Hindawi
Complexity
Volume 2021, Article ID 5540296, 13 pages
https://doi.org/10.1155/2021/5540296

Research Article

Lightweight Cryptographic Algorithms for Guessing Attack Protection in Complex Internet of Things Applications

Mohammad Kamrul Hasan,[1] Muhammad Shafiq,[2] Shayla Islam,[3] Bishwajeet Pandey,[4] Yousef A. Baker El-Ebiary,[5] Nazmus Shaker Nafi,[6] R. Ciro Rodriguez,[7] and Doris Esenarro Vargas[8]

[1] *Center form Cyber Security, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), 43600 Bangi, Selangor, Malaysia*
[2] *Cyberspace Institute of Advanced Technology, Guanghzou University, Gaungzhou, China*
[3] *Institute of Computer Science and Digital Innovation, UCSI University, 56000 Kuala Lumpur, Malaysia*
[4] *Department of Computer Science and Engineering, Birla Institute of Applied Science, Bhimtal, India*
[5] *Faculty of Informatics and Computing, University Sultan Zainal Abidin (UniSZA), Kuala Terengganu, Malaysia*
[6] *School of IT and Telecommunication Engineering, Melbourne Institute of Technology, Melbourne, Australia*
[7] *School of Software Engineering, National University Mayor de San Marcos, Lima, Peru*
[8] *Universidad Nacional Federico Villarreal UNFV(INERN), Lima, Peru*

Correspondence should be addressed to Mohammad Kamrul Hasan; hasankamrul@ieee.org, Shayla Islam; shayla@ucsiuniversity.edu.my, and Yousef A. Baker El-Ebiary; yousefelebiary@unisza.edu.my

Received 16 January 2021; Revised 3 March 2021; Accepted 10 March 2021; Published 5 April 2021

Academic Editor: Atif Khan

As the world keeps advancing, the need for automated interconnected devices has started to gain significance; to cater to the condition, a new concept Internet of Things (IoT) has been introduced that revolves around smart devices conception. These smart devices using IoT can communicate with each other through a network to attain particular objectives, i.e., automation and intelligent decision making. IoT has enabled the users to divide their household burden with machines as these complex machines look after the environment variables and control their behavior accordingly. As evident, these machines use sensors to collect vital information, which is then the complexity analyzed at a computational node that then smartly controls these devices operational behaviors. Deep learning-based guessing attack protection algorithms have been enhancing IoT security; however, it still has a critical challenge for the complex industries' IoT networks. One of the crucial aspects of such systems is the need to have a significant training time for processing a large dataset from the network s previous flow of data. Traditional deep learning approaches include decision trees, logistic regression, and support vector machines. However, it is essential to note that this convenience comes with a price that involves security vulnerabilities as IoT networks are prone to be interfered with by hackers who can access the sensor/communication data and later utilize it for malicious purposes. This paper presents the experimental study of cryptographic algorithms to classify the types of encryption algorithms into the asymmetric and asymmetric encryption algorithm. It presents a deep analysis of AES, DES, 3DES, RSA, and Blowfish based on timing complexity, size, encryption, and decryption performances. It has been assessed in terms of the guessing attack in real-time deep learning complex IoT applications. The assessment has been done using the simulation approach and it has been tested the speed of encryption and decryption of the selected encryption algorithms. For each encryption and decryption, the tests executed the same encryption using the same plaintext for five separate times, and the average time is compared. The key size used for each encryption algorithm is the maximum bytes the cipher can allow. To the comparison, the average time required to compute the algorithm by the three devices is used. For the experimental test, a set of plaintexts is used in the simulation—password-sized text and paragraph-sized text—that achieves target fair results compared to the existing algorithms in real-time deep learning networks for IoT applications.

1. Introduction

Our way of life changes with the continuous scientific developments in society, where life is now heavily driven by data. The advancements in semiconductor and communication technologies have led multiple devices to be interconnected to deliver communications and services to humans. This phenomenon is often referred to as the Internet of Everything (IoE) that includes the IoT as its subset. The IoE can be applied in various fields such as smart cities, smart homes, intelligent transportations, automated agriculture, and convenient healthcare (Figure 1). The IoE often suffers from its computation limitations in processing capabilities and fixed storage, leading to the lack of device safety, privacy, and performance [1–6]. Considering the ubiquitous application of IoE in our society, it is imperative to improve their security and performance Fig 1.

In the IoE/IoT domains' physical layer, the MAC layer and the physical layer control the security procedures mainly in GPRS applications, sensors, or RFID. IEEE 802.15.4. is used because of its low-cost and low-energy-consumption rates, but it sustains some limitations against the potential attacks. The network layer collects the data from the physical layer to partition a message into a bundle and to route the data packets from the source to the destination. With the rapid rise of IoT, IPv6 address loses its precedence to IPv4. AES, DES, or Inbuilt cryptography conventions are realizable by utilizing the IPsec in this layer. User Datagram Protocol (UDP) is used in IoT for end-to-end communication at the transport layer. However, Datagram Transport Layer Security (DTLS) is constructed in this layer because UDP is not reliable. The application layer is where the intelligence of IoT resides. The application layer can be used for social action, retail, wellbeing, or personal needs. Constrained Application Protocol (CoAP) [7] is employed to satisfy the IoT network's low resource restriction.

IoT faces challenges in security assurance, data reliability, and user confidentiality in its edge network. Furthermore, some of the challenges remain in IoT edge networks because of the lack of a mechanism to perform authorization, key management, authentication, and access control. Moreover, because the compelled edge devices interface with the Internet, fortifying the edge system is essential for the global IoT/IoE network. In an IoT wireless sensor network, there is much literature that explores the security vulnerabilities that cause attacks in eavesdropping, reply attack DoS/DDoS, and so on. Many applications can lose our private information on banking, health, and location services due to these security constraints. A security measure is required to secure communication in which the interception of messages by malicious users cannot harm our privacy [3, 8].

This work's main contribution is the experimental assessments on the technologies and cryptographic algorithms that can be used in the messages exchanged between the nodes to create a secure IoT network in a way that protects our communication. This article will conduct a comparative study of RSA, DES, AES, 3DES, and Blowfish encryption algorithms to protect the Internet of Things (IoT) applications. The experimental analysis includes the comparison of computational resources required versus the security improvement. The study can lead us to find an optimal tradeoff point between computational resources versus security performance in future IoT/IoE applications.

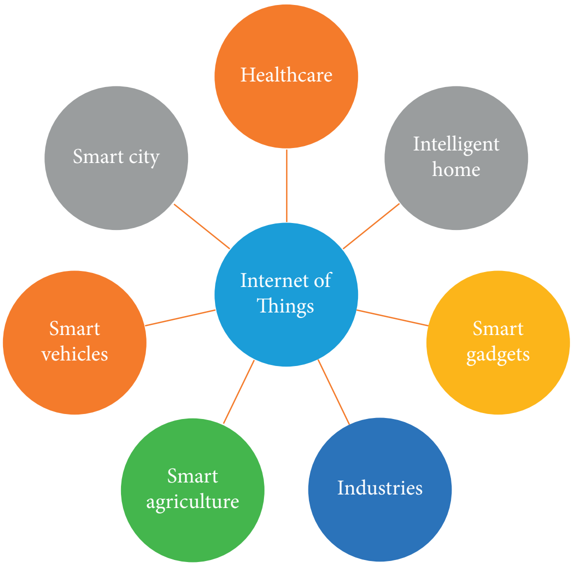

Figure 1: Various IoT applications.

