A Hybrid Encryption Solution to Improve Cloud Computing Security using Symmetric and Asymmetric Cryptography Algorithms

Hossein Abroshan
Department of Business Informatics and Operations Management, Ghent University
Ghent, Belgium

***Abstract*—Ensuring the security of cloud computing is one of the most critical challenges facing the cloud services sector. Dealing with data in a cloud environment, which uses shared resources, and providing reliable and secure cloud services, requires a robust encryption solution with no or low negative impact on performance. Thus, this study proposes an effective cryptography solution to improve security in cloud computing with a very low impact on performance. A complex cryptography algorithm is not helpful in cloud computing as computing speed is essential in this environment. Therefore, this solution uses an improved Blowfish algorithm in combination with an elliptic-curve-based algorithm. Blowfish will encrypt the data, and the elliptic curve algorithm will encrypt its key, which will increase security and performance. Moreover, a digital signature technique is used to ensure data integrity. The solution is evaluated, and the results show improvements in throughput, execution time, and memory consumption parameters.**

Keywords*—Cloud computing; security; cryptography; digital signature

I. Introduction

Many organisations and individuals use cloud computing services such as PAAS (Platform as a Service), IAAS (Infrastructure as a Service), and SAAS (Software as a Service). Cloud computing is a location-independent environment that shares calculations and resources to provide high-performance services [1]. Although cloud computing makes our lives easier, it comes with security challenges and the threat of cyberattacks such as the exploitation of authentication, sniffing, spoofing, resource manipulation, etc. [2]. These security challenges become more critical in cloud computing as resources and services are shared in an open environment between and within networks. Moreover, the user is storing data in third-party locations remotely, so its security is essential [3, 4]. Therefore, the users' data should be encrypted using cryptography algorithms to protect them against unauthorised accesses and to maintain the confidentiality and integrity of the data. Several cryptographic techniques have been used to protect data in different deployment models of cloud computing: public, private, and hybrid models and different cloud services models [5]. When cloud service providers and governments provide several concurrent services for large volumes of data (i.e., Big Data), security becomes even more critical, and guaranteeing data protection, especially its integrity, becomes a difficult task [6]. Thus, it is necessary to develop new techniques to improve security with low impact on performance and high execution time [7]. However, as cryptographic techniques are usually complex and have an overload on the processes and cloud environment (e.g., data in transit), we need better encryption solutions with the lowest possible impact on performance, hence less negative impact on the services, while providing high security and data protection. Therefore, this study proposes a hybrid cryptography solution based on an improved Blowfish algorithm.

The paper is structured as follows: the second section explains the research background and is followed by the new solution, which is explained in the third section. Finally, the proposed solution is evaluated in the fourth section, and the paper culminates with a conclusion based on the discussion thus far.

II. Research Background

Because of the importance of security in cloud computing, many studies have been conducted on this topic. For instance, AbdElminaam [8] proposed a hybrid cryptography technique using AES (Advanced Encryption Standard) and Blowfish. They suggest that using a combination of symmetric and asymmetric techniques will provide a high cloud security. Their results show a lower encryption time and better throughput then other techniques such as using combination of AES and RSA (Rivest–Shamir–Adleman) [9]. Thabit, et al. [10] proposed a lightweight cryptography technique for cloud computing security that uses a 128-bit block cipher and key to encrypt the data. They used Feistel and substitution permutation architectural methods to make the encryption more complex. The proposed algorithm has “flexibility in the length of the secret key and the number of turns”, and the results show a low encryption time. These solutions can improve security while decreasing the encryption time, but the problem with the proposed techniques is that they need to exchange the encryption key which can make confidentiality, and as a result the whole could computing security, risky. Another study [11] utilized different cryptography techniques to improve the cloud computing security. For this purpose, an AES based encryption and asynchronous key system is consolidate in their model. They also used an Elliptic curve cryptography to secure the communications between client and cloud storage system. The proposed solution randomly generates keys based on chaotic cryptography. However, it seems that the proposed technique did not consider the time needed for split/combine of data.

There are, however, studies that use other new techniques to improve the security of cloud computing. For example, Esposito, et al. [12] suggested using blockchain for security of cloud computing. In the proposed blockchain-based ecosystem for new data a new block is instantiated and distributed to all peers in the network, and the system will insert it in the chain when majority of the peers approve the new block. This solution can provide a good level of cloud computing security as blockchain is secure by design. Although the blockchain-based solution has many benefits such as high level if data integrity as, for instance, when data stored in the chain it is not possible to alter it, there are some challenges with this solution. For example, it might breach data protection regulations (such as GDPR[1]) as personal sensitive data shall be stored for the "shortest time possible" and a time limit to erase the data must be established [13, 14]. Moreover, as it needs to be approved by more than half of the peers, it will not be possible to undo the unwanted changes or fix mistakes. Another issue with the proposed solution is that blockchain is designed to store transactional data which are small, so the solution does not perfectly work for securing all cloud computing data as many of them are large data such as images. Another study evaluated the performance and effectiveness (e.g. execution time and memory consumption) of different symmetric cryptographic algorithms [15]. There results show that Blowfish and DES (Data Encryption Standard) are better in terms of required encryption/decryption time and memory. Some studies reviewed the most important and efficient cryptographic method for cloud computing security [5, 16]. The results of the previous studies show a need for new could computing techniques that can improve security and at the same time have no or as less as possible negative effect on performance, to make cloud computing more secure with no effect on its services.

