DOI: http://dx.doi.org/10.26483/ijarcs.v16i1.7181

ISSN No. 0976-5697

Volume 16, No. 1, January-February 2025

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

CIPHERSHIELD: PIONEERING MULTI-KEY CRYPTOGRAPHY FRAMEWORK FOR FILE ENCRYPTION USING DYNAMIC KEYS

Mohammed Abdul Lateef
Computer Science and Engineering
Jawaharlal Nehru Technological University Hyderabad
Hyderabad, India

Dr. P. Swetha
Computer Science and Engineering
Jawaharlal Nehru Technological University Hyderabad
Hyderabad, India

***Abstract*:** The exponential growth of digital content consumption, particularly in the form of video data, underscores the critical need for robust security measures to prevent unauthorized usage and copyright infringement of video data. This work introduces another level of hybrid cryptographic approach, specifically tailored to secure video files by leveraging a hybrid encryption model that combines the asymmetric power of RSA and dynamic AES key management. Unlike conventional methods, our approach dynamically generates unique AES keys for each chunk of video data, using system-unique identifiers and elliptic curve equations, thereby enhancing the security and adaptability of the encryption process. The proposed system is implemented through a sophisticated software solution that features dedicated modules for video encryption and decryption. This system ensures that video files are securely encrypted, with keys regenerated dynamically to maintain resilience against attacks. Rigorous testing and evaluation demonstrate the system's superiority in both security and performance metrics, proving it to be an effective solution for combating modern challenges such as copyright infringement and piracy. This research offers a significant contribution to the field of file security, presenting a cutting-edge cryptographic implementation that meets the evolving demands of digital content protection.

Keywords: Multi-Key Cryptography, Hybrid Cryptography, Elliptic Curve Cryptography, Real-time File Securing, Dynamic Encryption, Performance Metrics

I. INTRODUCTION

For the digital video content, there are many and plenty of data is built by information era brought on-line over different threats that Ciphers based on key-secret should certainly play a significant role protecting these types. Be it in order to protect the confidential content from being in-scoped by others without permission, or even from without distribution and tampering, video encryption is a must. Videos are much heavier than plain-text or individual static images, and so part of the reason why they can be hard to protect is due to their size; on top of that live streaming also demands real-time delivery. Given the particular context, existing encryption techniques might not be adequate for video data and they require to be made stronger with specific methods to successfully protect. Hence, we need some dedicated Video Data Aligned algorithms as further discussed.

The challenges with respect to manipulating video data are immense, hence encrypting the content adds another level of complexity. This is because the encryption algorithm to be used must have a throughput, with video files being large in size thus, making sure that there are no significant latencies arise during the implementation of security is equally crucial. Their use cases (like video streaming which requires real-time processing) are complex, and the architecture required to support such applications is a lot more complicated. While common encryption algorithms, like AES and RSA for example, while being secure, are computationally intensive and may not be optimized for the continuous, high-throughput nature of video streams. These issues have resulted in the development of models that are hybrid in nature, which leverage combined capabilities offered by symmetric and asymmetric cryptography to cater video related security challenges.

One promising approach to video encryption is the hybrid dynamic key cryptography approach. This method utilizes symmetric key algorithms for encrypting the bulk of the video data due to their efficiency and speed, while asymmetric key algorithms are used for securing the generated key through initial key encryption [10]. By dynamically generating and encrypting keys for each video chunk, this technique enhances security and ensures that compromising one chunk does not lead to the decryption of the entire video. The use of multiple keys also reduces the risk associated with key exposure, providing an additional layer of security.

The unique technique created for video encryption is capable of harnessing crucial aspects like the security and performance; to handle the distinctive issues associated with video data [11]. The approach makes sure that every video chunk it generates is encrypted with an exclusive key by utilizing a dynamic key generation mechanism. This greatly increases the difficulty of unauthorized decryption [9]. The method works fine enough, for today's real-time video streaming applications since it is also tuned to reduce latency [12]. The custom algorithm is a very effective technique to secure videos of any generation, in a variety of applications all being down to its combination of strong security features and lightning-fast algorithmic processing [13].

Hence this custom algorithm for the video-based encryption introduces a sophisticated approach to ensuring that video data is secure, and at the same time aims in addressing both the security and performance needs of modern applications. By the use of advanced cryptographic techniques and optimizing for the unique characteristics of video files, this algorithm offers a practical and efficient solution to the growing demand for secure video communication.