2. Cryptography and Encryption Algorithms and Its Challenges

This section will provide the related works in data encryption for IoT applications. Literature shows studies on power consumption, processing speed, packet size, data types, and avalanche effect in data encryption for IoT applications.

As per Gartner report (Stamford 2013), IoT can bring forth more than a three hundred billion US dollar revenue in 2020, excluding smartphones, tablets, and PCs. In addition, by 2020, amounts of smartphones and tablets reached over 7.3 billion units. For a large number of data communication over the network, a complex and massive network will be created. Many internet-based applications have been introduced, such as online shopping, instant payment, and electronic bill payment. Other than web applications, several new concepts are emerging in Cryptocurrency, Blockchain, and the Internet of Things (IoT).

In an IoT environment, the demand for using the appropriate cryptographic solution is increasing. Nevertheless, because of the limited battery life, low power computation, small memory, limited power supply, and small size of the edge devices suffer limitations in applying cryptography. A typical cryptographic primitive may not be suitable for these low-powered edge devices. For instance, an RFID tag cannot employ a 1204-bit RSA algorithm due to a lack of resources [9]. The current smart industry requires an intelligent cryptographic solution that can provide adequate security performance in pervasive computing and only resource-limited edge devices.

The classification of different encryption algorithms is illustrated in Figure 2 [10].

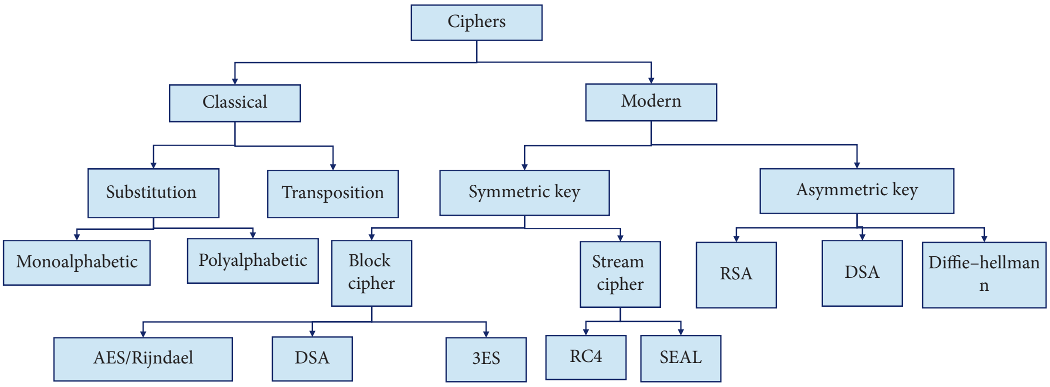

Figure 2: Classification of different encryption algorithms [10].

2.1. *Cryptography.*

Cryptography is a technique used to transform data or messages into an unreadable format. It protects unauthorized users from sending messages. As shown in Figure 3, two processes are associated with this—encryption and decryption [11]. They are used for protecting messages or data from fraud attacks on the network. Security of data is a significant issue in the Cloud IoT environment. Cryptography is addressed in some ways. Three types of cryptographic algorithms are as follows:

(i) Symmetric cryptography

(ii) Asymmetric password

(iii) Hash encryption

2.1.1. *Symmetric Cryptography.*

Symmetric cryptography (i.e., secret key cryptography) refers to cryptography that employs the same encryption key for plaintext encryption and decryption. The same key is shared between the two sides, which is a significant disadvantage of symmetric key encryption [12]. Compared with public key encryption (aka asymmetric key encryption shown in Figure 4), the main advantages of symmetric key encryption are that it does not consume too much energy, and the encryption speed is breakneck. It is divided into two categories: block ciphers and stream ciphers. Advanced Encryption Standard (AES), Data Encryption Standard (DES), Blowfish, Triple DES, etc., are some algorithms of standard symmetric key employed in cloud computing [11] as shown in Figure 5.

(1) Types of the symmetric algorithms: block cipher and stream cipher. In the block cipher, the secret message of any length is transformed into fixed blocks, and if the message length is smaller than the block size, then zero paddings are done. Next, on each block, an encryption algorithm and key are applied to generate the cipher message. The most preferred algorithms are DES, AES, Blowfish. Next, based on the algorithm's structure, a block cipher is classified into two types: substitution-permutation network (SPN) and Feistel network (FN). In the SPN network on the whole block, substitution and permutation layer is applied to generate the ciphertext, as shown in Figure 6. The plaintext and key XOR is done in the initial stage. Next, the XOR output is passed through s-box and p-layer. After that, on the fly is key generated for each round.

On the other side, in the Feistel network, the block is divided into two halves. Next, the functions f_1, f_2, and f_3 and key are applied to one-half of the block. Then the swapping function is functional with the repetitive process for all rounds, as shown in Figure 7 [13].

In the stream cipher, the cipher is generated by combining the message with the key using a simple transformation (e.g., XOR) as shown in Figure 8.

In the stream cipher, the block size is 1 bit long, and the algorithm's overall security depends on the key size (which is generated using key and initialization vector (IV)). The most preferred stream cipher is RC4 in the SSL/TLS and A5 in the GSM. Typically, stream cipher algorithms are fast, require fewer resources for encryption purposes, and also are preferred for encrypting the small message. Further, block cipher can be turned into stream cipher using various modes of operations, e.g., counter mode. This means that if you have a secure block cipher, you can build a fast stream cipher. Comparative analysis between the block and stream cipher is shown in Table 1.

Some examples of popular and well-respected symmetric algorithms include AES (aka Rijndael), Blowfish, DES, TDES, and IDEA, as an example of the symmetric algorithm (Figures 9–11).

DES: Data Encryption Standard (DES) is a symmetric key block cipher where the key length is 56 bits, and the block size is 64 bits in length [14]. When a weak key is used, it is vulnerable to key attacks. DES was discovered by IBM in 1972 using an algorithm for data encryption. It is approved by the US government as a standard algorithm for encryption. It starts with a 64-bit key, and then, the NSA limits the consumption of DES with a 56-bit key. Thus, DES removes 8 bits of the 64-bit key and subsequently utilizes a reduced 56-bit key obtained from the 64-bit key to 64-bit block size encrypted data. DES can function in different modes—CBC, ECB, CFB, and OFB, rendering it flexible. When a weak key is used, it is vulnerable to key attacks. In 1998, the

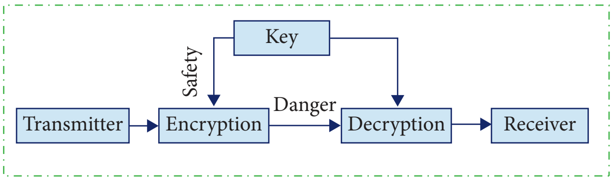

Figure 3: Encryption and decryption communication model [11].

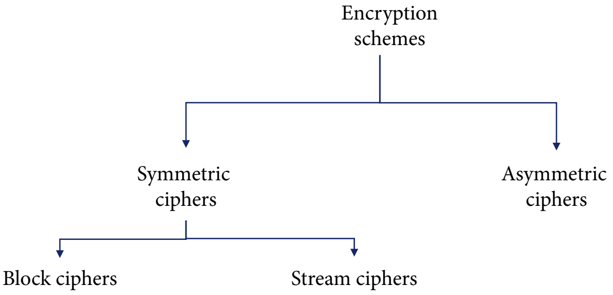

Figure 4: Type of encryption schemes.

