ASTES

Advances in Science, Technology and Engineering Systems Journal Vol. 6, No. 2, 622-628 (2021)

www.astesj.com

Special Issue on Multidisciplinary Sciences and Engineering

ASTESJ

ISSN: 2415-6698

Node-Node Data Exchange in IoT Devices Using Twofish and DHE

Bismark Tei Asare[*,1,2], Kester Quist-Aphetsi[2], Laurent Nana[1]

[1]*Univ Brest, Lab-STICC, CNRS, UMRS 6285, F-29200 Brest, France*

[2]*Ghana Communication Technology University, Computer Science Department, Ghana*

ARTICLE INFO

Article history:
Received: 24 December, 2020
Accepted: 08 March, 2021
Online: 20 March, 2021

Keywords:
Secure Cloud Communication
IoT Security
Twofish
Diffie-Hellman Key Exchange

ABSTRACT

Internet of Things provides the support for devices, people and things to collaborate in collecting, analyzing and sharing sensitive information from one device onto the other through the internet. The internet of things is thriving largely due to access, connectivity, artificial intelligence and machine learning approaches that it supports. The stability and enhanced speed of the internet is also attributable to the huge adoption rate that IoT continues to enjoy from Governments, industry and academia in recent times. The increased incidences of cyber-attacks on connected systems in recent times, has inspired the heightened efforts from Governments, industry practitioners and the research world towards improving existing approaches and the engineering of new innovative schemes of securing devices, the software or the platforms for the deployment of IoT. Security solution for Internet of things includes the use of secure ciphers and key exchange algorithms that ensures the provisioning of a security layer for the: devices or hardware, communication channels, cloud, and the life cycle management constituting the Internet of things. The use of key exchange algorithms in resilient cryptographic solution that have less computational requirements without compromising the security efficiency in the encryption of messages for IoT continues to be the preferred approach in securing messages in a node-node exchange of data. This paper aims at providing a cryptographic solution that uses a key exchange cryptographic primitive and a strong cipher in encrypting messages for exchange between nodes in an IoT. Towards achieving this goal, the Diffie-Hellman key exchange (DHE) protocol was used to provide a secure key exchange between the communicating nodes, whiles the Twofish block cipher was used in the encryption and decryption of messages, assuring the security, privacy and integrity of messages in a node-node IoT data exchange. The cryptographic solution has a high throughput.

1. Introduction

The use of sensors, actuators, radars and other diodes in the collection, intelligent analysis and communication of sensitive data in connected systems have driven the user specific needs for homes, businesses, public, private institutions as well as security agencies in expanding access to support the scaling of network resources to drive new value propositions. From surveillance to the aggregation of sensitive information such as humidity, pressure, temperature and in some case depth measurement in restricted environments like mining, construction and naval systems, IoT has been the preferred solution in achieving these desired goals.

This paper is an extension of work originally presented in the International Conference on Communications, Signal Processing and Networks/International Conference on Cyber Security and Internet-of-Things ICCSPN/ICSIoT. The relatively complex key schedule that Twofish cryptographic primitive supported together with the DHE provided an efficient and tamper-resistant security scheme for data exchange between nodes in IoT [1].

The security of communication in connected devices has understandably been of great concern to both governments, industry and academia due to the rather recent increase in the number of high profile cybersecurity attacks on these networks where several accounts with sensitive information have been tempered with and in some cases, holders of these accounts have lost huge sums of monies. The cases of cyber-attacks in the past

[*]Corresponding Author: Bismark Tei Asare, Ghana Communication Technology University, Ghana, +33638556158, btasare@gctu.edu.gh

five years have motivated several public and private organisations to increase their budgetary allocations and investments into providing a stronger security infrastructure to secure their networks [2]. The devices serving as the main actors for the internet-of-things could introduce additional sources and points for attacks that hackers could use in exploiting the vulnerabilities in IoT because most of these devices are not secure by design and hence lack the appropriate robust security configuration to ensure secure communication of sensitive data, yet these devices end up forming essential components for systems that aggregate sensitive data for communication across several devices through the internet. These devices are mostly susceptible to several cybersecurity challenges including the backdoor vulnerability where hackers could easily exploit to successfully manipulate these IoT systems to their skewed benefits [3]-[6].

