AL-RYADA JOURNAL FOR COMPUTATIONAL INTELLIGENCE AND TECHNOLOGY
(ARJCIT)

VOL. 1 (1) - 2024

A Comprehensive Survey on Digital Rights Management Systems (DRM) and Advanced Encryption Techniques

Dina M. Alhamalawy [a,b], Gamal M. Attiya [b], Zeiad M. Abdoun [b], Mokhtar A. A. Mohamed [b,c]

[a] *Comp. Systems Eng. Dept., Faculty of Engineering, MSA University, Egypt.*
[b] *Comp. Sci. & Eng. Dept., Faculty of Electronic Engineering, Menoufia University, Egypt.*
[c] *Faculty of Computers & Al, Ryada University for Science and Technology, Egypt.*

Article history:

Received: 14 – Sept – 2024

Revised: 14 – Sept – 2024

Accepted: 24 – Oct – 2024

Available online: 1 – Nov – 2024

DOI: 10.21608/ajcit.2024.320988.1006

Correspondence

Dina M. Alhamalawy,

Faculty of Electronic Engineering (FEE), Menofia University , Egypt.

Email: dinamagdy.1995@gmail.com

ABSTRACT:

The large conversion and revolution in the information world was accompanied by a wave of digital transformation which led us to various information formats, including written documents, images, videos, and audio files. This diversity has highlighted the need for ways to protect copyright and intellectual property rights to prevent unauthorized use and distribution. The DRM System has become an important tool for managing and protecting content by imposing usage rules and preserving the intellectual property rights of content creators. This research paper aims to discuss the evolution of the recent DRM systems, focusing on the traditional encryption algorithms they use and their vulnerability to hacking. The study aims to determine the strengths and weaknesses of each DRM system and recommends replacing traditional encryption methods with newer, more secure algorithms that are less known within the hacking community to improve overall system security and shows the best asymmetric algorithm from the recent survey papers to recommends it for key encryption to add another layer of security. Through a comprehensive review of existing DRM frameworks and an analysis of advanced cryptographic algorithms such as NLCA, SLIM, TIGRIS, and ISL-LWC, this paper provides insights into the future of DRM systems in safeguarding digital content.

KEYWORDS

digital right management; encryption algorithms; content protection ;

`

1. INTRODUCTION

Because of the recent global technological revolution and digital transformation, data is now produced digitally, and it can be presented in many different forms such as Written data, for example, Word, PDFs, TXT, and source code files, as well as videos, pictures, and audio. This data requires to have its copyright and intellectual property protected by the publishers, whether they are individuals or institutions. Because of increasing the population of using the different platforms of social media, it has become accessible for others to publish this data without the publisher's permission and to gain from it morally or materially without referring to him. In this manner, there was a requirement for a framework that oversees this information and circulates it to those who are entitled, and with what they are entitled to, whether it is copying, recording, or the like, and with what the publisher permits in terms of circulation rights for the data that has been distributed so The publisher therefore lacks complete control over the dissemination of content on the internet and social media, including the ability to govern all rights related to recording, copying, and other uses. As a result, it is essential for there to be an automated system developed to handle the control procedure for data access. We addressed the DRM system in this study article, but before we can do so, let's clarify what the DRM system is and how it operates. & How is the necessary security met, as well as the publisher's intellectual property and copyright, maintained for the data he publishes?

Digital rights management (DRM) is the use of technology to control and manage access to copyrighted material. Another DRM meaning is taking control of digital content away from the person who possesses it and handing it to a computer program. DRM aims to protect the copyright holder’s rights and prevents content from unauthorized distribution and modification.

The DRM system consists of three main entities, and they are user, rights and digital content. The user can be publisher, author or customer. As a result, there are differences in what the user can accomplish on the system. If the user was a publisher, writer, or other content producer, he could submit his work to the framework and specify the usage rules for it as well as the payment he would like to receive in exchange for this access. As a consumer, the user could purchase the rights to view the content he desires, depending on the publisher's granted access privileges. videos, audio, text files, and photos can all be included in the digital content. The creator's defined access rules determine whether or not it can be accessed. The access usage rules include many types of rules [1] like:

- Software Protection: Protect the software system from the reverse engineering.
- Date, location, count times expiry & self-destruct: Set expiry dates on media, which automatically prevents access, allow the access in specific location or limits the number of times they can access it.
- Prevent editing, copying, saving and printing: Restrict users from editing or saving, sharing, or forwarding, printing the uploaded documents.
- Encryption & decryption in RAM: The encryption technique must apply in RAM on materials to make it difficult to attack or hack.
- PDF & video protection: Converting unprotected content to a non-readable, encrypted format that can then only be unencrypted through our application.
- Password protection: The user must know a unique password to access a document.
- Device locking: Prevents users from opening a file unless they are on an approved device.

`

- Prevent screenshots & recording: Digital material automatically blocked or hidden in screenshots and screen shares.
- Watermark: Watermark documents and images to assert ownership and identity of content.

The DRM system consists of two modules, and they are content protection and right management. Because digital content uploaded to the framework is saved on the server until the customer requests access, and because the content is downloaded onto the customer's device, all of these processes put the content file at risk of being viewed or used by unauthorized users. For this reason, content protection is necessary to ensure that, even if it is found, it cannot be understood or translated. We must thus encrypt these files using a powerful encryption method that is unbreakable and keys that are unexpected and impossible to obtain. Two different kinds of encryption algorithms—symmetric and asymmetric—are used for this stage of the procedure [2,3]. As shown in figure 1, one technique that uses a single key for both encryption and decryption is the symmetric encryption algorithm. This is the best approach to encrypt content during the transmission procedures and save it on the server and consumer devices because it is thought to be more efficient and faster at encrypting blocks and huge amounts of data. Its speed can be attributed to the simple reason that two keys—one for encryption and the other for decryption—never need to be generated.

Using two keys—one for the encryption process and the other for the decryption algorithm—is a strategy used in asymmetric encryption techniques. Since this method is slower than symmetric encryption but safer overall, it is most commonly used for small-data encryption, such as symmetric encryption keys. A detailed comparison of the symmetric and asymmetric algorithms is provided in table 1.

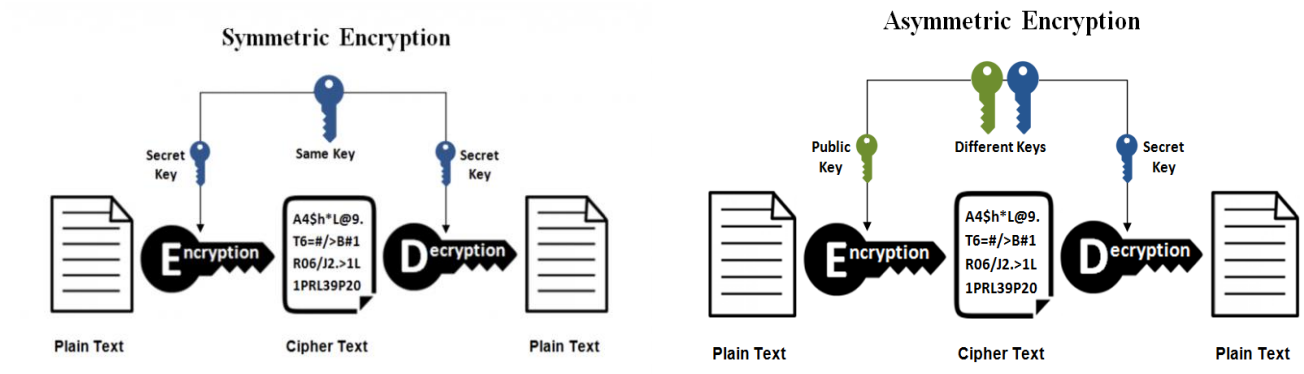


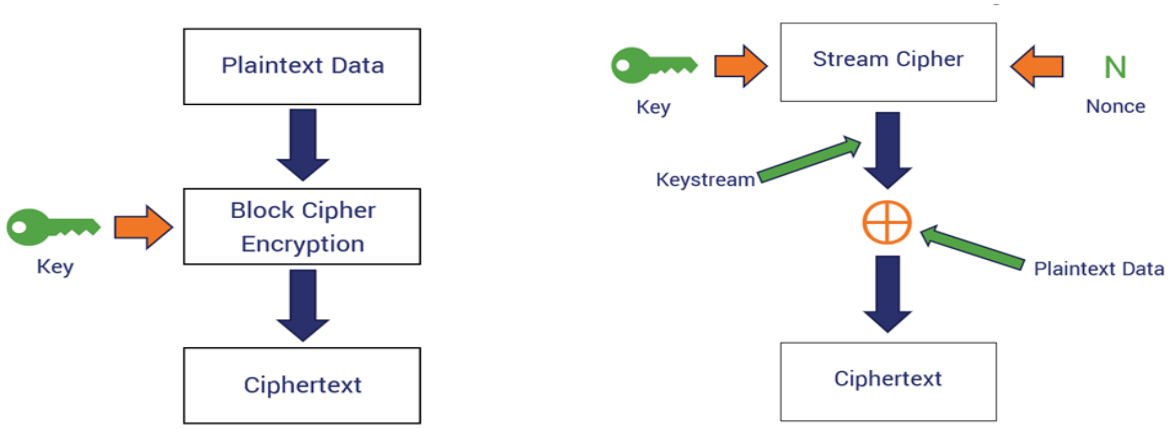
Figure 1: Symmetric vs Asymmetric encryption techniques [2].

Symmetric encryption algorithms [2] can be categorized into two types: block and stream symmetric encryption algorithms. As illustrated in Figure 2, both types use the same key for both encryption and decryption. The key difference lies in their approach: block symmetric encryption algorithms encrypt large amounts of data by processing it in fixed-size blocks, typically ranging from 64 to 128 bits, using a key of specific length and multiple steps to complete the encryption process. In contrast, stream symmetric encryption algorithms encrypt data byte by byte, utilizing a key stream and the XOR operation to produce the ciphertext. A detailed comparison of block and stream symmetric encryption algorithms can be found in Table 2.

`

Table 1: Comparison between symmetric and asymmetric encryption techniques [2].

	Symmetric	Asymmetric
Data Size	Used to transmit big data and efficient for secure large files exchange.	Used to transmit small data and efficient for secure key exchanges and digital signatures.
No. of Keys	It only requires a single key for both encryption and decryption.	It requires two keys, a public key and a private key, one to encrypt and the other to decrypt.
Speed	Very Fast Processing.	Slow Processing.
Key Lengths	The length of key used is 128 or 256 bits	The length of key used is 2048 or higher
Security	Security is lower as only one key is used for both encryption and decryption purposes.	Security is higher as two keys are used, one for encryption and the other for decryption.
Algorithms	RC4, AES, DES, 3DES, and QUAD.	Diffie-Hellman, ECC, El Gamal, DSA and RSA.



(a) (b)

Figure 2: Block vs Stream symmetric algorithms (a) Block, (b) Stream [3].

Regarding right management module, the publisher determines and selects access usage rules from what are provided by the host system. When the customer signs in the system and buys the license of any content, the host system will send a license key to him. At any moment, the customer needs to access it, he/she should provide license key, and he/she can access or download an encrypted version of this content on his own device after system verify the license key. Content decrypting and usage rules will be done by the user-based platform of digital system application.

`

The related work will demonstrate that all modern DRM systems are built using traditional symmetric and asymmetric encryption techniques, which are well-known to the hacker community. This renders any DRM system vulnerable to hacking and causes it to lose its fundamental significance, meaning, and goal. In summary, this paper aims to provide a comprehensive overview of current DRM systems and encryption techniques, highlighting their strengths, weaknesses, and areas for future research.

The paper is organized as follows: Section 2 reviews the most recently published DRM systems; Section 3 recommends and reviews the most recently published encryption algorithms; Section 4 discuss the previous comprehensive analysis research studies about the asymmetric encryption algorithms to deduce the most appropriate algorithm to encrypt the keys and Section 5 concludes with potential future directions. With this context in mind, the following section takes an in-depth review of the latest DRM systems and goes into all the strengths and weaknesses of each to get the characteristics of powerful DRM systems.