II. LITERATURE SURVEY

Y. Fouzar et. al. in 2023, proposed a multi-key solution that aims to separate the video into many parts and then encrypt each chunk with a different key. The receivers do not have access to these keys. The receiver application generates the key using the

encrypted chunks it has received. The receiver application starts the decryption process as soon as the video chunks are received because the suggested method is dynamic and automatic [1].

Iavich et. al. in 2018, introduced a new hybrid cryptographic algorithm model using combination of two cryptographic algorithms AES and ElGamal; and provide comparison between two symmetric, asymmetric algorithms and new hybrid model. They also portrayed an effectiveness and security of new hybrid model which makes the algorithm strong against vulnerabilities. Currently many encryption algorithms are available to secure the data but some algorithms consume lot of computing resources such as memory and CPU time. Their work shows comparative analysis experimental results on those encryption algorithms[2].

Y. Hu in 2023. proposed an efficient symmetric cryptographic algorithm based on garbled coding, and combined it with ECC asymmetric cryptographic algorithm and SHA-256 hash algorithm to propose an efficient hybrid encryption scheme suitable for encrypting electric power business data. This paper also analyzes that the proposed hybrid cryptographic algorithm has sufficiently high security, and the simulation verifies that the proposed method has high encryption and decryption efficiency with appropriate data chunking ratio [3].

C. Prashanth et. al. in 2021, illustrated mobile encryption using multiple algorithms (AES, Salsa20, fernet) as part of hybrid encryption approach. The data was split and encrypted with multiple algorithms. The selected file to be encrypted is separated into three parts, and it gets encrypted using AES, Salsa20 and Fernet. The private keys are then RSA(Rivest-Shamir-Adleman algorithm) encrypted. The RSA cipher file is stored in phone storage. To decrypt, the same RSA cipher file receives, and their correlating keys are received. And the AES, Salsa20 and Fernet algorithms decrypt their respective encrypted segments by keys. The decrypted segments are combined and the original file is obtained. Then the file is stored in the device [4].

T. Yue and et. al in 2019, proposed the hybrid encryption algorithm based on wireless sensor networks, to provide the analysis of existing wireless sensor networks susceptibility to security, which combines the high encryption efficiency characteristic of the symmetric encryption algorithm and asymmetric encryption's high encryption intensity. This algorithm works as follows: ciphertext blocks are generated, initially plaintext messages are grouped, and then the plaintext blocks are encrypted using AES and ECC, then data compression generates the cipher blocks and the MAC address is concatenated with the ECC-encrypted AES key to form the final cipher message. Through the description and implementation of the algorithm, the results show that the algorithm can reduce the encryption time, decryption time and total running time complexity without losing security [5].

Huahong Ma and et. al. a state-of-the-art survey on the topic of computational offloading techniques for video data in MEC systems with an early 2024 perspective to cover most up-to-date research was conducted (to our best knowledge), departing by re-thinking conventional task offloading schemes from mobility nature of edge server nodes, grasping existing static and dynamic designs offered using fixed or mobile-based MEC servers. Furthermore, they investigated how the video offloading methods vary in both single-MEC server and multiple-MEC servers. [6].

M. A. El-Mowafy and et. al. contributed to he first two algorithms of which were proposed in 2022 for compressed videos using the advanced H.264/AVC video coding are displayed. This is the first approach of algorithm which was implemented robust video encryption algorithms on chaos maps with random key then need to test for various attack. For video frames, a mixture of steganography and chaos-based cryptography pueblo (steg-chao cipher) has been applied by the second algorithm approach The algorithms are implemented on MATLAB platform, with luminance component Y of various resolution different yuv sequence videos. [7].

Q. Zhang, in 2023 provided a comprehensive review on security issues in 2021 for the information transmission and the method of hybrid encryption algorithms that will be widely used in the future. Also, the various characteristics of algorithms in different systems and some typical cases of hybrid encryption will be reviewed and analyzed to showcase the reinforcement by combining algorithms. Hybrid encryption algorithms will improve the security of the transmission without causing more other problems. Additionally, the way how the encryption algorithms combine to strength the security will be discussed with the aid of an example[8].

