Zhang and Wang
Journal of Engineering and Applied Science (2024) 71:138
https://doi.org/10.1186/s44147-024-00459-x

Journal of Engineering and Applied Science

RESEARCH Open Access

A hybrid encryption approach for efficient and secure data transmission in IoT devices

Limin Zhang[1] and Li Wang[2*]

*Correspondence:
jsjs9750@163.com

[1] College of Electrical and Information Engineering, Hunan Institute of Traffic Engineering, Hengyang 421001, Hunan, China
[2] Hunan Institute of Traffic Engineering, Hengyang 421001, Hunan, China

Abstract

Security is a crucial concern in the Internet of Things (IoT) ecosystem. Due to IoT devices' constrained processing and storage resources, providing reliable security solutions is challenging. Encryption is one of the most commonly used techniques to secure user data against unauthorized access. Therefore, it is essential to develop encryption solutions that have minimal impact on the performance of IoT devices. This study introduces a hybrid encryption approach that combines symmetric blowfish encryption with asymmetric elliptic curves. Blowfish encryption is used to encrypt large volumes of data, which could otherwise affect the execution time.

In contrast, elliptic curve cryptography is utilized to ensure the security of the private key, which has a small size and does not increase the execution time significantly. The suggested approach provides advantages of both asymmetric and symmetric encryption methods, leading to an improvement in throughput and a reduction in execution time. The proposed approach was evaluated, yielding promising results in comparison to other cryptographic algorithms. The results show the optimization of more than 15% in the execution time and the efficiency increase by the proposed solution. This improvement represents security with the least impact on processing resources.

Keywords: Internet of Things, Security, Encryption, Symmetric, Asymmetric algorithms

Introduction

The IoT is a novel idea within the field of information and communication technology [1–3]. It is a modern technology that enables any entity (human, animal, or thing) to send and receive data through communication networks, including the internet or intranet [4, 5]. Despite the numerous benefits of IoT, it also brings forth several challenges, including new security and privacy concerns. One major issue is the confidentiality, authenticity, and integrity of IoT devices' data collected, transmitted, and processed [6–8]. As a result, IoT environments are highly susceptible to various security attacks, which can compromise their security and create insecure IoT ecosystems. Many IoT devices lack secure configuration, which poses a potential threat to their users [9]. These devices can reveal their identity in various communication channels, such as URLs, requests, and responses to and from their servers, including their brand [10]. Furthermore, it has been demonstrated that IoT devices can be targeted by DDoS attacks. Even mobile devices can be exploited by configuring them as wireless access points to intercept traffic and

collect metadata transmitted by IoT devices to infer information [11]. Cryptography has long been a fundamental element of network and internet security, particularly for safeguarding sensitive data and information from unauthorized access. Cryptography is used in traditional networks to enable secure communication and data exchange [12–14].

IoT services are classified into 3 domains: low-level, data-related, and application. Each domain has distinct security risks and therefore requires different security technologies. However, all IoT devices must address the "Big 3" security concerns, namely confidentiality, integrity, and authentication, in addition to authorization and non-repudiation. The security challenges for each IoT service domain are listed in Table 1 for reference.

The new conditions of the environment and the different characteristics of the devices make IoT security particularly important, and various architectures are being developed for it. In many environments, user data is encrypted using various algorithms to ensure security against unauthorized access [15, 16]. However, since encryption techniques often create complexity and significant overhead, solutions for encryption in IoT require low execution time and better performance, in addition to ensuring security and minimizing negative effects on the available services and performance in the IoT environment. Based on this, this research proposes an improved combination encryption solution based on the blowfish and elliptic curves encryption (ECC) algorithms. The following are the main benefits of using the suggested solution in the IoT:

- Low memory requirement
- Relatively low execution time
- High throughput of the method
- The use of the ECC algorithm is the core of the solution, whose important advantages are low complexity and the need for low energy consumption.

The article's layout is designed to give a summary of the background research in the section that follows. In Sect. "Methods", the suggested remedy is provided; in Sect. "Evaluation", it is ultimately assessed and examined.

Related works

In [17], a method to maintain the security of information based on authentication with the help of a digital signature and also breaking the file into smaller parts is presented. Tests based on different numbers of transmission packets for the proposed

Table 1 Attacks in IoT domains

Attack/Issue	Low level	Data related	Application level
DoS attack	×		×
Node Replications	×		
Camouflage	×		
TagCloning		×	
Eves dropping		×	
Injecting Packets	×	×	×

method show that this method has a good performance criterion for different packets in terms of criteria such as energy consumption and accuracy due to the use of random encryption. Also, based on the proposed method, while authenticating the users, the confidentiality service and the accuracy of the information have been properly considered in the proposed method. The paper [18] states that security must be considered and established in the IoT infrastructure and all its layers. IoT security can be divided into 4 layers: application, transmission, network, and perception. In this article, the 2 concepts of IoT and IoT security were stated, and the important discussion of security in IoT is divided into 4 parts, privacy, authentication, trust, and access control. Each of these challenges and the solutions provided have been reviewed and categorized.

The purpose of the article [19] is to transfer data securely using AES encryption. Transferring data securely between 2 working devices is challenging. There are a number of encryption algorithms, such as DES, RSA, and AES. This article provides a secure data transfer mechanism where AES encryption is used to increase security. MATLAB software has been used to implement the proposed model, and criteria such as execution time and execution speed have been used to analyze the performance of this method. In [20], the authors introduce a lightweight algorithm that aims to enhance the security of computing environments. The algorithm utilizes a 16-byte block encryption technique, incorporating Feistel network and permutation architecture to increase the complexity of the encryption process. The algorithm is adjustable, as it can be implemented with varying private key lengths and number of rounds. The results of the evaluation demonstrate that the implementation time of the algorithm is low.

However, the authors acknowledge that one potential issue with solutions based on private keys is that the cryptographic keys must be exchanged, which poses a threat to the security and confidentiality of the system. In [21], multiple encryption techniques were employed to ensure IoT and fog computing storage space security. The researchers developed an encryption system utilizing the AES algorithm and an asynchronous key transfer system for secure data and information exchange. Additionally, they implemented an elliptical bend encryption technique to exchange information between the user and the server. The evaluation results indicate that the solution achieved a relatively favorable execution time during the process of sending and receiving data, despite the small volume of data evaluated in the study. A 2-step encryption solution is described in [22] as a means to ensure the security of stored data. The solution divides the main data into 2 parts, each encrypted by a common key generated based on the chaos theory model. This approach has the added benefit of reducing the time required for encryption and decryption operations while increasing overall security. In [23], several symmetric encryption algorithms were evaluated in terms of efficiency, focusing on execution time parameters and memory consumption. The investigation results showed that the DES and blowfish algorithms were more efficient than other options in terms of both encryption and decryption time as well as memory consumption. Authors in [24] and [25] are reviews that explore the most important encryption solutions for use in IoT and fog computing environments. The results of these reviews highlight the need for algorithms and solutions to balance security and service quality parameters to

minimize the negative impact of encryption techniques on the services provided in IoT environments.

