *B* resulting as "0". The algorithm *B* distinguishes the output of *Pb.X1* as:

$$\left| \begin{array}{r} \boldsymbol{P}_r[H(X_K) \leftarrow \mathbb{Z}_n^*; Pb \leftarrow (H(X_K) \times G); y \leftarrow Pb.X1 : \boldsymbol{B}(y) = 1] \\ -\boldsymbol{P}_r[y \leftarrow \mathbb{Z}_n^* : \boldsymbol{B}(y) = 1] \end{array} \right| \tag{12}$$

We will study each of these terms separately as: $P_1 \overset{\text{def}}{=} P_r[H(X_K) \leftarrow \mathbb{Z}_n^*; Pb \leftarrow (H(X_K) \times G); y \leftarrow Pb.X1 : B(y) = 1]$, and $P_2 \overset{\text{def}}{=} P_r[y \leftarrow \mathbb{Z}_n^* : B(y) = 1]$. In step 3, the algorithm *B* obtained the following:

$$\begin{array}{r} \boldsymbol{P}_1 = \boldsymbol{P}_r[H(X_K) \leftarrow \mathbb{Z}_n^*; Pb \leftarrow (H(X_K) \times G); y \leftarrow Pb.X1 : \\ b \in \{0,1\}; b' \leftarrow \boldsymbol{A}(Pb.X1 \times M_b) : b' = b] \end{array} \tag{13}$$

By using the condition on *b* gives:

$$\begin{array}{c} \boldsymbol{P_1} = P_r[H(X_K) \leftarrow \mathbb{Z}_n^*; y \leftarrow Pb.X1 : \boldsymbol{A}(Pb.X1 \times M_0) = 0] \times P_r[b = 0] \\ +P_r[H(X_K) \leftarrow \mathbb{Z}_n^*; y \leftarrow Pb.X1 : \boldsymbol{A}(Pb.X1 \times M_1) = 0] \times P_r[b = 1] \end{array} \tag{14}$$

With applying the fact:

$$P_r[b = 0] = P_r[b = 1] = \frac{1}{2}$$

and

$$\begin{array}{c} P_r[H(X_K) \leftarrow \mathbb{Z}_n^*; y \leftarrow Pb.X1 : \boldsymbol{A}(Pb.X1 \times M_1) = 1] = \\ 1 - P_r[H(X_K) \leftarrow \mathbb{Z}_n^*; y \leftarrow Pb.X1 : \boldsymbol{A}(Pb.X1 \times M_1) = 0] \end{array} \tag{15}$$

gives:

$$\boldsymbol{P_1} = \frac{1}{2} + \left[\frac{1}{2} \times \left(\begin{array}{c} P_r[H(X_K) \leftarrow \mathbb{Z}_n^*; y \leftarrow Pb.X1 : \boldsymbol{A}(Pb.X1 \times M_0) = 0] \\ -P_r[H(X_K) \leftarrow \mathbb{Z}_n^*; y \leftarrow Pb.X1 : \boldsymbol{A}(Pb.X1 \times M_1) = 0] \end{array} \right) \right] = \frac{1}{2} + \left(\frac{1}{2} \times \gamma(L) \right) \tag{16}$$

P_2 is calculated as:

$$\boldsymbol{P}_2 = P_r[y \leftarrow \mathbb{Z}_n^* : b \in \{0,1\}; b' \leftarrow A(Pb.X1 \times M_b) : b' = b] \tag{17}$$

As before, we eventually obtain:

$$\boldsymbol{P}_2 = \frac{1}{2} + \left[\frac{1}{2} \times \begin{pmatrix} P_r[y \leftarrow \mathbb{Z}_n^* : \boldsymbol{A}(Pb.X1 \times M_0) = 0] \\ -P_r[y \leftarrow \mathbb{Z}_n^* : \boldsymbol{A}(Pb.X1 \times M_1) = 0] \end{pmatrix}\right] \tag{18}$$

Since y is completely random and $Pb = H(X_K) \times G$, the probability of $\boldsymbol{A}$ wins when breaking the one-time pad is 0. Therefore, $\boldsymbol{P}_2$ is $1/2$. The final result after using all parameters together gives:

$$\begin{vmatrix} \boldsymbol{P}_r[H(X_K) \leftarrow \mathbb{Z}_n^*; Pb \leftarrow (H(X_K) \times G); \\ y \leftarrow Pb.X1 : \boldsymbol{B}(y) = 1] - \boldsymbol{P}_r[y \leftarrow \mathbb{Z}_n^* : \boldsymbol{B}(y) = 1] \end{vmatrix} = |\boldsymbol{P}_1 - \boldsymbol{P}_2| \\ = \left|\frac{1}{2} + \frac{\gamma(L)}{2} - \frac{1}{2}\right| = \frac{\gamma(L)}{2} \tag{19}$$

