Available online at sjuoz.uoz.edu.krd

Science Journal of University of Zakho

Vol. 11, No. 3, pp. 323– 332, July-September, 2023

journals.uoz.edu.krd

p-ISSN: 2663-628X
e-ISSN: 2663-6298

LIGHTWEIGHT CRYPTOGRAPHY METHOD IN THE INTERNET OF THINGS USING ELLIPTIC CURVE AND CROW SEARCH ALGORITHM

Arkan Abduljabbar Saffer [1*] , Soran Abdulkarim Pasha[1] , Ammar M. Aliakbr[2]

[1] Information Technology Department, Kalar Technical College, Sulaimani Polytechnic University, Kurdistan Region, Iraq - arkan.saffer@spu.edu.iq & Soran.pasha@spu.edu.iq

[2] Department of IT, College of Computer & IT, University of Garmian, Kalar, Kurdistan Region, Iraq. ammar.mohmmed@garmian.edu.krd

Received: 1 Nov., 2022 / Accepted: 3 Mar., 2023 / Published: 10 July 2023 *https://doi.org/10.25271/sjuoz.2023.11.3.1055*

ABSTRACT:

The Internet of Things (IoT) as an important technology consists of a heterogeneous and huge number of devices that generates an enormous amount of data in various applications. However, generating and transmitting huge amount of data in the IoT makes it crucial to implement a secure and safe data transmission scheme. Cryptography methods can secure the confidentiality, data integrity, access control, and authentication. Due to constrained resources in IoT devices, providing classical cryptography schemes isn't efficient for IoT applications, so a lightweight cryptography scheme is one of the most important solutions to overcome security challenges in IoT. In this paper, a new security scheme called ECCCSASHA256 based on Elliptic Curve Cryptography (ECC) and Secure Hash Algorithm (SHA-256) using Crow Search Algorithm (CSA) has been proposed for secure data transmission in IoT devices. The ECCCSASHA256 model uses the CSA for generating a private key to encode the elliptic curve. Furthermore, the proposed scheme uses SHA-256 model for hashing the incoming encoded data using ECC. The simulation results indicate that the average throughput of the proposed model was about 8.22% and 8.97% higher in encryption and 8.72% and 9.81% higher in decryption compared to 3DES&ECC&SHA-256 and RC4&ECC&SHA-256, respectively.

KEYWORDS: IoT, Data Security, Cryptography, CSA, ECC, SHA-256.

1. INTRODUCTION

IoT is an emerging and evolving phenomenon that has entered the human life cycle and business activities as an integral technology. On the other hand, IoT devices have many inherent limitations such as low computation power, low bandwidth, high latency, and low communication capabilities (Mousavi et al., 2021). IoT is a collection of interconnected objects that can interact with other components of the set (Jazebi & Ghaffari, 2020). The various applications of the IOT are growing daily and have important roles in our lifestyles from smart homes to smart cities (Seyfollahi & Ghaffari, 2020). For the security of IoT, cryptography algorithms have been proposed so far, all of which work based on two methods of symmetric and asymmetric cryptography (Kandhoul & Dhurandher, 2018). The method of asymmetric key cryptography, like elliptic curve cryptography (ECC), which is widely used for high security, has small and light computations for encrypting transmitted messages (Joglekar et al., 2019). Asymmetric key-based algorithms have a very high degree of security compared to symmetric keys that can secure IoT devices and prevent unauthorized access to data (Dhanda, Singh, & Jindal, 2020). Therefore, designing a lightweight and robust security algorithm against IoT attacks is a crucial issue (Sadhukhan, Ray, Biswas, Khan, & Dasgupta, 2021).

This paper indicates a new security method called ECCCSASHA256 using EC encryption (Miller, 1986) and the crow search algorithm (CSA) (Askarzadeh, 2016). In the proposed security scheme, the EC algorithm and SHA-256 are used to encrypt the data in the first phase and to hash the encrypted data, respectively. To generate a private key in an EC, the crow search algorithm is used. The purpose of generating a private key by CSA is to prevent the private key from extraction by hackers and to ensure data security. The purpose of using SHA-256 is to ensure the confidentiality of data. In SHA-256, a small change in the input of hashing scheme will change the produced hash (output data). On the other hand, in SHA-256 it is almost impossible to reverse engineer to reconstruct the original data (input data). These are advantages of SHA-2the 56 method. Therefore, we use SHA-256 method for hashing purposes in the proposed scheme (Fotohi & Aliee, 2021; Wang, Dong, Kang, & Chen, 2023; Wei, Jiang, & Deng, 2023).

In IoT devices, the emphasis is on the use of lightweight algorithms. This paper provides a model for information security in IoT based on a cryptography scheme, which will be pursued to achieve the following goals.

• Many important applications of IoT such as healthcare and smart home need secure data transactions. ECC algorithm is appropriate for supplying security services for IoT resource constrained devices. The proposed scheme uses ECC for encryption phase.

• Hashing the encrypted data using the SHA-256 algorithm. Sha-256 hashing scheme has high level security.

• Generating the private key for ECC scheme and data encryption phase using the crow search algorithm

The rest of the paper is organized as follows: Section 2 reviews the related works. The proposed scheme is described in Section 3. The performance of the proposed scheme is evaluated in Section 4. Finally, Section 5 concludes the paper and discusses future work.

2. BASIC CONCEPTS AND RELATED WORKS

2.1 Crow search algorithm

The crow search algorithm as one of the new meta-heuristic algorithms is taken from the social behavio~~u~~r of crows. We assume that the surrounding space contains *d* dimensions and *N* crows. The position of each row in each step of the iteration based

* Corresponding author

on the vector $x^{i,iter}$ is equal to $x^{i,iter} = [x_1^{i,iter}, x_2^{i,iter}, \dots, x_d^{i,iter}]$. Also, each crow has a memory that remembers where to hide food. In each step of the iteration, the hiding position of crow *i* is specified as $m^{i,iter}$. The said position is the best position that crow i has achieved so far. Meanwhile, the best place to hide food is stored in the memory of the crow, and the crows search the environment to find the best place to hide food when needed. Suppose that in each step, crow *j* wants to observe its food hiding place ($m^{j,iter}$). At this stage, crow *i* decides to follow crow *j* to reach its food place. Algorithm 1 shows the pseudo-code of the crow search algorithm.