Methods

The proposed method is a comprehensive security framework incorporating various encryption-based protection techniques. This concept combines symmetric and asymmetric encryption with an improved digital signature system that protects data integrity by using the SHA-256 hashing method. Overall, this proposed approach is designed to be highly effective at providing robust security for computing environments. The essential criterion that should be considered in the implementation of solutions based on cryptographic algorithms is the computational speed of algorithms, and the proportionality between speed and performance, asymmetric algorithms (public key) computationally require more time for the key generation process and cases [26]. Their speed is therefore comparatively slower than that of symmetric key techniques like blowfish; Therefore, it is a good idea to encrypt the primary message for transmission using a symmetric key approach that has faster processing. Accordingly, the improved blowfish algorithm has been used to encrypt the original data in this solution. Figure 1 shows the general outline of the solution.

First, the original data is hashed using the SHA-256 algorithm to ensure data integrity. Afterward, the Elliptic Curves (EC) encryption algorithm is utilized to create a digital signature and protect the SHA-256 code and private key. EC is chosen due to its high-security features and small key size, which reduces the execution time of the algorithm. Finally, the improved blowfish algorithm, a symmetric algorithm with a fast execution time, encrypts the original data. Each step's specifics are further elaborated on below.

Digital signature

In this paper, a hash function has been used along with a digital signature to ensure data integrity. This suggested solution uses hash functions to first summarize the message before signing the message digest because the original digital signature model is susceptible to attacks and makes it possible for an attacker to fabricate a digital signature by changing the verification process and disrupting the network [27, 28]. Therefore, with this solution, the attacker can only create a forged digital signature that does not match the contents related to the output of the message digest function, and the attacker

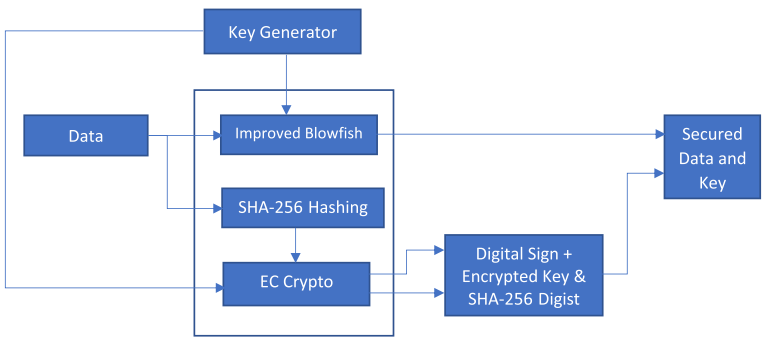

Fig. 1 Structure of the proposed method

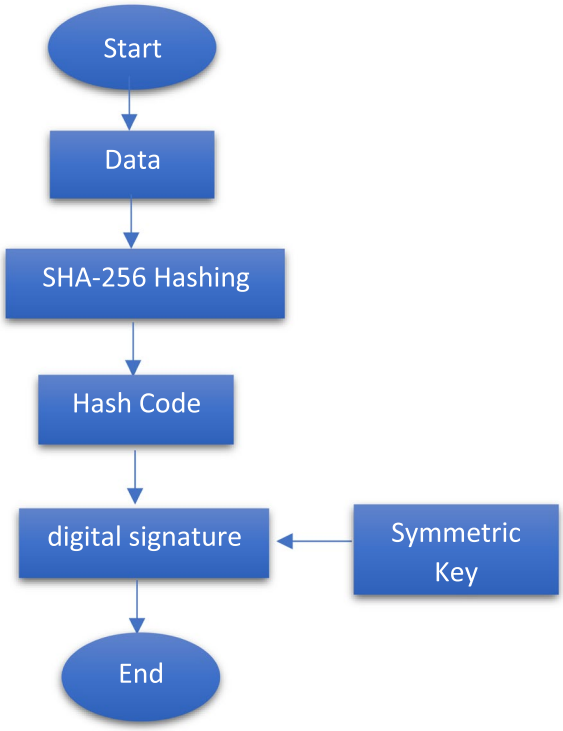

Fig. 2 Digital signature flowchart

cannot tamper with the message contents. Thus, the use of digital signatures can be a powerful tool for securing the IoT environment. The suggested solution's flowchart, which uses hash codes and digital signatures, is shown in Fig. 2. This diagram outlines the various steps involved in ensuring data integrity using the SHA-256 hashing function and digital signatures. By following this flowchart, users can effectively protect their data and ensure that it has not been tampered with or altered in any way.

Data encryption module

The main objective of this study is to present a secure encryption solution that is efficient and has a low execution time. To achieve this goal, the data encryption module employs an enhanced version of the blowfish algorithm. The algorithm has a key length that can vary from 32 to 448 bits and requires a large number of subkeys. These subkeys must be generated before the encryption or decryption of any data can take place. The blowfish algorithm is significantly faster and requires less memory than other common encryption algorithms [29]. It uses 4 S-Boxes, each with 256 inputs that are 32 bits in length. The selection process is repeated for the remaining inputs, where the first and second byte of a 32-bit input are used to select inputs in the first and second S-Boxes, respectively [30, 31]. The decryption process is similar to encryption but uses the subkeys in reverse order.