Since the term $\gamma(L)$ was non-negligible, the term $\frac{\gamma(L)}{2}$ is also non-negligible. As a result, A has a non-zero advantage in breaking $ELCA_E$ and hence $\boldsymbol{B}$ has a non-negligible advantage in breaking the *PRF* (i.e., distinguishing result of *Pb* from random). However, this contradicts the fact that *Pb is a **(t, ϵ)**-PRF*. Since no such A may exist, the assumption must be incorrect, thus $ELCA_E$ is secure *against IND-CPA*. □

4.3. Countermeasures Spoofing Attacks

ELCA can prevent spoofing attacks (e.g., replay attacks and the man-in-the-middle attacks) using the secure combination integration between encryption and authentication. Moreover, ELCA drops the reply packet from the intruders because of the following reasons:

- The MAC should be checked before performing the decryption process.
- The ephemeral shared secret key is computed at the source and destination.
- The three stages must be carried out by replay attacks before resending the intercepted communication. These steps—calculating the shared secret key, encrypting messages, and calculating the authentication tag—make it incredibly difficult to access information without compromising the shared secret key and hash function.

4.4. Countermeasures against Brute Force Attacks

ELCA addresses the weak bits issue and offers perfect forward secrecy because the shared secret key must change with each communication session. Additionally, the elliptic curve discrete logarithm problem (ECDLP), which requires $0.886 * \sqrt{k}$ steps, must be solved by the brute force attacker. This indicates that the security strength is 96, which will probably require a lot of computer power [37,40].

4.5. Countermeasures against Session Hijacking Attack

Secure hash functions such as SHA-2 and CMA are applied using the shared secret key in ELCA [29]. This method produces a random integer that can be used to create the session identification, such as the digest of a shared secret key after it has been hashed. In order to obtain access to the communication channel between the IoT parties, the attacker must determine the authentication code if he is successful in cracking the session ID. This is mostly because the verification process between the IoT sender and receiver of the session requires the authentication code.

4.6. Countermeasures against IoT Device Capture and Stolen-Verifier Attacks

The ELCA cryptographic system uses the built-in multifactor hash functions (e.g., CMA [29]) that are burned during programming sessions inside all IoT devices to protect against IoT device capture and stolen-verifier attacks. As stated in the assumption, the multifactor hash functions used in ELCA are flashed and transformed into low level source code language. Therefore, the stolen key will not function without disabling the hash algorithms, preventing the hacker from accessing any safe data in the IoT device.

5. Implementation and Performance Evaluation of ELCA on IoT

Based on the resource constraints in terms of computing cost, storage utilization, and power consumption, the security software in IoT platforms should be assessed. Therefore, ELCA adopted the concept of ECDH for exchanging the secret key advised by SECG/NIST (such as Secp192r1) [37]. Following are some reasons why utilizing the Secp192r1 standard elliptic curve in ELCA is advantageous:

- The size of the encryption and authentication keys is 24 bytes (192 bits), and the processing latency for the ECDH to generate and exchange the secret key has been assessed to be 0.576 s through experimental testing [31].
- It takes $0.886 * \sqrt{k}$ steps to determine the k-size of the acknowledged ideal algorithm for the ECDLP. In general, if the security system employs at least 2*k-bit key size, a k-bit security strength can be attained. Because of this, ELCA chose to employ the Secp192r1 curve, which can offer 96-bit security strength [37,40].
- The 6LowPAN protocol, which uses a 40-byte header to establish connections between IoT devices and sensor nodes, can be used to construct IoT devices with messages up to 127 bytes in size [41].

Since Mininet-IoT can replicate the IoT hardware and communication description, it is used in the assessment scenarios to implement and verify the performance of ELCA [42]. As can be shown in Figure 5, one IoT gateway (BaseST1), eight static IoT devices (sensors 1 through 8), two intruders (Intrudr6 and Intrudr7), and one mobile IoT device (IoTDev5) make up the experiment's IoT network topology. The adversary model that was covered in the previous part is mostly implemented by intruders. Each IoT hardware board includes two network interface cards, one for IPv4 and one for IPv6 communications with the IoT base station (i.e., 6LowPAN). Additionally, all sensors, IoTDev5, and BaseST1 have the suggested ELCA software uploaded. Additionally, all legitimate IoT devices exchange public keys and secure packets utilizing client–server socket programming in combination with ELCA code. BaseST1 implements the server code, and IoTDev5 and all sensors run the client code. The settings and setup of the experiment are shown in Table 3. In Mininet-IoT, the 6LowPAN protocol is implemented on the TCP/IP model using the 802.15.4 hwsim and 802.11 hwsim wireless models. Additionally, the wireless signal's propagation model is set up using a shadowing model, which depicts the actual signal degradation brought on by signal impairments including attenuation, noise, and interference. In the experiment, the grid network area measures 1000 m by 900 m, and random movement is used to construct the mobility model of mobile devices. To investigate the effectiveness of ELCA against intruders using dictionary and brute force attacks, the operating time of every experimental program is set to 1000 s.