Algorithm 1. Pseudocode of the crow search algorithm

Input: Problem objective–f(x), Problem dimension–D, Population size–N;
//Define the maximum number of iterations ($maxiter$)
//Define the Awareness Probability (AP) and flight length (fl)
//Define the upper (ub) and lower (lb) bounds of the search space
//Initialize the crow population within the prescribed boundaries
//valuate the position of each crow in the flock in terms of their corresponding fitness value
//Give the initial values to memory of each crow in the flock
While $iter < maxiter$ do
for i = 1 to N (each crow in the flock) do
    Choose a random crow from the flock to follow (crow j)
    Generate uniformly distributed Gaussian random numbers (rj and ri)
if $r_j \geq AP$
$x^{i,iter+1} = x^{i,iter} + r_i \times fl^{i,iter} \times (m^{j,iter} - x^{i,iter})$
else
$x^{i,iter}$ = generate a random position in the search space
end
end
Apply the boundary check mechanism and eliminate infeasible solutions
Assess the quality of each crow position
Update the memory of each crow
$iter = iter + 1$
end
Output: Best solution vector

2.2 Related works

Recently, many papers have presented based on cryptographic algorithms to secure IoT devices. This section explains related works.

In (Mousavi et al., 2021), the authors propose a security scheme based on a combination of RC4, EC, and SHA-256 algorithms to secure the important data of the intelligent irrigation system. This scheme uses an EC to improve the RC4 algorithm. The RC4 key is encrypted using the EC, and then the data is converted to SHA-256, creating a completely incomprehensible state of inaccessibility to the data. The results show that the encryption and decryption time of this scheme was shorter compared to other models and its throughput was higher than 50% compared to other models.

In (Elhoseny et al., 2020), the authors propose a security scheme for the IoT based on the particle swarm optimization algorithm and the locust algorithm to improve the key of the EC algorithm. The public key and the private key in the EC algorithm are generated by the particle swarm optimization algorithm and the propeller algorithm. The most important advantage of the hybrid design is the reduction of processor time compared to other models. The hybrid design was tested on a set of different images and the results showed that the amount of error and processing time was reduced. In (Davahli et al., 2020), the authors propose an IoT intrusion detection scheme based on a combination of genetic algorithms and the gray wolf. The combination of genetic algorithms and gray wolf was used to select the feature and the support vector machine was used to classify the data. The AWID dataset is used to detect the type of attacks on IoT devices. The test results showed a reduction in processing time and an increase in detection accuracy.

In (Reddy & Kumar, 2020), the authors propose a security scheme for data encryption based on the flower pollination algorithm and AES (Advance encryption standard). In their design, the flower pollination algorithm has been used to generate the key of the AES algorithm. The purpose of their design was to complicate the encryption and decryption of data. The proposed design was compared with AES, DES (Data encryption standard), and RC4 algorithms and the results showed that the memory time consumed in the proposed design was shorter and the penetration percentage was lower. In (Kota et al., 2018), the authors propose a security scheme based on particle swarm optimization on and cuckoo search algorithm to improve EC cryptography. The purpose of the hybrid model is to create a private key to encode the ECC. The production of the initial population is in the range of natural numbers, and particle updates are performed in each iteration to find the optimal key. The original text is also converted to ski codes.

In (Verma et al., 2020), the authors propose a security scheme for data encryption based on the locust algorithm and EC encryption. In this method, the locust algorithm is used to generate the private key in EC encryption. Each user selects a random number and encrypts the random number with the public key to generate encrypted text. The results on different files show that the encryption and decryption time in the proposed design was shorter compared to the genetic algorithm based on the EC and the EC encryption. In (Farah et al., 2020), the authors propose an improved security scheme by AES encryption using the genetic algorithm. The purpose of the proposal was to improve the random replacement box. Logistic turbulence mapping has also been used to add complexity and robustness to the proposed method. Merger and jump operations have been performed on the replacement box blocks.

In (Roselin et al., 2021), a security method is proposed by RC4 cryptography using a fruit fly optimization algorithm. This metaheuristic algorithm effectively detects the optimal locations of the pixels. At first, the image is divided into n 8 8 8 blocks on which a permutation combination is performed. The value of the pixels is converted to binary and then the XOR operation is performed by RC4 on the values. The results showed that the fruit fly optimization algorithm was more efficient than the genetic algorithm and particle community optimization. In (Kamal et al., 2021), the authors propose a security plan to improve S-DES based on three algorithms: cuckoo search algorithm, glow-worm algorithm, and black hole algorithm. The 10-bit key is an important input for cryptography generated by metaheuristic algorithms.

In (Raja Rajeshwari & Ramakrishnan, 2022), the authors proposed a lightweight cryptography method for securing resource-constrained devices that use Optimised ECC with Karatsuba algorithm (OECC-KA) for fast multiplication. Several efficient cryptographic algorithms have been widely used for resource constraint device; it may not be suitable with its limited computing power, bandwidth and CPU power. Elliptic curve cryptography is the best choice for limited power usage with high-level security. The proposed optimized ECC with Karatsuba algorithm increases the execution time with fast multiplication algorithm.

In (Das & Namasudra, 2022), the authors presented an encryption scheme using ECC, Advanced Encryption Standard (AES), and Serpent to secure healthcare data in IoT-enabled healthcare infrastructure. The proposed hybrid encryption scheme improves security measures for healthcare data by incorporating both symmetric and asymmetric-based encryption techniques.