Since the F function, which includes the adder and rotation operations, is the main operation of encryption in all rounds of the blowfish algorithm and is performed in the form of a module, it takes up the most execution time of this algorithm. Therefore, this

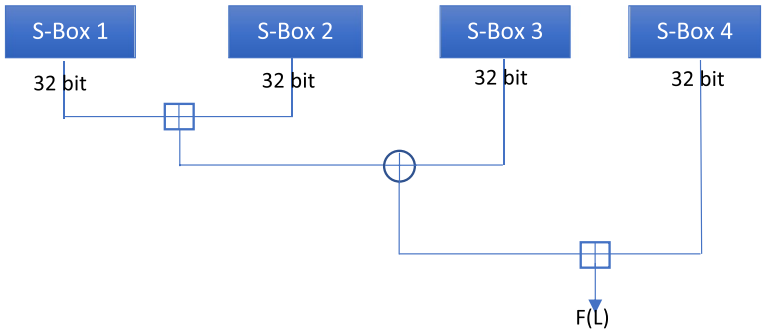

Fig. 3 Blowfish's F function module

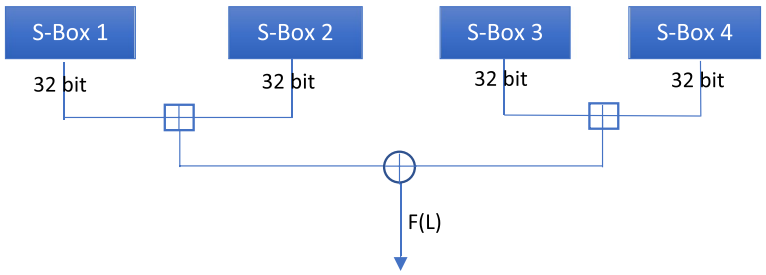

Fig. 4 Improved F-function module

research aims to reduce the execution time of the Blowfish algorithm by modifying the F module. The F module's general process in the traditional blowfish method is depicted in Fig. 3.

The optimization of an algorithm's execution time is intended to reduce its time complexity. As seen in Fig. 3, the value of function F in the standard blowfish is calculated using Eq. (1).

$$F(XL) = (S1, a + S2, bmod232)XORS3, c + (S4, dmod232) \tag{1}$$

Without compromising the security of the Blowfish algorithm, function F can be modified using Eq. (2):

$$F\prime(XL) = ((S1, a + S2, bmod232)(S3, c + S4, dmod232)) \tag{2}$$

Implementing this modification makes it possible to execute the 2 addition operations, ($S_{1,\,a} + S_{2,b}$ mod 2^{32}) and ($S_{3,c} + S_{4,d}$ mod 2^{32}) simultaneously. The amount of time needed to complete multiple operations can be decreased to only one operation with the use of parallelization. Since this technique has 16 rounds, each encryption and decryption process is completed 16 times faster. However, it should be noted that the security of the blowfish algorithm is based on the strength of its keys [32]. The modification made to the algorithm to perform the 2 addition operations in parallel does not have any negative impact on the security of the algorithm. Figure 4 also shows how this modification is done.

Additionally, Fig. 5 shows the encryption process's pseudocode for the updated blowfish technique, which makes use of the enhanced F function.

```
Input data d
Split d into two 32-bit halves: dL, dR
For k = 1 to 16:
        dL = dL XOR Pi
        dR = F(dL) XOR dR
        Swap dL , dR
Next k
dL and dR
dR = dR XOR P17
dL = dL XOR P18
Recombine dL and dR
Function F(dL)
Split dL into four 8bit quarters: a, b, c and d
F(dL) = ((S1,a + S2,b mod 2^32) XOR (S3,c + S4,d mod 2^32))
```

Fig. 5 The enhanced Blowfish method served as the basis for the encryption process's pseudocode

In the first step, the 64-bit input is divided into 2 equal 32-bit parts (left and right). The next step is to use the XOR operator, which is performed on the first part of the 32-bit block (L) and the first representation of P. Then the obtained 32-bit data is transferred to the F function so that after performing the corresponding operation, the resulting data is XORed again with the right 32-bit block (L) (which was the result of dividing the 64-bit block). The encryption process using the improved blowfish algorithm involves performing an initial key setup phase followed by multiple rounds of encryption and decryption using the modified F function. During each round, the 64-bit data is divided into 2 32-bit blocks, L and R, subjected to various operations, including XOR, modular addition, and S-Box substitution. After each round, L and R are swapped before the next round is applied. To decrypt the data, the same process is followed in reverse, using the keys in reverse order.

Secure symmetric key

As mentioned, to solve the problems of private key transmission in the blowfish algorithm, asymmetric encryption based on elliptic curves is used in this research. Elliptic curve cryptography (ECC) is a type of public key cryptography that utilizes elliptic curves in finite fields [33, 34]. One of the important capabilities of this algorithm is providing security through the use of a 164-bit key, and it has a higher efficiency compared to previous methods due to less energy consumption. Compared to other asymmetric encryption methods, it has the highest level of confidentiality, low power consumption, and requires less memory in the system [35]. One of the most important features of this encryption is to perform operations on finite fields. Cryptography based on elliptic

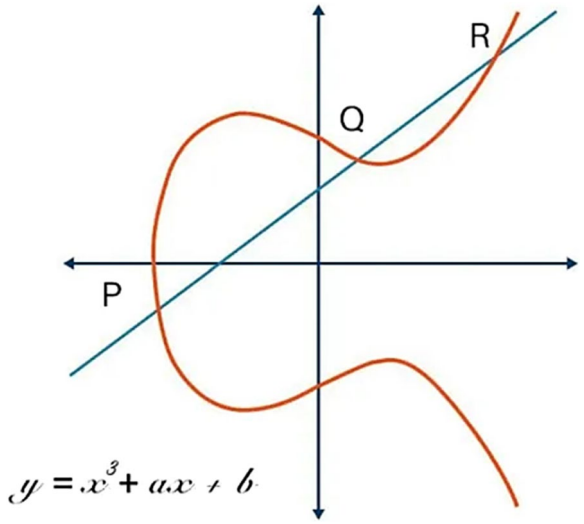

Fig. 6 Elliptical curves

curves involves solving the discrete elliptic curve logarithm problem, known as an NP-hard problem. An elliptic curve is described by an equation in Fig. 6:

$$y2 + xy = x3 + ax + b$$

The main feature of an elliptic curve is that in order to find the third point that lies on the curve, a rule can be defined to add 2 points to obtain the third point that lies on the curve. The points on an elliptic curve and the addition law on those points form a finite Abelian group. For the addition operation for 2 points to be well defined, to define an elliptic curve, a special point at infinity (denoted as O) is added to the curve, which is not on the curve itself but is treated as a point. This point serves as the additive identity element, which means that any point added to O will result in the same point. The number of points on an elliptic curve is called its order, which includes the point at infinity [8].