There exist various techniques for supporting secure node-to-node authentication that assure integrity and availability of sensitive data. Most of these techniques demand higher resource requirements for effective and efficient implementation. In the particular context of IoT devices, due to the resource constraint nature with regards to computational power, storage and power consumption, these encryption techniques and approaches are not adoptable for them, in most of the cases. In addition, since IoT devices automate their processes by eliminating human intervention in their operations, the use of some of the off the shelves solutions that exists are not practicably suited for them. Symmetric encryption involves the use of a shared key between two participating nodes in a communication. It allows these participating nodes to generate key pairs - a private key and a corresponding public key using a key generator - a trustee component in a key exchange protocol application that allows two nodes that have no prior knowledge of each other to jointly share secret for encryption and decryption of message. The public keys are common keys that are known by the participating nodes before any communication is even established. The public key is also known as a shared key. Both private (secret) key and public (shared or symmetric) keys are used in encrypting and decrypting data in a communication session between two participating nodes or parties. There are several key exchange protocols. The Diffie-Hellman key exchange (DHE) protocol allows two nodes to securely exchange session key (the public or shared key) prior to communicating messages between them [7]. The Diffie-Hellman key exchange protocol ensures the creation of a new session key for each message to be communicated, this provides another layer of security. Session keys are generated using random number generator schemes or algorithms. The key exchange protocol only supports secure sharing of session key and does not provide authentication of the source node for creating the message. Replay attacks, Man-in-the-middle attacks, Dictionary attacks, Key compromise impersonation attacks and ephemeral key compromise attacks are some of the vulnerabilities that DHE protocol suffers [1]. Cryptographic algorithms provide mechanisms for authentication using various approaches like digital signature schemes and public-key certificates.

The scalable nature of IoT and its ability to support the rapid generation volumes of traffic and their associated processing and temporary storage, places additional overload on the edge nodes used in the network. The nodes are originally constrained by design as well as operational capabilities particularly for large computational activities. Hence, complex security infractions and exploitations are constantly deployed on these IoT nodes using modern and sophisticated tools by hackers [8]-[11].

A key exchange algorithm like the DHE by itself is limited in use alone unless, it is fed into another protocol for undertaking encryption and decryption of messages. The use of the DHE in the Twofish encryption algorithm increases the confusing of the encryption key to enhance the diffusion of the distribution of the ciphertext to avoid redundancy. This improves the level of security in the cipher. The use of DHE within the Twofish cryptographic symmetric key block cipher for message encryption assured privacy and confidentiality of IoT node data.

The rest of the paper is structured into five sections. Section 2 presents key concepts needed for a better understanding of the article and related works. Section 3 is dedicated to the proposed methodology. Section 4 discusses results and Section 5 concludes the paper.

2. Background and Related Works

2.1. Background

A description of key notations and terminologies used in the paper is outlined as follows:

- Key – Is any random number as an output from a random number generator. It is used in encryption or decryption of messages. It is mainly referred to as a cryptography key.
- Public Key – It is also known as a shared or symmetric key. This key is a common key shared by two participating nodes prior to initiating a session for message communication. Two keys are generated for every participating node the public key and its corresponding private key.
- Private Key – The private key is a unique key for each participating node. The private key is used in combination with the public key of the sender for encryption of messages. During decryption of messages also, the private key of the receiver is used.
- Encryption – It is the process of encoding or hiding the content or message details by changing them from plaintext to a ciphertext using cryptographic keys. In a symmetric encryption, the private key is used to encrypt and decrypt messages.
- Decryption – It involves the use of cryptographic keys and approaches in changing ciphertexts to their original plain texts. In symmetric encryption algorithms, the private key of the destination node is used together with the public key of the source node for the decryption.
- Plaintext – It describes an unencrypted message in which the original message format and content are maintained.
- Ciphertext – It is a description for an encrypted message. Thus, the output or transformed text for any plaintext message on an encryption algorithm.
- Cipher – It is a cryptographic algorithm for transforming plaintext to ciphertext. It is used for encryption and decryption of messages.

- Algorithm – It involves a standard outline of sequential procedure or steps for undertaking encryption or decryption of messages.

The next section presents related works.

2.2. Related Work

2.2.1. Secure Cloud Communication

In our previous conference paper, we pointed out the security weakness of the DHE protocol in the provision of security solution against threats such as man-in-the-middle attacks due to vulnerabilities in the protocol to authenticate participating nodes using their unique identifying features in block stream cipher [1].

Cloud computing infrastructure services where enterprises manage and share resources between the local devices that are stationed in their premises and remote servers offer some convenience but have security challenges including data loss and tampering [12].

The absence of a trusted and robust security framework in the cloud to protect against security threats including access control management, authentication challenges, integrity of stored information that cloud services suffer is a concern that attracts equal attention from industry and academia to explore approaches to address the security vulnerabilities in these services to improve the security of the cloud to support secure remote processing and storage of information [13]-[16].