Table 3. Experiment Configuration.

Parameter	Values
MAC and PHY	802.15.14_hmsim and 802.11_hmsim
Propagation Model	Shadowing
Path loss exponent	3.0
Shadowing deviation (dB)	3.0
Event area	(1000 m × 900 m)
Number of IoT devices	12
Coverage of IoT device	150 m
Cover range of BaseST1	250 m
Traffic Emulator	TCP Socket client/server; 1000 messages.
Performance metrics	CPU execution time, storage cost, and energy consumption
ECDH curve	Secp192r1
Message Size	127 bytes
Key size	192 Bits
Emulation duration	1000 s

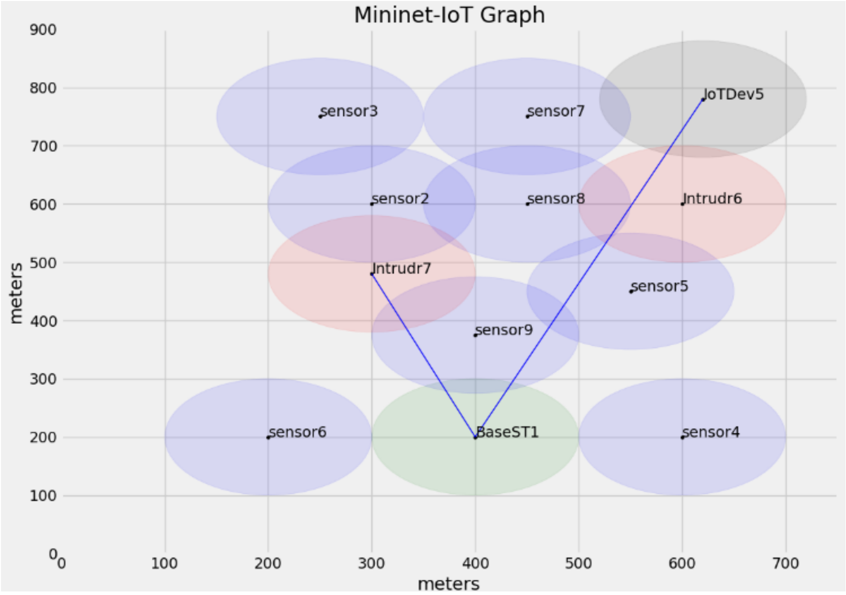


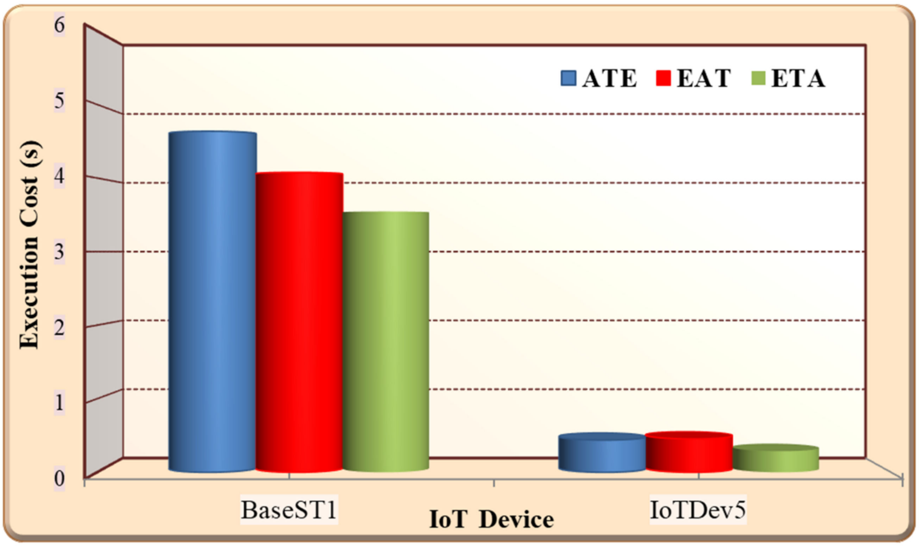
Figure 5. IoT mesh topology.