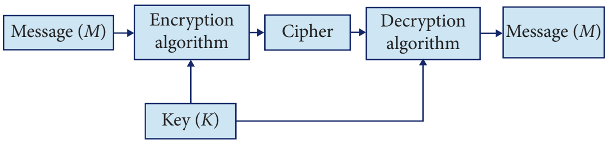

Figure 5: Block diagram of the symmetric algorithm.

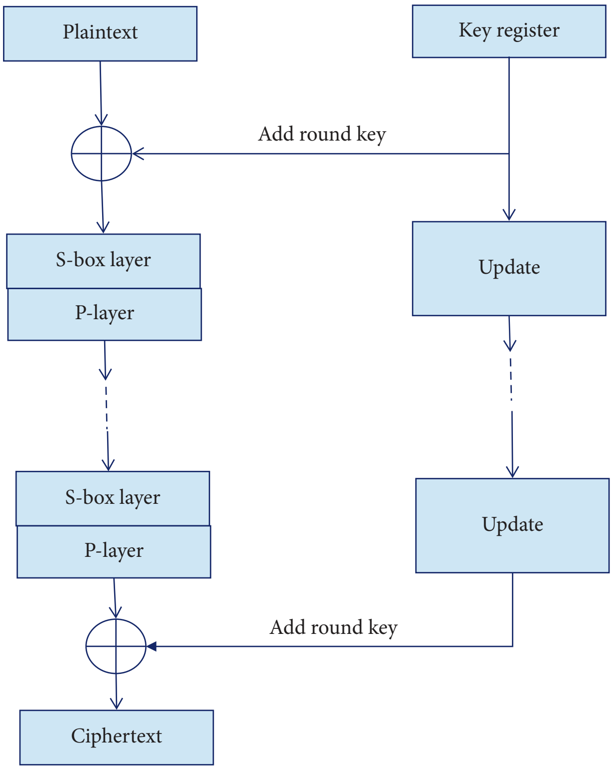

Figure 6: Substitution-permutation network.

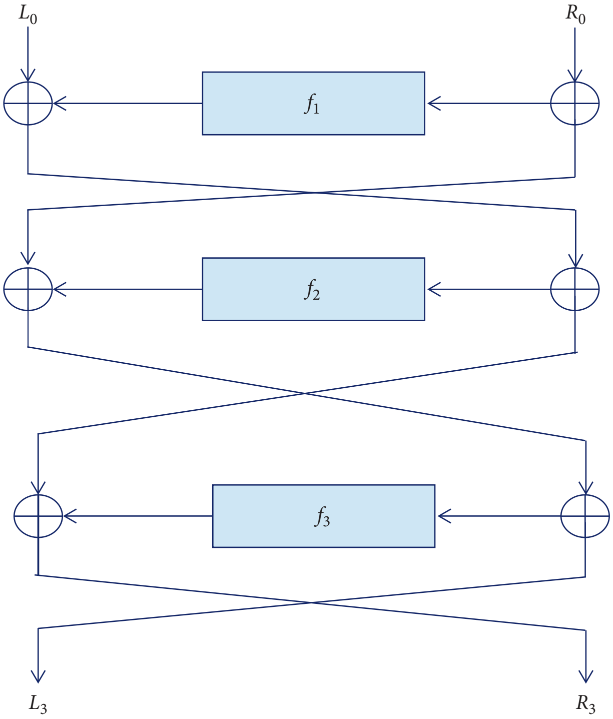

Figure 7: Feistel network [13].

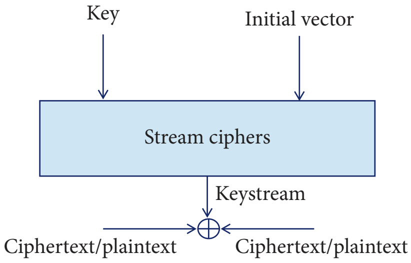

Figure 8: Block diagram of block cipher.

supercomputer DES cracking program cracked DES in 22 hours with hundreds of thousands of distributed PCs on the Internet.

Triple DES: Triple DES, a block cipher, is called a triple data encryption algorithm and cryptography. In 1998, the Triple Data Encryption Standard (3DES) was initially released. Hence, its name is like that because it uses three DES ciphers to each block of data, namely, “encryption and decryption—using DES encryption,” as shown in Figure 8. The key length is 112 bits or 168 bits, and the block size is 64 bits in length. As the computing power available today continues to increase, and the original DES ciphers’ capabilities are weak, it has suffered brute-force attacks and several crypt-analysis attacks. For providing a comparatively simple way of increasing the key size of DES, Triple DES aims to prevent such attacks with no design of a different block cipher algorithm. The encryption function used is $C = E\ (K_1,\ E\ (K_2,\ D\ (K_3,\ C)))$ and by using the same

Table 1: Comparative analysis between block cipher and stream cipher.

Parameter	Block cipher	Stream cipher
Block size	Large (128 bit)	Small (1 bit)
Key size	Fixed	Variable
Number of rounds	Large	—
Resource consumption	Large	Small
Robustness against error	Small	High
Application	To encrypt the bulk message	To encrypt the small message
Zero padding	Required	Not required
Examples	DES, AES, Blowfish	RC4, Grain, Trivium

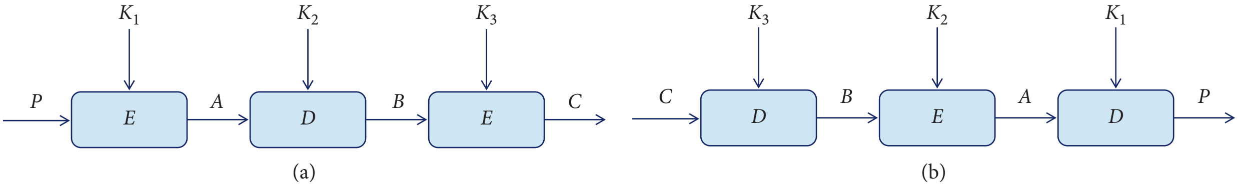

Figure 9: Structure of Triple DES: encryption (a) and decryption (b) [12, 15].

operation with keys reverse produces the decryption function of $P = D\ (K_1, E\ (K_2, D\ (K_3, C)))$. The 3DES is a formidable algorithm because DEA is an underlying cryptographic algorithm. Thus, similar resistance to cryptanalysis of DES can be claimed for 3DES. Besides, the 168-bit key length makes brute-force attacks effectively impossible.

International Data Encryption Algorithm (IDEA): International Data Encrypt Xuejia Lai and James L. Massey of ETH minor revision of an earlier cipher, PES (Proposed Encryption Standard); IDEA was originally called IPES (Improved PES early versions of the Pretty Good Privacy cryptosystem. K can be represented as TDES-EDE, which shows the structure of the triple DES. Encryption algorithm (IDEA) is a block cipher designed by ETH-Zürich and was first described in 1991. Based on the previous section's study, it can conclude that there are differences between the DES algorithm, Triple DES algorithm, and IDEA (shown in Figure 9). Table 2 shows the comparison between DES, Triple DES, and IDEA.