2.2.2. Internet-of-Things Security

In [17] an extensive survey on the security, privacy and authentication challenges in internet of things confirmed the fact that although there have been. improvement in the design and development of cryptographic solutions tailored for the individual elements within IoT, there is equally an increase in the number of exploitation of attacks in recent times. Therefore, authentication, confidentiality, and data integrity challenges exist in IoT domain making it demand a cryptographic solution that adopts an algorithm that provides efficient privacy and security.

In [18], the authors delved into the material agents used in the design and development of smart objects and internet of things systems. Majority of the devices used in an IoT system have sensing capabilities to independently collect data from its environment for transport to other devices within the network. A defective device by design is a threat by itself and introduces a weak link for attack exploitation to a system that has such a device as component. Compatibility challenges in intelligence sensing, interoperability, distributed intelligence and flexibility to adapt to a universal authentication solution for an IoT ecosystem have become fundamental in implementing IoT security solution.

An efficient encryption scheme that involves less computational overheads and runs on adequate power for constrained devices without compromising on the security, privacy and integrity of data in IoT is needed to in modern IoT security architectures [19]. The algorithm developed in [20] combined machine learning approaches to detect and control network congestion in IoT using the fitness function that is based on the grey wolf optimization algorithm (GWO), since network congestion could increase the execution overheads for cryptographic algorithms to run efficiently.

The intensity and size for IoT investments by enterprises keep increasing to improve the security of their network, as the number and the scale in the incidences of security attack on IoT keep rising [21], [22]. Modern and more sophisticated malware that targets IoT devices and the various components of IoT systems to exploit the security vulnerabilities in these devices to the skewed benefits of those attackers is on the increase [8].

2.2.3. Twofish

The authors in [23] adopted the Twofish encryption algorithm in securing data and maintaining confidentiality of the communication network and implemented the solution using the chilkat encryption activeX. Figure 1[24] illustrates the working of the Twofish encryption algorithm. The encryption involved splitting the input data into four halves of 128 bits, where the XOR operations were conducted on the bits input with a key. The key whitening and the XOR operation provided the needed security to the data to assure privacy of the data in the communication network.

The Twofish encryption algorithm was used in addition to other cryptographic primitives to enhance the security in Bluetooth encryption. The combined encryption approach supported the protection of data in Bluetooth transmission. The use of the Twofish cryptographic algorithm improved the security and efficiency of the encryption scheme in the Bluetooth communication [25].

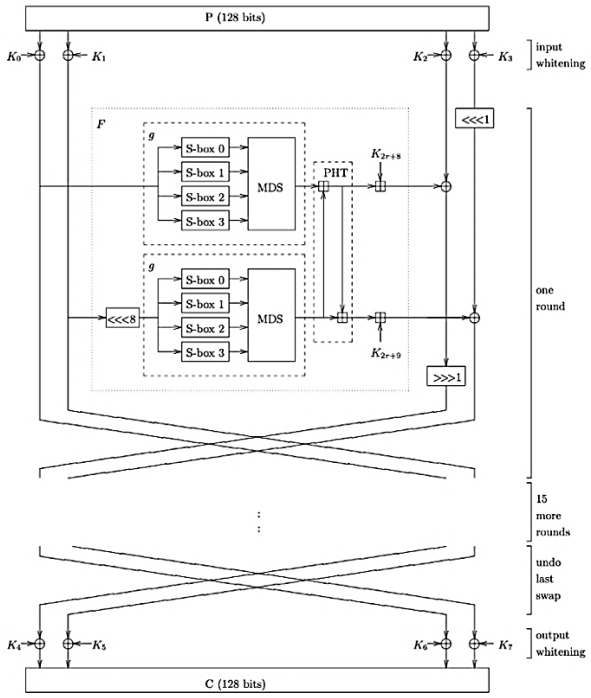

Figure 1: Twofish [24]

The Twofish cryptographic algorithm provided an enhanced security for a software to support secure communication of information over the internet. In their paper, the Twofish algorithm with 192-bits key space was implemented in encrypting messages.

The algorithm provided efficient and secure encryption of data for communication [26], [27].

In [28], a variant of the Twofish algorithm with an increased complexity for multi-level keys adoptable for dynamic bit size inputs to improve its cryptographic strength that made it resistant against the differential attacks was adopted for encrypting images.