Fixed curve selection

A huge prime number p is chosen, and a finite field GF(p) is created using a set of integers fewer than p in order to construct elliptic curve cryptography. As a result, by removing the term xy from Eq. (3) and the condition $4a^3+27b^2\neq 0$, the elliptic curve will be equal to:

$$y2 = x3 + ax2 + b \tag{3}$$

In order to encrypt based on elliptic curves, first, a random number like k is selected in the range [1,n-1]; it should be noted that this range belongs to the field, and the random number in this range is assumed as the private key will be next, the public key R is calculated as $R=F_k$, F is a point on the ellipse curve. In this case, the calculation of k according to points Q and F has exponential time complexity. To enhance security, a fixed curve and point are selected such that the order of the fixed point F is a large

prime number. This value is determined using Schoff's algorithm based on the order of the curve. Using elliptic curve cryptography provides smaller public keys and signatures compared to asymmetric algorithms such as RSA while maintaining the same level of security. This is because if the fixed-point order of F is an n-bit prime number, calculating k from k_F and F requires approximately $2^{(n/2)}$ operations.

Encrypt using ECC

Suppose 3 users, Alice, Bob, Cathy, and David, have agreed on an elliptic curve (non-confidential) and a fixed curve point (non-confidential) f. To send a message to Bob, Alice must first obtain Bob's public key. Once she has it, she can encrypt the message using Bob's public key and send the ciphertext to Bob. In elliptic curve cryptography, Bob's public key is a curve point Bp on the same curve Alice and Bob agreed upon. Bob has also chosen a secret integer Bk, and his public key is calculated as Bp = BkF, where F is the fixed point on the curve. To encrypt the message, Alice generates a random number, r, and calculates the point R = rF on the curve. She then calculates the ciphertext C = M + rBp, where M is the plaintext message. She sends the pair (R, C) to Bob as the ciphertext. Bob can then use his secret key Bk to recover the plaintext message. He calculates the point S = rBp = rBkF and subtracts it from C to obtain M = C—S = M + rBp—rBkF. In this case, Alice first calculates A_kB_p and uses its result as a key for encryption. Bob can also determine the same number by calculating B_kA_p. Because:

$$B_kA_p = B_k.(A_kF) = A_k.(B_kF) = A_kB_p \tag{4}$$

The method's security relies on the presumption that the computation of k in relation to F and kF will be exceedingly challenging.

Due to the fact that the operation in this method is based on the field; as a result, the use of scaled multiplication algorithms can greatly help to improve the efficiency of encryption based on elliptic curves. Accordingly, the general steps of the method will be as follows:

- **Step 1**: The MD-5 hash algorithm is used to create a hash code for the original material before it is encrypted. This is done to enhance the digital signature's effectiveness and verify the data's integrity.
- **Step 2**: Utilizing the data's digital signature and encrypted private key, the hash code is created in the preceding phase.
- **Step 3**: Elliptic curve encryption is used to encrypt the blowfish algorithm's private key.
- **Step 4**: Prior to transmitting the data, the intended information is encrypted using a symmetric encryption algorithm (Blowfish). In fact, at this stage, the symmetric key is used for the encryption operation and to generate the encrypted data string from the original data. Then the encrypted data is sent along with the encrypted strings in the previous step.
- **Step 5**: Applying the reverse process on the receiver side; In this case, the received information is decrypted using the private key.

- **Step 6**: The blowfish algorithm's private key is used to decode the original data. Then, the review and validation process is performed with the help of the hashing function (digital signature).

Evaluation

The implementation of the proposed solution has been done in the Eclipse software environment and through the use of the Java development kit version 7. In the Java development kit, encryption functions are possible with the help of 2 main libraries, JCA and JCE. JCA functions are fully integrated with the core Java APIs and are able to provide most of the basic cryptographic capabilities. JCE is also used to provide advanced encryption operations. Also, the hardware used has an Intel Core2 Duo processor with a frequency of 2.5 GHz. All evaluations have been performed in the Windows 7 environment. In order to evaluate, the proposed method with AES, 3DES, DES, RSA, PRESENT, and ECDH cryptographic algorithms [19, 20] in artificial criteria, including execution time, throughput, and the amount of memory consumed, have been compared. At first, the solutions with a small data size (50 KB) were evaluated in order to be able to see and check how they work when the data size is small. Then with a data size of 1024 KB (1 MB) and 2048 kilobytes (2 MB), the solutions have been compared and evaluated.

Evaluation criteria

A set of criteria is required to gauge the efficacy of the solution, and the effectiveness of the solution may then be assessed in accordance with these criteria. For this purpose, the following criteria have been selected.

Execution time: It is equal to the time required to perform encryption operations. The execution time is computed using the following relationship:

$$Executiontime = EncryptStarttime - endtime \tag{5}$$

In the above relation, the execution time is done through a System timestamp, which is actually a timer in the Java language, so that it will start with the start of the encryption operation and will stop at the end of it. Subtracting these 2 times from each other actually determines the time of implementing the solution.

Throughput: It is equal to the number of tasks that have been processed and encrypted per unit of time. Also, in order to calculate the throughput of solutions, the following relationship is used.

$$Throughtput = \frac{DataSize}{ExecutionTime} \tag{6}$$

In the above relationship, throughput is obtained by dividing the evaluated data size by the required execution time calculated through the previous relationship.

Results and discussion

The first evaluation is based on the data size of 50 kilobytes, as seen in Fig. 7. The execution time compared to RSA, AES, DES, 3DES, and PRESENT algorithms is 800,

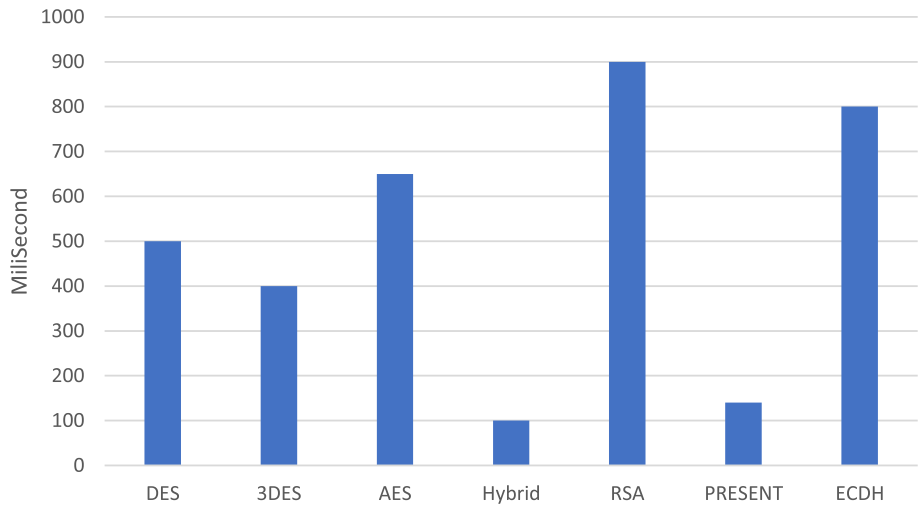

Fig. 7 Execution time—milliseconds (data size: 50 KB)