Lateef and Kavitha (2025) reinforced smart contract security using bytecode analysis to detect vulnerabilities. Similarly, our dynamic key generation and AES-based chunk encryption enhance video security, mitigating risks of unauthorized access and cryptographic attacks[15].

Lateef and Amri (2023) presents a security framework for online learning platforms using blockchain, OAuth, and MFA. Blockchain ensures immutable academic records, OAuth enables secure authentication, and MFA adds an extra layer of protection. Similarly, in video encryption, our approach enhances security by employing AES-based chunk encryption and dynamic key generation, mitigating risks of unauthorized access and ensuring content integrity[16].

Lateef et al. (2023) introduced the C2MAP protocol using Chebyshev chaotic maps for secure mutual authentication in wireless networks, improving efficiency over traditional ECC-based methods. Similarly, in video encryption, our approach leverages dynamic key generation and AES-based chunk encryption to enhance security, ensuring robust protection against unauthorized access while maintaining computational efficiency.[17]

III. PROPOSED SYSTEM

The increasing amount of video data transmitted across the internet require strong encryption methods, at scale with high-speed performance. There are a few problems when using traditional encryption methods like AES and RSA for video data. Even though, AES is efficient but the problem here is that just one key was used for both encryption and decryption which if compromised has introduced a big security hole. However, RSA is computationally expensive and cannot encrypt large amounts of data as pre-shared key pairs due to its slow processing speed. The challenge is to create a framework that secures the data but also have enough performance at disposal for real-time applications like video streaming and secure file storage.

CipherShield answers these questions by providing a multi-key, dynamic encryption model that increases both security and performance. In this work, the system developed for it, is intended to fulfil such need using an algorithm called Hybrid Dynamic Key Video Encryption Algorithm (HDKVEA) which has been designed and implemented. The approach that is considered by HDKVE Algorithm aims to achieve video file protection that involves an initial key encryption using RSA algorithm and a dynamic chunk-wise AES algorithm. Such a hybrid method can be used to play off the strength and weakness

of each cryptographic technique in order achieve security with strong efficiency. The primary requirement of the system is to securely encrypt and decrypt video files while maintaining the integrity and confidentiality of the content. The system must handle large video files by dividing them into smaller chunks, each encrypted with a dynamically generated key.

This is the two-step process HDKVE Algorithm uses to encrypt and decrypt. First, a master key is encrypted using RSA, which provides us the security of having different keys used for encryption and decryption. Then each chunk of the video file will be encrypted with AES and dynamically generated keys through the same AES. These keys are generated from a pre-defined equation and are unique per chunk, so that it enhances security to the full extent.

The most basic video safeguarding system must guarantee that it is a process for securely encrypting and decrypting video files as well in maintaining confidence that this content arrive unaltered. The second is very tricky since the system needs to encrypt big video files by breaking them down into chunks and each chunk containing in an encrypted file is based on a dynamic key that was generated using chunk wise input. A two-step process for encryption and decryption is used by the HDKVE Algorithm. The process starts by getting a master key encrypted using RSA. After that, each of these chunks are encrypted using AES and a runtime generated keys. These keys are generated through a predefined function and they differ for each chunk which also makes it secure to use.

Here, we present an algorithm to encrypt the video content for enhanced security, where HDKVE Algorithm helps in resolving common complication of just applying symmetric and asymmetric cryptography by using its own unique way of creating keys. This hybrid approach reinforces security, yet allows you to gain all the necessary optimal performance for real-time video applications. This approach optimizes the encryption process enough to handle larger video files while still being suitable for streaming and other high-throughput use-cases.

Encrypting video content poses unique challenges due to the large size of video files and the need for real-time processing. Traditional encryption methods, such as AES (Advanced Encryption Standard) and RSA (Rivest-Shamir-Adleman), have limitations when applied to video data. AES, while efficient, relies on a single key, posing a security risk if the key is compromised. RSA provides secure key exchange but is computationally intensive and not suited for large data volumes.

IV. CipherShield Algorithm With Formulation

The formulation for CipherSield Algorithm for the encryption and decryption process is portrayed below.