The standard Feistel encryption algorithm involving several rounds of encryption with key whitening techniques to improve the security of the encryption is what the Twofish encryption algorithm offers. The input and output data are XOR-ed with eight sub-keys. The encryption in the Twofish encryption algorithm involves encrypting a message over 16-rounds in a Feistel network that uses a bijective function comprising four-byte wide pre-computed key dependent substitution (S) boxes; a matrix; a key schedule, and a Pseudo Hadamard Transform of bitwise rotation. After each round of encryption, the ciphertext generated is swapped and fed as input into the next round such that the left encrypted text is interchanged with the right counterpart and vice versa. The final round of encryption produces a ciphertext in two halves-the right and left halves of cipher texts. The resultant ciphertext for the Twofish encryption is achieved by interchanging the positions of the ciphertexts and combining both as the ciphertext for the message [23], [24]

2.2.4 Diffie-Hellman Key Exchange (DHE)

The communicating nodes through a key generator acquire two sets of key pairs (a private key and a public key) to enable them authenticate for communicating data. The DHE is susceptible to several attacks including key compromise impersonation attacks [23], [29].

The authors in [30] adopted an algorithm that is based on the Linear Feedback Shift Register (LFSR)-dependent correlation technique to supplement the Diffie-Hellman key exchange protocol to maintain the privacy of message communication between the cloud and the local device. The LFSR correlation algorithm detected and verified digital signatures from a digital signature pool between the devices and the cloud by using the correctional framework for digital signature analysis through the calculation of linear complexities between the cloud and the local devices. This technique helped with the detection of errors with digital signatures of messages thereby maintaining the privacy of the information shared between the cloud and the local devices, eliminating the possibility of key compromise or impersonation and the related attacks.

In [31], cryptographic-based public key infrastructure approaches provided at the device levels were adopted to support secure communication of message in an end-to-end encryption that ensured privacy, confidentiality and device integrity. The Elliptic Curve Diffie-Hellman key exchange (ECDH) guaranteed a secure key sharing between communicating devices on a chat application that operated on the android environment. The RC4 encryption scheme was used to encrypt the multimedia messages for communication between the devices used on the chatting application.

The use of authentication scheme based on the Diffie-Hellman model supported the efficient key exchange and management for communication devices with more than one identity. The authors in [32] used a key agreement and management protocol adaptable for IoT communicating devices with more than one identity. The key agreement and management protocol supported the efficient selection and initialization of session key pairs from devices with multiple identities and helped authenticate the communicating devices in the IoT system.

The next section deals with the proposed methodology.

3. Methodology

The proposed methodology is a combined cryptographic primitive consisting of a key exchange protocol or key agreement protocol such as the Diffie-Hellman Key Exchange (DHE) protocol illustrated in figure 2, and a cryptographic cipher for encryption and decryption such as the Twofish cryptographic algorithm illustrated in figure 1.

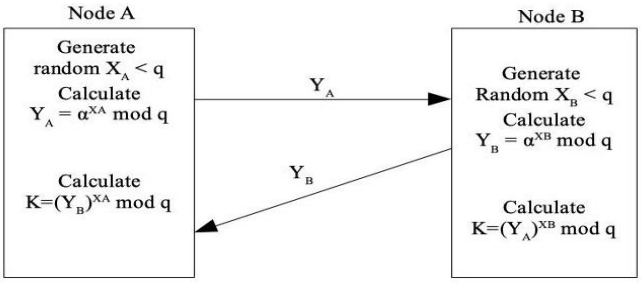

Figure 2: Diffie-Hellman Key Exchange between Node A and Node B.

The Diffie-Hellman Key Exchange (DHE) protocol is a public-key cryptographic primitive with an implementation involving two unknown nodes or devices to securely establish communication by generating and exchanging shared secrets between them. The securely shared secret is used in performing a symmetric key encryption and decryption. The DHE is structured on the discrete logarithm problem which is based on a one-way function; finding the factorization of the product of two prime numbers p and g.

Such that p is a prime number and g is a primitive root mod (p).

where: p and g (generator) are two large integer number.

$1 < n \leqslant (p-1)$; where $n = g^k \bmod p$

Thus, for every number n between 1 and p-1 inclusive, there is a power k of g such that $n=g^k \bmod p$.

Publicly available numbers of p and g are to be used by any two untrusting and unknown devices or nodes to generate their key pairs: namely the public key (Pu) and private key (Pv).

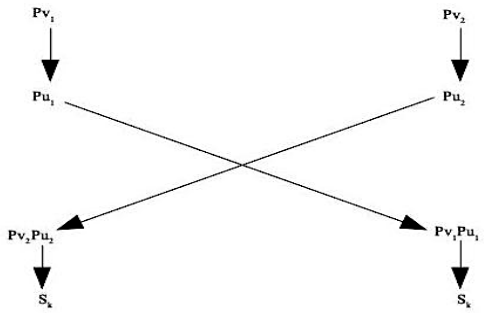

Figure 3: The generator for the shared key in DHE

As shown in Figure 3, the notations for Pv_1 , Pu_1 and Pv_2 , Pu_2 represent the Private key and shared key for the source node and the receiver node respectively. The DHE protocol used four numbers in all. The first two numbers are to generate the shared key (of same value) $Pu_1 = Pu_2$. Thus, both sender and receiver have the same value for the shared key Pu_1.