III. Proposed Solution

The proposed solution is an effective technique that uses different cryptographic based data protection algorithms. The solution used symmetric and asymmetric encryption combined with an improved digital signature using an MD5[2] hashing function to ensure data integrity. It is necessary to consider the algorithm's encryption speed, and the balance between performance and speed. Asymmetric key generation algorithms (i.e., public key) need more generation time than symmetric algorithms such as Blowfish, so they have a lower speed. Thus, we use an improved Blowfish (symmetric key generation) algorithm for encrypting the data in this solution. Fig. 1 illustrates a high-level diagram of the solution.

First, the hash code of the original data will be generated using MD5, which will be used to verify data integrity. In the second step, an Elliptic Curve (EC) algorithm will be used for digital signature generation and securing the MD5 code and private key. We need to have a high level of security and low length of the key and hash code, and as a result, the execution time of the EC will be decreased. Finally, the original data will be encrypted using an improved Blowfish algorithm, a symmetric algorithm with low encryption/decryption time. In the following sections, each step of the solution is described in detail.

A. Digital Signature

This study used a digital signature together with a hashing function to assure data integrity. However, as the initial digital signature model is vulnerable to cyberattacks (i.e., the cyber attacker can create a fake digital signature by manipulating the digital signature verification process), in this solution, we first hash the message and subsequently sign the hashed message. Thus, creating a fake signature does not work for the attacker as the signature does not match the hashed message's outcome, and the attacker cannot manipulate or damage the content of the message. Therefore, using a digital signature can be considered a robust tool for securing cloud computing [17]. Fig. 2 shows the flowchart of using a digital signature and hash code in the proposed solution.

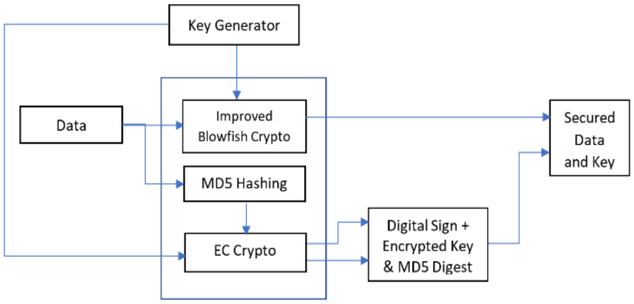


Fig. 1. High-Level Diagram of the Proposed Solution.

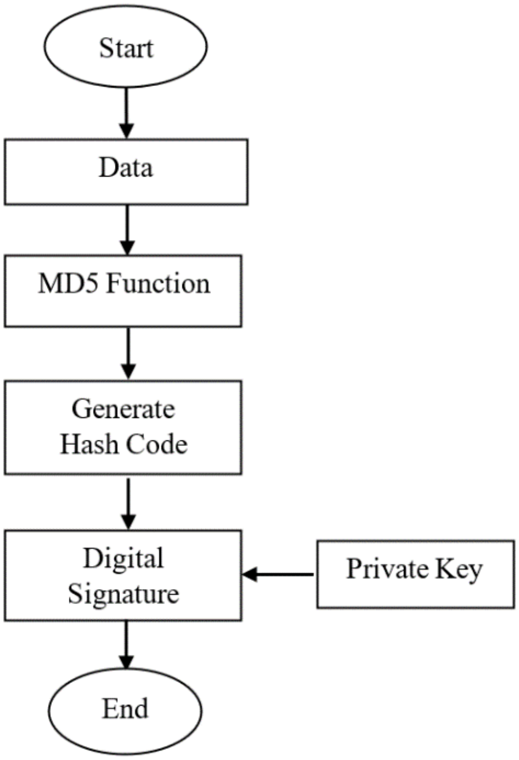


Fig. 2. Hash Code and Digital Signature Flowchart.

[1] General Data Protection Regulation is European personal data protection regulation.