Moreover, this method also ensures data integrity by using the elliptic curve-based digital signature. To prove the efficiency of the proposed scheme, both formal security analysis and performance comparisons are presented in this paper. Results and discussion prove the effectiveness of the proposed scheme.
In (Kumar & Koti, 2021), the authors proposed a hybrid Elliptic Curve Digital Signature Algorithm (ECDSA), RC4, and SHA-256 for securing the reliable data which is transmitted from the IoT based healthcare systems. In this scheme, the ECDSA is used for improving the RC4 in which the encryption of key is processed by ECDSA for performing the XOR operation in RC4. Furthermore, the security is ensured by transforming the incoming data using the SHA-256 technique.

3. THE PROPOSED METHOD

The ECCCSASHA256 model contains ECC, crow search algorithm and SHA-256 algorithms. The CSA is used to create a private key for the EC algorithm, and finally the SHA-256 algorithm is used to increase the security and confidentiality of data. Figure 1 indicates the flowchart of the ECCCSASHA256 model.

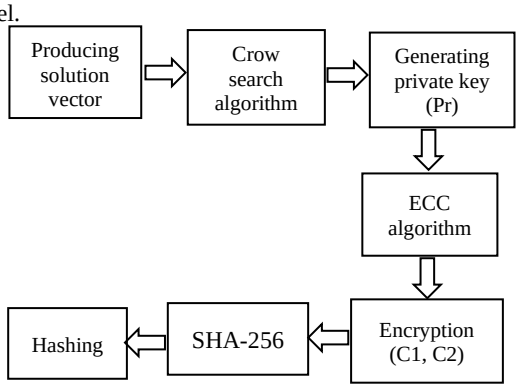

Figure 1. Block diagram of ECCCSASHA256

3.1 Elliptic curve algorithm

Assume that p is a prime number and the finite field F_p contains a set of numbers smaller than p, the elliptic curve E with two-dimensional coordinates is defined according to Eq. (1) (Miller, 1986).

$$y^2 = x^3 + ax + b \qquad (1)$$

The values a and b are changed for different elliptic curve equations. In Eq. (1), $a, b \in F_p$ and $4a^3 + 27b^2 \neq 0 \pmod p$. If the coordinates (x, y) of a point in Eq. (1) are true, that point belongs to the elliptic bend. Also, $E(F_p)$ and G are the points set on the elliptic bend (correct points) and the point on E, respectively. For encryption in the EC, the random number x in the interval [1, n-1] and belonging to the field F_p is selected and considered as a private key. Then, the public key P_u is estimated as $P_u = P_r G$, G is the point on the elliptical bend and P_r is the private key. In the EC algorithm, each character is converted to bytes, then the bytes are converted to dots in the shape of (x, y), then the dots must be encrypted on the EC, and then all the coded dots must be converted to bytes.
The steps for encoding an EC are as follows:
1) Start-up step: The sides of the elliptic curve E and the generator G are selected from order p.
2) Public key production stage: The public key is formed as $P_u = P_r G$. In this equation, P_u and P_r are the public and private keys respectively.
3) Encryption step: To encryption, the number r is randomly selected and encryption operation is performed by Eq. (2). The sender sends the value C to the recipient to send the message m.

$$C = Enc(m) = \begin{cases} c_1 = rG \\ c_2 = m + rP_u \end{cases} \rightarrow Enc(m) = (c_1, c_2) \qquad (2)$$

4) Decryption step: The receiver receives the value of message m by receiving C and using the private key P_r r, through Eq. (3).

$$Dec(C) = c_2 - P_r . c_1 = m + rP_u - P_r rG = m + rP_r G - P_r rG = m \qquad (3)$$

3.2 Crow search algorithm to generate private key

The crow search algorithm starts with n populations (crows) in a group. In each iteration of the algorithm, the i's crow position provides an optimal solution to the optimization problem. Figure 2 indicates the block diagram of the crow search algorithm for private production.

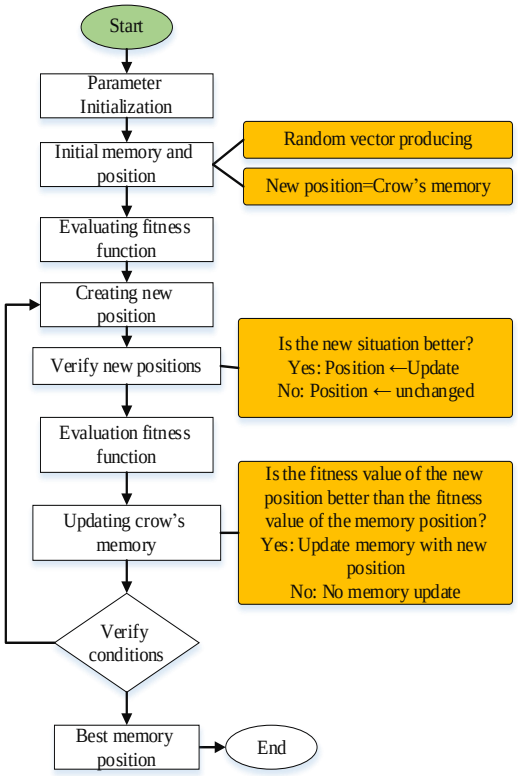

Figure 2. CSA for private key generation

The steps to implement the crow search algorithm to generate a private key are as follows:
1) The value of crow search algorithm parameters such as number of crows, search space dimension, flight length, probability of awareness, and maximum number of repetitions.
2) According to the number of crows, the position of each crow in the d-dimensional search space is defined according to Eq. (4). The matrix of Eq. (4) contains the number of crows (rows) and the number of dimensions (columns). In a next d problem, the position vector in the search space for crow i is $[x_1^i, x_2^i, \dots, x_d^i]$.

$$Crows = \begin{bmatrix} x_1^1 & x_2^1 & \dots & x_d^1 \\ x_1^2 & x_2^2 & \dots & x_d^2 \\ \vdots & \vdots & \vdots & \vdots \\ x_1^N & x_2^N & \cdots & x_d^N \end{bmatrix} \qquad (4)$$

3) The initialization of the positions for each crow is done according to Eq. (5), where L and U have a range of low and high values in the problem space and j are the dimensions of the problem.

$$x_{ij} = L_{ij} + rand \times (U_{ij} - L_{ij}); \quad i = 1,2,\dots,NP;\ j = 1,2,\dots,d \qquad (5)$$

The memory of each crow is randomly defined according to Eq. (6). Crows have no experience at first. Therefore, it is assumed that crows hid their food in the early stages.

$$\text{memory} = \begin{bmatrix} m_1^1 & m_2^1 & \dots & m_d^1 \\ m_1^2 & m_2^2 & \dots & m_d^2 \\ \vdots & \vdots & \vdots & \vdots \\ m_1^N & m_2^N & \cdots & m_d^N \end{bmatrix} \quad (6)$$

The evaluation of the objective function for each crow is performed by placing the position of each crow (decision variables) inside the objective function.