The DHE and Twofish Algorithm:

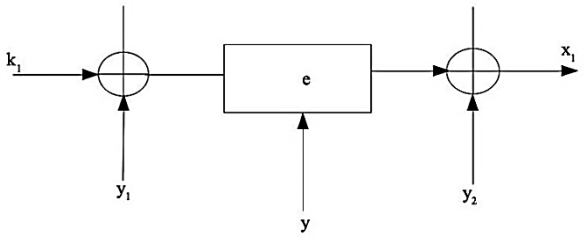

Figure 4: Twofish Encryption Algorithm based on the DHE Protocol

As shown in Figure 4, (k_1, y_1) represents the private key and public key respectively for the source node.

(e), represents the Twofish cryptographic function.

(x_1, y_2) represents the private and shared keys respectively for the receiver node.

The shared key for the communicating nodes is $(y) = y_1 = y_2$.

Plaintext of arbitrary size data up to 128 bits. The input bits are grouped into four sections, thus four parts of each constituting a 32-bit part. The four sections are divided into halves. The first two sections of 32 bits each forms the right component of the input bits and the other two sections of two 32 bits, form the left component [23].

The four keys for whitening the encryption are applied on the Bit-XOR input.

$R_{0,i} = P \oplus K_i \; ; i = 0, \dots, 3$

Where:

R denotes the rounds of encryption in the Feistel network.

K denotes the key (Ki) where i represents the sub-key for whitening. There are four sub-keys. 0, 1, 2, 3.

Encryption: $e_k, k_1, y_1(M), = e_k(M \oplus k_1) \oplus y_1 = C_T$

Decryption: $e_k^{-1}, x_1, y_2(C_T), = e_k^{-1}(C_T \oplus x_1) \oplus y_2 = M$

where:

M represents the plaintext.

C_T denotes the ciphertext or encrypted message.

e_k represents the Twofish encryption function.

e_k^{-1} represents the Twofish decryption function.

k_1 denotes the private key for the source node.

x_1 represents the private key for the receiver node.

y_1 denotes the public key of the source node.

y_2 denotes the public key of the receiver node.

$y_1 = y_2 = y$, denotes the shared secret for encryption and decryption of the DHE-Twofish cryptographic solution.

Encryption at the source node is implemented using the shared key with the Twofish algorithm encryption function.

Algorithm 1: Secure Node-Node Data Exchange - *Encryption* - Twofish (M, S_k)

Input : M, S_k
Output: C_T
Begin
  Split M into four 32-bit parts $M_{i=0,1,2,3}$
  Split S_k into four 32-bit partial keys key $K_{i=0,1,2,3}$
  For i=0..1 **loop**
    $R_{i,0}$ = M XOR K_{2*i}
    $L_{i,0}$ = M XOR K_{2*i+1}
    For j=1..3 **loop**
      $L_{i,j} = R_{i,j-1}$
      $R_{i,j} = L_{i,j-1}$ XOR (((2 $R_{i,j-1}$ * K_j)^ x) % (2^32-1))
    End loop
  End loop
  Combine L_{13}, R_{13}, L_{03} and R_{03} into C_T
  Return C_T
End

Algorithm 2: Secure Node-Node Data Exchange - *Decryption* - Twofish (CT, S_k)

Input : C_T, S_k
Output: M
Begin
  Split C_T into four 32-bit parts $C_{T\ i=0,1,2,3}$
  Split S_k into four 32-bit partial keys key $K_{i=0,1,2,3}$
  For i=0..1 **loop**
    $R_{i,3}$ = C_T XOR K_{2*i}
    $L_{i,3}$ = C_T XOR K_{2*i+1}
    For j=2..0 **loop**
      $R_{i,j} = L_{i,j+1}$
      $L_{i,j} = R_{i,j+1}$ XOR (((2 $L_{i,j+1}$ * K_j)^ x) % (2^32-1))
    End loop
  End loop
  Combine L_{10}, R_{10}, L_{00} and R_{00} into M
  Return M
End

The Algorithm 1 above is the encryption procedure for the Twofish cipher at the source node.

The Algorithm 2 above is used for the decryption of the ciphertext at the receiving node using the Twofish cipher.

In Algorithms 1 and 2:

M represents the plaintext.