5.1. Performance Evaluation and Results Discussion

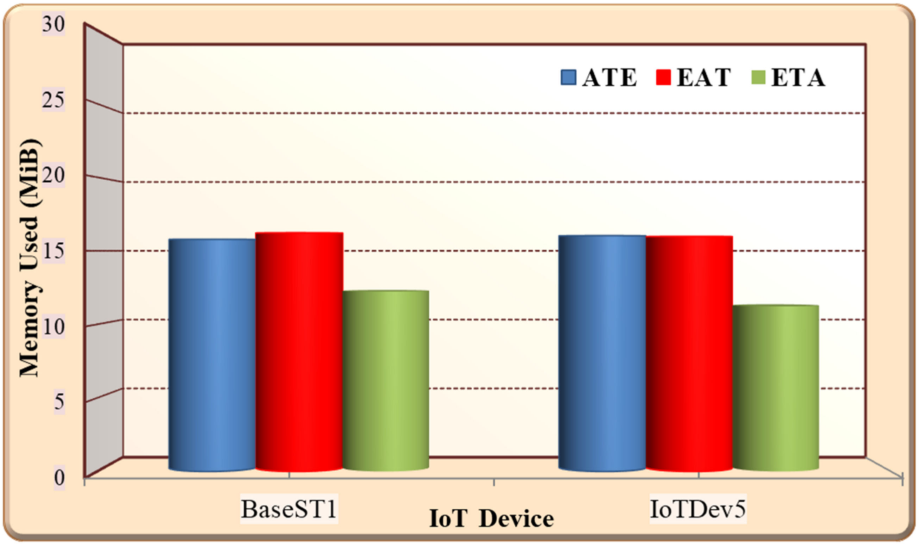
In terms of CPU execution time, memory utilization, and power consumption expenses, the suggested integration of encryption and authentication (for example, ELCA) was evaluated in terms of performance. For the three combinations of authentication and encryption shown in Figure 2, a comparison of performance analysis was investigated. Additionally, ECIES AES and ECIES Ra (RFC4503), two benchmark security algorithms, were used to compare ELCA's performance. Python is used throughout the source code and is implemented in the Mininet-IoT emulator. Additionally, all baseline algorithms' primary source codes can be downloaded from the security website [43]. Numerous scenarios were run, and each testbed was repeated ten times while exchanging 1000 packets. Finally, using the mean and standard deviation as inputs and accepting 5% variation errors in the sample, the average findings were determined with a confidence interval that exceeds 95%. Furthermore, the memory profiler and cProfile programs offer deterministic cost profiling of the baseline methods and ELCA. Memory profiler can be used to calculate an algorithm's execution time, storage expense, and energy usage. The product of CPU execution time and the quantity of steps per execution (s/e) can be used to evaluate the entire cost of CPU execution time. Additionally, the total cost of communication (send/received message) data, sensed data, and the cost of the source code in a time unit can be used to calculate the storage cost in each IoT device. Additionally, the total energy required by IoT devices (mJ) can be calculated as the total energy used to carry out the security algorithm's source code plus any packet overhead [44].

5.1.1. Comparison between Integration Methods of Authentication and Encryption

In this experiment, the performance of using ELCA in three methods of integration between authentication and encryption was evaluated. ELCA was implemented using the three combination approaches (e.g., ATE, EAT and ETA) illustrated in Figure 2. Generally, the results in Figure 6 show that the performance cost of BaseST1 in three combination is higher than IoTDev5. This is mainly due to the type of connection in the IoT system is many-to-one that means all sensor devices send the environment data to the sink (BaseST1). The sink in Figure 6 manipulated the security for all data in the IoT system. As shown in Figure 6a, the ELCA with ETA experiences on average 30.74% less CPU execution time compared to ELCA with ATE, and it experiences on average 15% less CPU execution time compared to ELCA with EAT. Moreover, Figure 6b illustrates that ELCA with ETA experiences on average 22.5% less memory usage compared to ELCA with ATE, and it experiences on average 32.63% less memory usage compared to ELCA with EAT. Moreover, Figure 6c shows that ELCA with ETA consumes on average 68.7% less energy consumption compared to ELCA with ATE, and it consumes on average 52.5% less energy consumption compared to ELCA with EAT. The results presented in Figure 6 show that the impressive performance of the ELCA with ETA algorithm is mainly achieved due to the following reasons: Firstly, ELCA with ETA uses fewer steps of call functions due to the verification of authentication being implemented before the decryption, which causes a reduced CPU execution time, less memory to be used, and reduced power consumption. However, ATE and EAT must implement decryption and verification of authentication with all received ciphertexts and tags, which consumes more resources in term of energy consumption, storage cost, and CPU execution time. Finally, ATE and EAT consume higher call functions, execution time, and communication overheads due to the frequent uses of scalar multiplication and the inverse modular multiplicative in the decryption process.