4) Generating a new position is done by randomly selecting a crow and tracking its position using Eq. (7).

$$x^{i,iter+1} = \begin{cases} x^{i,iter} + r_i \times fl^{i,iter} \times (m^{j,iter} - x^{i,iter}) & r_j \geq AP^{j,iter} \\ \text{a random position} & \text{otherwise} \end{cases} \quad (7)$$

In CSA, flight length has a major effect on increasing the performance of the algorithm to calculate the crows' hidden positions. If the flight length value is constant, then the entire search space must be covered to find optimal solutions. If the flight length value is low, then only the solutions in the flight path are explored and if the flight length value is high, then the out-of-flight solutions are explored. Therefore, in order to accelerate convergence, exploration and exploitation, a variable method is used for equilibrium. In this paper, Eq. (8) is used to change the flight duration.

$$fl^{i,iter} = fl_{max}.\exp\left(\log\left(\frac{fl_{min}}{fl_{max}}\right).\left(\frac{iter}{iter_{max}}\right)\right) \quad (8)$$

The value of this parameter is important in this algorithm, because the convergence behavior of the algorithm and the amount of exploration and extraction depend on this parameter. If the value of this parameter is considered high, it increases the variety of solutions in the search space and increases the exploration action, and if the value of this parameter is considered small, most of the work is spent on extraction, which in the end does not achieve good results. Therefore, it is better to determine its value by Eq. (8). In such a way that at first it is so large that the exploration operation is done well and after a while its amount is gradually reduced so that the extraction operation is done well and proper convergence is achieved.

If the value of the AP parameter is high, the variability increases and if the value of the AP parameter is low, the resonance increases. Intensifying the algorithm's ability to focus on points and areas of the search space where the optimal value of the function is located. It should be noted that a powerful algorithm is an algorithm that achieves the best combination of these two components. Therefore, this paper uses a new method to adjust the probability of awareness according to Eq. (9).

$$AP^{i,iter} = AP_{min} + (AP_{max} - AP_{min}) \times \exp\left(\ln\left(\frac{AP_{min}}{AP_{max}}\right)^2.\left(\frac{iter}{iter_{max}}\right)\right) \quad (9)$$

In the ECCCSASHA256 model, each solution vector must be converted to a binary state, where $x^{i,iter+1}$ is the value of the vector x in the continuous state, and the parameter r is a random value in the range zero to one. The sigmoid function according to Eq. (10) and Eq. (11) is used for binary. In the ECCCSASHA256 model, the dimensions of each vector can be zero or one.

$$S(x^{i,iter+1}) = \frac{1}{1+e^{-x^{i,iter+1}}} \quad (10)$$

$$x^{i,iter+1} = \begin{cases} 1 & \text{if } S(x^{i,iter+1}) \geq r \\ 0 & \text{otherwise} \end{cases} \quad (11)$$

The objective function for each vector is defined based on Eq. (12). In Eq. (12), L_v is a vector. Objective The objective function is maximization, and the vector with the highest value is selected as the appropriate vector.

$$F = \frac{1}{\sqrt{L_v}}\left|\left(\frac{1+\sqrt{L_v}}{\text{Sum of 1}}\right) - \left(\frac{|1-\sqrt{L_v}|}{\text{Sum of 0}}\right)\right| \quad (12)$$

6) Calculate the fitness function.

7) If the new position of the crows fits better than the current position, the crows' memory will be updated using Eq. (13).

$$m^{i,iter+1} = \begin{cases} x^{i,iter+1} \leftarrow f(x^{i,iter}) \text{is(better)than } (f(m^{i,iter})) \\ m^{i,iter} \leftarrow \text{otherwise} \end{cases} \quad (13)$$

Steps 4 to 7 are repeated until the end of the program.

In the ECCCSASHA256 model, the solution vectors include the numbers 0 and 1. The best vector to choose from is the vector whose number of units is in the optimal state, that is, contains the least number one. The value obtained as d is the optimal value for the private key of the elliptic curve (EC) algorithm with length 7, and the EC will have the necessary power for encryption. The XOR operation ($d \oplus 1$) is then performed on the resulting key. The purpose of the XOR operation is to solidify the key and change the bits for non-recognition. The value obtained as a private key (d) is multiplied by the base point (G) in the EC algorithm.

In this paper, crow-based cryptographic algorithm is combined with EC algorithm and an asymmetric cryptographic system is proposed for encryption. In the ECCCSASHA256 model, the goal is to generate a private key in an EC encryption system. In the first step, cryptography is based on an EC. In the second step, the encrypted text is encrypted by the EC using the SHA-256 algorithm to increase security and confidentiality.

4. PERFORMANCE EVALUATION

This section evaluates the ECCCSASHA256 model based on criteria such as encryption time, decryption time, throughput, and security. Evaluations and results in the Sharp 2017 programming environment were performed on a system with the specifications of a quad-core processor with 2 GHz, 6 GB of memory and Windows 8 (64-bit). The key length in the ECCCSASHA256 model is 128 bits. Table 1 shows the simulation parameters.