[2] Message-digest algorithm

B. Data Encryption

This study aims to propose a secure cryptography technique with low execution time and high throughput, and we used an improved Blowfish algorithm in the core of the data encryption. This algorithm uses a variable key length from 32 up to 448-bits and used several subkeys [18]. The subkeys must be pre-computed before encryption/decryption of the data. This algorithm is much faster than other algorithms and requires less memory. The blowfish algorithm uses four 256 S-BOX containing a total of 1024 32-bit entries. The first byte from the first 32-bit entry will be used to find an entry in the first S-BOX, the second byte to find an entry in the second S-BOX and the same for all other entries (Modified blowfish algorithm). The same goes for the decryption procedure, starting with the cyphertext as an input, except that the subkeys will be used in a reverse way.

The F-function is the most time-consuming part of the encryption as in all rounds of the Blowfish algorithm, the F-function does the main calculation, including Adder and Rotation, in a modular format. Thus, in this study, we reduce the Blowfish's execution time by changing the F-function's module. Fig. 3 shows the overall process of the F-function module in a standard Blowfish.

To improve the algorithm, we decrease the complexity of execution time. Equation (1) calculates the value of F-function in a standard Blowfish.

$$F(X_L) = ((S_{1,a} + S_{2,b} \bmod 2^{32}) \text{ XOR } S_{3,c}) + S_{4,d} \bmod 2^{32} \quad (1)$$

We can change the F-function to equation (2), without reducing the security of the Blowfish algorithm.

$$F'(X_L) = (S_{1,a} + S_{2,b} \bmod 2^{32})\ (S_{3,c} + S_{4,d} \bmod 2^{32}) \quad (2)$$

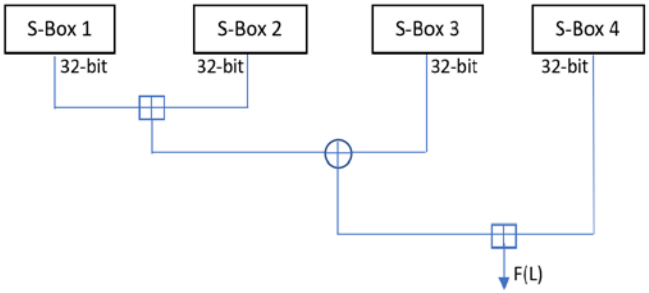


Fig. 3. F-Function Module in Blowfish.

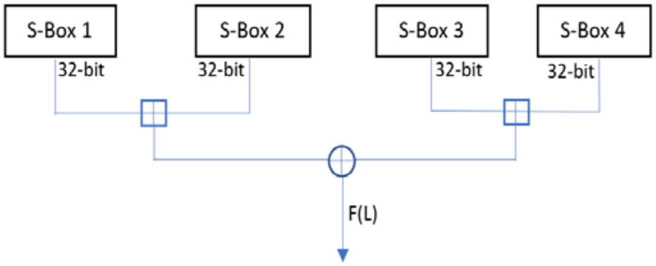
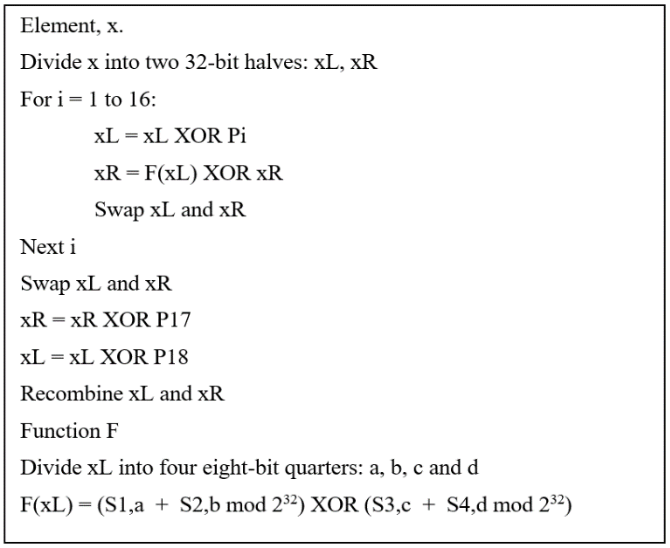


Fig. 4. The Improved F-Function Module.

This change will allow us to do both the ($S_{1,a}$ + $S_{2,b}$ mod 2^{32}) and ($S_{3,c}$ + $S_{4,d}$ mod 2^{32}) addition operations in parallel. This parallel operation will reduce the execution time as it will reduce the needed time for two operations to one operation. As Blowfish has 16 rounds, this change should improve the execution time 16 times in each encryption and decryption. Furthermore, as the security of Blowfish is related to its keys, this change will not negatively impact the algorithm's security. Fig. 4 illustrates the mentioned change.