Table 2: Comparison between block and stream symmetric algorithms [3].

	Block	**Stream**
Plain text size per time	• Convert data in plaintext into ciphertext in fixed-size blocks. • The block size (64-bit or 128-bit blocks). • If the plaintext length is not a multiple of 8, the encryption scheme uses padding to ensure complete blocks.	• Convert data in plaintext into ciphertext in bit or byte of data.
Speed	Slow Processing.	Fast Processing.
Key Lengths	Key size can be 128, 192, 256 and 512 bits	Infinite stream of pseudorandom bits as the key.
Reversibility	Reverse encrypted text is hard.	Reverse encrypted text is easy.
Techniques	Works on transposition techniques like rail-fence technique, columnar transposition technique, etc.	Works on substitution techniques like Caesar cipher, polygram substitution cipher, etc.
Algorithms	DES, 3DES, AES, TwoFish, Serpent, and Blowfish.	RC4, SEAL, Salsa20, PANAMA, Scream, Rabbit, HC-256, and Grain.

2. RELATED WORK OF DRM SYSTEMS

In this section, we examine the existing literature on contemporary Digital Rights Management (DRM) systems. This review is crucial for gaining insight into the current landscape of DRM technologies, recognizing emerging trends, and identifying gaps that this survey seeks to address. The related work is structured to provide an overview of each system, outlining their features and limitations, and ultimately identifying the key characteristics that should be considered in the design of an ideal DRM system for future implementation.

`

2.1 Blockchain and Homomorphic encryption for digital copyright protection

This system is not actually a DRM system, it auctions system for the digital content that the copyright owner (or "publisher) offers his content and copyright demanders (or "customer") buys its copyright with the highest price. It was implemented in 2020 using blockchain framework with cloud and homomorphic encryption technique [4]. They introduce the blockchain for decentralization and non-tampering features and cloud for large data storage and homomorphic encryption techniques for data security in outsource cloud.

As shown in figure 3, the process begins when a copyright demander ("customer") seeks copyrighted content using keywords to search on the storage cloud platform for the desired content, which contains the content ciphertext that was encrypted using the copyright owner (or "publisher) public key. From the search results, the copyright demander (or "customer") selects to participate in a copyright auction hosted on the blockchain frame by the copyright owner (or "publisher"). After the copyright owner (or "publisher") selects the winner among the copyright demanders (or "customer"), the copyright owner uses the winner's public key to re-encrypt the data and uploads it to the cloud. The content that is encrypted is downloaded from the cloud by the winner copyright demander, who then uses their private key to decrypt it first. They built the project on the Truffle framework.

The system not only protects the copyrighted content but also protects the copyright owner and copyright demanders privacy. The transaction can be done without two parties knowing each other.

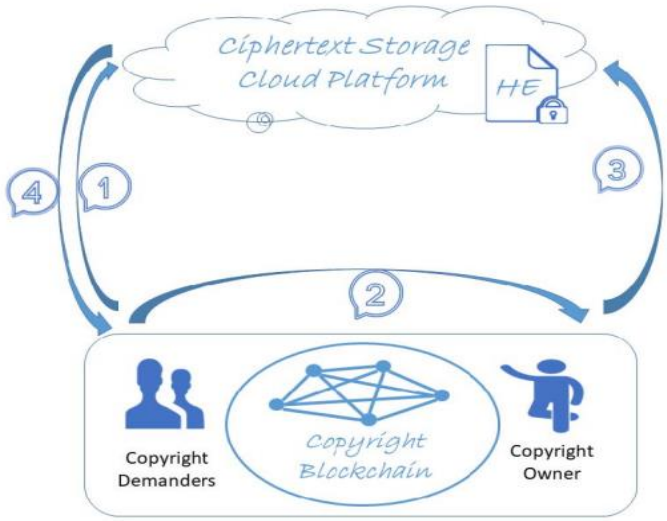


Figure 3: System structure flowchart [4].

Homomorphic encryption techniques are the encryption techniques that support homomorphism which is we can make operations on ciphertexts without need to decrypt it. It has three types are additive, multiplicative and full which is mix between additive and multiplicative. We can proof the additive homomorphic by taking two different plaintexts and their ciphertexts then add the ciphertexts together then decrypts the result it will give us the summation of the two plaintexts. In the same way we can proof the multiplicative and full homomorphic.

For data security, they use a homomorphic encryption technique for its important feature which is more suitable than traditional algorithms in cloud computing because we can share the sensitive data without decrypts it first. They improved exist homomorphic encryption algorithm called Paillier and introduce large prime number

`

(LPN) algorithm, which is better because it has more complex structure, Paillier is additive homomorphism and LPN is full homomorphism that mix the additive and multiplicative homomorphism. They use in LPN a high size of prime numbers that exceeds 512 bits which makes the algorithm uncrackable.

It was tested the two algorithms that had the same nature of homomorphism like LPN and Paillier to show which algorithm is more suitable to their application and the test was on key size 2048 bits and 2000 bit as a data block and they found that Paillier faster than LPN in key generation but LPN faster than Paillier in encryption and decryption processes as shown in figure 4.

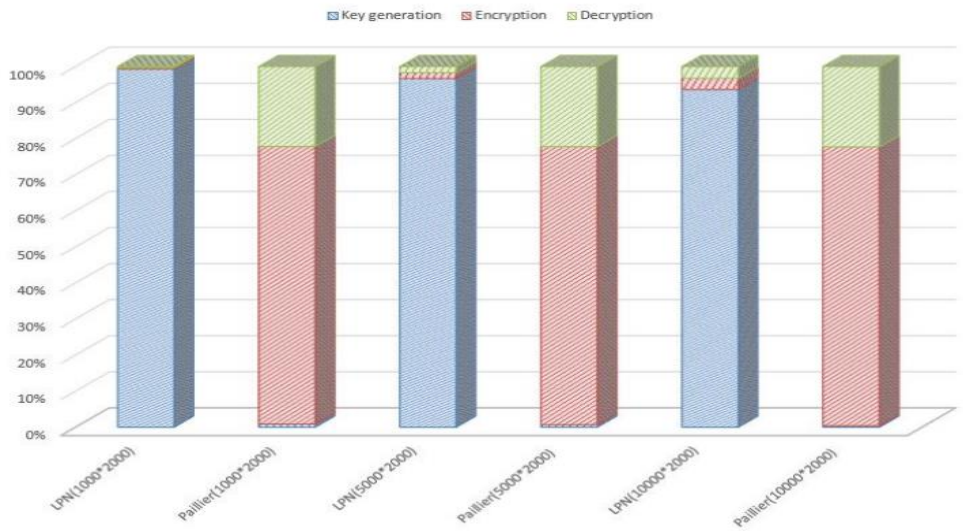


Figure 4: Comparisons between LPN and Paillier of time performance in key generation, encryption and decryption [4].

2.2 A robust computational DRM framework for protecting multimedia contents using AES and ECC

This DRM system was implemented in 2018 using Advanced Encryption Standard (AES-256) and Elliptic Curve Cryptography (ECC) as ciphering techniques [5]. AES-256 with 14 rounds is used to protect the content and ECC for protecting the ciphering keys. The DRM system intend to protect the content as a video (mp4, avi, mkv, 3gp, flv, m2ts) and audio (mp3) only that will be received by the reader as a file. As shown in figure 5, the system consists of the following parts: desktop application, web application and system administration. The desktop application that is installed on the customer's device receives the encrypted content in the.drm extension from the server, then checks the license in cooperation with the server. The desktop application begins in the process of playing the loaded encrypted video or audio in the memory RAM; it decrypts a fragment and plays it, then encrypts this fragment again, then decrypts another one, and so on.

The web application that allows the customer to explore the available content to buy the license also allows the publisher to share their content on the framework. For the license, it was shared with customer when he pays it from the website in JavaScript Object Notation (JSON) format and validated using ECDSA to authenticate the origin and integrity of the license. System administration is used to manage the users, contents, and devices as

`

adding, deleting and editing. The system is used to protect the content using the following usage rules: number of registered devices (up to 5 devices for each user), no copy, expiry date to access the data and no screenshots or recording and they allow the payments in the following modes: daily, weekly, monthly and unlimited.

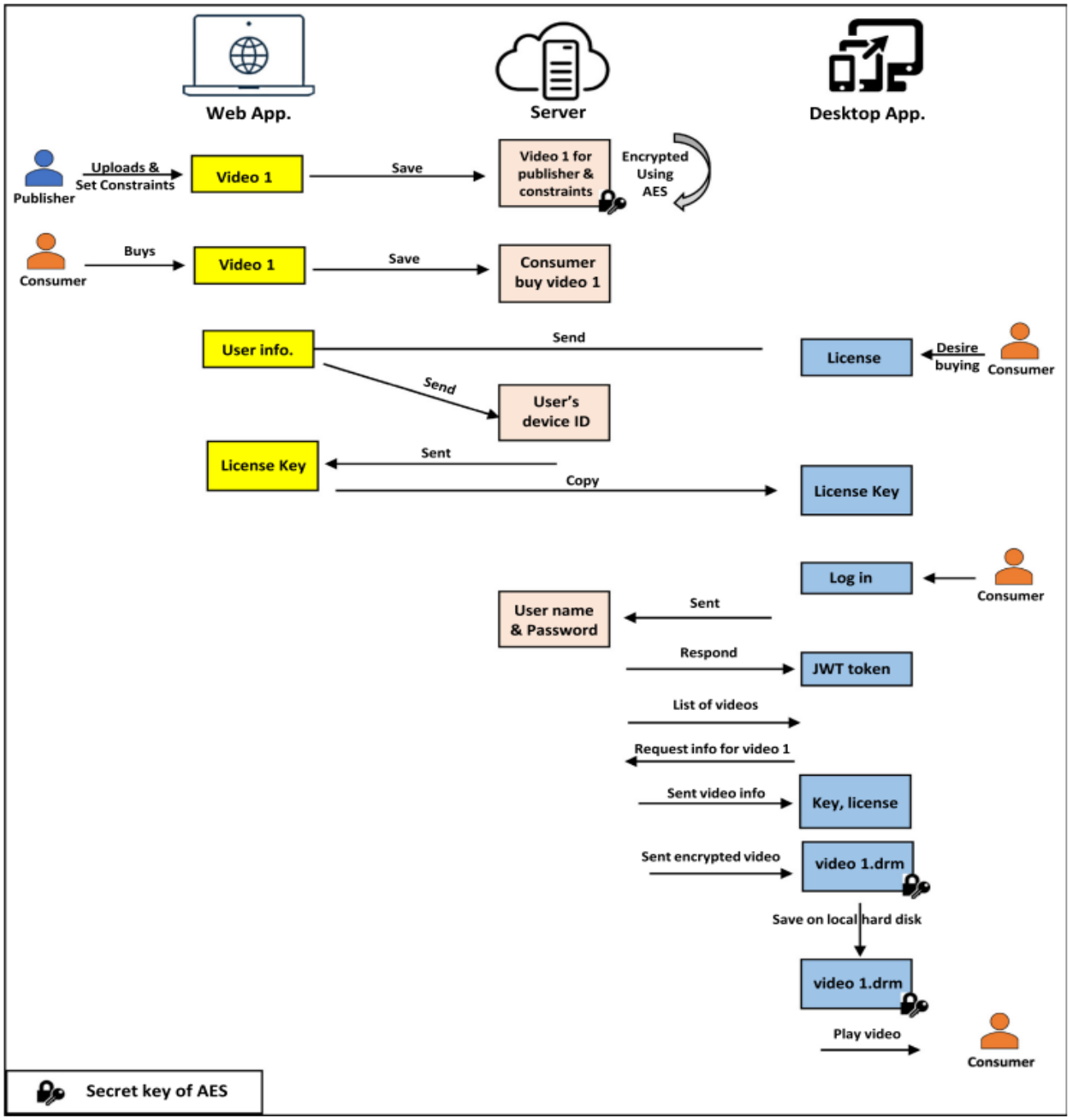


Figure 5: The detailed sequence of DRMs using AES and ECC [5].