Table 1. Simulation parameters

Configuration	2017C#	Cryptography class
Models	Key length (bit)	Block size (byte)
3DES	128	64
ECC	128	-
RC4	128	-
CSA	Number of crows	50
	Fly length	0.5-2
	Awareness probability	0.5-2

4.1 Encryption and decryption time

Table 2 shows the results of encryption time (milliseconds) on the ECCCSASHA256 model based on the number of repetitions with 10 runs. The initial population for the experiment is 30. According to the results, if the number of iterations is 60 and the file size is 100 MB, the encryption time is 7125 ms. If the number of iterations is 150 and the file size is 100 MB, then the encryption time is 6884 milliseconds. If the number of iterations is 200 and the file size is 100 MB, then the encryption time is 6676 milliseconds.

Table 2. Encryption time based on number of repetitions

Size of input file (KB)	Encryption time based on number of repetitions							
	30	**60**	**80**	**100**	**120**	**150**	**180**	**200**
1000	64	67	71	66	73	75	76	76
2000	126	125	127	127	127	129	131	142
3000	193	188	193	188	190	188	200	193
3500	236	225	221	226	227	223	239	234
4000	265	265	280	255	252	270	255	291
4500	278	286	295	286	293	294	287	331
5000	313	317	318	315	319	315	340	315
8000	517	536	541	498	520	509	507	537
9000	567	639	596	577	561	574	571	564
10000	638	659	638	623	625	630	641	660
20000	1318	1337	1310	1344	1290	1309	1302	1356
50000	3419	3374	3248	3316	3219	3411	3232	3236
100000	6902	7125	6889	6547	6628	6884	6661	6676
Average	1141.231	1164.861	1132.842	1105.231	**1101.846**	1139.806	1110.923	1132.924

Table 3 indicates the decoding time on the ECCCSASHA256 model based on the number of repetitions and with 10 executions. The initial population for the experiment is 30. The results show that if the number of iterations is equal to 80, then the decoding time is shorter. If the number of duplicates is equal to 60 and the file size is equal to 100,000 KB, then the encryption time is equal to 235/1056. If the number of iterations is equal to 100 and the file size is equal to 100,000 KB, then the encryption time is equal to 1085.308 ms. If the number of iterations is equal to 150 and the file size is equal to 100,000 KB then the encryption time is equal to 1106.385 ms.

Table 3. Encryption time of ECCCSASHA256

Size of input file (KB)	Decryption time							
	30	**60**	**80**	**100**	**120**	**150**	**180**	**200**
1000	57	57	56	55	56	62	65	66
2000	116	114	107	109	110	109	111	107
3000	169	167	163	164	168	169	170	173
3500	194	197	192	205	194	192	198	201
4000	222	228	229	221	220	223	224	265
4500	274	250	256	264	276	266	255	283
5000	297	288	307	309	292	295	279	287
8000	481	499	496	488	479	499	474	498
9000	522	599	539	541	539	544	569	533
10000	574	589	592	627	619	597	595	641
20000	1268	1231	1236	1233	1279	1295	1245	1127
50000	3051	3125	3219	3309	3360	3287	3475	3376
100000	6342	6387	6148	6584	6555	6845	6760	6644
Average	1043.615	1056.231	**1041.538**	1085.308	1088.231	1106.385	1109.231	1092.385

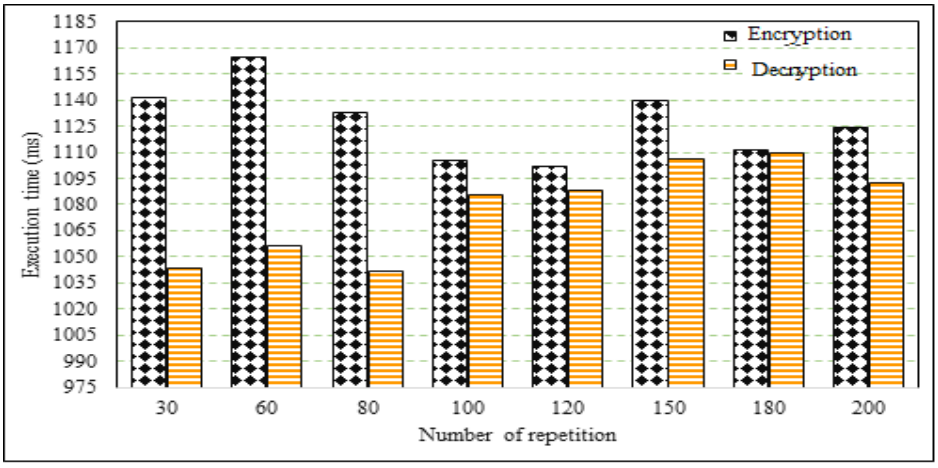

Figure 3. Encryption and decryption time

Figure 3 indicates the encryption and decryption diagram of the ECCCSASHA256 model based on the average time and number of iterations.

4.2 Security analysis

In this section, the ECCCSASHA256 scheme is evaluated and analyzed based on various security parameters and attacks such as middle man attack, confirmation of theft attack, response attack, identity forgery and key reconstruction (Dhillon & Kalra, 2019). Table 4 shows a comparison of security factors.

Man-in-the-Middle (MIM) Attack: In the middle man attack, the intruder sits between a network of objects and eavesdrops on information that is being exchanged between objects and users so that he can access that information. For example, when a user stores data on an IoT server, if the connection is not encrypted and is insecure, the attacker can view it before the login information reaches the site server.

Burglar Attack: Principles of Burglar Attack are designed based on the intruder accessing information such as passwords, etc. through the server database and using this information as a tool for an authorized user. Assuming that the attacker has access to the (I (key), s/he will not succeed in extracting the key, because a hash layer has been created using the SHA-256 hash function (hash (I (key))). And the key is protected against intruders. Additionally, the attacker could not create a fake login request because the encryption key was changed by the XOR operation. Hence, the ECCCSASHA256 is resistant to the attack of a stolen verifier.