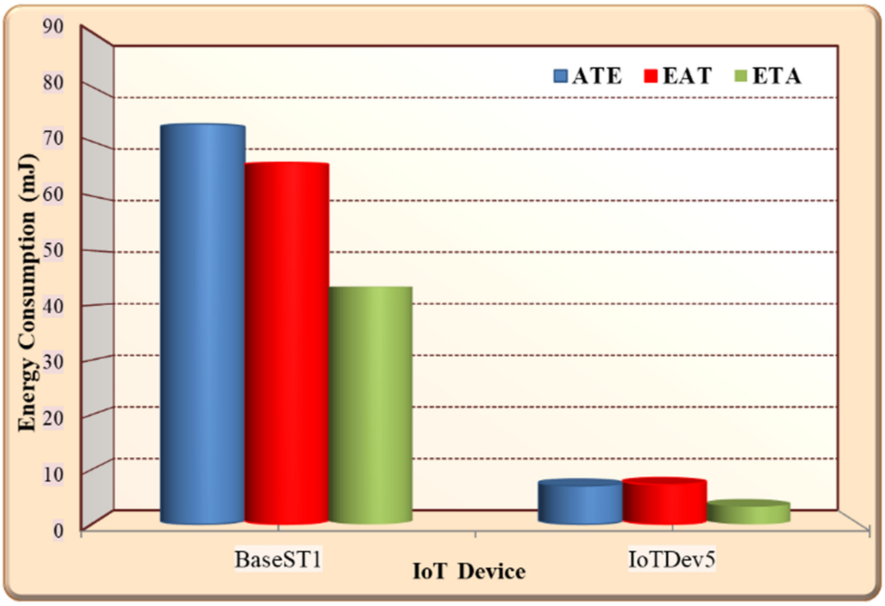


(**a**)



(**b**)

Figure 6. *Cont.*

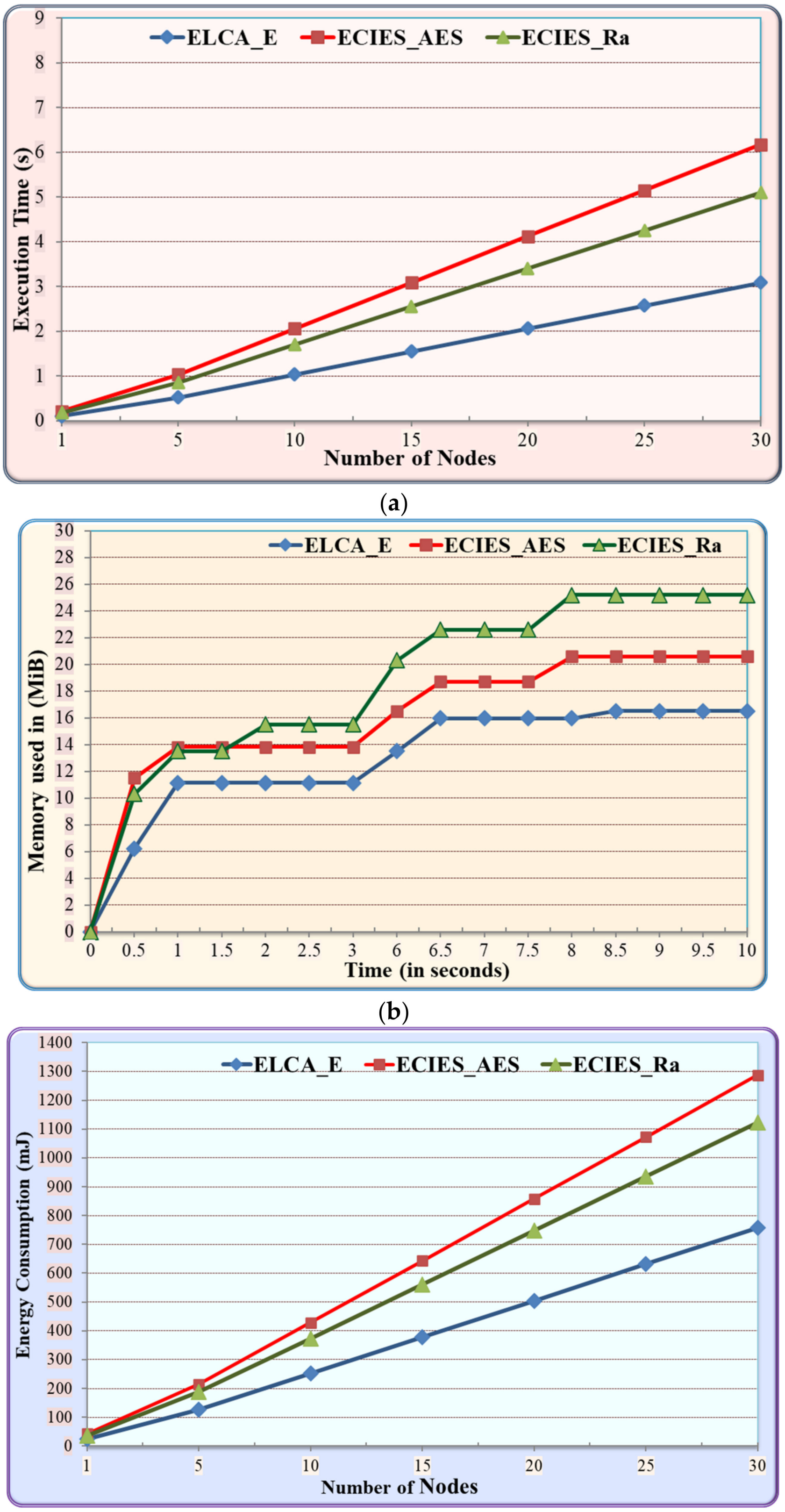


(**c**)

Figure 6. Comparison between integration methods of authentication and encryption. (**a**) Execution cost; (**b**) storage cost; (**c**) energy consumption.