A. Encryption Process

1. Let (V) be the original video file of size (S).

$$V = \{C_1, C_2, \dots, C_S\}$$

2. Divide (V) into (n) chunks (C_i) such that:

$$\sum_{i=1}^{n} |C_i| = V$$

3. Initialize VID from first 16 bytes of first video chunk.

$$VID = \{C_1[1], C_1[2], \dots, C_1[16]\}$$

4. Generate a master AES key using the ECC equation with VID. The modified fundamental ECC equation is,

$$y = \sqrt{(x^3 + ax + b) mod(p)}$$

And the master AES key is generated as below with int converted value of GUID

$$K_{master} = \sqrt{(x^3 + VID * x + GUID) mod(2^{256})}$$

Where x is derived as below for first chunk,

$$x = int\big(sha256(VID)\big)$$

5. Encrypt the first chunk using master AES key (K_{master})

$$E_0 = E_{AES}(C_0, K_{master})$$

6. Encrypt the symmetric key (K_{master}) using the public RSA key (K_{RSA_Pub})

$$K_{ENC\_master} = E_{RSA}\big(K_{master}, K_{RSA\_pub}\big)$$

7. Concatenate the encrypted master key with the first encrypted video chunk.

$$E_0 = E_0 \parallel K_{ENC\_master}$$

8. For encrypting chunks other than first chunks we use modified GUID to make key generated independent.

$$\mathrm{GUID_{new}} = \mathrm{GUID} \oplus \big(\mathrm{SHA256}\big(\mathrm{K_{prev}}\big)[:16]\big)$$

9. For the rest of video chunks to be encrypted, the key would be derived from previous key K_{Prev} and GUID is as follows. Also (K_{Prev}= K_{ENC_master})when the second chunk is being encrypted, where $i >= 1$.

$$K_{i-1}^{*} = \sqrt{(x^3 + K_{Prev} * x + GUID_{new}) mod(2^{256})}$$

Where, x is obtained as,

$$x = int\big(sha256(Key_{Prev})\big)$$

10. Encrypt each video chunk (C_i) using its corresponding symmetric key (K_i) with the AES algorithm.

$$E_i = E_{AES}(C_i, K_i)$$

11. Store or transmit the encrypted video chunks (E_i) to the receiver having the final cipher as C_{Final}

$$\sum_{i=1}^{n} |E_i| = C_{Final}$$

The encryption process of HDKVE algorithm of CipherShield is depicted in Table[I].

TABLE I. PSEUDOCODE FOR ENCRYPTION PROCESS USING HDKVE ALGORITHM

	Pseudo code
Step 1	Initialize GUID from Receiver and Load RSA public key.
Step 2	Divide the video into chunks.
Step 3	Initialize VID from first 16 bytes of first video chunk.
Step 4	Generate a master AES key using ECC equation with VID.
Step 5	Encrypt First chunk using master AES key and generate the next key using ECC equation with Previous AES key.
Step 6	Encrypt the master AES key using RSA public key and concatenate it with first encrypted video chunk
Step 7	For each chunk:
Step 7.1	-Generate a dynamic key using the ECC equation and previous AES Key.
Step 7.2	-Encrypt the chunk using the dynamic key.
Step 8	Save/Transmit Encrypted chunks.

Note: The encrypted data being stored or transmitted would be more than the actual size of video file as it involves concatenation of the master AES key.

B. Decryption Process

1. Retrieve the encrypted video chunks (E_{Final}) and the GUID information.

2. Extract the first encrypted chunk and the encrypted master AES key i.e. (E_0) and (K_{ENC_master})from (E_0).

$$E_0 = E_0[:256] \text{ and } K_{ENC\_master} = K_0[256:]$$

3. Decrypt the encrypted symmetric key (K_{ENC_master}) using the private RSA key (K_{RSA_Priv}).

$$K_{master} = D_{RSA}(K_{ENC\_master}, P_{Priv})$$

4. Derive the chunk data by decrypting each encrypted video chunk using the key.

* If the first video chunk is to be decrypted the key would be derived as follows. Here *x* is integer from the hash of VID i.e.

$$C_0 = D_{AES}(K_{master}, E_0)$$

Here ($K_{master} = K_0$)i.e. the first key generated and (K_1) is the subsequent key generated.

$$K_1 = \sqrt{(x^3 + VID * x + GUID) mod (2^{256})}$$

*For encrypting chunks other than first chunks we use modified GUID to make key generated independent.

$$GUID_{new} = GUID \oplus (SHA256(K_{prev})[:16])$$

5. For the rest of encrypted video chunks key would be derived from previous key K_{Prev} and GUID is as follows for *i>=1*.

$$K_{i-1}^{*} = x^3 + K_{Prev} * x + GUID_{new}.mod 2^{256}$$

6. Decrypt each of the encrypted video chunks (E_i) using its corresponding symmetric keys (K_i) with the AES algorithm.

$$C_i = D_{AES}(E_i, K_i)$$

7. Reassemble the decrypted video chunks (C_i) to reconstruct the original video file (V).

$$V = \sum_{i=0}^{n} C_i$$

The decryption process of HDKVE algorithm of CipherShield is depicted in Table[II].