Response Attack: In a response attack, the attacker hears and records the requests sent by the user to the server, which later sends the same packages and requests to the server. Assuming that the attacker repeats each of the previous communication messages, the server is in this situation that it needs the public and private key and the value of r (a random value), i.e., u = (public key, private key, r). Therefore, access to the database through an insecure channel is not possible. Even if the attacker can find the public key, he will not be able to guess the private key. Because it is intelligently generated by the crow search algorithm and stored in a vector.

Authentication Attack: In an authentication attack, the attacker enters the IoT network as an authorized user, using the identity of an authorized user. Because the ECCCSASHA256 model uses a crow search algorithm to generate a private key, security is enhanced. In this attack, it is impossible for the intruder to discover the private key.

Unknown key attack: Upon successful completion of a key agreement, user A believes that a session key has been created correctly for user B, but this may not be true for user B, and the session key is set by an intruder instead of user A. Is. Note that this may not be the case with the ECCCSASHA256, as they both compute the common session key. Therefore, the ECCCSASHA256 is immune to key unknown attacks.

Key Recovery: If a new key is created for each communication session, it will be possible to keep the session key anonymous, and it will also be invalidated if the session key of any previous communication is compromised. In the ECCCSASHA256 model, each session key specified includes the points of the EC algorithm and the crow search algorithm, which are generated from fresh random numbers.

Attack on Original Text: An attack on the original text occurs when an attacker tries to obtain a key based on the original text and the encrypted text. To guess the key, the attacker analyzes the relationship between plain text and encrypted text. This type of attack is powerful because the attacker can use encrypted text to guess the key. In the ECCCSASHA256 model, an attacker receives encrypted text and tries to guess the secret key, but due to the use of SHA-256 and XOR, it is not possible to set the message in the ECCCSASHA256 model. Hence, this model resists this attack.

Table 4. Comparison ECCCSASHA256 model based on security factors

Attacks type	3DES&ECC& SHA-256(Kumar & Koti, 2021)	RC4&ECC& SHA-256(Kumar & Koti, 2021)	ECCC SASH A-256
MIM	×	×	√
Phishing	×	×	√
Reply	×	×	√
Spoofing	×	×	√
Unknown key	√	√	√
Key recovery	×	×	√
Cipher-text only	×	×	√
Integrity	√	√	√
Confidentiality	√	√	√

4.3 Encryption and decryption time

Table 5 shows the comparison of ECCCSASHA256 model with RSA-AES model (Zou, Ni, Huang, Shi, & Li, 2020a). The key length in the ECCCSASHA256 model and the RSA-AES model is 128 bits due to the similarity.

Comparison results show that the ECCCSASHA256 model spends less time encrypting and decrypting than the RSA-AES. The encryption time on the 25 MB file is 3.61 in the RSA-AES model and 1.47 in the ECCCSASHA256 model. The decryption time on the 50 MB file is 31.07 in the RSA-AES model and 3.24 in the ECCCSASHA256 model.

Table 5. Comparison of the proposed scheme with RSA-AES model

Size of input file (KB)	RSA-AES（Zou, Ni, Huang, Shi, & Li, 2020b）		ECCCSASHA256 (Sec)	
	Encryption	Decryption	Encryption	Decryption
1.19	1.68	18.38	0.06	0.05
3.57	2.07	19.11	0.20	0.17
7.14	2.11	18.42	0.43	0.38
10.7	2.83	20.98	0.62	0.60
17.8	3.49	21.61	0.71	0.51
21.4	3.54	27.67	1.35	1.18
25	3.16	31.95	1.47	1.22
28.5	4.81	31.62	1.69	1.48
32.1	4.22	30.88	1.55	1.51
35.7	5.23	9727.	1.96	1.84
39.2	4.77	27.55	2.08	2.03
42.8	5.46	27.64	2.32	2.17
46.4	5.15	29.40	2.65	2.47
50	6.27	31.07	3.53	3.24

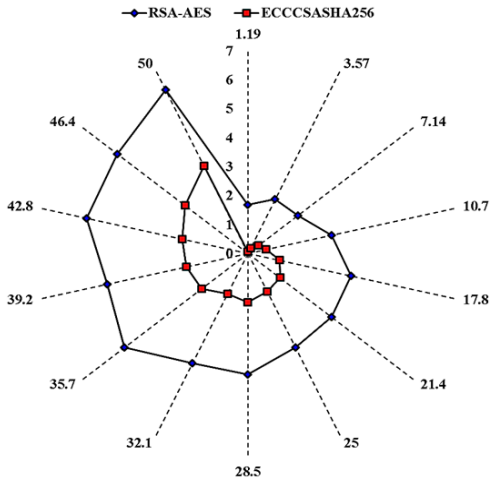

Figure 4. Encryption time of ECCCSASHA256 and RSA-AES

Figure 4 shows the cryptographic comparison chart between the ECCCSASHA256 model and the RSA-AES hybrid model. According to the obtained diagram, it can be said that the encryption time in ECCCSASHA256 model is shorter compared to RSA-AES. The distance between the two-line graphs is very large and the ECCCSASHA256 line graph with the red square is closer to the center. The numbers at the end of the lines indicate the file size.

Figure 5 compares the decoding time between the ECCCSASHA256 model and the RSA-AES hybrid model. Figure 5 shows that the decoding time in ECCCSASHA256 is shorter than in RSA-AES. The RSA algorithm requires a lot of computation in the encryption and decryption phase and its speed is very low.

Figure 6 and Figure 7 show a comparison of the ECCCSASHA256 scheme with different hybrid models based on cryptography and decryption. As shown in Figure 6 and Figure 7, it is clear that the ECCCSASHA256 scheme, even if the file

size is large, has a shorter encryption and decryption time and has been able to perform more powerfully than the other model.

In order to show the efficiency of the ECCCSASHA256 scheme, we used two hybrid models. We implemented the 3DES & ECC & SHA-256 and RC4 & ECC & SHA-256 hybrid algorithms on the current system, and the results of millisecond encryption and decryption are shown in Table 6. The purpose of combining the algorithms were to estimate the time to show that the ECCCSASHA256 model has better performance despite three different algorithms.