Pseudocode:

Pseudocode for the Blowfish algorithm based on the improved F-function



```
Element, x.
Divide x into two 32-bit halves: xL, xR
For i = 1 to 16:
        xL = xL XOR Pi
        xR = F(xL) XOR xR
        Swap xL and xR
Next i
Swap xL and xR
xR = xR XOR P17
xL = xL XOR P18
Recombine xL and xR
Function F
Divide xL into four eight-bit quarters: a, b, c and d
F(xL) = (S1,a + S2,b mod $2^{32}$) XOR (S3,c + S4,d mod $2^{32}$)
```

First, the 64-bit entry splits into two left (L) and right (R) 32-bit. The next step is a XOR operation on the first 32-bit block (L). In the third step, the calculated 32-bit data will be transferred to F-function to be XOR'd with the other 32-bit block (R). Then, the L and R will be swapped to be used in the next Blowfish rounds. The decryption operation is the same except that the P1, P2, P18 will be used in a reverse way.

C. Securing the Symmetric-Key

Symmetric-key based algorithms, such as Blowfish, need to transfer the private key, which can be risky, as attackers might steal the key (e.g., man in the middle attacks). Thus, we use an elliptic-curve-based asymmetric cryptography algorithm to secure the private key and hash code in this solution. In this method, a 164-bit key will be used to provide a higher performance than other solutions [19]. The elliptic-curve-based cryptography algorithms have a high level of security and need low memory and bandwidth [20]. One of the benefits of this encryption is cryptography over finite fields. So, the assumption is that the p is a prime number, and the finite field contains a set of numbers less than p. Following is the explanation of the two-dimensional Elliptic curve (E).

If the p>3 is an odd prime, and $a, \in Fp$ and the $4a^3 + 27b^2 \neq 0 \bmod p$, then the Elliptic curve E(Fp) is equal to the equation (3).

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

Using element Q, which can create a group in the E(Fp) set, and considering p=(x1,y1) and $\{p, Q\} \in E(F_p)$

For point adding $P + Q = (x_3, y_3) and\ P \pm Q$

$x_3 = \lambda^2 - x_1 - x_2$

$y_3 = \lambda(x_1 - x_3) - y_1$

$\lambda = (y_2 - y_1)/(x_2 - x_1)$

For point doubling $P + P = 2P(x_3, y_3)$

$x_3 = \lambda^2 - 2x_1$

$y_3 = \lambda(x_1 - x_3) - y_1$

$\lambda = (3x_1^2 + a)/2y_1$

Adding two different points on the elliptic curve needs six additions, one square, two multiplications, and one inverse operation in Fp. Also, point doubling on the elliptic curve needs eight additions, two square, two multiplications, and one inverse operation. For cryptography using the elliptic curve, we first choose a random k number in a range that belongs to the field and consider it as the private key. Then, the public key Q will be calculated by Q=kP. In this situation, P is a point on the elliptic curve. The elliptic-curve cryptography is based on the difficulty of the discrete logarithm problem, so calculating k from the P and Q points has computational complexity, which is one of the main benefits of elliptic-curve cryptography. This method uses the following scalar multiplication.

$$Q = k.P = P + P + ... + P \tag{4}$$

Because the operations in this method are based on the field, using Modular Multiplication can help to improve the performance of elliptic-curve cryptography. Fig. 5 shows the flowchart of this method, and the following steps describe the method in detail.

Step 1. Before the encryption, a hash code (using MD5) for the original data will be generated. This code will be used to improving the efficiency of the digital signature and data integrity verification.

Step 2. The hash code will be encrypted using the private key, and the digital signature of the data will be issued.

Step 3. The Blowfish's private key will be encrypted using elliptic-curve cryptography.

Step 4. The data will be encrypted by the symmetric cryptography algorithm (Blowfish). We use the symmetric key to encrypt the original data. Then, the encrypted data in this step and the previous steps will be transferred.

Step 5. The receiver decrypts the received data using a reverse process and the private key.

Step 6. The original data will be decrypted using Blowfish's private key, and the verification and validation process will be run using the hash function (digital signature).