TABLE II. PSEUDOCODE FOR DECRYPTION PROCESS USING HDKVE ALGORITHM

	Pseudo code
Step 1	Initialize RSA private key and load GUID data.
Step 2	Extract Encrypted master AES Key, GUID from first encrypted video chunk.
Step 3	Decrypt the master AES key using RSA private key.
Step 4	For each encrypted chunk:
Step 5	-Generate the corresponding dynamic key using the predefined equation.
Step 6	-Decrypt the chunk using the dynamic key
Step 7	Combine the decrypted chunks to reconstruct the video file.
Step 8	Save/transmit the decrypted video.

V. IMPLEMENTATION

To validate the effectiveness and performance of the CipherShield algorithm, a comprehensive web application was developed. This web application was designed to manage the entire life cycle of video encryption and decryption, from the initial upload and chunking of video files to the final decryption and playback. The application architecture was modular, enabling independent testing and optimization of each component. The application is structured into several distinct modules, each responsible for a specific aspect of the encryption and decryption process. This modular design ensures that each component can be independently developed, tested, and optimized, enhancing the overall robustness and efficiency of the system.

A Video Upload and Chunking Module handles the initial upload of video files. Once a video file is uploaded, it is divided into smaller chunks. The chunking process is crucial for parallel processing, enabling efficient encryption and decryption. The size of each chunk is configurable but the default assumed is 1024 kilo bytes or 1mb, allowing the system to balance between security and performance. Upon chunking the video, the Dynamic Key Generation Module helps the system generates unique symmetric keys for each chunk. These keys are generated using a cryptographically secure random number generator to ensure unpredictability and enhance security. The dynamic

nature of key generation ensures that each chunk is encrypted with a distinct key, mitigating the risk of key compromise.

A Symmetric Encryption Module uses the Advanced Encryption Standard (AES) algorithm to encrypt each video chunk with its corresponding symmetric key. AES is chosen for its efficiency and robustness in handling large data volumes. The encryption process ensures that each chunk is securely encrypted, making it difficult for unauthorized entities to access the content. The symmetric keys generated for each chunk are encrypted using the RSA algorithm in the Asymmetric Key Encryption Module. RSA provides secure key exchange, ensuring that the symmetric keys can be safely transmitted and stored. This module ensures that even if the encrypted video chunks are intercepted, the symmetric keys remain protected.

The encrypted video chunks and their corresponding encrypted symmetric keys can be stored or transmitted securely. This module manages the storage and retrieval of encrypted data, ensuring data integrity and confidentiality. Additionally, it handles the secure transmission of data, ensuring that the encrypted chunks and keys can be efficiently transmitted over potentially insecure channels.