`

They use Python programming language and Django for back-end system development, ReactJS framework for front-end development, and Java programming language and JavaFX for desktop application development. Figure 6 shows that the system takes less time in encryption and decryption processes against the previous system which makes it faster.

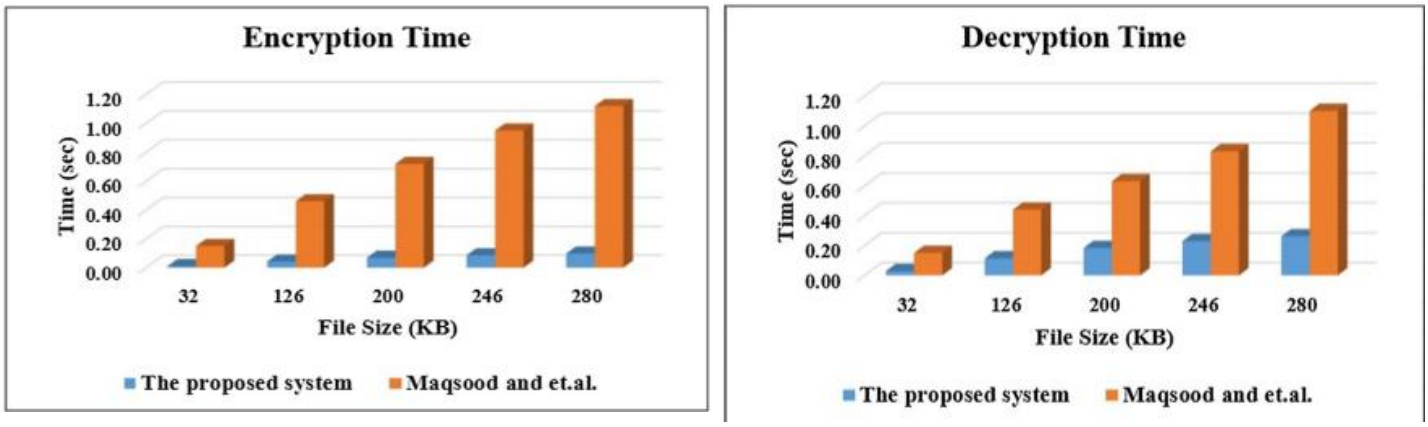


Figure 6: Encryption and decryption time for DRMs using AES and ECC and previous system [5].

2.3 A flexible and lightweight user-demand DRM system for multimedia contents over multiple portable device platforms

The DRM system was implemented in 2017 using AES, RSA and SHA-256 as a ciphering technique [6]. They proposed an architecture for DRM system that suitable to implement on any operating system (Windows, Linux, Mac OS, Android and IOS platforms). They intend to protect any type of files, but they implemented testing it with video files only. As shown in figure 7, the system consists of the following parts: DRM host and DRM client. The host, that is, the content provider, controls the content encryption and the issuing of the content key(s) where the client side, that is, the end device, shall have valid license and key(s) decryption capability to obtain through the media distribution.

The DRM host uploads the content on the system and issuing the content key that will be used to encrypt the content using AES. When the DRM client buy the license from the system, it will generate a license file that contain the devices IDs, expiry date, product key and content keys.

To generate license file, it will firstly protect the content keys it will be encrypted with the product key using AES-256 then the license content that mentioned previously grouped then hash the file using SHA-256 and digitally signed using RSA and then concatenate the hashed file and the signed file with a special delimiter and encrypt the result using AES-256.

When the DRM client requests to access a content, it must first check from his license if it validated or not. Validated means that the license is not fake, not expired and device ID identified. If DRM client validated, he will receive the encrypted data with a header that will help him with the license to decrypt the data and show it.

The system is used to protect the content with the expiry date of playback and device identification for number of devices as a usage rules. The system can be implemented in any operating system on any type of device via any open-source interface.

`

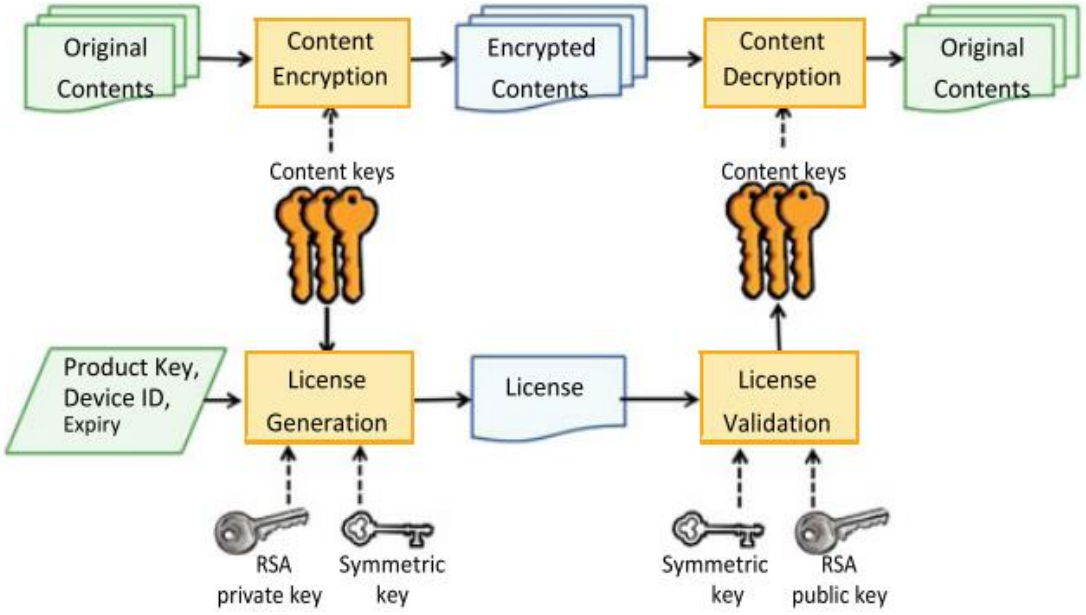


Figure 7: Block diagram of the DRM system [6].

2.4 An Approach to Mobile Multimedia Digital Rights Management Based on Android "MDRM"

This DRM system was implemented in 2014 to solve the problems in protecting the audio and video for Android mobile systems [7]. They implement their system using 3DES as a ciphering technique.

The system consists of two parties: Server and Mobile terminal as shown in figure 8. The server side is composed of the management systems for content, key, and license. The content management system is responsible for content collection, encrypting using the 3DES algorithm and packaging the content then distributes this encrypted content. The key management system is responsible for generating the key that was used to encrypt the content and another key that was used to generate a license. Each generated key was different for each original content. The license management that responsible to generate and distribute the license for authorized users and manage the usage rules for them. The usage rules in this system are device identification for the no. of devices and no of playbacks. The license is composed of the decryption key for the protected content and right of the user to this content.

For the mobile terminal, they built their own player for the digital content "video and audio" called MDRM player that needs from the customer to install on Android platform. The content saved in this terminal is in protected format. When the client opens this application, it shows him a list of the names of content. When a client chooses what he wants to open, the system starts to check the validation of this content for this user from the license. If it validates, the system starts to decrypt the content as a temporary file and play it for the customer. After the content finished playing, the system cleared the temporary decrypted file.

In reviewing the most recent DRM systems, it is clear that all of them are supporting video and audio files or video files only. All of them support only some usage rules not all of them and they are using two types of encryption algorithms one is symmetric for the content bulks and other for asymmetric for secure key exchange but all of them

`

are traditional and known encryption algorithms to hackers' community which is not secured at all. As shown in table 3 a comparison between the mentioned DRM systems in different criteria.

.

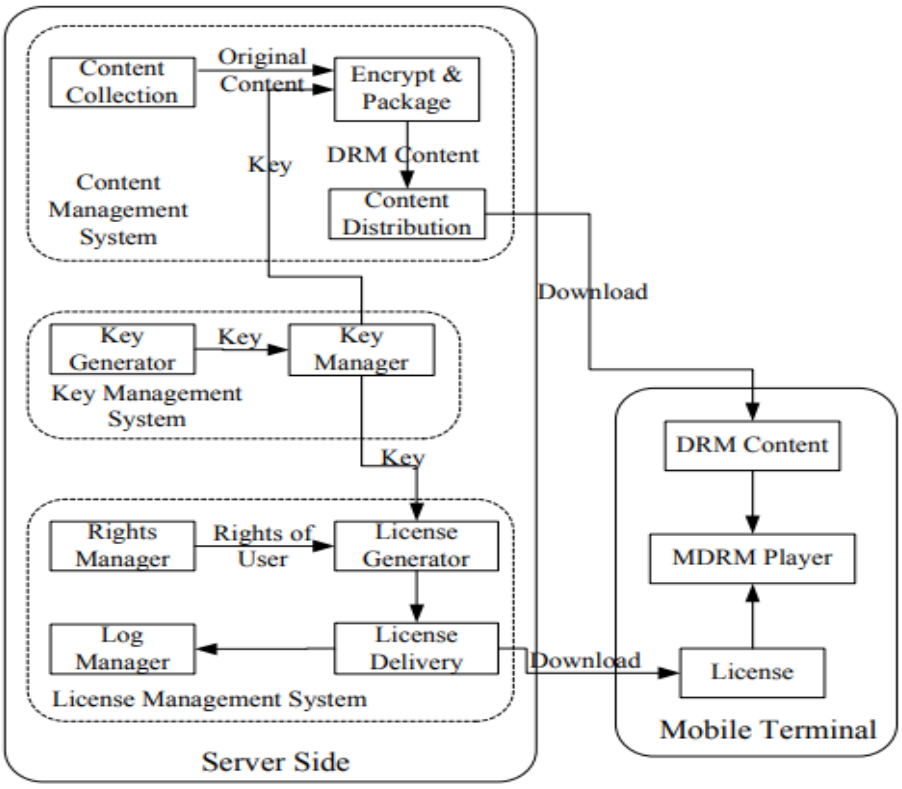


Figure 8: System architecture of the MDRM system [7].

Table 3: Comparison between different recent DRM systems.

	Blockchain and Homomorphic Encryption for Digital Copyright Protection [4]	**A robust computational DRM framework for protecting multimedia contents using AES and ECC [5]**	**A flexible and lightweight user-demand DRM system for multimedia contents over multiple portable device platforms [6]**	**An Approach to Mobile Multimedia Digital Rights Management Based on Android "MDRM" [7]**
Year	2020	2019	2017	2014
Application Type	Their own website on Truffle framework	Their own desktop application	VLC media player	Android application
Encryption Algorithms	LPN	AES-256, ECC	AES, RSA and SHA-256	3DES

`

Supported Content	All the types.	Video, Audio	Video	Video, Audio
Delivered Content	Files	Files	Files	Files
Usage Rules	• Gives the customer all the access on the content.	• Number of registered devices • No Copy • Expiry date of playback • No screen shots or recording	• Expiry date of playback • Device identification for number of devices	• Content stored in encrypted form and decrypted when we play it and then delete the decrypted temporary file. • No. of playback.
License Description	Not Applicable	JSON	JSON	Not mentioned
Strengths	• Decentralization which offers from blockchain. • High security by introducing LPN homomorphic algorithm which can make operations on content without decrypting it. • Transparency and Immutability that supported from blockchain that ensures when data is uploaded, it cannot be modified.	• High security from using AES to encrypt the content and ECC to encrypt the encryption key. • Accessible to all the OS and devices because it was implemented on cloud platform. • Adding the digital signature using ECDSA makes it more secured against repudiation.	• Flexibility and customization in the structure which can be implemented on different OS and different devices. • High security by using three different encryption algorithms to encrypt the content and keeping it secured.	• High speed decryption process which is suitable to mobile environment due to its constraints in resources. • Easy to implement for its easy structure. • High security in sharing the license file encrypted.
Limitation	• Full access to customers without any restrictions on the content. • Complex to implement.	• Using traditional encryption algorithms makes the system more vulnerable to hacking. • Complexity in generating the license may affect the ease of use.	• Using traditional encryption algorithms makes the system more vulnerable to hacking. • Complex in implementation • Dependency on external tools like VLC player. • Using multiple encryption algorithms gives us a higher security but reduce the performance on the end-user devices.	• Using traditional encryption algorithms makes the system more vulnerable to hacking. • Customized to one mobile operating system and can't implement on other operating systems. • Less restrictions on decrypted file and usage rules which make the content vulnerable to stolen.