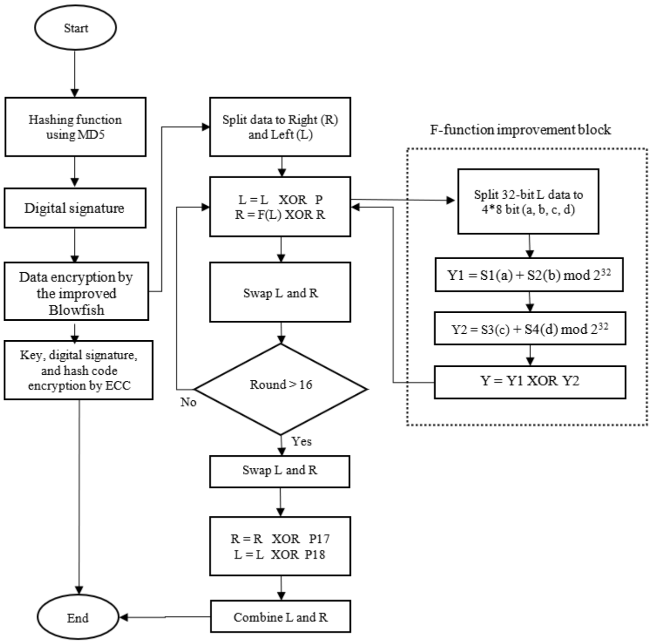

Start
Hashing function using MD5
Digital signature
Data encryption by the improved Blowfish
Key, digital signature, and hash code encryption by ECC
Split data to Right (R) and Left (L)
L = L XOR P
R = F(L) XOR R
Swap L and R
Round > 16
No
Yes
Swap L and R
R = R XOR P17
L = L XOR P18
Combine L and R
End
F-function improvement block
Split 32-bit L data to 4*8 bit (a, b, c, d)
Y1 = S1(a) + S2(b) mod 2^32
Y2 = S3(c) + S4(d) mod 2^32
Y = Y1 XOR Y2

Fig. 5. The Proposed Solution's Flowchart.

IV. Evaluation and Discussion

We implemented the proposed solution on Eclipse and by using Java development kit (JDK) version 7. The cryptography functions are available in two Java Cryptography Architecture (JCA) and Java Cryptography Extension (JCE) libraries in the JDK. The JCA provides most of the fundamental cryptography requirements, while the JCE can be used for advanced cryptography operations. The hardware used in the solution evaluation had an Intel Core2 Duo 2.5 GHz processor. All the evaluations were run in Windows 7.

Different parameters such as memory use, throughput, and execution time of the proposed solution were compared with AES, 3DES, DES, and RSA algorithms [21]. We initially evaluated the solution with a low size data (50 KB) to analyse its performance when the data is small and then evaluated the solution with larger data (i.e., 1024 KB and 2048 KB).

A. *Evaluation Results – Throughput and Execution Time*

The first evaluation is based on 50KB data. As fig. 6 illustrates, the execution time of the proposed technique (Hybrid) reduced 800, 550, 300 and 400 milliseconds in comparison with RSA, AES, DES, and 3DES (in order). This means that the encryption process took less time due to using the Blowfish algorithm (which is a fast algorithm) and changing the F-function. Furthermore, the asymmetric EC algorithm had no impact on the execution time as it was only used to encrypt the key and digest it.

In Fig. 7, the throughput of different techniques, with 50 KB data, are compared. As it is presented, the throughput of the proposed technique is better than the others.

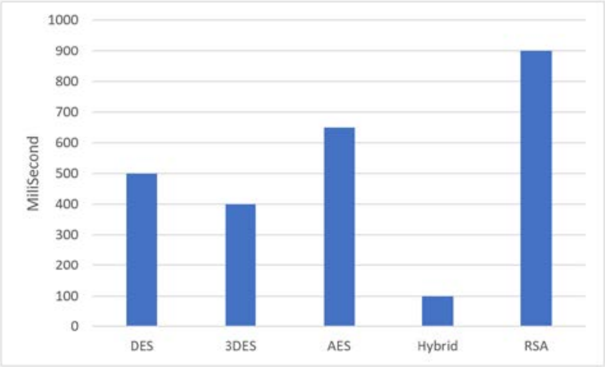


Fig. 6. Execution Time with 50KB Data.

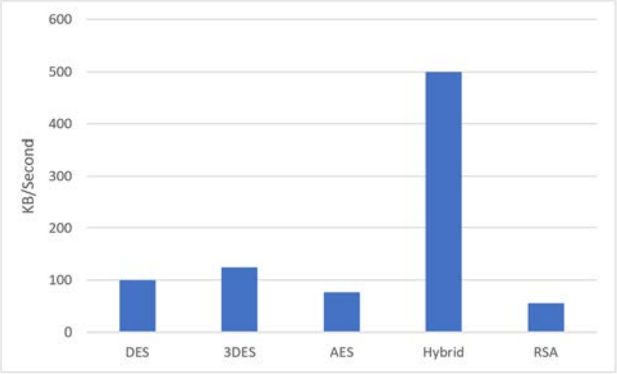


Fig. 7. Throughput with 50KB Data.

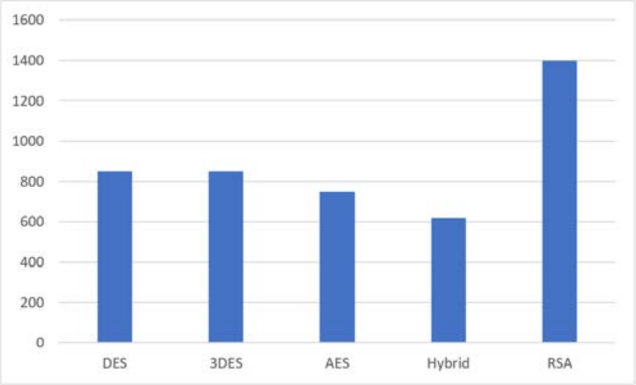


Fig. 8. Execution Time with 1024KB Data.