The AES algorithm is a symmetric key block cipher which is established by Joan Daemen and Vincent Rijmen in 1998. The AES algorithm strengthens any amalgamation of data with key lengths of 128, 192, and 256 bits, as shown in Figure 10. AES allows 128-bit data length, which can be divided into four basic operation units. These units are regarded as byte arrays and arranged into a matrix with 4×4, also known as states, and undergo various transformations through rounds. For full encryption, for cases where the key lengths are 128, 192, and 256, the number of rounds used is the variable $N = 10$, 12, and 14. Each round of AES utilizes permutation and replacement networks and is fitting for hardware and software implementations.

Blowfish: Blowfish was originally released in 1993. It is a symmetric key block cipher with key lengths ranging from 32 bits to 448 bits and a block size of 64 bits. Its composition is the Festival Network. As a symmetric block cipher, Blowfish can be exploited as a casual alternative to DES or IDEA. It uses a variable-length key ranging from 32 bit to 448 bit, making it ideal for home and business use. Devised by Bruce Schneier, Blowfish is a speedy, free complementary of prevailing encryption algorithms. Since then, it has been extensively investigated, and as a robust encryption algorithm, it is gradually gaining popularity. Blowfish is not patented, has a free license, and is free for all uses. The process of the Blowfish encryption algorithm is shown in Figure 11.

Lightweight cryptography has been essential for the last few years, driven by the lack of primitives capable of running on devices with deficient computing power [22]. One of the most ciphers in lightweight cryptography is the PRESENT algorithm.

(2) PRESENT algorithm: the PRESENT algorithm is an asymmetric cryptography algorithm that is based on the substitution permutation network. The PRESENT algorithm has a block size of 64 bits and supports two key sizes 80 bit and 128 bit and required 32 rounds for the data encryption. In the initial phase, the plaintext and key XOR operation is performed which transforms the original bits. Next, the XOR operation output is given to the substitution layer, which transforms the actual bits. The 64 bits are processed in the 4-bit chunk. Thus, $2^4 = 16$ combination is required in the look-up table for the s-box, as shown in Table 3. The authors do not disclose the s-box mathematical modeling. Therefore, the algorithm is secure and preferred in the number of applications. Next, s-box output-input to the permutation layer shuffles the bits at bit level as shown in Table 4.

The PRESENT algorithm's permutation layer consumes a large number of cycles due to the bit-level permutation. Further, a layer of key scheduling is performed in the round

Table 2: Comparison between DES, Triple DES, and IDEA.

	DES	Triple DES	IDEA
Key size	56 bits	112 (2TDES) or 168 bits (3TDES)	128 bits
Block size	64 bits	64 bits	64 bits
Structure	Feistel network	Feistel network	Substitution-permutation network
Round used	15	48	8.5

as shown in Figure 12. The PRESENT algorithm key scheduling is most preferred in the other lightweight ciphers due to the more straightforward key scheduling step.

Therefore, other lightweight cipher algorithms are explored in work, providing better security and consuming less permutation and encryption purposes. Next, in Table 5, a comparative analysis of various conventional and lightweight cryptography algorithms is done.

The comparative analysis found that, in conventional cryptography, AES is the most recommended NIST algorithm and preferred in several applications such as e-commerce, social media, and Internet banking. On the other side, in the lightweight algorithm, NIST recommended the PRESENT algorithm. Even up to now, no benchmark algorithm is proposed which are used for validating the lightweight algorithm. Due to the PRESENT algorithm's popularity, we have studied the PRESENT algorithm and found a large number of cycles for encryption.

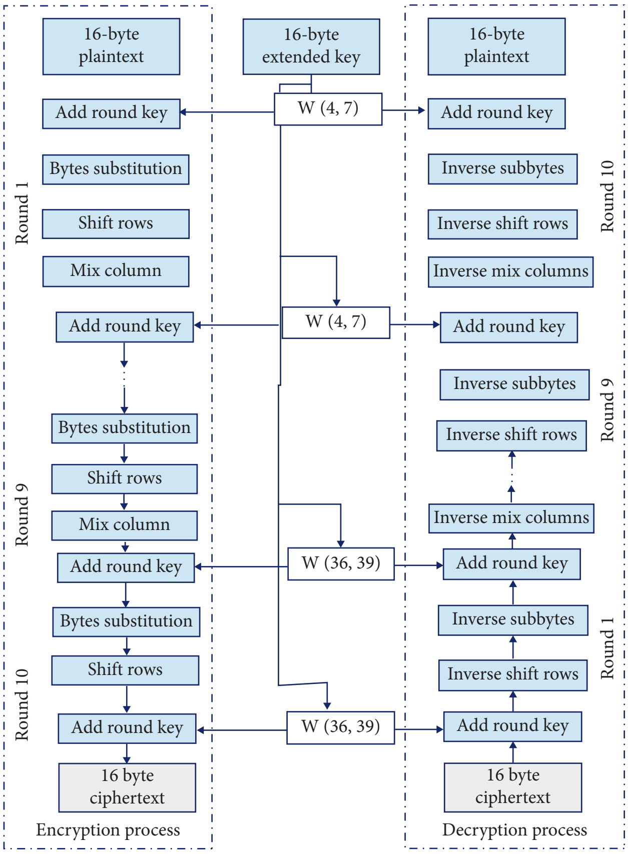

Figure 10: Structure of AES [11, 14–20].

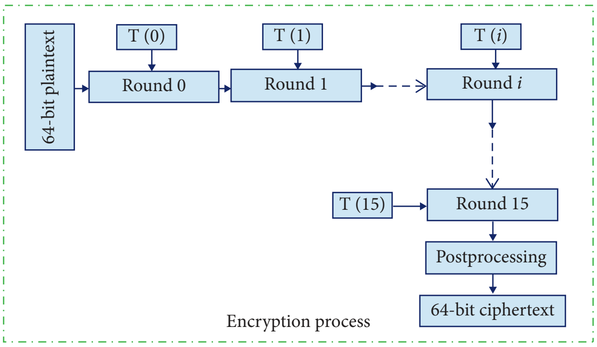

Figure 11: Blowfish encryption algorithm [12–15, 17, 21].

2.1.2. *Asymmetric Password.*

Asymmetric key algorithms (secret key algorithms) use different keys for plaintext encryption and ciphertext decryption. It consists of two keys: a private key and a public key. The public key is used to encrypt the sender everyone knows, and the private key is used for the decryption of the confidential receiver [12]. Unlike symmetric ciphers, which share different keys, this is one of the main advantages of asymmetric ciphers. However, the main disadvantage of asymmetric encryption is that it consumes too much energy, and it is not as fast as symmetric encryption. Some popular asymmetric key algorithms used in cloud computing are Rivest–Shamir–Adleman (RSA), Elliptic Curve Cryptography (ECC) [11], as shown in Figure 13.

(1) RSA. Established in 1977, RSA is a public key cryptosystem. RSA is an asymmetric cryptographic algorithm named after its founder Rivest, Shamir, and Adelman. It produces two keys: a public key to encrypt and a private key to decrypt messages. There are three steps in the RSA algorithm. At the first step, the key generation is performed, which is operated as a key to encrypt and decrypt data. The next step is to encrypt, where the actual process is performing conversion from plaintext to ciphertext. Finally, the third step is to decrypt. At this step, the encrypted text is translated to plaintext on the other end. RSA is established on the problem of retrieving the product of two large prime numbers. 1024 to 4096 bits are found for the key size. To secure the key on the Internet, the original key and public key are given to the RSA algorithm, which generates the encrypted key in the output [23]. The detailed description of the RSA algorithm key creation, encryption, and decryption is shown in Table 6.