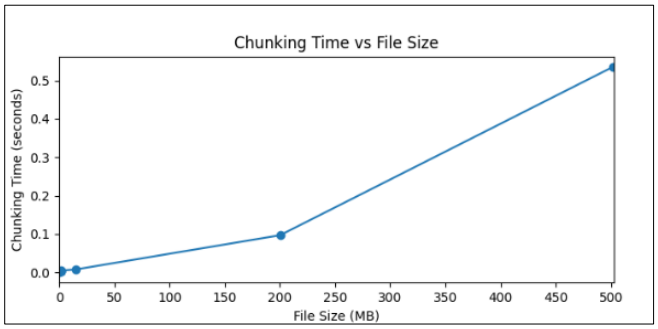

Fig. 1. Time taken to Chunk the video

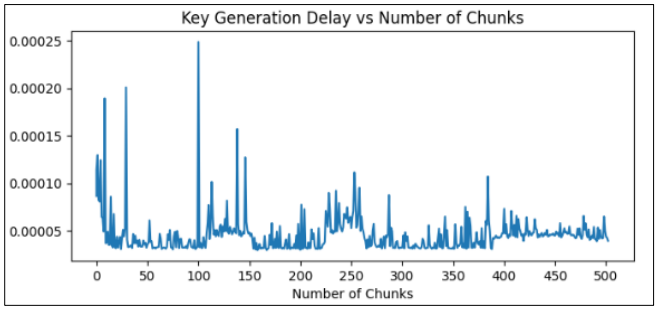

Fig. 2. Time taken generate dynamic keys for the chunks created

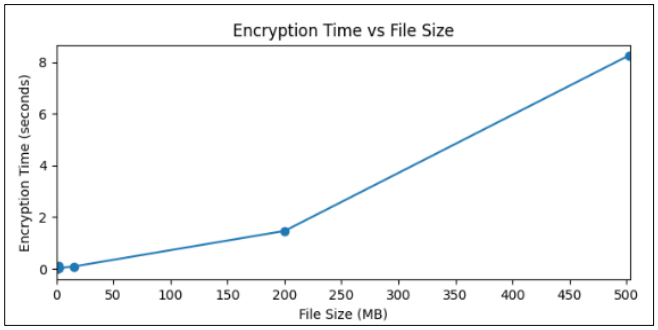

Fig. 3. Time taken to Encrypt the video file using dynamic keys generated.

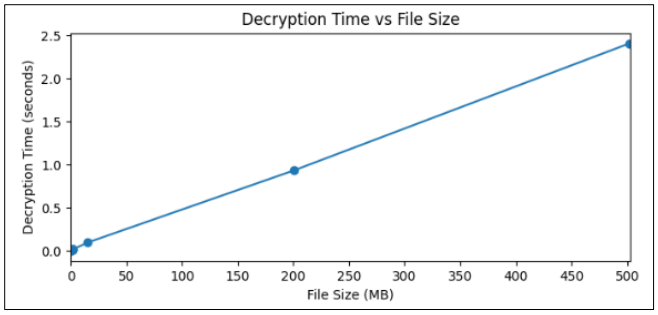

Fig. 4. Time taken to Decrypt the video chunks

When a user requests to view a video, the system retrieves the encrypted chunks, the encrypted symmetric key and the GUID of receiver stored. The decryption process involves first decrypting the encrypted symmetric key using the private RSA key. Once the symmetric key is retrieved, each video chunk is decrypted using its corresponding decrypted symmetric key. Each encrypted video chunk is decrypted using its corresponding symmetric key. The decryption process uses the AES algorithm, resulting in the original chunk. The decrypted chunks are then reassembled to reconstruct the original video file. The reassembly process ensures that the video is reconstructed accurately and can be played back seamlessly.

A. Delay for splitting file into chunks

The time required to divide a video file into smaller chunks, a critical component of our video encryption process. Our implementation utilizes a custom chunking algorithm rather than relying on utilities that are computationally heavy like FFMpeg[1]. The chunk size is dynamically determined based on the video content, aiming for efficient processing and encryption. The time taken by our video processing module to split the video file into chunks is illustrated in Fig.[1]. The results indicate that as the file size increases, the delay also increases gradually, which is expected. This delay is essential for achieving high security in video transmission, despite the overhead introduced by chunk formation.

B. Time to Generate Encryption Keys

To assess the impact of our multi-key encryption approach, this metric calculates the time required to generate each key during the encryption process. Our custom key generation method uses Elliptic Curve Equation(ECE) combined with system-specific information and video-specific data, ensuring each key is unique and secure. Fig[2] shows the time required to generate each key. The first key is derived from the video identifier, while subsequent keys are generated using a combination of the previous key and additional system-specific data. The consistency in parameter lengths results in minimal variation in key generation times, though the precise time captured allows for detecting even small delays. Here the plot is made for a video file of over 500mb in order to determine the delay between key generation of over 500 keys as each chunk is of 1mb. The model determines the delay between 0.00005 to 0.00025s or 0.05 to 0.25 milliseconds.

C. Time to Encrypt Video Chunks

This section examines the time taken to encrypt individual video chunks. Each chunk, once formed, is encrypted using AES with the dynamically generated keys. Unlike traditional methods where chunk sizes are fixed, our implementation allows for varying chunk sizes to optimize encryption efficiency. Fig[3] depicts the delay involved in encrypting each chunk. The chunks are of 1mb in size, leading to encryption times ranging from 0.9

to 1.2 seconds for lower video sizes and is directly proportional to higher video sizes.