`

3. RELATED WORK OF ENCRYPTION ALGORITHMS

The previous DRM systems have a strong structure but depend on classical encryption algorithms that are already known to the hacker's community. So, for this reason we recommend replacing these algorithms with another unknown, but they proved their efficiency and superiority over other algorithms. In this section we will list the most recent and efficient published encryption algorithms and describe the structure, strength and weakness for each one then give a short description.

3.1. Lightweight Encryption Algorithm "ISL-LWC"

This algorithm is a new block cipher design called ISL-LWC that was proposed by Ardabek Khompysh in 2023 [8]. The algorithm used 8 bytes as a block size and with a key size 10 bytes. It encrypts the data in 16 rounds. It was inspired by the Substitution Permutation network (SP) architectural methods. It was designed to support data encryption on devices with low resources. It was compared with another lightweight encryption algorithms "Present and Speck" by statistical tests developed by NIST and D. Knuth. It was testing the encryption and key generation for the three algorithms on Arduino Uno R3 board.

For the encryption process, as we see in figure 9, firstly we will XORed the plain text block ("64 bits = 8 byte") with round key then divide the result to 4 blocks of data ("16 bits = 2 byte"). From the left side the first subblock rotated by 5 then XORed with the second subblock. The result from the rotation and the XOR will be swapped and then enter to SP transformation. In the third stage, the third and fourth subblocks each undergo a substitution (S) transformation and are then XORed with the results obtained from the previous stage, following a specific scheme. Finally, the results from the previous stages are swapped according to the encryption algorithm's scheme.

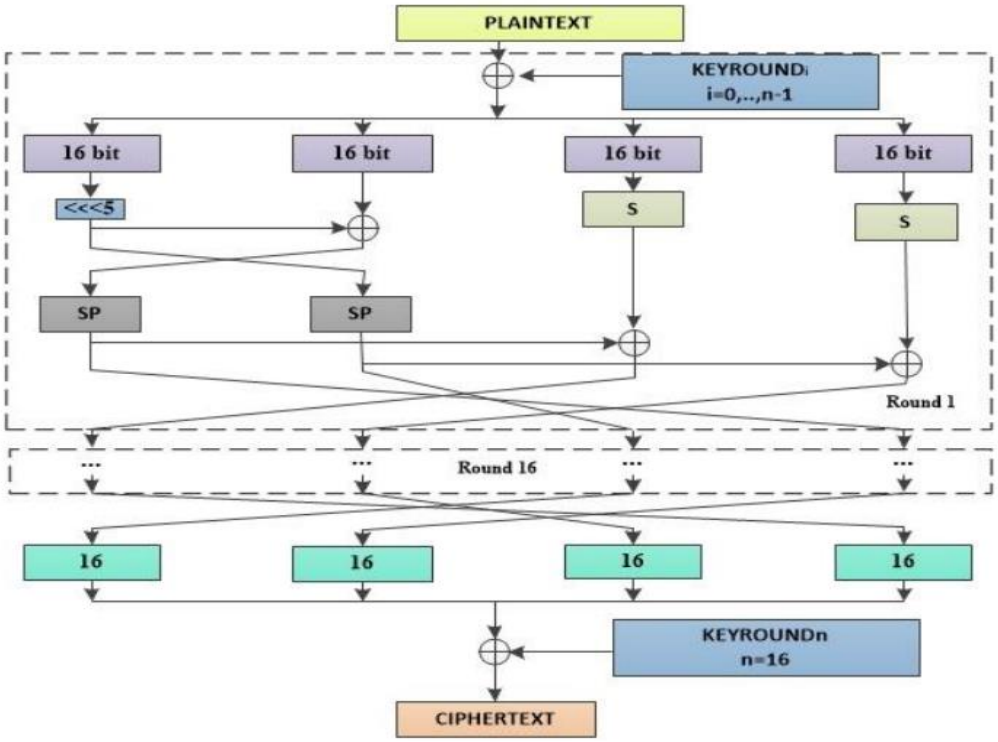


Figure 9: ISL-LWC Architecture [8].

For the SP transformation, as we see in figure 10, each 4-bit subblock is fed into different transformation boxes. The first two subblocks go through S-box transformations (S box1 and S box2), which typically involve substitution operations to enhance security by introducing non-linearity. The next two subblocks are processed by P-box transformations (P box), which usually perform permutations to diffuse the input bits, ensuring that the output bits depend on multiple input bits. After these transformations, the results are crossed over and connected to another layer of S-boxes and P-boxes. This crossover likely represents a swap or mixing step to further increase complexity and security. The final output consists of transformed 16-bit blocks that have undergone both substitution and permutation transformations, enhancing the encryption strength by thoroughly mixing and substituting the bits of the original input.

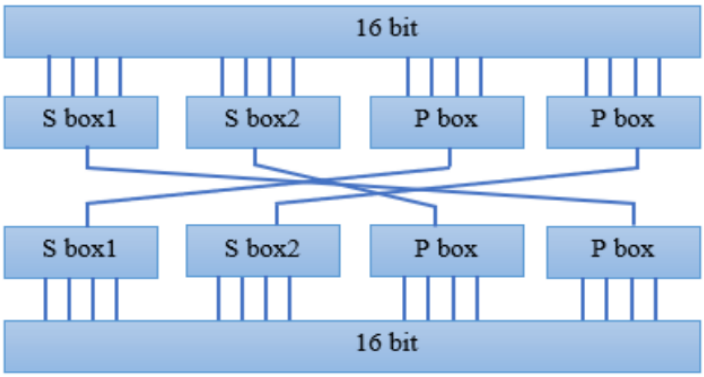


Figure 10: SP Transformation Architecture [8].

For the round key generation, as we see in figure 11, we have a base key with ("80 bit = 10 bytes") as a size and it will divide to 5 subblock with ("16 bit = 4 bytes") as a size then we will enter the 1st and 3rd subblocks to S transformation boxes which provides the nonlinearity to the key generation. The result from the transformed 1st subblock will XORed with the 2nd subblock to generate the new 2nd subblock and the result from the transformed 3rd subblock will XORed with the 4th subblock to generate the new 4th subblock and the 5th subblock will XORed with the new 2nd subblock. As a last step of the rounds keys generation the subblocks will swapped together and take the resulted first 4 subblocks with ("64 bit = 8 bytes") as round key.

NIST tests are a set of tests designed to assess the randomness of binary sequences produced by cryptographic algorithms and random number generators. They test the NIST tests on ISL_LWC, Present and Speck algorithms and compare them together. As a final result, the algorithms achieved the following percentages of successfully passed tests: ISL-LWC at 99%, Present at 97.5%, and Speck at 97%.

D. Knuth statistical tests is a series of tests designed to assess the quality of random number generators. It was testing them on the same algorithms and achieved the following percentages: ISL-LWC at 93.5%, Present at 99%, and Speck at 99%. As a time of encryption and key generation processes, ISL_LWC was larger than Speck but smaller than Present in encryption time and ISL_LWC smallest than Present and Speck in the key generation time.

`

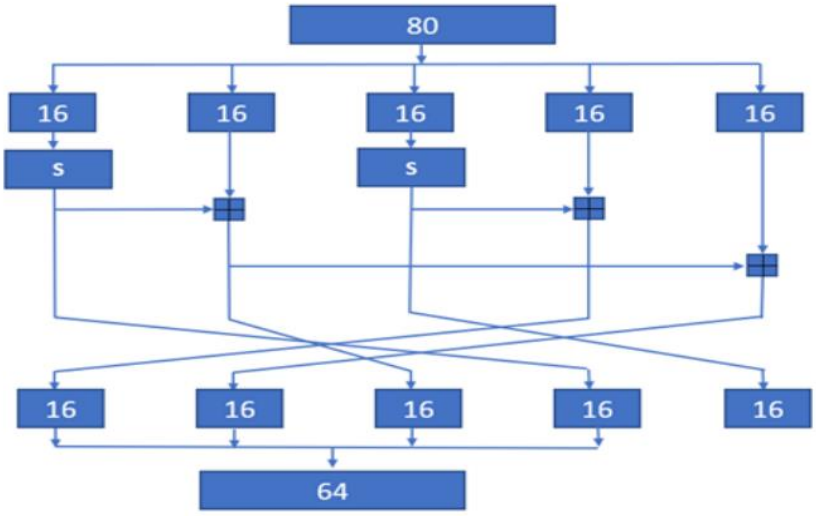


Figure 12: Round keys generation process in ISL-LWC [8].

3.2 New Lightweight Cryptography Algorithm "NLCA"

This algorithm is a new lightweight cryptographic NLCA for enhancing data security in cloud computing that was proposed by Fursan Thabit in 2021 [9]. The algorithm used 16 bytes as a block size and also 16 bytes for key size and can change the block and key size, but it must be equal. Encrypt the data in 5 rounds only which give the algorithm some strength points like: lower computational complexity and decrease the power of processing. It was inspired from the Feistel and Substitution Permutation network (SP) architectural methods to improve the complexity of the encryption. It has flexibility in block and key length and no rounds.

For encryption algorithm, as we see in figure 12, we divide each block of data "128 bits = 16 byte" to 4 small blocks with "32 bits = 4 byte" then we take the first and last blocks and XNORed with round key, this we will call it P1 and P4. Now we will apply on them F-function that contains the activity of substitute (S boxes), AND, OR, and left shift (LS) and will call the results as EFL and EFR. We will XORed EFL with P3 that produced P2 and XORed EFR with P2 that produced P3. As a final step in each round, we will exchange P1 with P2 and exchange P3 with P4. We will repeat this process with each round.

For the key generation process, as we see in figure 13, we divide the original key "128 bits = 16 byte" to 8 small blocks with "16 bits = 2 byte" and shift each block then feed each one to f-function. F- function is a linear and non-linear confusion and diffusion transformations composed of Permutation and substitution boxes. Each output from F function will produce 16 bits arranged as 4x4 matrix then the matrices using transposition and rail-fence will generate the rounds keys. Each of two matrices will be a round key. In this case we will generate only four keys for four rounds, for the last round will generate the key from XORed the other four round keys. The paper shows in figure 14 the processing time of NLCA compared with the most famous symmetric block algorithms and it shows a definite high speed with a different file size.