2.1.3. *Hash Encryption.*

Hash is a numerical function that transforms any type of data into distinctive string bits. Any form or extent of data can be hashed. A unidirectional

Table 3: S-box for the PRESENT algorithm.

Input	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S (input)	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

Table 4: P-box for the PRESENT algorithm.

Input	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
P (input)	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51
Input	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
S (input)	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55
Input	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
S (input)	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59
Input	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
S (input)	12	28	44	60	13	29	45	61	14	30	46	62	15	31	47	63

$$[K_{79}K_{78}\text{.........}K_1K_0] = [K_{18}K_{17}\text{.........}K_{20}K_{19}]$$

$$[K_{79}K_{78}K_{77}K_{76}] = S[K_{79}K_{78}K_{77}K_{76}]$$

$$[K_{19}K_{18}K_{17}K_{16}K_{15}] = [K_{19}K_{18}K_{17}K_{16}K_{15}] \text{ XOR round\_counter}$$

Figure 12: Key scheduling.

Table 5: Comparative analysis of conventional and lightweight cryptography algorithms.

Algorithm	Block size (in bits)	Key size (in bits)	No. of rounds	Structure	Year published
Conventional algorithms					
DES Wong (Wark and Dawson, 1998)	64	56	16	FN	1975
3DES (Patil et al. 2016)	64	168/112/56	48	FN	1998
AES (Daemen and Rijmen, 2013)	128	128/192/256	10/12/14	SPN	2001
Blowfish (Singh and Singh, 2013)	64	32-448	16	FN	1993
Lightweight algorithms					
PRESENT (Bogdanov et al. 2007)	64	80/128	32	SPN	2007
Clefia (Shirai et al. 2007)	128	128/192/256	18/22/26	FN	2007
HIGHT (Hong et al. 2006)	64	128	32	FN	2016
RECTANGLE (Zhang et al. 2015)	64	80/128	25	SPN	2015
PICO (Bansod et al. 2016)	64	80/128	32	SPN	2016
BORON (Bansod et al. 2017)	64	80/128	25	SPN	2017

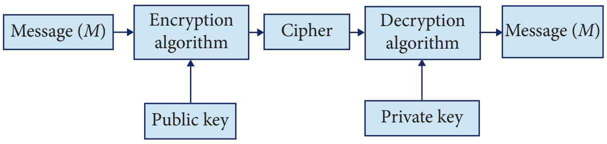

Figure 13: Block diagram of asymmetric cryptography.

Table 6: RSA key creation, encryption, and decryption.

Sender	Receiver
Key creation	
Choose two secret prime numbers p and q. Choose encryption exponent e with GCD $(e, (p-1)\ (q-1)) = 1$. Publish $N = pq$, and e.	
Encryption	
	Choose plaintext m. Use Bob s public key (N, e) to compute $c = m^e$ mod N. Send ciphertext c to Bob
Decryption	
Compute d satisfying Ed $= 1$(mod $(p-1)\ (q-1)$) Complexity compute $m' = c^d$ mod N Then m' equals the plaintext m.	

process puts data into a hash algorithm and gets a unique text string. Hash functions are fundamental tools in modern cryptography. In hash encryption, the identical message continually results in an equal hash value. It can also quickly calculate the hash value of any delivered message [24]. Also, minor changes to the message will adjust the hash value. It is not possible to get the identical hash value for two separate messages. Secure Hash Functions (SHA-1 and SHA-256) and Message Digests (MD5) [25] are some of the popular hash encryption technologies employed in cloud computing, as shown in Figure 14.

2.1.4. Comparative Time Complexity Analysis of the Symmetric, Asymmetric, and Hash Cryptography. In this section, based on the previous study, a comparative analysis between symmetric, asymmetric, and hash cryptography is presented. Table 7 shows that symmetric algorithm is faster in encryption compared to asymmetric or hash function. Thus, the symmetric algorithm is often preferred in steganography.

3. Related Work

This section examines earlier work in data encryption in terms of power consumption, processing speed, data type, throughput, avalanche effect, and packet size. This section reviews the literature in cryptography algorithms [18–20, 26–43]. Internet of Things (IoT) has made it imperative to have several devices on a network. Most of these are computers, but there are sensors, digital tools, and vehicles. The large size of the network of devices and anonymous or uncontrolled Internet structure is imperative to consider. Protecting data and communication systems is essential in IoT security [26–43]. The studies have used machine learning-based neural network algorithms to solve security issues. Some other authors have been studied the performance of various security algorithms on a single processor and cloud networks for different input sizes [18–20, 26–35, 41].

The purpose of this article is to get quantitative terms such as speedup ratios that help to implement secure algorithms (MD5, RSA, and AES) using cloud resources, which companies can use for encrypting considerable amounts of data. Three distinct algorithms are utilized—AES (symmetric encryption algorithm), RSA (asymmetric encryption algorithm), and MD5 (hash algorithm) [40–43]. Results stated in this article determine that algorithms realized in a cloud environment (i.e., Google App) are more effective than applying them on a single system. For single-processor (on-premises) and cloud (Appengine) environments, MD5 consumes the least time, whereas RSA consumes the most. In the case of low input file sizes, the highest speedup can be obtained in AES, and as the input file size increases, the speedup ratio drops dramatically. AES is the highest in speeding up, followed by MD5, and RSA has the lowest speedup in the case of each input size.

Three algorithms are compared and analyzed. RSA, AES, and DES consider specific parameters such as calculation time, output bytes, and memory usage. These parameters are the main concerns in any sort of encryption algorithm [17]. Experimental findings demonstrate that, in the case of AES and DES algorithms, the DES algorithm consumes the least encryption time. In contrast, the AES algorithm uses the least memory, and the difference in encryption time is small. RSA uses the lengthiest encryption time while the memory usage is also high, but in the RSA algorithm, the output bytes are minimal.

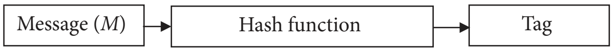

FIGURE 14: Block diagram of hash cryptography.

The performance of symmetric encryption algorithms is studied. This article presents an assessment of the six most popular encryption algorithms: 3DES, AES (Rijndael), DES, RC2, RC6, and Blowfish [18]. Comparisons have been made for each algorithm under different settings, such as different data block sizes, various data types, battery power use, various key sizes, and final encryption/decryption speed. The investigational simulation demonstrates the following results. When the results are displayed in hexadecimal base encoding or base 64 encodings, there is no significant difference [18]. In the case of altering the packet size, it is observed that RC6 takes lesser time than other algorithms except Blowfish. In the case of changing data types (e.g., images in place of text), RC2, RC6, and Blowfish were found to be disadvantageous in terms of time consumption over other algorithms. Moreover, compared to the algorithm DES, the performance of 3DES is still very low. Lastly, in the case of altering the key size (only feasible in RC6 and AES algorithms), it can be observed that larger key sizes can cause significant changes in battery and time consumption.

To evaluate the performance of various cryptographic algorithms, we applied various cryptographic algorithms to encrypt video files. We calculated the encryption and decryption time for various video file formats (including .vob and .DAT) with the file size ranging from 1 MB to 1100 MB. The results show that the AES algorithm performs adequately with less processing time than DES but more time than Blowfish [20]. More in-depth analysis is presented in the following section.