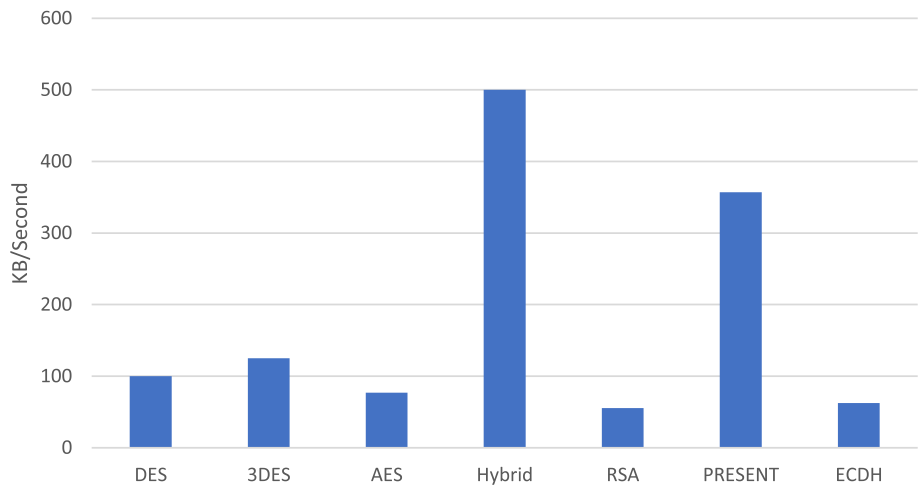

Fig. 8 Throughput—kilobytes per second (data size: 50 kilobytes)

550, 300, 400, and 800 respectively. The milliseconds have decreased, which means that encryption operations have been performed in less time. This sum results from a decrease in execution time brought about by the development and use of the blowfish algorithm for primary data encryption. Blowfish is one of the fastest cryptography methods, considering that it is in the category of fragment codes. Also, through the change made in the F function section, its speed and efficiency have increased. Considering that EC asymmetric algorithm is used in the proposed solution only for key encryption and hash code summary, as a result, it did not have much effect on the execution time.

In Fig. 8, the solutions are compared in terms of throughput in the data size of 50 KB. As demonstrated in this assessment, the proposed method exhibits higher throughput than alternative solutions. Accordingly, compared to RSA, AES, 3DES, DES, and ECDH algorithms, the optimality rate has reached 100%. Given that the implementation time

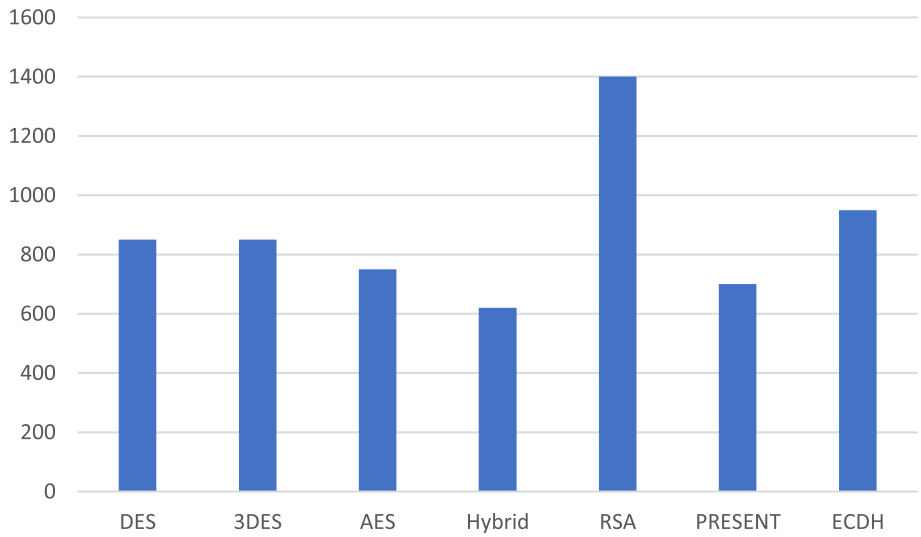

Fig. 9 Execution time—milliseconds (data size: 1024 KB)

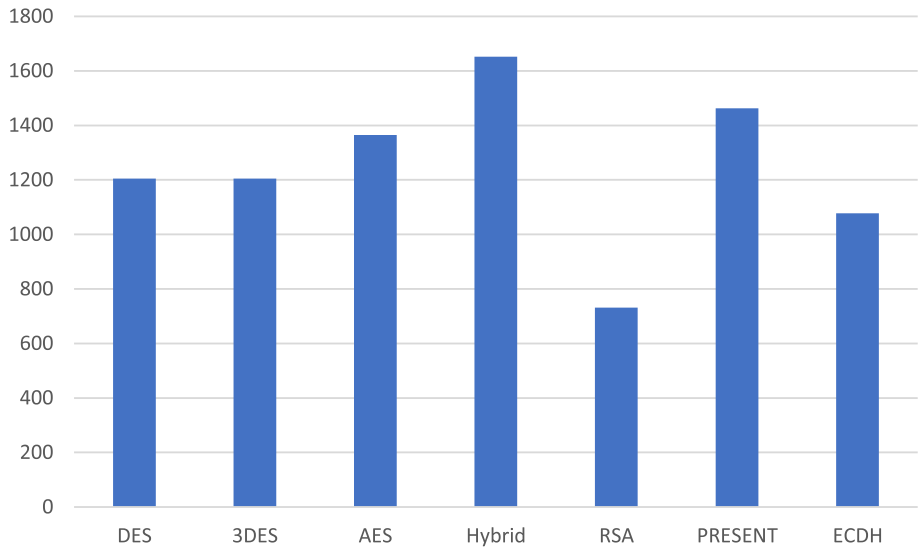

Fig. 10 Throughput—kilobytes/second (data size: 1024 kilobytes)

for this solution was very low in the evaluation, it was expected to achieve this level of throughput.