`

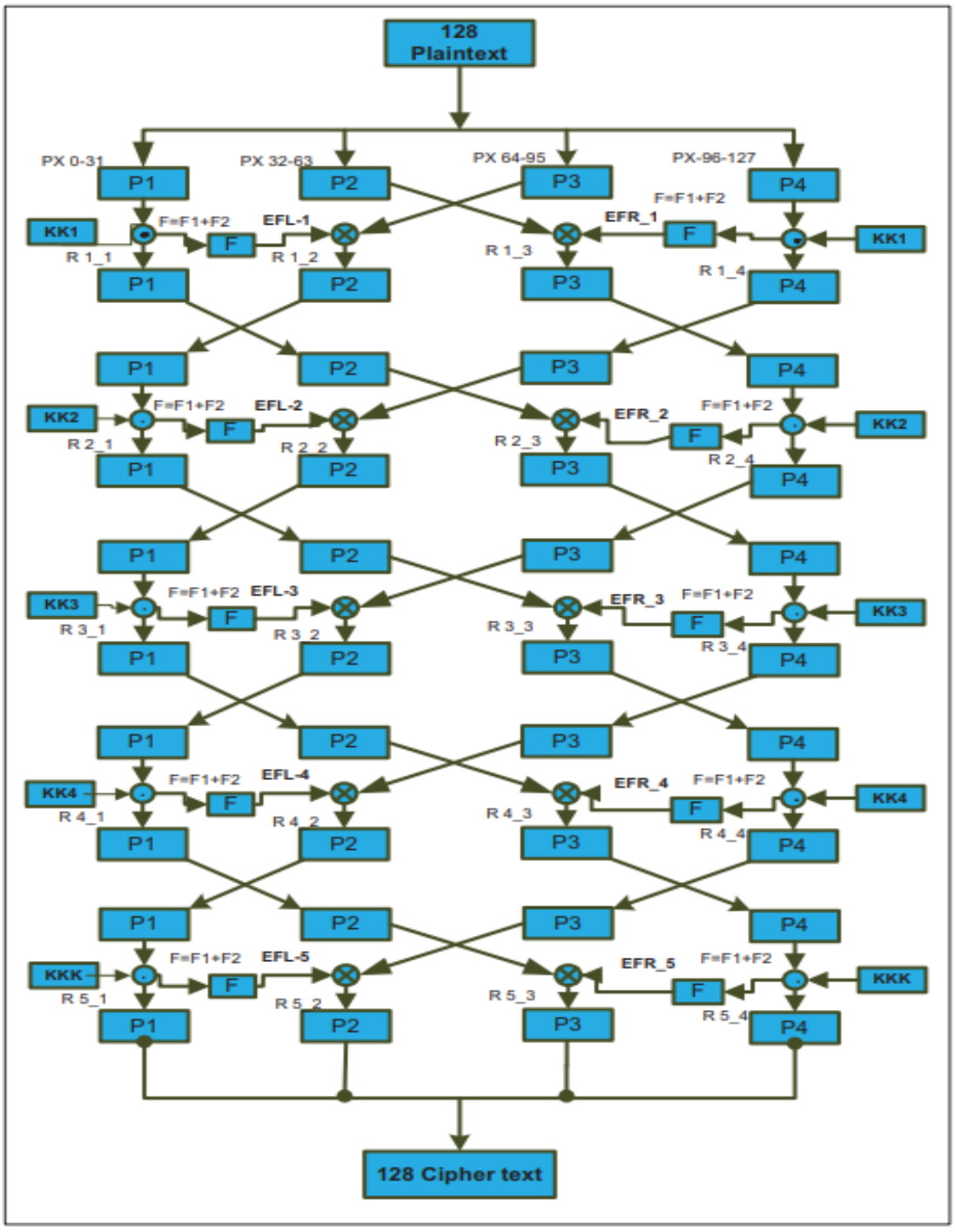


Figure 12: NLCA Architecture [9].

`

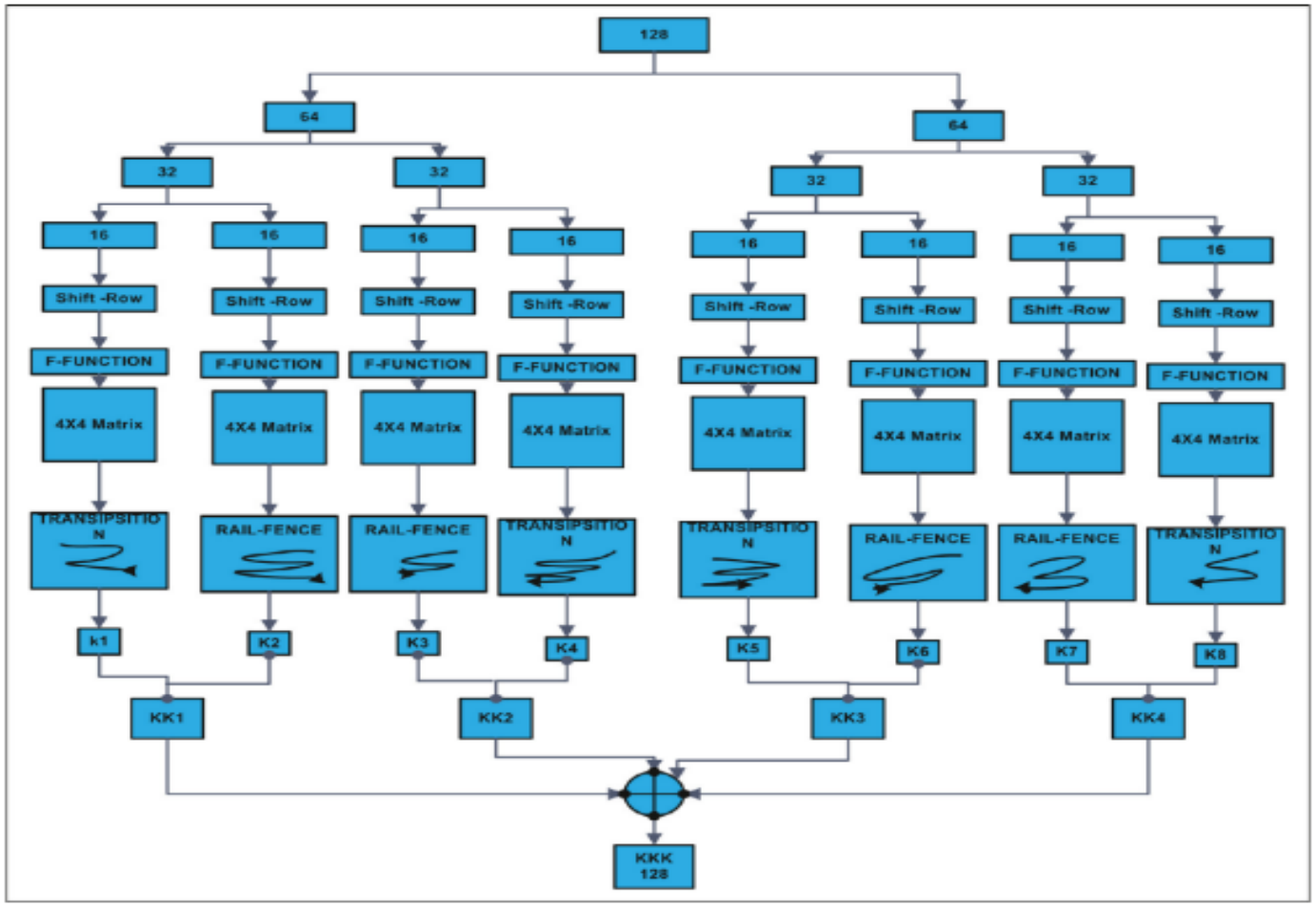


Figure 13: Round keys generation process in NLCA [9].

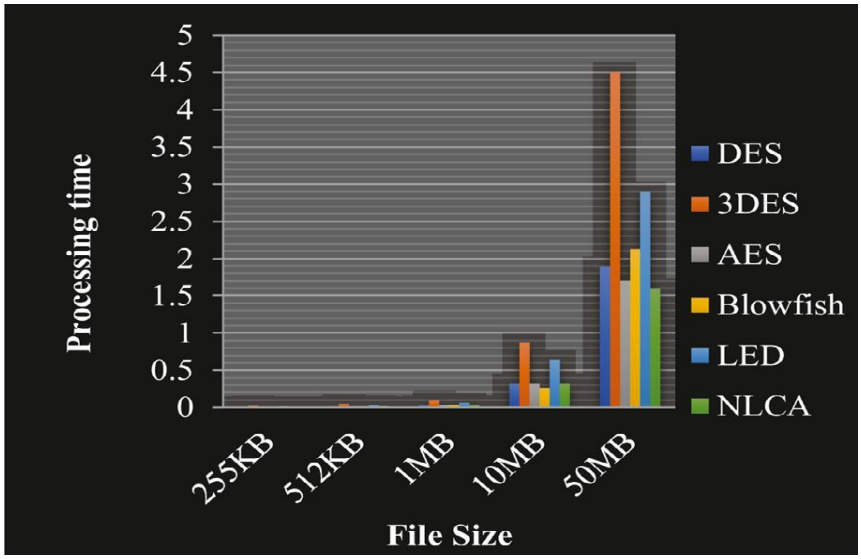


Figure 14: Processing time of different encryption algorithms including NLCA with different file sizes [9].

`

3.3 SLIM

This algorithm is a new lightweight block cipher called SLIM for internet for health things that was proposed by Bassam Aboushosha in 2020 [10]. The algorithm used 4 bytes as a block size and with a 10 byte as a key size. It encrypts the data in 32 rounds. It was inspired by the Feistel architectural methods. It can be implemented easily with both software and hardware. It was implemented on RFID systems which have a small memory and power source and shows an efficient performance and high security. It consists of substitution and permutation boxes designed to provide nonlinearity and the makes the algorithm very secured in the face of attacks. It provides a simple key generation design that uses the shift left and rotate right in this process which reduces the computational power. It shows good security with a different type of attacks when they cryptanalyzed it with the linear and differential cryptanalysis.

For the encryption process, as we see in figure 15, we divide each block of data (“32 bits = 4 byte") by 2 (“16 bits = 2 byte"). We will XOR the right side with the round key, then we will enter each 4 bits into a shift box. The bits generated from the s-boxes will enter a permutation box. The 16-bits generated from P-box will be XORed with the left side of the message, and this will be L’. Before finishing any round, we will swap L’ and the original right R.

For round key generation process, as we see in figure 16, for 32 rounds and a block of 32-bit, it is required 32 sub-keys (16-bit), which are generated from the 80-bit encryption key. The first 5 round keys are the LSB in the original key. The first 16 bits are K1 and so on. For the other round keys, The KeyMSB XORs the output after the KeyLSB performs a circular left shift of two bits. The substitution layer receives the XORing operation's output. To create the round sub-key, the XOR operation is used to alter the output of the S- boxes and the rotated KeyMSB (KeyMSB ≤ 3).

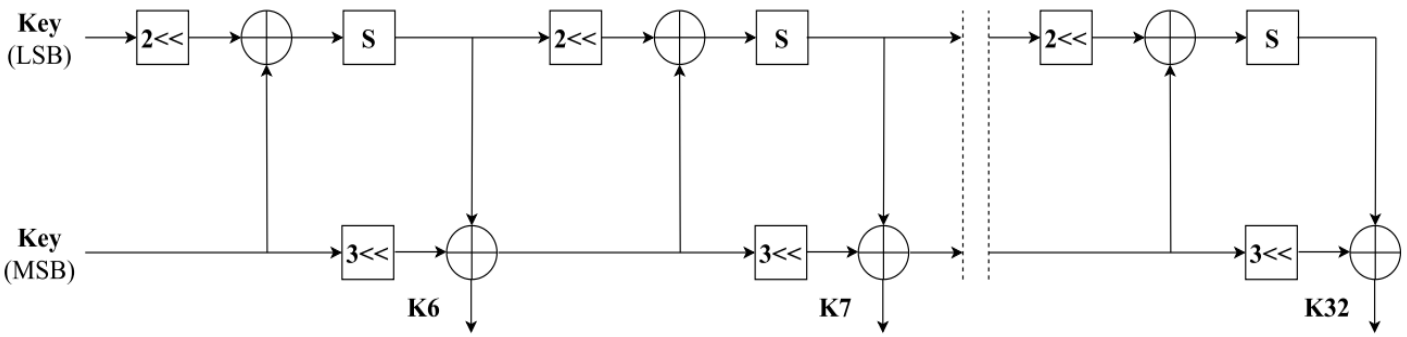


Figure 16: Round keys generation process in SLIM [10].