4. Performance Analysis with Result and Discussion

4.1. Symmetric Cryptographic Algorithm. Symmetric cryptography is the most widely used and most frequently used encryption algorithm today. It is used in the software industry, but it is also in the hardware industry [10, 22, 23, 25–32, 44, 45]. When various infrastructures are involved in security requirements, symmetric encryption algorithms are given priority. For most symmetric cryptographic algorithms, the encryption and decryption processes are reversed. The features are as follows:

(a) Low execution time, fast encryption speed, high encryption efficiency: however, both parties use the

Table 7: Comparative analysis of symmetric, asymmetric, and hash cryptography based on the various parameters [25, 26].

Parameters	Symmetric	Asymmetric	Hash
Block size	Fixed	—	Variable
Key	Required	Required	Required
Resource consumption	Low	High	Low
Computation time complexity	Low	High	Low
Application	Encryption	To create secure channel/authentication	Authentication

same key, and security is not guaranteed. There are two types of symmetric ciphers: stream ciphers and block ciphers, but block ciphers are now commonly used:

Block cipher-working mode:

ECB: electronic codebook
CBC: ciphertext link
CFB: ciphertext feedback
OFB: output feedback
CTR: counter
Block password filling method
No padding
PKCS5 padding
ISO10126 padding

(b) Comparisons: Table 8 presents the comparison of the various symmetric cryptographic algorithms.

Table 8: Comparison of the lightweight algorithms.

Algorithm	Key length	Default key length	Operating mode	Cipher algorithm padding
DES	56	56	ECB, CBC, PCBC, CTR, CTS, CFB, CFB8-CFB128, OFB, OFB8-OFB128	No padding, PKCS5 padding, ISO10126 padding
3DES	112,168	168	ECB, CBC, PCBC, CTR, CTS, CFB, CFB8-CFB128, OFB, OFB8-OFB128	No padding, PKCS5 padding, ISO10126 padding
AES	128,192,256	128	ECB, CBC, PCBC, CTR, CTS, CFB, CFB8-CFB128, OFB, OFB8-OFB128	No padding, PKCS5 padding, ISO10126 padding
Blowfish	32,448	64	ECB, CBC	

For a fair comparison, a common C# language was used to test the encryption methods. We present our testing of symmetric encryptions using DES, 3DES, and AES/Blowfish in the following section.

4.1.1. Symmetric Cryptographic Algorithm Simulation. To evaluate the symmetric cryptography algorithms' efficiency, a simulation has been conducted on 3 separate computers. The experiments used C# running on Microsoft .NET Framework.

Table 9 shows the details of the devices used in the simulation.

The simulation tests the speed of encryption and decryption of the selected encryption algorithms. For each encryption and decryption, the tests will execute the same encryption using the same plaintext for 5 separate times, and the average time is compared. The key size used for each encryption algorithm is the maximum bytes the cipher can allow. To make a fair comparison, the average time required to compute the algorithm by the 3 devices is used.

A set of plaintexts are used in the simulation—password-sized text and paragraph-sized text—which would give a fair comparison between the algorithms in real-time deep learning networks for IoT.

Password-sized plaintext:

K86a1uZEJ

Paragraph-sized plaintext:

In the tree, there was something. From the ground, it was difficult to tell but rachael could see movement. Her eyes were squinted, and peered towards the movement, trying to decipher exactly what she had spied. With the increase of her peering, she increasingly thought it might be a figment of her imagination. Anything seemed not to move until she started to take her eyes off the tree. Then, in the corner of her eye, she would find the movement once again and start staring again. Headphones were on. They had been used on intention. She could listen to her mother shouting in the background, but could not make out exactly what the shouting was about. So, she had put them on.

Table 10 shows the simulation result on device 1, and Table 11 presents the simulation results on device 2, Table 12 summarizes the simulation results on device 3, and device 4 results are presented in Table 13.

The results show that the AES performs at a much faster rate in both encryption and decryption. This was more prominent in encrypting and decrypting a larger size plaintext. Intel's proprietary hardware acceleration can explain AES's fast encryption rate for AES–AES-NI [15]. This fast encryption speed makes the encryption algorithm the

Table 9: Simulation configurations.

	Processor	Number of logical processors	Number of cores	Frequency (MHz)
Device 1 details	Intel Core i5–8250U	8	4	1800
Device 2 details	Intel Core i5–7200U	4	2	2500
Device 3 details	Intel Core i5–7200U	4	2	2700

Table 10: Simulation results on device 1.

	DES	Triple DES	AES	Blowfish
Encryption (paragraph-sized)				
Time (milliseconds)	2.40	1.02	0.06	1.17
Key size (bytes)	8	24	32	56
Ciphertext size (characters)	1016	1016	1024	1504
Decryption (paragraph-sized)				
Time (milliseconds)	0.306	0.024	0.017	0.446
Encryption (password-sized)				
Time (milliseconds)	0.179	0.032	0.036	0.175
Key size (bytes)	8	24	32	56
Ciphertext size (characters)	24	24	24	48
Decryption (password-sized)				
Time (milliseconds)	0.214	0.019	0.020	0.292

Table 11: Simulation results on device 2.

	DES	Triple DES	AES	Blowfish
Encryption (paragraph-sized)				
Time (milliseconds)	4.0	1.277	0.162	1.523
Key size (bytes)	8	24	32	56
Ciphertext size (characters)	1016	1016	1024	1504
Decryption (paragraph-sized)				
Time (milliseconds)	0.773	0.0236	0.0256	0.561
Encryption (password-sized)				
Time (milliseconds)	0.961	0.0209	0.0253	0.174
Key size (bytes)	8	24	32	56
Ciphertext size (characters)	24	24	24	48
Decryption (password-sized)				
Time (milliseconds)	1.249	0.0183	0.0277	0.188

Table 12: Simulation results on device 3.

	DES	Triple DES	AES	Blowfish
Encryption (paragraph-sized)				
Time (milliseconds)	7.75	4.95	0.142	3.38
Key size (bytes)	8	24	32	56
Ciphertext size (characters)	1016	1016	1024	1504
Decryption (paragraph-sized)				
Time (milliseconds)	0.222	0.039	0.05	1.28
Encryption (password-sized)				
Time (milliseconds)	0.227	0.0414	0.0679	0.529
Key size (bytes)	8	24	32	56
Ciphertext size (characters)	24	24	24	48
Decryption (paragraph-sized)				
Time (milliseconds)	0.558	0.045	0.068	0.777
Ciphertext size (characters)	1016	1016	1024	1504

Table 13: Simulation results on device 4.

	DES	Triple DES	AES	Blowfish
Encryption (paragraph-sized)				
Time (milliseconds)	4.716	2.416	0.121	2.024
Key size (bytes)	8	24	32	56
Ciphertext size (characters)	1016	1016	1024	1504
Decryption (paragraph-sized)				
Time (milliseconds)	0.434	0.0289	0.0309	0.762
Encryption (password-sized)				
Time (milliseconds)	0.456	0.0314	0.043	0.293
Key size (bytes)	8	24	32	56
Ciphertext size (characters)	24	24	24	48
Decryption (password-sized)				
Time (milliseconds)	0.211	0.0274	0.0386	0.419

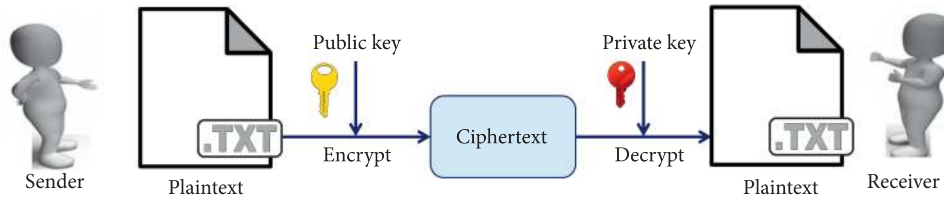

Figure 15: Symmetric encryption model.