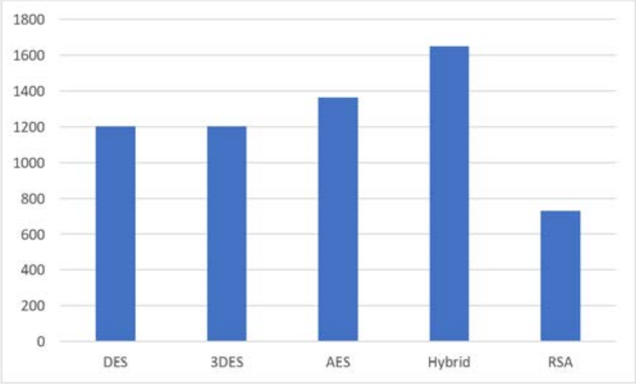


Fig. 9. Throughput with 1024KB Data.

Because the data size can impact the execution time, we increase the data size to 1024 KB to evaluate the solution with larger data. As shown in Fig. 8, the proposed solution has a lower execution time than the other techniques, even with a larger data size (i.e., the 1024 KB). The proposed technique (Hybrid) reduced 230, 230, 130 milliseconds in comparison with symmetric algorithms (i.e., DES, 3DES, and AES), and 731 milliseconds in comparison with the asymmetric algorithm (i.e., RSA). This shows that the proposed technique is faster than the other techniques.

Fig. 9 shows the throughputs of the selected techniques and the proposed solution with 1024 KB data. As shown, the proposed solution has better throughputs, which could be expected as the execution time of the solution is lower. The solution's throughput is, on average, 16% higher than the symmetric algorithms (AES, 3DES, and DES) and over 50% higher than the asymmetric algorithm (RSA).

Finally, we evaluate the selected algorithms with a 2049 data size. Fig. 10 and 11 show the execution time and throughput results. Increasing the data size increases the execution time, but as it was shown, the proposed solution has a lower execution time than DES, 3DES, and RSA. The execution time is 470ms lower than 3DES, 320ms lower than DES, and 670ms lower than RSA. However, the proposed solution's execution time does not work better than the AES. However, considering the other benefits such as the digital signature and hashing, we might ignore the increased execution time, as the time difference is likely to have a very low effect in the cloud computing environment. Future studies should evaluate the proposed solution in the cloud computing environment to determine how it works with larger data and if the mentioned benefits can overshadow the execution time (compared to the AES).

As Fig. 11 shows, the throughput of the proposed solution is higher than DES, 3DES, and RSA. This is because of the changes we applied to the F-function and, in fact, parallel processing in the core of Blowfish encryption. Furthermore, the data encryption and key encryption are separated in this solution, so parallel usage of Blowfish and EC did not affect the execution time and throughput as only private key hash codes are encrypted by EC, and because of their small size, this had almost no effect on the execution time. Moreover, the original data, which has a larger size, is encrypted by the improved Blowfish algorithm, which is faster and has a high performance.

Several studies proposed hybrid and multilevel encryption solutions and/or modified cryptography algorithms to improve cloud security [7, 22-26]. However, our proposed solution benefits using both symmetric and asymmetric algorithms together with a digital signature. Combining the utilised algorithms and improving the Blowfish’s F-function provides high data protection and signature integrity with high-performance encryption. However, like other cloud security methods, our proposed solution should be tested and evaluated in cloud computing environments with extensive data.

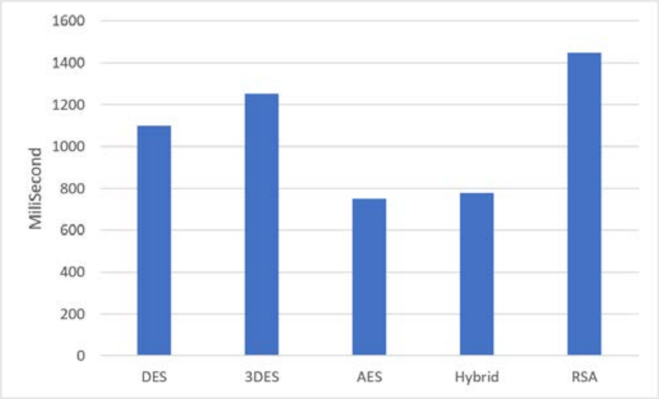


Fig. 10. Execution Time with 2048KB Data.