4.4 Encryption and decryption throughput

Throughput is one of the most important QoS (quality of service) and security parameters in cryptography schemes. The throughput of encryption and decryption are defined as input file/encryption time and input file/decryption time, respectively and can be calculated according to Eq. (14) and Eq. (15). Higher throughput indicates higher performance and strong cryptographic algorithm. If the total size of the files is 22MB, then the encryption capacity of the ECCCSASHA256 is 15.35 and the decryption capacity is 16.24, which is higher than the different models. The encryption capacity of 3DES&ECC&HA-256 is 7.13 and the decryption capacity is 7.52. The encryption throughput of the RC4&ECC&SHA-256 is 6.38 and the decryption throughput is 6.43. The encryption and decryption performance of the ECCCSASHA256 scheme has been improved by about 8.22% and 8.72% compared to 3DES&ECC&SHA-256. The encryption and decryption throughput of the ECCCSASHA256 method has been improved by about 8.97% and 9.81% compared to the RC4&ECC&SHA-256.

$$Encryption\ Throughput\ =\ \Sigma(Input\ File)\ /\ \Sigma(Encryption\ Time) \quad (14)$$

$$Decryption\ Throughput\ =\ \boldsymbol{\Sigma(Input\ File)}\ /\ \boldsymbol{\Sigma(Decryption\ Time)} \quad (15)$$

According to the results, ECCCSASHA256 model had higher performance compared to different models. Experiments of the ECCCSASHA256 model based on the number of different iterations and files of different sizes showed that the cryptography time of the ECCCSASHA256 scheme is low.

Figure 5. Decryption time of ECCCSASHA256 and RSA-AES

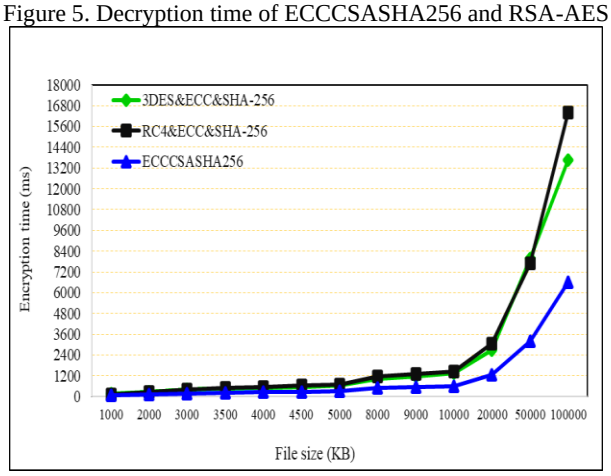

Figure 6. Encryption time of ECCCSASHA256 and other schemes

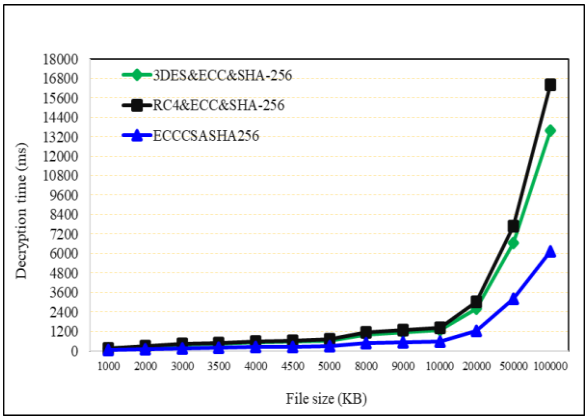

Figure 7. Decryption time of ECCCSASHA256 and other schemes

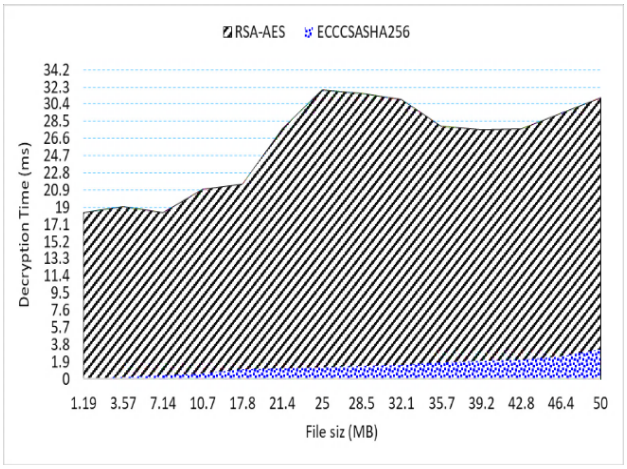

Table 6. Comparison of the proposed scheme with different model

Size of the input file (KB)	**3DES&ECC&SHA-256**(Kumar & Koti, 2021)		**RC4&ECC&SHA-256**(Kumar & Koti, 2021)		**ECCCSASHA256**	
	Encryption	**Decryption**	**Encryption**	**Decryption**	**Encryption**	**Decryption**
1000	150	135	147	145	73	56
2000	275	260	288	276	127	107
3000	335	320	433	417	190	163
3500	410	395	521	492	227	192
4000	465	450	579	566	252	229
4500	510	495	646	633	293	256
5000	745	745	721	700	319	307
8000	790	775	1179	1122	520	496
9000	865	850	1330	1278	561	539
10000	920	905	1472	1423	625	592
20000	2667	2615	3049	3013	1290	1236
50000	7968	6660	7680	7710	3219	3219
100000	13642	13592	16419	16391	6628	6148
220000	30831	29219	34464	34166	14324	13540
Throughput	7.13	7.52	6.38	6.43	**15.35**	**16.24**

5. CONCLUSION AND FUTURE WORKS

In recent years, IoT devices have been used in various applications such as healthcare, intelligent city, battlefield and smart homes with many security concerns. Lightweight security and cryptography schemes are essential for resource constrained devices of IoT. In this paper, a new security model using the EC algorithm and SHA-256 for data security of IoT-based applications is proposed. In the proposed model, the CSA was used to generate the private key in the EC algorithm. ECC and SHA-256 schemes were used for encryption and hashing the encrypted data, respectively. Combination of ECC, CSA, and SHA-256 generated a high level of data transmission security. The simulation results showed that the average throughput of decryption and encryption time in the propose scheme outperforms when compared with other models. Based on the results and analysis, the proposed ECCCSASHA256 scheme improved security of data transmission. For future work, we intend to propose different models for generating keys based on machine learning algorithms.