`

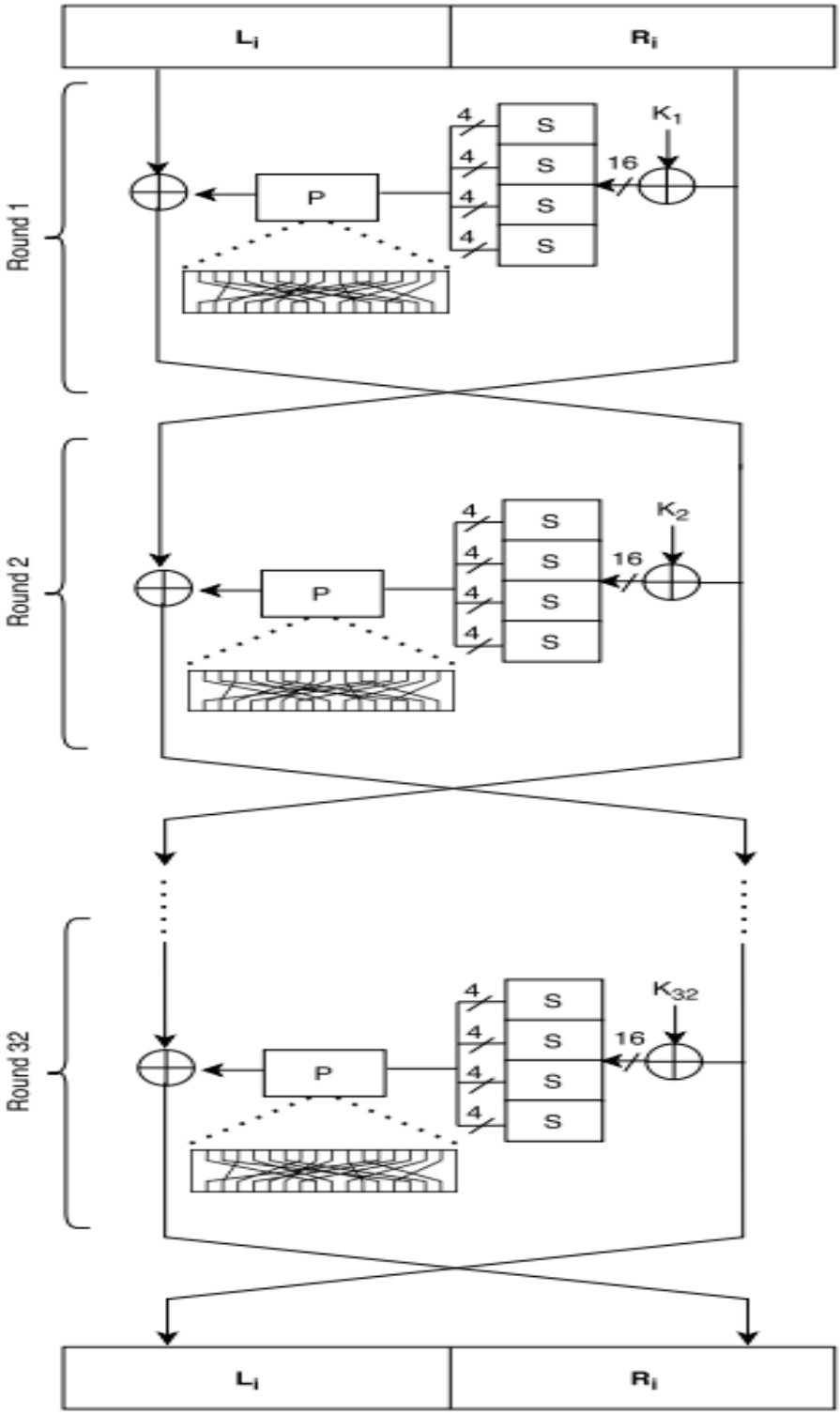


Figure 15: SLIM Architecture [10].

`

3.4 TIGRIS

This algorithm is a new block cipher design called TIGIRS that was proposed by Omar A. Dawood in 2015 [11]. It's a new variant of Advanced Encryption standard (AES). The algorithm used ("128 bit = 16 bytes") as a block size and with a three variable key size which ("128, 192 or 256 bits = 16, 24 or 32 bytes"). It encrypts the data in 16 rounds or multiples of 4 rounds. It was inspired by the architectural methods to improve the complexity of the encryption. It can be implemented easily with both software and hardware. It provides an efficient and elegant key schedule (key setup) that expands in two directions in rows/columns states which lead to increase the complexity of key generation. It works on both rows and columns to the bytes in the same row or column in plain text on not the same in cipher text. The algorithm is designed to work with a small amount of memory. It shows good security with different types of attacks like shortcut, brute and dictionary attacks.

For the encryption process, as we see in the figure 17, the encryption process can be multiples of rounds to convert the plain text to cipher text. Each round consists of 4 stages: SubByte, ReversibleShiftRow, ShiftingMixColumn and AddRoundKey. We will apply the stages on the message in 4*4 bytes matrix. For SubByte stage uses a custom S-box designed to increase non-linearity further. It employs a different irreducible polynomial and affine matrix compared to AES, aiming to improve security and make the cipher more resistant to various cryptanalytic attacks. For ReversibleShiftRow stage as shown in figure 18, the rows in the matrix shifting in different way than AES. The shifting in TIGRIS is reversible and nonlinear as shown in the figure. For ShiftingMixColumn stage we will multiplying each column with a fixed matrix after that we will shifting or rotating the elements within the column that adds another layer of diffusion, ensuring that the data is spread out even more across the state matrix unlike the static MixColumns in AES. For AddRoundKey stage we will XOR the output from the previous stage with the round key.

For the key Scheduling process, it's like the rounds keys generation process in AES but with some changes to make it more complex and higher security makes it more resistant to advanced cryptanalytic attacks. These changes are using a unique method of expanding keys in both row and column directions within the message matrix and employs two constant vectors derived from the golden ratio and a natural base algorithm to eliminate weak and semi-weak keys. These vectors are integrated into the round key generation process to ensure a high level of diffusion of cipher key differences. You can see the key expansion process in figure 19.

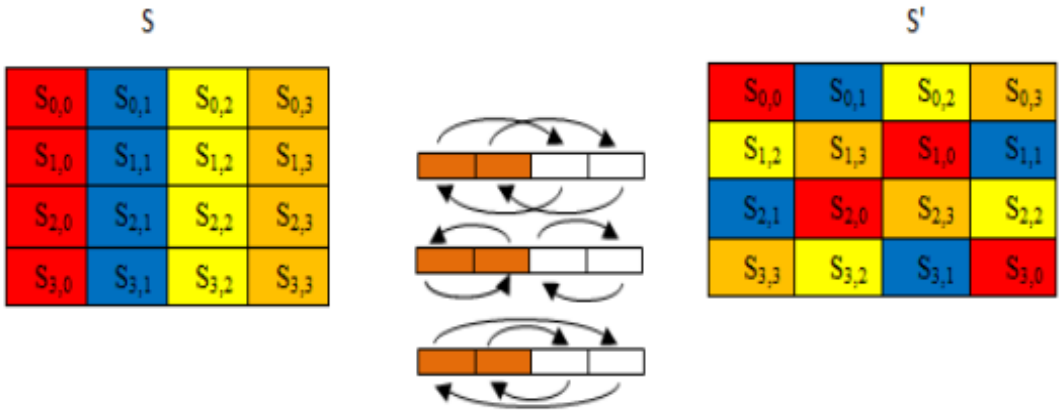


Figure 18: ReversibleShiftRows stage in TIGRIS [11].

`

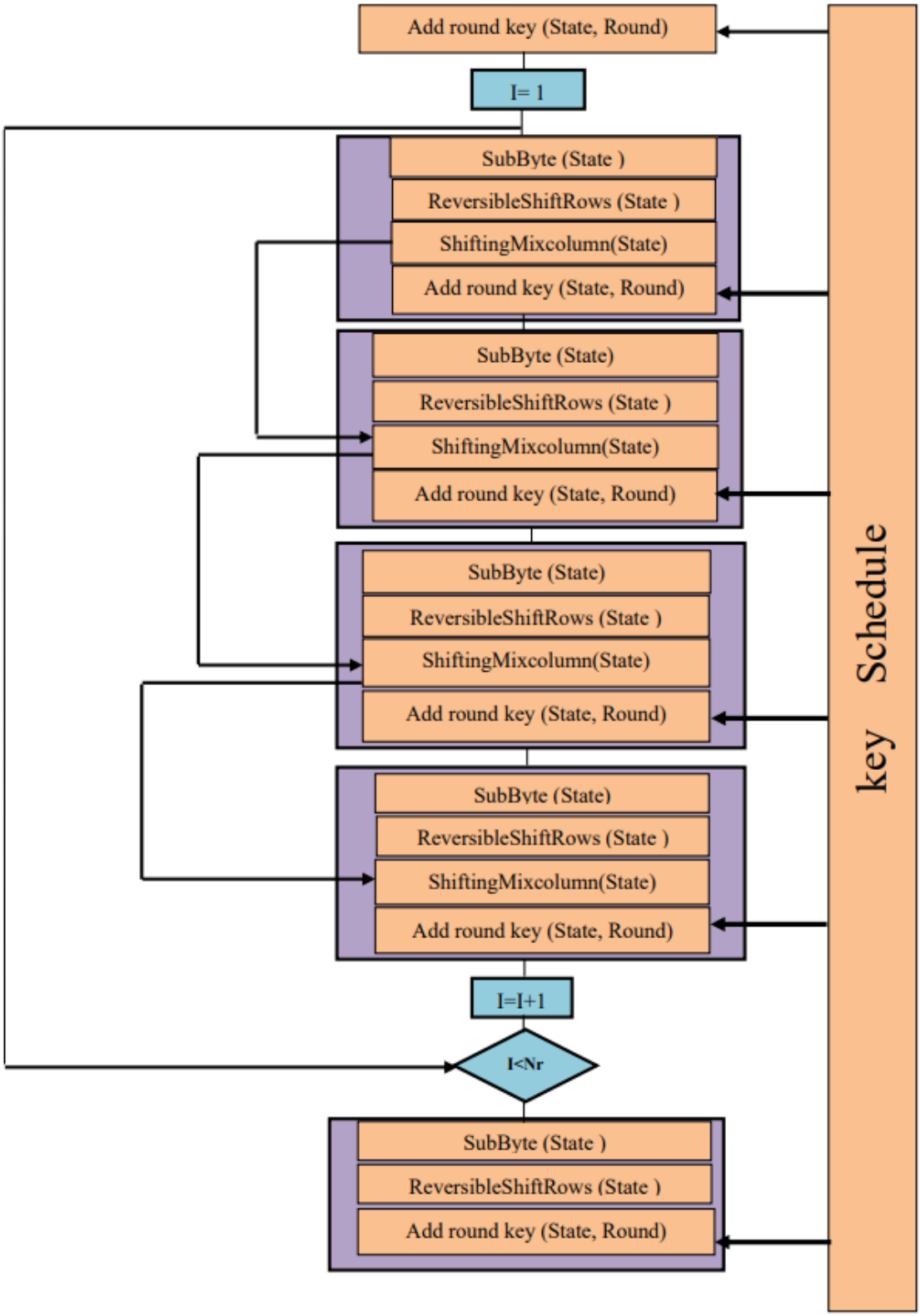


Figure 17: TIGRIS Architecture [11].