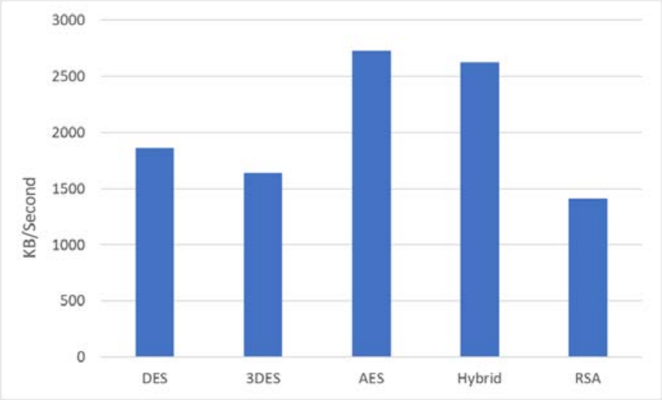


Fig. 11. Throughput with 2048KB Data.

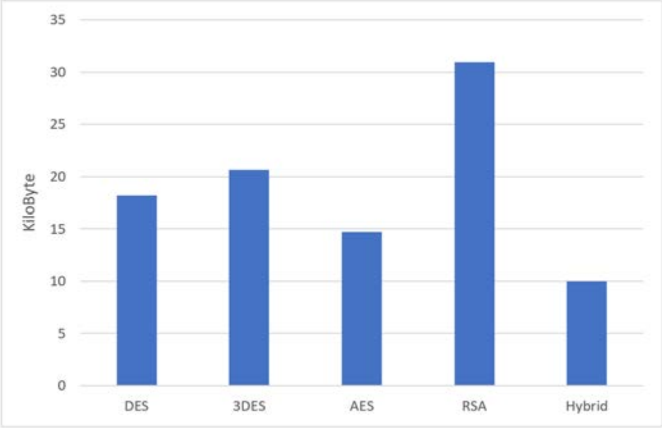


Fig. 12. Memory Usage.

B. Memory Consumption

Memory consumption is one of the important metrics in evaluating encryption techniques. Fig. 12 shows the used memory in the proposed solution and other selected solutions. The proposed solution used lower memory in comparison with the other techniques. The main factors that impact memory usage are the number and type of operations, size of the key and initialisation vectors [21], so we tested the memory consumption with 50KB data only. The lower memory usage of the proposed solution is because of using the Blowfish algorithm as one of the most memory-efficient cryptography algorithms [21]. The other reason for lower memory usage is applied changes in the core of Blowfish in the proposed solution. In addition, using EC, which uses a 164-bit key that needs a low memory [27], will allow us to use the proposed solution in infrastructures with low memory resources, such as old ones.

V. Conclusion

This study developed an improved cryptography solution that can provide efficient and high-performance, secure cloud computing. Because the core of this solution is based on an improved Blowfish (a symmetric algorithm) which decreases the needed time for encryption, the execution time and throughput of the solution are improved. Moreover, using the EC asymmetric cryptography algorithm to encrypt the key, reduces the security challenges of symmetric key exchange algorithms, such as stealing the key in transit. In addition to the mentioned benefits, the proposed solution uses MD5-based digital signatures to provide data integrity. The solution evaluation shows an overall improvement in comparison with AES, DES, 3DES, and RSA. The throughput, memory usage, and execution time of the proposed solution were, on average, better than the other solutions. However, we found that AES was slightly faster and had higher throughput when we increased the data size. We suggest that future studies test the solution in different infrastructures, especially cloud computing environments, with a high volume of data.

References

[1] N. Subramanian and A. Jeyaraj, "Recent security challenges in cloud computing," Computers & Electrical Engineering, vol. 71, pp. 28-42, 2018.

[2] H. Tabrizchi and M. K. Rafsanjani, "A survey on security challenges in cloud computing: issues, threats, and solutions," The journal of supercomputing, vol. 76, no. 12, pp. 9493-9532, 2020.

[3] Z. Balani and H. Varol, "Cloud Computing Security Challenges and Threats," in 2020 8th International Symposium on Digital Forensics and Security (ISDFS), 2020: IEEE, pp. 1-4.

[4] A. Orobosade, T. Aderonke, A. Boniface, and A. J. Gabriel, "Cloud application security using hybrid encryption," Communications, vol. 7, pp. 25-31, 2020.

[5] V. Agarwal, A. K. Kaushal, and L. Chouhan, "A Survey on Cloud Computing Security Issues and Cryptographic Techniques," in Social Networking and Computational Intelligence: Springer, 2020, pp. 119-134.

[6] S. Ray, K. N. Mishra, and S. Dutta, "Big Data Security Issues from the Perspective of IoT and Cloud Compu-ting: A Review," Recent Advances in Computer Science and Communications, vol. 12, pp. 1-00, 2020.