REFERENCES

Askarzadeh, A. (2016). A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. *Computers & structures, 169*, 1-12.

Das, S., & Namasudra, S. (2022). A Novel Hybrid Encryption Method to Secure Healthcare Data in IoT-enabled Healthcare Infrastructure. *Computers and Electrical Engineering, 101*, 107991.

Davahli, A., Shamsi, M., & Abaei, G. (2020). Hybridizing genetic algorithm and grey wolf optimizer to advance an intelligent and lightweight intrusion detection system for IoT wireless networks. *Journal of Ambient Intelligence and Humanized Computing, 11*(11), 5581-5609.

Dhanda, S. S., Singh, B., & Jindal, P. (2020). Lightweight cryptography: a solution to secure IoT. *Wireless Personal Communications, 112*(3), 1947-1980.

Dhillon, P. K., & Kalra, S. (2019). Secure and efficient ECC based SIP authentication scheme for VoIP communications in internet of things. *Multimedia Tools and Applications, 78*(16), 22199-22222.

Elhoseny, M., Shankar, K., Lakshmanaprabu, S., Maseleno, A., & Arunkumar, N. (2020). Hybrid optimization with cryptography encryption for medical image security in Internet of Things. *Neural computing and applications, 32*(15), 10979-10993.

Farah, M., Guesmi, R., Kachouri, A., & Samet, M. (2020). A new design of cryptosystem based on S-box and chaotic permutation. *Multimedia Tools and Applications, 79*(27), 19129-19150.

Fotohi, R., & Aliee, F. S. (2021). Securing communication between things using blockchain technology based on authentication and SHA-256 to improving scalability in large-scale IoT. *Computer Networks, 197*, 108331.

Jazebi, S. J., & Ghaffari, A. (2020). RISA: routing scheme for Internet of Things using shuffled frog leaping optimization algorithm. *Journal of Ambient Intelligence and Humanized Computing, 11*(10), 4273-4283.

Joglekar, J., Bhutani, S., Patel, N., & Soman, P. (2019). *Lightweight Elliptical Curve Cryptography (ECC) for Data Integrity and User Authentication in Smart Transportation IoT System.* Paper presented at the International Conference on Sustainable Communication Networks and Application.

Kamal, R., Bag, M., & Kule, M. (2021). On the cryptanalysis of S-DES using nature inspired optimization algorithms. *Evolutionary Intelligence, 14*(1), 163-173.

Kandhoul, N., & Dhurandher, S. K. (2018). *An asymmetric RSA-based security approach for opportunistic IoT.* Paper presented at the International Conference on Wireless Intelligent and Distributed Environment for Communication.

Kota, S., Padmanabhuni, V. N. R., & Budda, K. (2018). Authentication and encryption using modified elliptic curve cryptography with particle swarm optimization and cuckoo search algorithm. *Journal of The Institution of Engineers (India): Series B, 99*(4), 343-351.

Kumar, S. S., & Koti, M. S. (2021). An hybrid security framework using internet of things for healthcare system. *Network Modeling Analysis in Health Informatics and Bioinformatics, 10*(1), 1-10.

Miller, V. S. (1986). *Use of elliptic curves in cryptography.* Paper presented at the Conference on the theory and application of cryptographic techniques.

Mousavi, S. K., Ghaffari, A., Besharat, S., & Afshari, H. (2021). Improving the security of internet of things using cryptographic algorithms: a case of smart irrigation systems. *Journal of Ambient Intelligence and Humanized Computing, 12*(2), 2033-2051.

Raja Rajeshwari, K., & Ramakrishnan, M. (2022). Trivial Cryptographic Protocol for Resource-Constraint IoT Device Security Using OECC-KA. In *Mobile Computing and Sustainable Informatics* (pp. 531-538): Springer.

Reddy, M. I., & Kumar, A. (2020). An efficient data transmission approach using IAES-BE. *Cluster Computing, 23*(3), 1633-1645.

Roselin Kiruba, R., & Sree Sharmila, T. (2021). Secure data hiding by fruit fly optimization improved hybridized seeker algorithm. *Multidimensional systems and signal processing, 32*(2), 405-430.

Sadhukhan, D., Ray, S., Biswas, G., Khan, M. K., & Dasgupta, M. (2021). A lightweight remote user authentication scheme for IoT communication using elliptic curve cryptography. *The Journal of Supercomputing, 77*(2), 1114-1151.

Seyfollahi, A., & Ghaffari, A. (2020). Reliable data dissemination for the Internet of Things using Harris hawks optimization. *Peer-to-Peer Networking and Applications, 13*(6), 1886-1902.

Verma, O. P., Jain, N., & Pal, S. K. (2020). Design and analysis of an optimal ECC algorithm with effective access control mechanism for big data. *Multimedia Tools and Applications, 79*(15), 9757-9783.

Wang, Z., Dong, X., Kang, Y., & Chen, H. (2023). Parallel SHA-256 on SW26010 many-core processor for hashing of multiple messages. *The Journal of Supercomputing, 79*(2), 2332-2355.

Wei, D., Jiang, M., & Deng, Y. (2023). A secure image encryption algorithm based on hyper-chaotic and bit-level permutation. *Expert Systems with Applications, 213*, 119074.

Zou, L., Ni, M., Huang, Y., Shi, W., & Li, X. (2020a). *Hybrid encryption algorithm based on AES and RSA in file encryption.* Paper presented at the International Conference on Frontier Computing.

Zou, L., Ni, M., Huang, Y., Shi, W., & Li, X. (2020b, 2020//). *Hybrid Encryption Algorithm Based on AES and RSA in File Encryption.* Paper presented at the Frontier Computing, Singapore.