Due to the size of the encrypted data being potentially effective during the execution of the operation, in the second experiment, the data size was increased to 1 megabyte (1024 kilobytes) to compare the efficiency of the algorithms at this volume of data. As can be seen in Fig. 9, the proposed solution has performed better than other algorithms. The execution time has been significantly reduced, with reductions of 240, 240, 130, 80, and 330 ms compared to the DES, 3DES, AES, PRESENT, and ECDH algorithms, which indicates that the proposed solution works faster than these

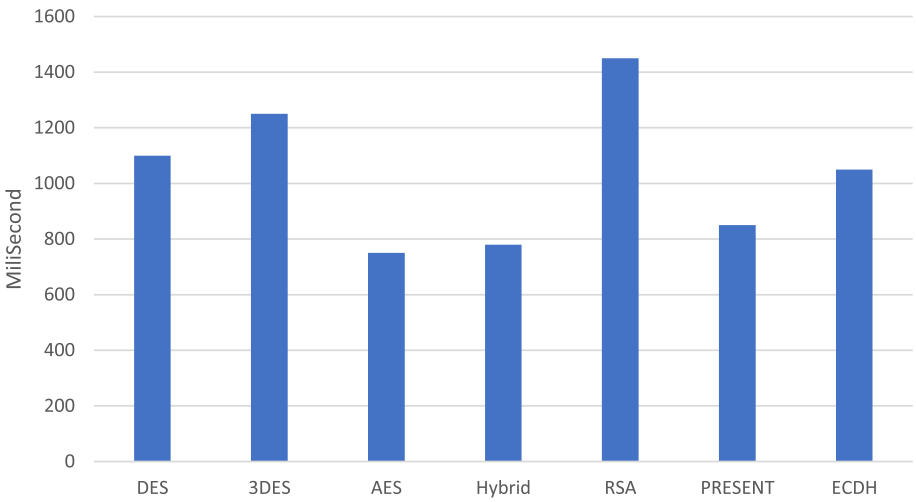

Fig. 11 Execution time—milliseconds (data size: 2048 KB)

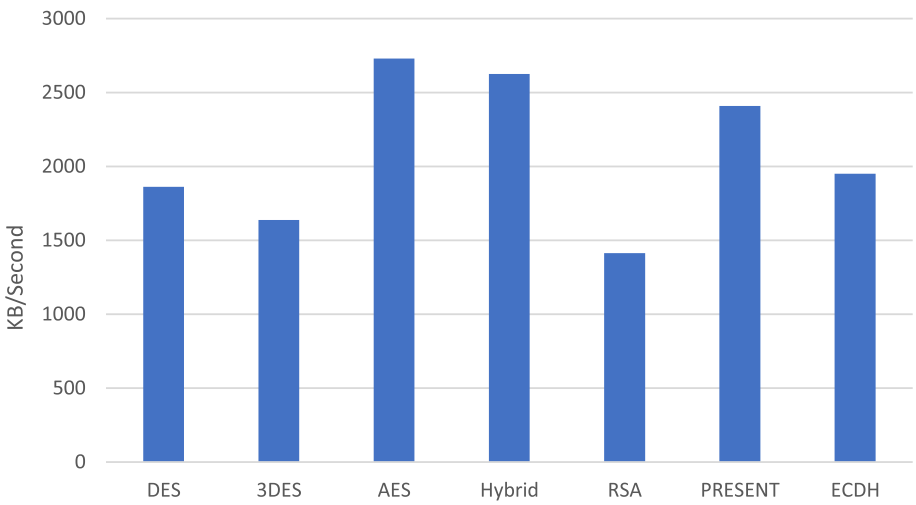

Fig. 12 Throughput—kilobytes per second (data size: 2048 kilobytes)

3 solutions. Also, compared to the RSA asymmetric algorithm, the execution time has been reduced by more than 731 ms, indicating the solution's faster implementation.

Figure 10 also shows the throughput rate of the solutions, which due to the lower execution time, it was predictable that the throughput rate would be higher. Accordingly, the throughput is more than 16% on average compared to AES, 3DES, and DES. It has also reached more than 50% compared to the RSA asymmetric algorithm. In the presented solution, the most execution time is related to the encryption of the main data, which is done by the algorithm. As a result of parallelization and improvement of this part of the solution, it has greatly contributed to reducing the execution time and, consequently, increasing the throughput.

Finally, in the last test, the desired algorithms were evaluated with a data volume of 2048 kilobytes (2 megabytes), which Figs. 11 and 12 clearly show that the proposed solution is more optimal. As can be seen, as the amount of data increases, it affects the

duration of encryption and causes more time to be needed. However, in this evaluation, the proposed solution was able to complete the operation in less time than DES, 3Des, and RSA algorithms. Accordingly, the execution time compared to the 2 symmetric algorithms 3DES and DES is reduced between 470 and 320 ms, respectively, and on the other hand, the throughput is much more optimized, which can be seen in (11) and (12). In the case of the asymmetric RSA algorithm, the execution time has decreased by more than 670 ms. However, in this experiment, compared to the symmetric AES algorithm, no optimization has been achieved, and considering that the proposed solution offers other capabilities, such as digital signature and hashing, compared to the standard AES algorithm, this amount of increase in execution time can be invisible and will not have much impact on the efficiency of the IoT infrastructure.

As seen in Fig. 12, with the increase in data size, the throughput of the solution has also increased, which is due to changes in the F function and the parallelization done in the blowfish encryption core. Its throughput may be much higher than other algorithms. In addition, due to the fact that in the proposed freezing method, data encryption is separated from key encryption; as a result, the simultaneous use of blowfish and EC has not had much effect on performance loss and execution time. Because in this case, only the private key and the hash code are encrypted by EC, and due to their very low volume, it has the least impact on the execution time. On the other hand, the main data, which is quite sizable, is encrypted using enhanced blowfish cryptography, which, due to its high efficiency, execution time, and throughput, is much higher than 3DES, DES, and RSA algorithms.

Memory consumption

They consider the amount of memory consumed by encryption algorithms also important. As a result, in this section, the amount of memory consumption of the proposed