S_k shared key, represents the Key K

C_T represents the ciphertext.

K_i represents the partial key or the subkeys where i represents the indexes from (0,1,2,3)

Li and Ri are Left and Right partitioned 64-bits block size
f represents the Twofish cipher.

% represents the Modulo operator.

Li represents the Leftmost partitioned part of the Feistel structure

Ri represents the Rightmost partitioned part of the Feistel structure

x represents the total number of rounds, which is 16 in this case.

4. Results and Discussion

At the source node, using the DHE through a handshake request with the destination node established a shared key for encryption and decryption.

The Twofish encryption uses the data, and the shared key to produce the ciphertext. Decryption of ciphertext at the destination node is also implemented using the Twofish on the shared key and ciphertext.

Table 1: Displayed results

ID	SK	Data	Ciphertext	Recovered
1	262262262	2345	OXEVxUcVBXZk lzbf0F1D+A==	2345
2	232232232	1291	7efDJnVQrVwEV qBD3rguLA==	1291
3	452452452	4672	+gJXYsjBJP/Y/UE djUMIWQ==	4672
4	232232232	1456	RXec4NLFwt1nmx eWqQkm3g==	1456
5	232232232	2121	IeTtTlkRgQuvjD0h AFQFIg==	2121

The ID denotes the unique node identity number which represented the order of arriving data for the encryption. The 1st, 2nd, 3rd, 4th, 5th showed the first data, second data, third data, fourth data, and firth data to arrive in that order.

The SK represents the shared key by the DHE for message encryption.

Data denotes the plaintext.

Ciphertext represents the output of the plaintext with the Twofish cryptographic algorithm.

Recovered Data represents the decrypted data.

The Diffie Hellman key exchange protocol that took the unique ID of the communicating devices for establishing the shared key using the key generator helped for encryption of node data.

Table 2: Average Encryption Throughput (MiB) in the ARMv7-a of Samsung and Xiaomi Devices [33]

Algorithm/ Pack Size	1 MiB	5 MiB	10 MiB
AES	77.539	78.058	77.586
RC6	51.84	53.556	53.1065
Twofish	47.7135	48.6145	47.1835

As presented in Table 2 [33], the average encryption throughput for data packet sizes 1MiB, 5MiB and 10 MiB shows an improved throughput for Twofish algorithm. The management information base (MiB) values catalogued the properties and data objects from the encryption of implementation of AES, RC6 and Twofish respectively for the block ciphers. Across all the pack sizes of the MiB values, the encryption throughput for Twofish was higher than the AES and the RC6.

Table 3: Average Decryption Throughput (MiB) in the ARMv7-a of Samsung and Xiaomi Devices [33]

Algorithm/ Pack Size	1 MiB	5 MiB	10 MiB
AES	66.671	68.541	69.026
RC6	52.0065	53.3635	52.877
Twofish	47.785	48.6215	47.0225

As seen in Table 3 [33], the average decryption throughput for data packet sizes of 1MiB, 5MiB and 10 MiB shows an improved throughput for Twofish algorithm. The management information base (MiB) values catalogued the properties and data objects from the encryption of implementation of AES, RC6 and Twofish respectively for the block ciphers. Across all the pack sizes of the MiB values, the encryption throughput for Twofish was higher than the AES and the RC6.

Table 4: Speed of AES Candidates for different Key Lengths [34]

Algorithm Name	Key Setup	Encryption
MARS [BCD+98]	Constant	Constant
RC6 [BCD+98]	Constant	Constant
Rijndael [DR98]	Increasing	128: 10 rounds 192: 20% slower 256: 40% slower
SERPENT [ABK98]	Constant	Constant
Twofish [SKW+98, SKW+99a]	Increasing	Constant

As shown in Table 4 [34], the speed of the AES candidates for different key lengths for encryption is presented. The Twofish algorithm adopts an increasing key setup but encrypts and decrypts at a speed independent of the length. The performance and security of an encryption is as a function of the key space or key length.

5. Conclusion

The combined cryptographic scheme consisted of the DHE, an algorithm based on mathematical approaches for exchanging a shared secret between the communicating nodes and the Twofish. The Twofish encryption algorithm provided a relatively better encryption time than the AES and RC6. The use of the DHE increased the confusion and diffusion of the key for the encryption algorithm to improve the strength of the cryptographic algorithm of Twofish in eliminating the possibilities of relative key attacks that could result in man-in-the-middle and its associated attacks.