[7] K. Sajay, S. S. Babu, and Y. Vijayalakshmi, "Enhancing the security of cloud data using hybrid encryption algorithm," Journal of Ambient Intelligence and Humanized Computing, pp. 1-10, 2019.

[8] D. S. AbdElminaam, "Improving the security of cloud computing by building new hybrid cryptography algorithms," International Journal of Electronics and Information Engineering, vol. 8, no. 1, pp. 40-48, 2018.

[9] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Communications of the ACM, vol. 21, no. 2, pp. 120-126, 1978.

[10] F. Thabit, S. Alhomdy, A. H. Al-Ahdal, and S. Jagtap, "A new lightweight cryptographic algorithm for enhancing data security in cloud computing," Global Transitions Proceedings, vol. 2, no. 1, pp. 91-99, 2021.

[11] A. Hussain, C. Xu, and M. Ali, "Security of cloud storage system using various cryptographic techniques," Int. J. Math. Trends Technol, vol. 60, no. 1, pp. 45-51, 2018.

[12] C. Esposito, A. De Santis, G. Tortora, H. Chang, and K.-K. R. Choo, "Blockchain: A panacea for healthcare cloud-based data security and privacy?," IEEE Cloud Computing, vol. 5, no. 1, pp. 31-37, 2018.

[13] EC. "For how long can data be kept and is it necessary to update it?" https://ec.europa.eu/info/law/law-topic/data-protection/reform/rules-business-and-organisations/principles-gdpr/how-long-can-data-be-kept-and-it-necessary-update-it_en (accessed 18/05/2021, 2021).

[14] ICO. "Principle (e): Storage limitation." https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/principles/storage-limitation (accessed 18/05/2021, 2021).

[15] A. R. Wani, Q. Rana, and N. Pandey, "Performance Evaluation and Analysis of Advanced Symmetric Key Cryptographic Algorithms for Cloud Computing Security," in Soft Computing: Theories and Applications: Springer, 2019, pp. 261-271.

[16] B. UMAPATHY and D. KALPANA, "A SURVEY ON CRYPTOGRAPHIC ALGORITHM FOR DATA SECURITY IN CLOUD STORAGE ENVIRONMENT," European Journal of Molecular & Clinical Medicine, vol. 7, no. 09, p. 2020.

[17] I. Ahmed, "A brief review: security issues in cloud computing and their solutions," Telkomnika, vol. 17, no. 6, 2019.

[18] M. R. Asassfeh, M. Qatawneh, and F. M. AL-Azzeh, "Performance evaluation of blowfish algorithm on supercomputer iman1," International Journal of Computer Networks & Communications (IJCNC), vol. 10, no. 2, 2018.

[19] S. Chandra, S. Paira, S. S. Alam, and G. Sanyal, "A comparative survey of symmetric and asymmetric key cryptography," in 2014 international conference on electronics, communication and computational engineering (ICECCE), 2014: IEEE, pp. 83-93.

[20] T. Wollinger, J. Pelzl, V. Wittelsberger, C. Paar, G. Saldamli, and Ç. K. Koç, "Elliptic and hyperelliptic curves on embedded μP," ACM Transactions on Embedded Computing Systems (TECS), vol. 3, no. 3, pp. 509-533, 2004.

[21] P. Patil, P. Narayankar, D. Narayan, and S. M. Meena, "A comprehensive evaluation of cryptographic algorithms: DES, 3DES, AES, RSA and Blowfish," Procedia Computer Science, vol. 78, pp. 617-624, 2016.

[22] A. PanimalarS, N. Dharani, R. Aiswarya, and P. Shailesh, "Cloud Data Security Using Elliptic Curve Cryptography," 2017.

[23] V. K. R. Gangireddy, S. Kannan, and K. Subburathinam, "Implementation of enhanced blowfish algorithm in cloud environment," Journal of Ambient Intelligence and Humanized Computing, pp. 1-7, 2020.

[24] U. Gupta, M. S. Saluja, and M. T. Tiwari, "Enhancement of Cloud Security and removal of anti-patterns using multilevel encryption algorithms," International Journal of Recent Research Aspects, vol. 5, no. 1, pp. 55-61, 2018.

[25] A. Chauhan and J. Gupta, "A novel technique of cloud security based on hybrid encryption by Blowfish and MD5," in 2017 4th International conference on signal processing, computing and control (ISPCC), 2017: IEEE, pp. 349-355.

[26] V. Saranya and K. Kavitha, "A modified blowfish algorithm for improving the cloud security," Elsiyum J, vol. 4, no. 3, pp. 1-6, 2017.

[27] D. Mahto and D. K. Yadav, "Performance Analysis of RSA and Elliptic Curve Cryptography," IJ Network Security, vol. 20, no. 4, pp. 625-635, 2018.