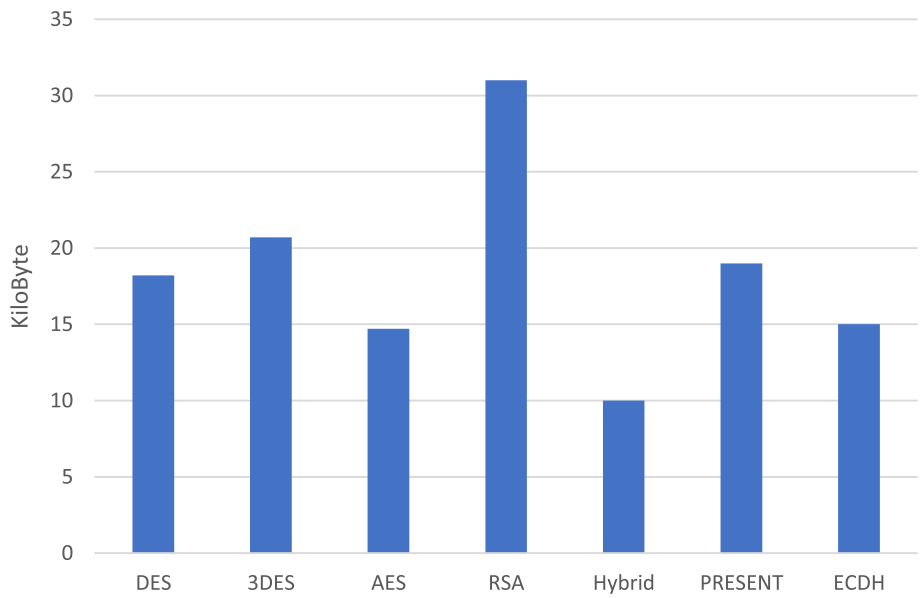

Fig. 13 The amount of memory consumed by solutions

Table 2 Memory consumption in compared methods

Algorithm	Memory consume
DES	18.2
3DES	20.7
AES	14.7
RSA	31
Hybrid	**10**
PRESENT	19
ECDH	15

Table 3 Cryptanalysis for ECC illustrating complexity of attack algorithm

Algorithm	Time complexity	Space complexity
Brute-Force	$O(n) = O(2^{192})$	O(l)
Baby-Step, Giant-Step	O(l)	$O(\sqrt{n}) = O(2^{192/2})$
Pollard's Rho	$O(\sqrt{n}) = O(2^{192/2})$	O(l)

method has been evaluated and investigated in comparison with other common encryption solutions. Next, the results of the evaluation are shown in Fig. 13. In this test, the data size used for evaluation is considered equal to 50 kilobytes.

As seen in Fig. 13 and Table 2, the proposed method has the lowest amount of memory consumption. The reason for this high level of efficiency is the optimization of the cryptographic core, which is improved based on the blowfish algorithm, and this has increased throughput and reduced memory consumption because, in normal conditions, the blowfish algorithm is one of the most optimal algorithms in the field of memory occupation and execution time. Also, the EC algorithm is one of the most optimal asymmetric algorithms in the field of memory consumption due to the use of a 164-bit key. This amount of low memory requirement does not cause any problem in using the proposed method in old infrastructures, but it can also be used in the smallest embedded systems.

Table 3 summarizes the robustness of various algorithms. Assuming the key size to be *N*, this is described as what is the hardness in the algorithm. These comparisons are based on the time complexity needed by a breaking attack. A row in this table illustrates the required key length for an equivalent security strength. Actual key sizes used are 128, 192, 256 for AES, and 80, 128 for Hybrid and PRESENT algorithms.

While the proposed method employs the ECC algorithm for secure private key exchange, Table 3 presents cryptanalysis for ECC, illustrating the complexity of the attack algorithm [21].

As seen in Table 3, the time complexity of the ECC is very high, which can make this algorithm and solutions based on it safe against various attacks, including Brute-Force, Baby-Step, and Pollard's Rho.

Conclusion

In this research, an optimal encryption solution is proposed to enhance security in the IoT computing infrastructure while improving efficiency. Since symmetric encryption forms the basis of the solution, blowfish has been enhanced; thus, the execution time has dropped and the throughput has increased as a consequence of the encryption process taking less time. On the other hand, using EC asymmetric encryption to secure the key makes the key-sending operation without security challenges related to symmetric algorithms. Also, a digital signature based on SHA-256 was employed to ensure the integrity of the data. This allows for continuous verification and confirmation of the data's completeness during the transmission and receipt. In the evaluation section, the performance of encryption compared to other common encryption algorithms (AES, DES, 3DES, and RSA) was investigated. Because the higher the speed and throughput in this part, which is the core of the operation, the overall performance of the solution will be higher in the IoT infrastructure. The evaluations conducted with various data sizes demonstrate the solution's optimality with regard to execution time, throughput, and memory usage. In the end, it should be mentioned, although it was shown during the evaluations that the proposed solution has a significant effect on improving the efficiency in addition to increasing the security in the IoT infrastructure. Due to the use of symmetric and asymmetric encryption, there are some problems related to these algorithms, the most important of which are as follows:

- The encryption process requires larger key sizes to achieve the best level of security.
- Since blowfish encryption requires the same key for both encryption and decryption, it is difficult to scale it to large numbers of IoT devices.

Abbreviations

IoT	Internet of Things
EC	Elliptic Curves
DES	Data Encryption Standard
AES	Advanced Encryption Standard
SHA	Secure Hashing Algorithm
MD-5	Message-Digest-5
JCE	Java Cryptographic Extensions
JCA	Java Cryptography Architecture

Acknowledgements

I would like to take this opportunity to acknowledge that there are no individuals or organizations that require acknowledgment for their contributions to this work.

Authors' contributions

Limin Zhang:Presented methodology and validated the formal analysis and was a major contributor to writing the manuscript. Li Wang, Writing-Original draft preparation, Conceptualization, Supervision, and project administration. All authors read and approved the final manuscript.

Funding

No Funding.

Availability of data and materials

Data will be available on request.

Declarations

Ethics approval and consent to participate
Disclosure of potential conflicts of interest: The authors declare no competing interests.
Research involving Human Participants and/or Animals: The observational study conducted on medical staff needs no ethical code. Therefore, the above study was not required to acquire an ethical code.
Informed consent: This option is not necessary due to that the data were collected from the references.

Consent for publication
This option is not necessary due to that the data were collected from the references.

Competing interests
The authors declare no competing interests.