The Twofish cryptographic algorithm complemented by the pre-shared secret key protocol in the DHE key exchange provided authentication for the nodes as well as the creation of a secure channel between the communicating nodes to guarantee a secure

node-to-node exchange of IoT messages. It helped in assuring the integrity of the content of messages between the communicating nodes since the DHE, in generating the shared key, only included and engaged the intended receiver node prior to the actual message communication. The complex key exchange logic provided in DHE algorithm improved the key diffusion in the Twofish cryptographic symmetric key cipher assuring an improved security with a relatively high throughput for end-to-end encryption for secure communication within IoT systems. As presented in Tables 2, 3 and 4 [33], [34], the Twofish encryption algorithm produced an encryption and decryption with a high throughput [32], [33], [35].

An analysis on the throughput, battery drain, key space and its impact on the security of related symmetric key ciphers for IoT constrained devices would be explored for future works.

Conflict of Interest

The authors declare no conflict of interest.

References

[1] B.T. Asare, K. Quist-Aphetsi, L. Nana, "Secure data exchange between nodes in IoT using TwoFish and DHE," Proceedings - 2019 International Conference on Cyber Security and Internet of Things, ICSIoT 2019, 101–104, 2019, doi:10.1109/ICSIoT47925.2019.00024.

[2] G. Strategy, L. Council, "Internet of Things : Where Your Competitors Are Investing Overview," Gartner, 2020.

[3] S. Siboni, V. Sachidananda, Y. Meidan, M. Bohadana, Y. Mathov, S. Bhairav, A. Shabtai, Y. Elovici, "Security Testbed for Internet-of-Things Devices," IEEE Transactions on Reliability, 2019, doi:10.1109/TR.2018.2864536.

[4] C. DeWitt, J. Ellis, "Control system cyber security: Staying ahead of the evolving threats," in Proceedings of the Annual Offshore Technology Conference, 2013, doi:10.4043/24393-ms.

[5] S.M. Saeed, S.S. Ali, O. Sinanoglu, Scan design: Basics, advancements, and vulnerabilities, 2017, doi:10.1007/978-3-319-44318-8_6.

[6] P. Chan, T. Barnett, A.-H. Badawy, P.W. Jungwirth, "Cyber defense through hardware security," 2018, doi:10.1117/12.2302805.

[7] M.R. Mishra, J. Kar, "A Study on Diffie-Hellman Key Exchange Protocols," International Journal of Pure and Apllied Mathematics, **114**(2), 2017, doi:10.12732/ijpam.v114i2.2.

[8] C. Kolias, G. Kambourakis, A. Stavrou, J. Voas, "DDoS in the IoT: Mirai and other botnets," Computer, **50**(7), 80–84, 2017, doi:10.1109/MC.2017.201.

[9] P. Wang, S. Sparks, C.C. Zou, "An advanced hybrid peer-to-peer botnet," IEEE Transactions on Dependable and Secure Computing, 2010, doi:10.1109/TDSC.2008.35.

[10] E.B. Beigi, H.H. Jazi, N. Stakhanova, A.A. Ghorbani, "Towards effective feature selection in machine learning-based botnet detection approaches," in 2014 IEEE Conference on Communications and Network Security, CNS 2014, 2014, doi:10.1109/CNS.2014.6997492.

[11] D. Zhao, I. Traore, B. Sayed, W. Lu, S. Saad, A. Ghorbani, D. Garant, "Botnet detection based on traffic behavior analysis and flow intervals," Computers and Security, 2013, doi:10.1016/j.cose.2013.04.007.

[12] F. Sabahi, "Cloud computing security threats and responses," 2011 IEEE 3rd International Conference on Communication Software and Networks, ICCSN 2011, 245–249, 2011, doi:10.1109/ICCSN.2011.6014715.

[13] H. Sato, A. Kanai, S. Tanimoto, "A cloud trust model in a security aware cloud," Proceedings - 2010 10th Annual International Symposium on Applications and the Internet, SAINT 2010, 121–124, 2010, doi:10.1109/SAINT.2010.13.

[14] A. Behl, "Emerging security challenges in cloud computing: An insight to cloud security challenges and their mitigation," Proceedings of the 2011 World Congress on Information and Communication Technologies, WICT 2011, 217–222, 2011, doi:10.1109/WICT.2011.6141247.

[15] M.D. Ryan, "Cloud computing security: The scientific challenge, and a survey of solutions," Journal of Systems and Software, **86**(9), 2263–2268, 2013, doi:10.1016/j.jss.2012.12.025.

[16] C. Rong, S.T. Nguyen, M.G. Jaatun, "Beyond lightning: A survey on security challenges in cloud computing," Computers and Electrical Engineering, **39**(1), 47–54, 2013, doi:10.1016/j.compeleceng.2012.04.015.