5.1.2. Performance of Cryptographic Algorithms

It has been determined how well ELCA encryption (ELCA_E) performs in comparison to ECIES_Ra and ECIES_AES. As can be seen in Figure 7a, ELCA_E executes with an average execution time that is 50% lower than that of EDIDS_AES and averages 39.4% lower than that of ECIES_Ra. Additionally, Figure 7b shows that ELCA_E uses memory on average 19.6% and 32% less efficiently than ECIES_AES and ECIES_Ra. Additionally, Figure 7c demonstrates that ELCA_E uses an average of 32.6% less energy than ECIES_Ra and there is a difference of 41.2% between ECIES_AES and ELCA_E. The aforementioned results show that ELCA E outperforms ECIES_AES and ECIES_Ra in terms of CPU time execution, storage cost, and energy usage. This is mostly because of the following factors: Firstly, ELCA_E uses less computing power and energy during encryption and decryption because it is based on an effective mathematical random function. For each session between IoT devices, ELCA_E generates an overall shared secret key that ensures perfect forward secrecy of the encrypted message. Second, because fewer functions are called and there are fewer execution steps for each function, ELCA_E uses less storage space. Finally, ECIES_AES and ECIES_Ra employ more difficult and inefficient encryption and decryption techniques than ELCA_E. In conclusion, the experimental findings demonstrate that the suggested integration of authentication and encryption in ELCA is efficient, lightweight, and offers exceptional performance in terms of CPU execution time, storage cost, and energy consumption. More crucially, it fixes the issues with symmetric cryptography's key distribution and the verification of the sender's identity in digital signatures.



(**a**)

(**b**)

(**c**)

Figure 7. Comparison between ELCA encryption (ELCA_E) and baseline cryptographic algorithms on IoT. (**a**) Execution cost; (**b**) storage cost; (**c**) energy consumption.

6. Conclusions and Future Work

The proposed secure integration between encryption and authentication (e.g., ELCA) algorithm was presented and compared with standard lightweight cryptographic schemes. ELCA utilized ECDH to implement key distribution, while the weak bits problem in the shared secret key is resolved. The security of ELCA was proven mathematically using the IoT adversary model and the random oracle model. The finding in the experimental results shows the efficiency and effectiveness of ELCA performance in terms of a reduced CPU execution time by 50%, reduced storage cost by 32–19.6%, and reduced energy consumption by 41% compared to the baseline cryptographic algorithms. The future work of this research will focus on developing an unforgeable digital signature based on the three steps of hash function inspections for IoT networks. Moreover, the weak bit problem will be resolved using advanced key generation without concerns about the IoT key selection.

Author Contributions: Conceptualization, A.A.A. (Adel A. Ahmed) and W.A.; methodology, A.A.A. (Adel A. Ahmed); software, A.A.A. (Adel A. Ahmed); validation, S.J.M., A.A.A. (Ahmed A. Alzahrani) and W.A.; formal analysis, S.J.M.; investigation, W.A.; resources, A.A.A. (Adel A. Ahmed); data curation, A.A.A. (Ahmed A. Alzahrani); writing—original draft preparation, A.A.A. (Adel A. Ahmed); writing—review and editing, W.A.; visualization, A.A.A. (Ahmed A. Alzahrani); supervision, A.A.A. (Adel A. Ahmed); project administration, A.A.A. (Adel A. Ahmed); funding acquisition, A.A.A. (Adel A. Ahmed). All authors have read and agreed to the published version of the manuscript.

Funding: This research work was funded by institutional Fund Projects under grant no. (IFPIP:324-830-1443). The authors gratefully acknowledge technical and financial support provided by the Ministry of Education and King Abdulaziz University, DSR, Jeddah, Saudi Arabia.

Data Availability Statement: Not applicable.