Table 14: Encryption principles.

	DES	Triple DES	AES	Blowfish
Encryption (paragraph-sized)				
Time (milliseconds)	4.716	2.416	0.121	2.024
Key size (bytes)	8	24	32	56
Ciphertext size (characters)	1016	1016	1024	1504
Decryption (paragraph-sized)				
Time (milliseconds)	0.434	0.0289	0.0309	0.762
Encryption (password-sized)				
Time (milliseconds)	0.456	0.0314	0.043	0.293
Key size (bytes)	8	24	32	56
Ciphertext size (characters)	24	24	24	48
Decryption (password-sized)				
Time (milliseconds)	0.211	0.0274	0.0386	0.419

Table 15: Simulation results for RSA devices 1, 2, and 3, RSA average, Triple DES, and DES.

Technique	Encryption speed (milliseconds)	Decryption speed (milliseconds)	Private key size (bytes)
RSA device 1	0.321	1.213	128 bytes
RSA device 2	0.793	1.748	128 bytes
RSA device 3	0.570	2.317	128 bytes
RSA average	0.561	1.759	128 bytes
Triple DES	0.0314	0.0274	—
DES	0.456	0.211	—

most reliable in IoT applications, where a large amount of data is computed in real time. Although Triple DES works surprisingly faster than AES in encrypting short-length plaintext commonly used in password or signature encryptions, it is no longer reliable encryption as it has been deprecated by NIST due to the Sweet32 vulnerability [46].

4.2. Asymmetric Cryptographic Algorithm. The symmetric encryption algorithm utilizes the identical secret key for encryption and decryption; the asymmetric encryption algorithm needs two keys for encryption and decryption (as shown in Figure 15).

RSA is a commonly used encryption mode. The encryption principle can be briefly discussed with the instances presented in Table 14.

4.2.1. Simulation Result. Simulation results are presented in Table 15.

The above simulation used password-sized plaintext as a sample for encryption and decryption. The purpose is to understand the performance of the RSA algorithm. Although RSA has been the most commonly used asymmetric encryption algorithm, it shows that RSA performs relatively slow compared to symmetric encryption algorithms. Thus, it should be used only when asymmetric encryption is essential in practice as it will incur extra overhead in both encryption and decryption [47].

5. Conclusion and Recommendations

This article studied and tested several encryption methods on independent computing devices with the C# programming language. Symmetric encryption and decryption were faster but not highly secure as the keys need to be shared between the computing devices (which render it insecure). Asymmetric encryption utilizes a pair of keys, i.e., public and private keys. Thus, it has higher security, yet both encryption and decryption were comparatively slower (than its symmetric counterparts). A recommended solution is to encrypt the symmetric encryption key with the asymmetric encryption public key. The receiver utilizes the private key to decrypt the symmetric encryption key. The asymmetric encryption/decryption only occurs to exchange the keys, therefore not requiring significant computing resources (suitable for IoT/IoE applications). Then, the two edge devices can utilize symmetric encryption in their further communications. The simulated results show that the Blowfish offers better performance than the rest of the commonly used encryption algorithms. Because the Blowfish has no known security weaknesses, it can be a good candidate for standard encryption algorithms. Compared to the other algorithms, AES showed poor performance because it required a heavy-duty computing process. The IoT/IoE application will benefit primarily with Blowfish for data encryption and decryption between the edge devices from the perspective of execution time and cost.

Data Availability

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

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the research grant of the Universiti Kebangsaan Malaysia (UKM) under the grants FRGS/1/2020/ICT03/UKM/02/6 and GGPM 2020-028.

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): a vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] J. Henry, "3DES is officially being retired," *Cryptomathic*, 2020.

[3] M. A. Majid and K. A. Z. Ariffin, "Success factors for cyber security operation centre (SOC) establishment," in *Proceedings of the 1st International Conference on Informatics, Engineering, Science and Technology, INCITEST*, Bandung, Indonesia, July 2019.

[4] S. Kamil, "Challenges in multi-layer data security for video steganography revisited," *Asia-Pacific Journal of Information Technology and Multimedia*, vol. 7, no. 2-2, pp. 53–62, 2018.

[5] N. Varish, A. K. Pal, R. Hassan et al., "Image retrieval scheme using quantized bins of color image components and Adaptive tetrolet transform," *IEEE Access*, vol. 8, pp. 117639–117665, 2020.

[6] M. K. Hasan, M. M. Ahmed, A. H. A. Hashim, A. Razzaque, S. Islam, and B. Pandey, "A novel artificial intelligence based timing synchronization scheme for smart grid applications," *Wireless Personal Communications*, vol. 114, no. 3, pp. 1–18, 2020.

[7] Z. Shelby, K. Hartke, and C. Bormann, "The constrained application protocol (CoAP)," *Internet Engineering Task Force*, pp. 1–110, 2014.

[8] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: a survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[9] B. Padmavathi and S. Ranjitha Kumari, "A survey on performance analysis of DES, AES and RSA algorithm along with LSB substitution," *International Journal of Science and Research*, vol. 2, no. 4, pp. 170–174, 2013.

[10] R. Hassan, A. S. Ahmed, and N. E. Osman, "Enhancing security for IPv6 neighbor discovery protocol using cryptography," *American Journal of Applied Sciences*, vol. 11, no. 9, pp. 1472–1479, 2014.

[11] M. J. Islam, M. Mahin, A. Khatun, S. Roy, S. Kabir, and B. C. Debnath, "A comprehensive data security and forensic investigation framework for cloud-iot ecosystem," *Gub Journal of Science and Engineering*, vol. 4, no. 1, 2017.

[12] S. A. Shakir and M. M. Deris, "A survey on the cryptographic encryption algorithms," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 11, pp. 333–344, 2017.

[13] K. Nyberg, "Generalized Feistel networks," in *Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security*, pp. 91–104, Kyongju, Korea, November 1996.

[14] *Tutorial point Learn Cryptography.*, https://www.tutorialspoint.com/cryptography/data_encryption_standard.htm, 2020.

[15] P. Patila, P. Narayankar, D. G. Narayan, and S. M. Meena, "A Comprehensive Evaluation of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish," in *Proceedings of the International Conference on Information Security & Privacy (ICISP 2015)*, Nagpur, India, December 2015.

[16] P. Arora, A. Singh, and H. Tiyagi, “Evaluation and comparison of security issues on cloud computing environment,” *World of Computer Science and Information Technology Journal*, vol. 2, no. 5, pp. 179–183, 2012.

[17] R. Shashi Mehrotra Seth, “Comparative analysis of encryption algorithms for data communication,” *International Journal of Computer Science and Technology*, vol. 2, no. 2, pp. 292–294, 2011.

[18] D. S. Abdul, H. M. A. Kader, and M. M. Hadhoud, “Performance evaluation of symmetric encryption algorithms,” *IJCSNS International Journal of Computer Science and Network Security*, vol. 8, no. 12, pp. 280–286, 2008.