`

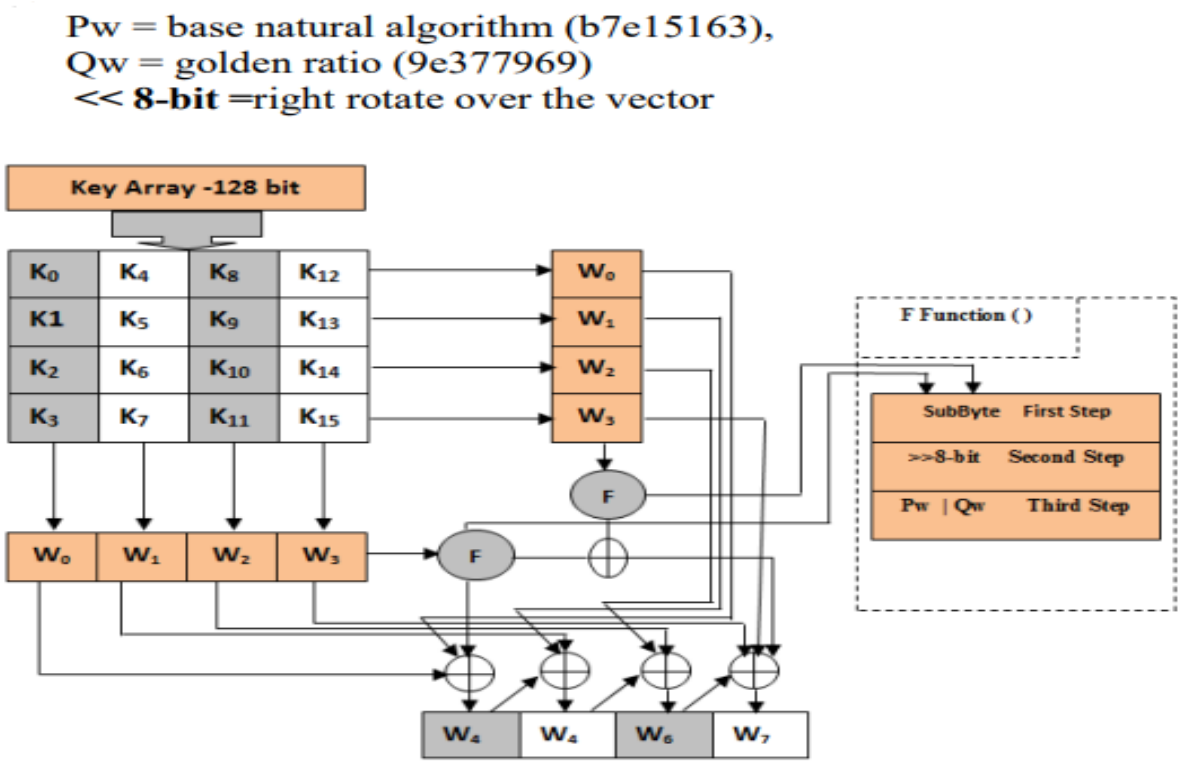


Figure 19: Key expansion process in TIGRIS [11].

In reviewing the most recent recommended encryption algorithms, it is clear that all of them are block symmetric algorithms that will be using to encrypt the content as a replacement of the old encryption algorithms in DRM systems and all of them used to be lightweight to be suitable to encrypt and decrypt the data in RAM. As shown in table 6 a comparison between the mentioned algorithms in different criteria.

Table 6: Comparison between different recommended recent encryption algorithms.

	ISL-LWC [8]	**NLCA [9]**	**SLIM [10]**	**TIGRIS [11]**
Year	2023	2021	2020	2015
Type	All of them block symmetric algorithms			
Structure	Substitution Permutation network (SP)	Feistel and Substitution Permutation network (SP)	Feistel	Substitution Permutation network (SP)
Block size	64 bits	128 OR 256 bits	32 bits	128 bits
Key size	80 bits	128 OR 256 bits	80 bits	128, 192 or 256 bits

`

Possible keys	2^{80}	$2^{128} or\ 2^{256}$	2^{80}	$2^{128}, 2^{192} or\ 2^{256}$
Rounds No.	16	4	32	16 or multiples of 4
SP Structure	2 S-Boxes	4 S-Boxes	4 S-Boxes	4 S-Boxes
S-Box Size	4x4	4x4	4x4	8x8
Mathematical Operations	XOR, Permutation, Substitution, Shifting	XOR, XNOR, Addition, Shifting, Substitution	XOR, Permutation, Substitution, Shifting	XOR, Permutation, Substitution, Shifting
Testing	Tested against Present and Spark algorithms on • NIST statistical tests Percentage of successfully passing: ISL-LWC – 99%, Present – 97.5%, Speck – 97%. • D.Knuth statistical tests Percentage of successfully passing: ISL-LWC – 93.5%, Present – 99%, Speck – 99%. • Encryption and Key setting process time: For encryption ISL_LWC was slower than Speck but much faster than Present. For Key setting ISL_LWC faster than Speck and Present.	Tested against DES, 3DES, AES, BlowFish and LED For processing time NLCA was faster than all the other algorithms.	Tested against another 14 lightweight block encryption algorithms like: AES, Clefia, Speck and Simon on the same hardware technology and was found that SLIM takes less area than most of them and there were some algorithms takes less area like Speck but use smaller key size which makes it vulnerable to know and attack. It was tested against linear and differential cryptanalysis and was found that SLIM is secured against cryptanalysis attackers.	Tested against AES For processing time TIGRIS was slower than AES with little milliseconds.
Features	• Using small no. of s-boxes which suitable to limited resources and add the nonlinearity to the process.	• Using Feistel and SP together makes the structure more complex than others and strong resistance to cryptanalysis.	• Large no. of rounds provides resistance to differential and linear cryptanalysis. • S-boxes are non-linear and crafted to	• The improvement on S-Boxes against AES makes it high non-linearity, providing strong resistance against linear and

`

	• Adding a combination of substitution (S-boxes) and permutation (P-boxes) achieves diffusion and confusion. • Statistical testing revealed that it effectively creates a strong avalanche effect, making the encrypted data appear almost random and statistically secure.	• Including a different mathematical operation, such as XNOR, increases the complexity and makes it more difficult for attacks and attackers. • A small number of encryption rounds reduces the time complexity. • Ensures strong diffusion and confusion by utilizing transpose and swap procedures. • The dynamic key generation process, utilizing matrix and f-function extension, prevents brute-force attacks and ensures robust key management.	withstand common cryptanalytic attacks and using multiple S-boxes across different rounds enhances security by making transformation predictions more difficult. • The iterative structure of Feistel rounds ensures comprehensive mixing of data, resulting in strong diffusion and confusion. • It used a simple key scheduling process for the limited resource devices but with adding s-boxes for nonlinearity in addition to shifting and XORing.	differential cryptanalysis. • Achieves diffusion through the ShiftingMixColumn operation and confusion through the S-boxes and ReversibleShiftRows. The combined effect of these operations ensures that small changes in the plaintext or key result in significant changes in the ciphertext. • Using golden ratio-based constants in key expansion, ensuring that each round key is unique and not prone to weaknesses such as related-key attacks. • It has a flexibility in block size, key size and n. of rounds.
Recommended Area	Resource-constrained devices	Secure application in cloud computing	Resource-constrained devices especially in Radio frequency identification (RFID) systems	Limited resource environment such as sensors, mobile devices and networks

4. COMPARATIVE ANALYSIS BETWEEN DIFFERENT ASYMMETRIC ALGORITHMS

We covered the DRM systems in the earlier sections, highlighting one of its primary advantages—the extra protection provided by key encryption. For this reason, asymmetric encryption algorithms have become the best technique since they are more resistant to hacking than symmetric algorithms. This benefit is especially pertinent when encrypting small-sized data, like encryption keys, as the dual-key approach of asymmetric encryption—one key for encryption and another for decryption—provides higher security. As a result, the most recent research papers evaluating several asymmetric algorithms will be reviewed in this section to show their advantages and disadvantages, in order to determine which algorithm is best for securing encryption keys.

`

4.1 Cryptographic algorithms for enhancing security in cloud computing

It was published in 2023 to provide an overview of existing symmetric and asymmetric encryption algorithms and show a full comparison between all of them [12]. They also mentioned the previous cloud systems and how symmetric and asymmetric algorithms were employed in these systems, and how each of them performed. As shown in table 7, They compare some of the popular symmetric and asymmetric algorithms to each other and they find that RSA and ElGamal are low battery consumption which is suitable to work with resource-constrained devices. They show also that asymmetric algorithms are slow compared to symmetric algorithms when used to encrypt data with large size. For cloud computing systems, they mentioned 16 different systems some of them works with symmetric algorithms only but most of them works with hybrid between symmetric and asymmetric algorithms. They found two systems that have high level of security, lightweight and significant improvement in encryption execution time, memory usage, and throughput was hybrid systems. The first system was proposed by Fursan Thabit in 2022 [13]. It was a lightweight homomorphic cryptographic algorithm with two layers of encryption. The first layer uses the new effective, light-weight cryptographic algorithm, and the second layer employs multiplicative homomorphic schemes. The second system was proposed by Noha El-Attar in 2021 [14]. It was used RSA, Twofish, AES, and DES together to keep the cloud storage secured and show high security and quick data processing.

Table 7: Comparison between different symmetric and asymmetric encryption algorithms. [12]

Parameters	AES	DES	TDES	Blowfish	Twofish	RC2	RSA	DHKE	El-gamal
Type	symmetric	symmetric	symmetric	symmetric	symmetric	symmetric	asymmetric	asymmetric	asymmetric
Structure	Substitution - Permutation	Feistel	Feistel	Feistel	Feistel	Feistel	Factorization	Festial & Substitution	discrete logarithm
Key used	Secret key	Secret key	Secret key	Secret key	Secret key	Secret key	Different keys	Different keys	Different keys
Key size	128- 192 and 256 bits	56- bits	192 bits, with only 168 bits being used (3 times 56)	From 32 to 448 bits	128-192and 256 bits	8 to 1024 bits;	1024 to 4096 bits.	1024 to 2048 bits	1024 bits
Round for encryption	10,12 and 14 rounds	16 rounds	48 rounds	16-round	16 rounds	16 (Mixing) +2 (Mashing)	one-round	one-round	one-round
Throughput of encryption and decryption	fast	fast	fast	fast	fast	fast	slow	slow	slow
Battery Consumption	high	medium	high	Very low	low	high	low	high	low

`

4.2 Analysis of most common encryption algorithms

It was published in 2022 [14], and they were aim to full analysis to the mostly and commonly used encryption algorithms with both types symmetric and asymmetric encryption algorithms. They emphasized the importance of asymmetric encryption algorithms in the process of exchanging keys for symmetric encryption keys on the Internet or large network. They compared Elliptic-curve cryptography (ECC), ElGamal and Rivest-Shamir-Adleman (RSA) as asymmetric algorithms and they summarized their analysis in table 8. They tested them on the same plaintext and found that Elgamal is the fastest in the encryption process and RSA is the slower due to the large size of the key. They recommend ECC as a perfect asymmetric encryption algorithm for resource-constrained devices like smart cards and smart phones.

Table 8: Comparison between ECC, ElGamal and RSA algorithms [14].

Parameters for Comparison	Elliptic Curve	Elgamal	RSA
Developers	Neal Koblitz, Victor S. Miller	Taher Elgamal	Rivest, Shamir, and Adleman
Emerging Year	1985	1985	1977
Size of Key	Up to 512 bits	521 bits	Up to 15360 bits
Algorithm Type	Algebraic Structure of elliptic curve over a finite field	Diffie Hellman key exchange	Exponentiation is a finite field over integers including prime numbers
Encryption Time (ms)	390	297	531

4.3 An experimental study on performance evaluation of asymmetric encryption algorithms

It was published in 2012 to test the performance of some asymmetric encryption algorithms such as ElGamal, Rivest-Shamir-Adleman (RSA) and Paillier [15]. They accomplish various experiments to compare their encryption and decryption time, memory usage and throughput over variable text file and test them in the same environment (software and hardware). They choose the private key size for ElGamal and Paillier are 160 bits and RSA 1024 bit to make them give the same security level of the other algorithms. RSA showed better performance over ElGamal and Paillier in term of encryption time and ElGamal showed better performance over RSA and Paillier in decryption time as shown in Figure 19. RSA showed better throughput over ElGamal and Paillier in encryption process and Elgamal showed better throughput over RSA and Paillier in decryption process as shown in figure 20. Paillier is worse than RSA and ElGamal in the encrypted file size as shown in figure 21. They recommend that RSA algorithm is better than ElGamal and Paillier for its high performance in most of the experiments that were conducted.