[17] P.M. Chanal, M.S. Kakkasageri, "Security and Privacy in IoT: A Survey," Wireless Personal Communications, **115**(2), 1667–1693, 2020, doi:10.1007/s11277-020-07649-9.

[18] C. Savaglio, M. Ganzha, M. Paprzycki, C. Bădică, M. Ivanović, G. Fortino, "Agent-based Internet of Things : State-of-the-art and research challenges," Future Generation Computer Systems, **102**, 1038–1053, 2020, doi:10.1016/j.future.2019.09.016.

[19] P. Sudhakaran, C. Malathy, "Energy efficient distributed lightweight authentication and encryption technique for IoT security," International Journal of Communication Systems, (August), 1–10, 2019, doi:10.1002/dac.4198.

[20] M.S. Manshahia, "Grey Wolf Algorithm based Energy-Efficient Data Transmission in Grey Wolf Algorithm based Energy-Efficient Data Transmission in Internet of Things Internet of Things," Procedia Computer Science, **160**, 604–609, 2019, doi:10.1016/j.procs.2019.11.040.

[21] I. Lee, K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," Business Horizons, 2015, doi:10.1016/j.bushor.2015.03.008.

[22] A. Mayuri, Bhabad, B. Sudhir, T., "Internet of Things: Architecture, Security Issues and Countermeasures," Future Generation Computer Systems, **10**(6), 1–4, 2015, doi:10.5120/ijca2015906251.

[23] M.A. Muslim, B. Prasetiyo, Alamsyah, "Implementation twofish algorithm for data security in a communication network using library chilkat encryption activex," Journal of Theoretical and Applied Information Technology, **84**(3), 370–375, 2016.

[24] S. Bruce, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson, Twofish : A 128-Bit Block Cipher, John Wiley & Sons Inc, 1999.

[25] M.A. Albahar, O. Olawumi, K. Haataja, P. Toivanen, "Novel Hybrid Encryption Algorithm Based on Aes, RSA, and Twofish for Bluetooth Encryption," Journal of Information Security, **09**(02), 168–176, 2018, doi:10.4236/jis.2018.92012.

[26] P. Gehlot, S. R. Biradar, B. P. Singh, "Implementation of Modified Twofish Algorithm using 128 and 192-bit keys on VHDL," International Journal of Computer Applications, 2013, doi:10.5120/12024-8087.

[27] D. Smekal, J. Hajny, Z. Martinasek, "Hardware-Accelerated Twofish Core for FPGA," in 2018 41st International Conference on Telecommunications and Signal Processing, TSP 2018, 2018, doi:10.1109/TSP.2018.8441386.

[28] S.M. Kareem, A.M.S. Rahma, "A novel approach for the development of the Twofish algorithm based on multi-level key space," Journal of Information Security and Applications, 2020, doi:10.1016/j.jisa.2019.102410.

[29] N. Li, "Research on diffie-hellman key exchange protocol," ICCET 2010 - 2010 International Conference on Computer Engineering and Technology, Proceedings, **4**(4), V4-634-V4-637, 2010, doi:10.1109/ICCET.2010.5485276.

[30] M.P. Rewagad, M.Y. Pawar, "Use of digital signature with diffie hellman key exchange and aes encryption algorithm to enhance data security in cloud computing," Proceedings - 2013 International Conference on Communication Systems and Network Technologies, CSNT 2013, 437–439, 2013, doi:10.1109/CSNT.2013.97.

[31] M. Begum, M.A. Waheed, "Protecting Data Privacy in Cloud," International Journal of Engineering and Advanced Technology (IJEAT), (9), 2249–8958, 2020, doi:10.35940/ijeat.C6096.029320.

[32] A. Ali, A. Sagheer, "Design of Secure Chatting Application with End to End Encryption for Android Platform," Iraqi Journal for Computers and Informatics, 2017, doi:10.25195/ijci.v43i1.73.

[33] D.A.F. Saraiva, V.R.Q. Leithardt, D. de Paula, A.S. Mendes, G.V. González, P. Crocker, "PRISEC: Comparison of symmetric key algorithms for IoT devices," Sensors (Switzerland), **19**(19), 1–23, 2019, doi:10.3390/s19194312.

[34] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson, "On the twofish key schedule," Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), **1556**, 27–42, 1999, doi:10.1007/3-540-48892-8_3.

[35] A. Ghosh, "Comparison of Encryption Algorithms : AES , Blowfish and Twofish for Security of Wireless Networks," International Research Journal of Engineering and Technology, (June), 4656–4659, 2020, doi:10.13140/RG.2.2.31024.38401.