[19] S. Pavithra and M.. E. Ramadevi, “Performance evaluation of symmetric algorithms,” *Journal of Global Research in Computer Science*, vol. 3, no. 8, pp. 43–45, 2012.

[20] H. O. Alanazi, B. B. Zaidan, A. A. Zaidan, H. A. Jalab, M. Shabbir, and Y. Al-Nabhani, “New comparative study between DES, 3DES and AES within nine factors,” *Journal of Computing*, vol. 2, no. 3, pp. 152–157, 2010.

[21] P. C. Kocher, “Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems,” in *Proceedings of the Annual International Cryptology Conference*, pp. 104–113, Santa Barbara, CA, USA, August 1996.

[22] S. Kamil, M. Ayob, S. N. H. S. Abdullah, and Z. Ahmad, “Lightweight and optimized multi-layer data hiding using video steganography paper,” *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 12, pp. 256–262, 2018.

[23] S. Kamil, “Optimized data hiding in complemented or non-complemented form in video steganography,” in *Proceedings of the 2018 Cyber Resilience Conference (CRC)*, Putrajaya, Malaysia, November 2018.

[24] M. B. Yassein, S. Aljawarneh, E. Qawasmeh, W. Mardini, and Y. Khamayseh, “Comprehensive study of symmetric key and asymmetric key encryption algorithms,” in *Proceedings of the 2017 International Conference on Engineering and Technology (ICET)*, pp. 1–7, Antalya, Turkey, August 2017.

[25] S. Kamil, M. Ayob, S. N. H. Sheikhabdullah, and Z. Ahmad, “Challenges in multi-layer data security for video steganography revisited,” *Asia-Pacific Journal of Information Technology and Multimedia*, vol. 7, no. 2-2, pp. 53–62, 2018.

[26] G. Singh, “A study of encryption algorithms (RSA, DES, 3DES and AES) for information security,” *International Journal of Computer Applications*, vol. 67, no. 19, 2013.

[27] S. Choudhary, K. Kumar, B. Pandey et al., “Soil moisture and environmental temperature and humidity sensor-based data breaches in IoT enable irrigation system design using Arduino and FPGA,” *Gyancity Journal of Engineering and Technology*, vol. 6, no. 2, pp. 1–12, 2020.

[28] A. H. R. M. Shaaban, M. Hamed, I. M. H. Rabie et al., “Moderation in spending and its impact on achieving social security in the era of the IT revolution,” *Journal of Critical Reviews*, vol. 7, no. 16, pp. 1121–1132, 2020.

[29] S. Bamansoor, S. I. A. Saany, and Y. A. B. El-Ebiary, “The S-commerce usage and acceptance modelling in Malaysia,” *3C TIC: Cuadernos de desarrollo aplicados a las TIC*, vol. 9, no. 1, pp. 99–115, 2020.

[30] M. Shafiq, Z. Tian, A. K. Bashir, and X. Du, “IoT malicious traffic identification using wrapper-based feature selection machanisms,” *Computer & Security*, vol. 94, Article ID 101863, 2020.

[31] M. Akhtaruzzaman, M. K. Hasan, S. R. Kabir, S. N. Abdullah, M. J. Sadeq, and E. Hossain, “HSIC bottleneck based distributed deep learning model for load forecasting in smart grid with A comprehensive survey,” *IEEE Access*, vol. 8, pp. 222977–223008, 2020.

[32] I. Memon, R. A. Shaikh, M. K. Hasan, R. Hassan, A. U. Haq, and K. A. Zainol, “Protect mobile travelers information in sensitive region based on fuzzy logic in IoT technology,” *Security and Communication Networks*, vol. 2020, Article ID 8897098, , 2020.

[33] Z. E. Ahmed, M. K. Hasan, R. A Saeed et al., “Optimizing energy consumption for cloud internet of things,” *Frontiers in Physics*, vol. 8, p. 358.

[34] M. Shafiq, X. Yu, and A. K. Bashir, “A machine learning approache for feature selection network traffic classification using security analysis,” *The Journal of Supercomputing*, vol. 74, pp. 4867–4892, 2018.

[35] M. K. Hasan, A. F. Ismail, A.-H. Abdalla, H. A. M. Ramli, W. Hashim, and S. Islam, “Throughput maximization for the cross-tier interference in heterogeneous network,” *Advanced Science Letters*, vol. 22, no. 10, pp. 2785–2789, 2016.

[36] M. K. Hasan, S. H. Yousoff, M. M. Ahmed, A. H. Hashim, A. F. Ismail, and S. Islam, “Phase offset analysis of asymmetric communications infrastructure in smart grid,” *Elektronika ir Elektrotechnika*, vol. 25, no. 2, pp. 67–71, 2019.

[37] M. Shafiq, Z. Tian, A. K. Bashir, and X. Du, “CorrAUC: a malicious bot-IoT traffic detection method in IoT network using machine learning techniques,” *IEEE Internet of Things*, vol. 99, no. 1, 2020.

[38] S. Islam, A.-H. Abdalla, and M. Kamrul Hasan, “Novel multihoming-based flow mobility scheme for proxy NEMO environment: a numerical approach to analyse handoff performance,” *ScienceAsia*, vol. 43S, no. 1, pp. 27–34, 2017.

[39] M. Shafiq, Z. Tian, Y. Sun, and X. Du, “Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for Internet of Things in smart city,” *Future Generation Computer Systems*, vol. 107, no. 4, 2020.

[40] S. Islam, O. O. Khalifa, A. H. Hashim, M. K. Hasan, M. A. Razzaque, and B. Pandey, “Design and evaluation of a multihoming-based mobility management scheme to support inter technology handoff in PNEMO,” *Wireless Personal Communications*, vol. 114, no. 2, 2020.

[41] M. K. Hasan, “Lightweight encryption technique to enhance medical image security on internet of medical things applications,” *IEEE Access*, vol. 23 pages, 2021.

[42] M. Shafiq, Z. Tian, A. K. Bashir, A. Jolfaei, and X. Yu, “Data mining and machine learning method for sustainable cities traffic classification,” *Sustainable Cities and Society*, vol. 60, Article ID 102177, 2020.

[43] D. Singh, D. Patel, B. Borisaniya, and C. Modi, “Collaborative IDS framework for cloud,” *International Journal of Network Security*, vol. 18, no. 4, pp. 699–709, 2016.

[44] M. Shafiq, Z. Tian, A. K. Bashir, K. Cengiz, and A. Tahir, “SoftSystem: smart edge computing device selection method for IoT based on soft set technique,” *Wireless Communication & Mobile Computing*, vol. 2020, Article ID 8864301, 2020.

[45] E. S. Ismail and M. S. Hijazi, “New cryptosystem using multiple cryptographic assumptions,” *Journal of Computer Science*, vol. 7, no. 12, p. 1765, 2011.

[46] P. Schmid and A. Roos, *AES-NI Performance Analyzed; Limited To 32nm Core i5 CPUs*, Tom s Hardware, 2010, https://www.tomshardware.com/reviews/clarkdale-aes-ni-encryption,2538.html.

[47] T. M. Ghazal, M. K. Hasan, R. Hassan et al., “Security vulnerabilities, attacks, threats and the proposed countermeasures for the Internet of Things applications,” *Solid State Technology*, vol. 63, no. 1s, pp. 2513–2521, 2020.