Acknowledgments: This research work was funded by institutional Fund Projects under grant no. (IFPIP:324-830-1443). The authors gratefully acknowledge technical and financial support provided by the Ministry of Education and King Abdulaziz University, DSR, Jeddah, Saudi Arabia.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Malina, L.; Hajny, J.; Fujdiak, R.; Hosek, J.J. On perspective of security and privacy-preserving solutions in the internet of things. *Comput. Netw.* **2016**, *102*, 83–95. [CrossRef]
2. Hussain, S.; Ullah, S.S.; Ali, I.; Xie, J.; Inukollu, V.N. Certificateless signature schemes in Industrial Internet of Things: A comparative survey. *Comput. Commun.* **2022**, *181*, 116–131. [CrossRef]
3. Qiu, J.; Tian, Z.; Du, C.; Zuo, Q.; Su, S.; Fang, B. A survey on access control in the age of internet of things. *IEEE Internet Things J.* **2020**, *7*, 4682–4696. [CrossRef]
4. Li, S.; Zhang, T.; Yu, B.; He, K. A Provably Secure and Practical PUF-Based End-to-End Mutual Authentication and Key Exchange Protocol for IoT. *IEEE Sens. J.* **2021**, *21*, 5487–5501. [CrossRef]
5. Arne, B.; Le, N.; Dominik, S.; Stephan, S.; Lars, C.W. Security Properties of Gait for Mobile Device Pairing. *IEEE Trans. Mob. Comput.* **2019**, *19*, 697–710.
6. Attarian, R.; Hashemi, S. An anonymity communication protocol for security and privacy of clients in IoT-based mobile health transactions. *Comput. Netw.* **2021**, *190*, 107976. [CrossRef]
7. Almajed, H.N.; Almogren, A.S. SE-Enc: A Secure and Efficient Encoding Scheme Using Elliptic Curve Cryptography. *IEEE Access* **2019**, *7*, 175865–175878. [CrossRef]
8. Bu, L.; Isakov, M.; Kinsy, M.A. A secure and robust scheme for sharing confidential information in IoT systems. *Ad Hoc Netw.* **2019**, *92*, 101762. [CrossRef]
9. Hendaoui, F.; Eltaief, H.; Youssef, H. UAP: A unified authentication platform for IoT environment. *Comput. Netw.* **2021**, *188*, 107811. [CrossRef]
10. Vidya, R.; Prema, K.V. Lightweight hashing method for user authentication in Internet-of-Things. *Ad Hoc Netw.* **2019**, *89*, 97–106.
11. Katz, J.; Yehuda, L. *Introduction to Modern Cryptography*; CRC Press: Boca Raton, FL, USA, 2007.
12. Barker, E. Recommendation for Key Management. In *Computer Security*; NIST Special Publication 800-57 Part 1, Revision 5; USA, Department of Commerce: Washington, DC, USA, 20 May 2020. [CrossRef]
13. Chuang, Y.-H.; Lo, N.-W.; Yang, C.-Y.; Tang, S.-W. A Lightweight Continuous Authentication Protocol for the Internet of Things. *Sensors* **2018**, *18*, 1104. [CrossRef] [PubMed]