`

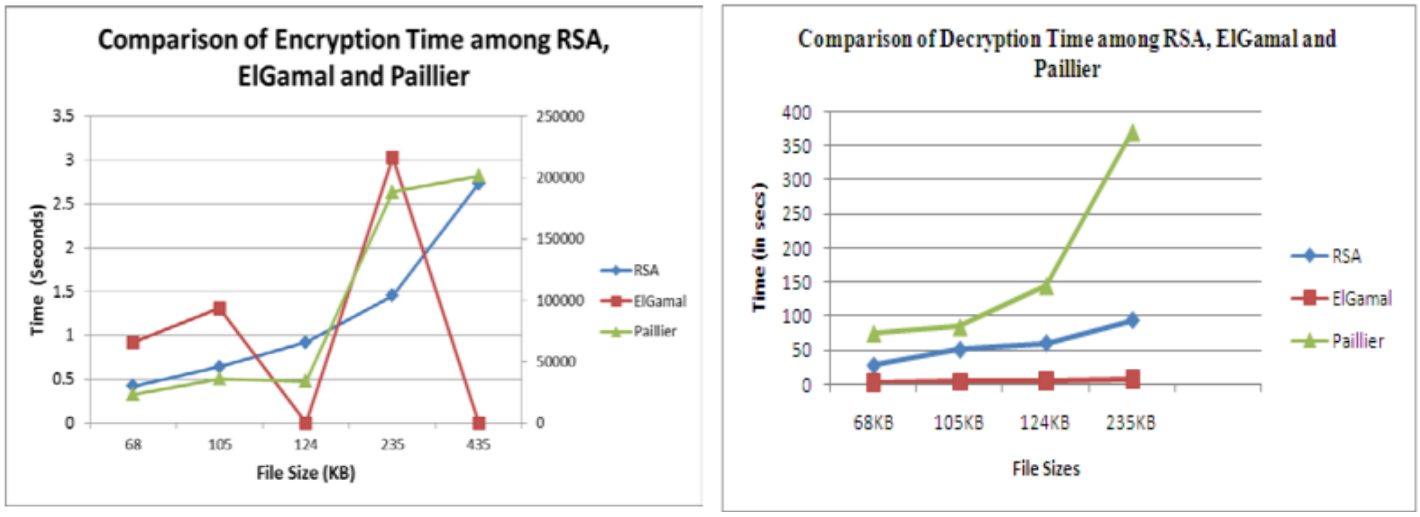


Figure 19: Encryption and decryption time of RSA, ElGamal and Paillier [15].

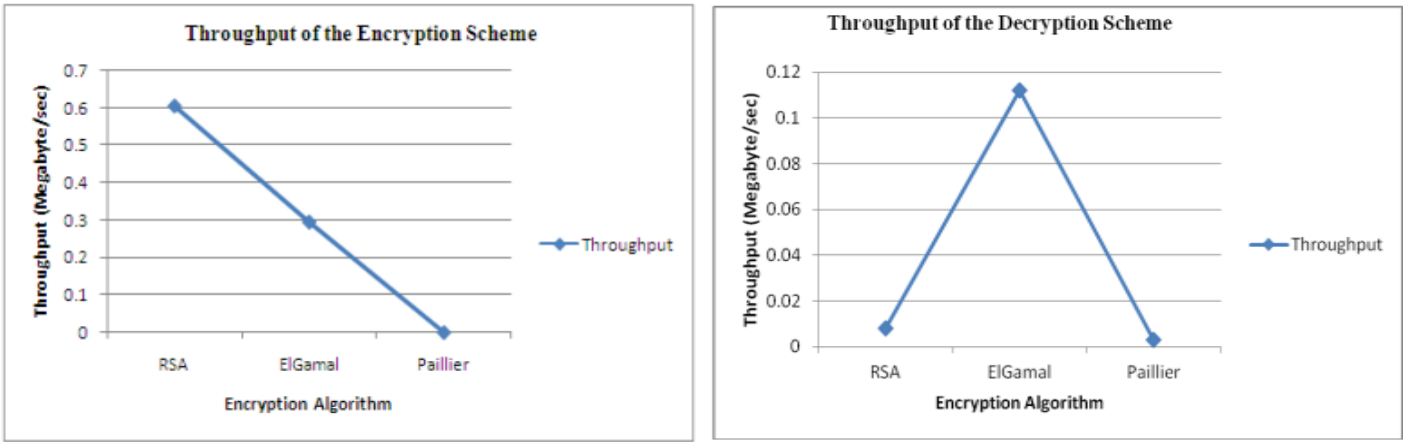


Figure 20: Throughput of Encryption and decryption of RSA , ElGamal and Paillier [15].

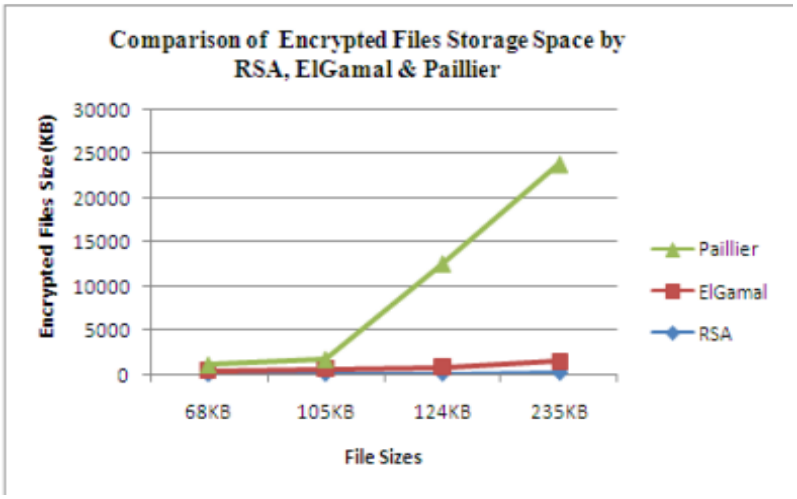


Figure 21: File sizes in Encryption process of RSA, ElGamal and Paillier [15].

5. CONCLUSION

In conclusion, the ongoing growth of digital content has driven the need for advanced digital rights management (DRM) systems to effectively oversee and safeguard digital media and intellectual property rights. While traditional DRM systems have been successful in managing digital content, a major drawback has been their dependence on conventional, widely known, and vulnerable encryption algorithms. To address this, newer and less familiar encryption algorithms like NLCA, SLIM, TIGRIS, and ISL-LWC have been introduced. Integrating these algorithms into DRM systems enhances their robustness and security against cyber-attacks. These modern algorithms offer notable improvements in efficiency and resilience against various cryptographic attacks. We have reviewed also the most recent survey papers on asymmetric algorithms, providing an in-depth analysis of their strengths and weaknesses to determine the most suitable choice for encrypting the keys, thereby adding an additional layer of security to the system. Future DRM systems must incorporate these advancements to ensure the integrity and security of digital content in an increasingly interconnected and digital world.

REFERENCES

[1] C. Roach, Data Insider Digital Guardian's Blog, "What is Digital Rights Management (DRM)? (The Definitive Guide)", Digital Guardian, Aug,22,2024, [Online], Available: https://www.digitalguardian.com/blog/what-digital-rights-management.

[2] N. Gupta, SSL2buy, "Symmetric vs. Asymmetric Encryption – What are differences?", SSL2buy, [Online], Available: https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences.

[3] C. Crane, hashedout, " Block Cipher vs Stream Cipher: What They Are & How They Work", hashedout, Jan,14,2021, [Online]. Available: https://www.thesslstore.com/blog/block-cipher-vs-stream-cipher/.

[4] W. Sun, H. Fang, S. Zheng, and Q. Qian, "Blockchain and Homomorphic Encryption for Digital Copyright Protection", IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/ SustainCom), pp. 754-761, 2020, DOI: 10.1109/ISPA-BDCloud-SocialCom-SustainCom51426.2020.00120 .

[5] H. Hassan, M. Tahoun and Gh.S. ElTaweel, "A robust computational DRM framework for protecting multimedia contents using AES and ECC", Alexandria Engineering Journal, vol. 59, no. 3, pp. 1275-1286, 2020, DOI: 10.1016/j.aej.2020.02.020.

[6] E. H. Wu, S. Chuang, C. Shih, H. Hsueh, S. Huang and H. Huang, "A flexible and lightweight user-demand DRM system for multimedia contents over multiple portable device platforms", Software - Practice and Experience, vol. 47, no. 10, pp. 1417-1441, 2017, DOI: 10.1002/spe.2479.

[7] Z. Wang, Z. Zhang, Y. Chang, and M. Xu, "An Approach to Mobile Multimedia Digital Rights Management Based on Android", In: Pan, JS., Krömer, P., Snášel, V. (eds) Genetic and Evolutionary Computing. Advances in Intelligent Systems and Computing, vol 238. Springer, Cham, 2014 DOI: https://doi.org/10.1007/978-3-319-01796-9_25 6.

[8] A. Khompysh, N. Kapalova, O. Lizunov, D. Dyusenbayev, and K. Sakan, "Development of a New Lightweight Encryption Algorithm", International Journal of Advanced Computer Science and Applications (IJACSA), vol. 14, no. 5, pp. 452-459, 2023, DOI: 10.14569/IJACSA.2023.0140548.

[9] F. Thabit, S. Alhomdy, A. H.A. Al-Ahdal, and S. Jagtapa, "A new lightweight cryptographic algorithm for enhancing data security in cloud computing", Global Transitions Proceedings, vol. 2, no. 1, pp. 91-99, 2021, DOI: 10.1016/j.gltp.2021.01.013.

`

[10] B. Aboushosha, R. A. Ramadan, A. D. Dwivedi, A. El-Sayed, and M. M. Dessouky, "SLIM: A Lightweight Block Cipher for Internet of Health Things", IEEE Access, vol. 8, pp. 203747- 203757, 2020, DOI: 10.1109/ACCESS.2020.3036589.

[11] O. A. Dawood, A.M. S. Rahma, and A.M. J. Abdul Hossen, "The New Block Cipher Design (Tigris Cipher)", International Journal of Computer Network and Information Security, vol. 7, No. 12, pp. 10-18, 2015, DOI:10.5815/ijcnis.2015.12.02.

[12] D. S. El-Morshedy, N. E. El-Attar, I. M. Hanafy and W. A. Awad, "Cryptographic Algorithms for Enhancing Security in Cloud Computing", Alfarama Journal of Basic & Applied Sciences, Faculty of Science Port Said University, vol. 4, no. 3, pp. 433-455, 2022, DOI: 10.21608/ajbas.2 022.141544.1105.

[13] F. Thabit, O. Can, S. Alhomdy, G. H. Al-Gaphari and S. Jagtap, "A Novel Effective Lightweight Homomorphic Cryptographic Algorithm for data security in cloud computing", MDPI Cryptogrphy, vol. 5, no. 37, pp. 1-20, 2021, DOI: 10.1016/j.ijin.2022.04.001.

[14] N. E. El-Attar, D. S. El-Morshedy, and W. A. Awad, "A New Hybrid Automated Security Framework to Cloud Storage System", International Journal of Intelligent Networks, vol. 5, no. 4, pp. 16-30, 2022, DOI: 10.3390/cryptography5040037.

[15] A. Ahmed, and M. Naeem, "Analysis of Most Common Encryption Algorithms", International Journal of Engineering and Applied Computer Science (IJEACS), vol. 4, no. 2, pp. 9-13, 2022, DOI: 10.24032/IJEACS/0402/003.

[16] S. Farah, M. Y. Javed, A. Shamim and T. Nawaz, "An experimental study on Performance Evaluation of Asymmetric Encryption Algorithms", Recent Advances in Information Science, Proceeding of the 3rd European Conf. of Computer Science (EECS-12), pp. 121-124, 2012.

`