Received: 20 August 2023 Accepted: 25 May 2024
Published online: 20 June 2024

References

1. Rana M, Mamun Q, Islam R (2022) Lightweight cryptography in IoT networks: A survey. Futur Gener Comput Syst 129:77–89
2. Prasanalakshmi B, Murugan K, Srinivasan K, Shridevi S, Shamsudheen S, Hu YC (2022) Improved authentication and computation of medical data transmission in the secure IoT using hyperelliptic curve cryptography. J Supercomput 78(1):361–378
3. Windarta S, Suryadi S, Ramli K, Pranggono B, Gunawan TS (2022) Lightweight Cryptographic Hash Functions: Design Trends, Comparative Study, and Future Directions. IEEE Access 10:82272–82294
4. Kumar V, Malik N, Singla J, Jhanjhi NZ, Amsaad F, Razaque A (2022) Lightweight authentication scheme for smart home iot devices. Cryptography 6(3):37
5. Ahmed AA, Malebary SJ, Ali W, Alzahrani AA (2023) A Provable Secure Cybersecurity Mechanism Based on Combination of Lightweight Cryptography and Authentication for Internet of Things. Mathematics 11(1):220
6. Zhang L, Hu S, Trik M, Liang S, Li D (2024) M2M communication performance for a noisy channel based on latency-aware source-based LTE network measurements. Alexandria Eng J 99:47–63
7. Liao Y, Tang Z, Gao K, Trik M (2024) Optimization of resources in intelligent electronic health systems based on Internet of Things to predict heart diseases via artificial neural network. Heliyon.
8. Panahi U, Bayılmış C (2023) Enabling secure data transmission for wireless sensor networks-based IoT applications. Ain Shams Engineering Journal 14(2):101866
9. Khezri E, Zeinali E, Sargolzaey H (2023) SGHRP: Secure Greedy Highway Routing Protocol with authentication and increased privacy in vehicular ad hoc networks. PLoS ONE 18(4):e0282031
10. Trik M, Akhavan H, Bidgoli AM, Molk AMNG, Vashani H, Mozaffari SP (2023) A new adaptive selection strategy for reducing latency in networks on chip. Integration 89:9–24
11. Ghafourian E, Samadifam F, Fadavian H, Jerfi Canatalay P, Tajally A, Channumsin S (2023) An ensemble model for the diagnosis of brain tumors through MRIs. Diagnostics 13(3):561
12. Trik, M., Molk, A. M. N. G., Ghasemi, F., & Pouryeganeh, P. (2022). A hybrid selection strategy based on traffic analysis for improving performance in networks on chip. J Sensors
13. Samiei M, Hassani A, Sarspy S, Komari IE, Trik M, Hassanpour F (2023) Classification of skin cancer stages using an AHP fuzzy technique within the context of big data healthcare. J Cancer Res Clin Oncol 149(11):8743–8757
14. Sun J, Zhang Y, Trik M (2024) PBPHS: a profile-based predictive handover strategy for 5G networks. Cybern Syst 55(5):1041–1062
15. Trik M, Mozaffari SP, Bidgoli AM (2021) Providing an adaptive routing along with a hybrid selection strategy to increase efficiency in NoC-based neuromorphic systems. Comput Intell Neurosci. 2021:8338903
16. Wang Z, Jin Z, Yang Z, Zhao W, Trik M (2023) Increasing efficiency for routing in the Internet of Things using binary gray wolf optimization and fuzzy logic. Journal of King Saud University-Computer and Information Sciences 35(9):101732
17. Wang G, Wu J, Trik M (2023) A novel approach to reduce video traffic based on understanding user demand and D2D communication in 5G networks. IETE J Res 46(12):1–17
18. Fakhri PS, Asghari O, Sarspy S, Marand MB, Moshaver P, Trik M (2023) A fuzzy decision-making system for video tracking with multiple objects in non-stationary conditions. Heliyon. 9(11):e22156
19. Khosravi M, Trik M, Ansari A (2024) Diagnosis and classification of disturbances in the power distribution network by phasor measurement unit based on fuzzy intelligent system. The Journal of Engineering 2024(1):e12322
20. Li Y, Wang H, Trik M (2024) Design and simulation of a new current mirror circuit with low power consumption and high performance and output impedance. Analog Integr Circuits Signal Process. 119(25):1–13
21. Trick M, Boukani B (2014) Placement algorithms and logic on logic (LOL) 3D integration. Journal of mathematics and computer science 8(2):128–136
22. Trik M, Boukani B, Ansari B, Emtiyaz S, Azar SR, Mohammadi F (2014) An Overview of through-silicon–based 3-dimensional integrated circuits (3D IC) to placement to optimize timing. Research Journal of Recent Sciences, ISSN 2277:2502
23. Mokhlesi Ghanevati D, Khorami E, Boukani B, Trik M (2020) Improve replica placement in content distribution networks with a hybrid technique. Journal of Advances in Computer Research 11(1):87–99
24. Khezri E, Zeinali E, Sargolzaey H (2022) A novel highway routing protocol in vehicular ad hoc networks using VMaSC-LTE and DBA-MAC protocols. Wireless Commun Mobile Comput 11(2):2022

25. Khalafi M, Boob D. (2023). Accelerated primal-dual methods for convex-strongly-concave saddle point problems. In International Conference on Machine Learning. PMLR 202:16250–16270
26. Xiao L, Cao Y, Gai Y, Khezri E, Liu J, Yang M (2023) Recognizing sports activities from video frames using deformable convolution and adaptive multiscale features. Journal of Cloud Computing 12(1):167
27. Khezri E, Yahya RO, Hassanzadeh H, Mohaidat M, Ahmadi S, Trik M (2024) DLJSF: Data-Locality Aware Job Scheduling IoT tasks in fog-cloud computing environments. Results in Engineering 21:101780
28. Ding X, Yao R, Khezri E (2023) An efficient algorithm for optimal route node sensing in smart tourism Urban traffic based on priority constraints. Wireless Netw. 16(10):1–18
29. Zhu J, Hu C, Khezri E, Ghazali MMM (2024) Edge intelligence-assisted animation design with large models: a survey. J Cloud Computing. 13(1):48
30. Hosseini A, Rahaeifard M, Mojahedi M (2020) Analytical and numerical investigations of the ultrasonic microprobe considering size effects. Mech Adv Mater Struct 27(24):2043–2051
31. Karabulut E, Gholizadeh F, Akhavan-Tabatabaei R (2022) The value of adaptive menu sizes in peer-to-peer platforms. Transportation Research Part C: Emerging Technologies 145:103948
32. Saidabad MY, Hassanzadeh H, Ebrahimi SHS, Khezri E, Rahimi MR, Trik M (2024) An efficient approach for multi-label classification based on an Advanced Kernel-Based Learning System. Intelligent Systems with Applications 21:200332
33. Behzad Behbahani A, Lages WS, Kelliher A (2019) A Multisensory Design Probe: An Approach for Reducing Technostress. In Proc Thirteenth Int Conf Tangible Embedded Embodied Interact. 10(9):459–466
34. Goyal TK, Sahula V, Kumawat D (2022) Energy efficient lightweight cryptography algorithms for IoT devices. IETE J Res 68(3):1722–1735
35. Goyal TK, Sahula V (2019) Lightweight security algorithm for low-power IoT devices. International conference on advances in computing, communications, and informatics (ICACCI). IEEE, Jaipur, pp 1725–1729

Publisher's Note