14. Alaba, F.A.; Othman, M.; Hashem, I.A.T.; Alotaibi, F. Internet of Things security: A survey. *J. Netw. Comput. Appl.* **2017**, *88*, 10–28. [CrossRef]
15. Riad, K.; Huang, T.; Ke, L. A dynamic and hierarchical access control for IoT in multi-authority cloud storage. *J. Netw. Comput. Appl.* **2020**, *160*, 102633. [CrossRef]
16. Alexander, J.M.; Kueffer, C.; Daehler, C.; Edwards, P.J.; Pauchard, A.; Seipel, T.; Arévalo, R.J.; Cavieres, L.A.; Dietz, H.; Jakobs, G.; et al. NETRA: Enhancing IoT Security Using NFV-Based Edge Traffic Analysis. *IEEE Sens. J.* **2019**, *19*, 4660–4671. [CrossRef]
17. Hellaoui, H.; Koudil, M.; Bouabdallah, A. Energy-efficient mechanisms in security of the internet of things: A survey. *Comput. Netw.* **2017**, *127*, 173–189. [CrossRef]
18. Magdich, R.; Jemal, H.; Ayed, M. A resilient Trust Management framework towards trust related attacks in the Social Internet of Things. *Comput. Commun.* **2022**, *191*, 92–107. [CrossRef]
19. Liu, X.; Yu, W.; Liang, F.; Griffith, D.; Golmie, N. On deep reinforcement learning security for Industrial Internet of Things. *Comput. Commun.* **2021**, *168*, 20–32. [CrossRef]
20. Li, X.; Niu, J.W.; Ma, J.; Wang, W.D.; Liu, C.L. Cryptanalysis and improvement of a biometrics-based remote user authentication scheme using smart cards. *J. Netw. Comput. Appl.* **2011**, *34*, 73–79. [CrossRef]
21. Al-Karaki, J.N.; Gawanmeh, A.; Almalkawi, I.T.; Alfandi, O. Probabilistic analysis of security attacks in cloud environment using hidden Markov models. *Trans. Emerg. Telecommun. Technol.* **2022**, *33*, e3915. [CrossRef]
22. Wang, Y.; Yang, G.; Li, T.; Li, F.; Tian, Y.; Yu, X. Belief and fairness: A secure two-party protocol toward the view of entropy for IoT devices. *J. Netw. Comput. Appl.* **2020**, *161*, 102641. [CrossRef]
23. Ahmed, A.A. Lightweight Digital Certificate Management and Efficacious Symmetric Cryptographic Mechanism over Industrial Internet of Things. *Sensors* **2021**, *21*, 2810. [CrossRef] [PubMed]
24. NIST Computer Security Resource Center. Lightweight Cryptography Project. Available online: https://csrc.nist.gov/projects/lightweight-cryptography (accessed on 13 March 2022).
25. Seok, B.; Sicato, J.C.S.; Erzhena, T.; Xuan, C.; Pan, Y.; Park, J.H. Secure D2D Communication for 5G IoT Network Based on Lightweight Cryptography. *Appl. Sci.* **2020**, *10*, 217. [CrossRef]
26. Mohseni-Ejiyeh, A.; Ashouri-Talouki, M.; Mahdavi, M. An Incentive-Aware Lightweight Secure Data Sharing Scheme for D2D Communication in 5G Cellular Networks. *ISeCure* **2018**, *10*, 15–27.
27. Abro, A.; Deng, Z.; Memon, K.A. A Lightweight Elliptic-Elgamal-Based Authentication Scheme for Secure Device-to-Device Communication. *Future Internet* **2019**, *11*, 108. [CrossRef]
28. Javed, Y.; Khan, A.S.; Qahar, A.; Abdullah, J. EEoP: A lightweight security scheme over PKI in D2D cellular networks. *J. Telecommun. Electron. Comput. Eng.* **2017**, *9*, 99–105.
29. Ahmed, A.A.; Ahmed, W.A. An Effective Multifactor Authentication Mechanism Based on Combiners of Hash Function over Internet of Things. *Sensors* **2019**, *19*, 3663. [CrossRef]
30. Sciancalepore, S.; Piro, G.; Boggia, G.; Bianchi, G. Public Key Authentication and Key Agreement in IoT Devices with Minimal Airtime Consumption. *IEEE Embed. Syst. Lett.* **2017**, *9*, 1–4. [CrossRef]
31. Khan, M.A.; Quasim, M.T.; Alghamdi, N.S.; Khan, M.Y. A Secure Framework for Authentication and Encryption Using Improved ECC for IoT-Based Medical Sensor Data. *IEEE Access* **2020**, *8*, 52018–52027. [CrossRef]
32. Muhammad, U.; Ahmed, I.; Imran, M.A.; Shujaat, K.; Usman, A.S. SIT: A lightweight encryption algorithm for secure internet of things. *Int. J. Adv. Comput. Sci. Appl.* **2017**, *8*, 402–411.
33. Shah, R.H.; Salapurkar, D.P. A multifactor authentication system using secret splitting in the perspective of Cloud of Things. In Proceedings of the International Conference on Emerging Trends & Innovation in ICT (ICEI), Pune, India, 3–5 February 2017; pp. 1–4.
34. Hammi, B.; Fayad, A.; Khatoun, R.; Zeadally, S.; Begriche, Y. A Lightweight ECC-Based Authentication Scheme for Internet of Things (IoT). *IEEE Syst. J.* **2020**, *14*, 3440–3450. [CrossRef]
35. Rangwani, D.; Sadhukhan, D.; Ray, S.; Khan, M.K.; Dasgupta, M. A robust provable-secure privacy-preserving authentication protocol for Industrial Internet of Things. *Peer-to-Peer Netw. Appl.* **2021**, *14*, 1548–1571. [CrossRef]
36. Lochter, M.; Merkle, J. *RFC 5639: Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation*; IETF: Felemon, CA, USA, 2010; ISSN 2070-1721.
37. NIST. *Fips Publication 180-2: Secure Hash Standard*; Technical Report; National Institute of Standards and Technology (NIST): Gaithersburg, MD, USA, 2003.
38. Biryukov, A. Adaptive Chosen Plaintext Attack. In *Encyclopedia of Cryptography and Security*; Van Tilborg, H.C.A., Jajodia, S., Eds.; Springer: Boston, MA, USA, 2011.
39. Biryukov, A. Related Key Attack. In *Encyclopedia of Cryptography and Security*; Van Tilborg, H.C.A., Jajodia, S., Eds.; Springer: Boston, MA, USA, 2011.
40. Silverma, J.H. *An Introduction to the Theory of Elliptic Curves, Summer School on Computational Number Theory and Applications to Cryptography*; Brown University: Providence, RI, USA, 2006.
41. IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals. Available online: http://www.ietf.org/rfc/rfc4919.txt (accessed on 27 November 2022).
42. Mininet-IoT Emulator of Internet of Things. Available online: https://github.com/ramonfontes/mininet-iot (accessed on 27 November 2022).

43. A Security Site. Available online: https://asecuritysite.com/encryption (accessed on 27 November 2022).
44. Ahmed, A.A. An optimal complexity H. 264/AVC encoding for video streaming over next generation of wireless multimedia sensor networks. *Signal Image Video Process.* **2016**, *10*, 1143–1150. [CrossRef]

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