D. Number of Keys Generated

This metric tracks the number of encryption keys generated during the video processing. Given our multi-key approach, each chunk is encrypted with a unique key, enhancing security. The total number of keys generated corresponds to the number of chunks produced from the video file. Since the keys are temporarily stored and used only once, the increase in key quantity does not impact memory usage or introduce additional delays in the encryption process.

E. End-To-End Decryption Delay

This section analyses the time taken by the receiver's application to process encrypted video chunks. Upon receiving the encrypted chunks, the application decrypts them sequentially and combines them into the original video. The total decryption time, including chunk reassembly, is depicted in Fig[4]. The decryption delay generally mirrors the encryption delay, although transmission medium factors may introduce additional delays. However, the decryption ensures that the video is ready for playback without significant lag.

F. Comparison of Results

To ensure a fair and consistent comparison, we adhered to similar test conditions as those described in [1]. This included using video files of comparable sizes and applying the encryption and decryption processes within controlled environments. The primary metric under consideration is the time taken to encrypt and decrypt video files. We captured this data for various file sizes, ensuring that the comparison is comprehensive and covers a wide range of scenarios. The results of our comparison are illustrated in the form of a double bar graph. In this graph, each video file size is represented on the x-axis, while the y-axis denotes the time required (in seconds) to complete the encryption and decryption processes as shown Fig[5].

One bar in each pair of represents the encryption time reported in [1], while the other bar indicates the encryption time recorded by our implementation. The results indicate that our algorithm consistently outperforms the base paper's technique in terms of encryption speed, particularly as the video file size increases. This performance improvement is largely attributed to our dynamic key generation mechanism and efficient chunk processing. Similarly, the decryption times are plotted side by side. Our algorithm demonstrates a lower decryption time across all test cases, emphasizing its efficiency in real-time video playback scenarios. The optimized key retrieval and chunk reassembly processes are key factors contributing to this performance gain was demonstrated on a 15.4mb video file and higher ones as well but the below plot can be observed for the same.

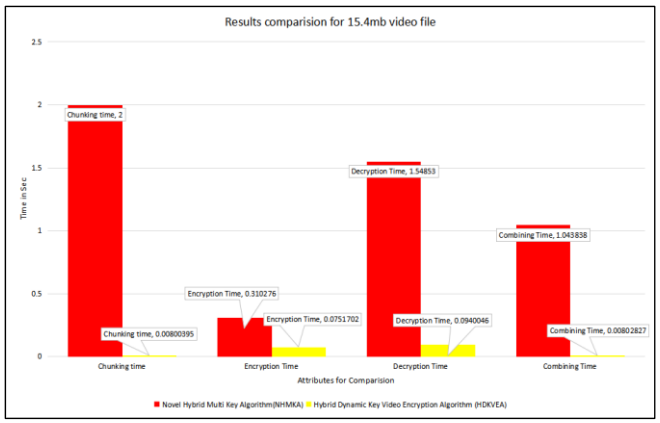

Fig. 5. Results comparison using proposed algorithm and existing algorithm[1] for 15.4mb video file.

VI. CONCLUSION

In this paper, we introduce a secure and efficient approach for video encryption/decryption targeting on protecting the contents of video data. The system developed uses sophisticated cryptographic approaches such as dynamic key production and chunk wise encryption to make sure video data is safe at the time of decryption and transfer. The proposed system utilizes these innovative ways to not only secure video files but with limited process time which makes it feasible for real-time applications.

This shows both secure and high-performance armouring against the state-of-the art techniques. These two methods, combined with the key generation method employed with ECC equation demonstrates on-the-fly nature of keys and allows to decrypt each chunk securely. Because the approach deals with large video files without compromising security in handling these data, make it potentially a useful tool for several applications such as secure video conferencing.

This proposed work is an efficient methodology for video encryption which may be considered as a required advancement within the region of cryptography aimed toward presenting strong and speedy algorithm. With its adaptable scripting system and robust security measures, it becomes a valuable asset when seeking to safeguard video content in an age that is ever more digital. By integrating the system with other security protocols, this kind of approach would help a long way to making it more useful for deployment in different environments involving large data to be transferred securely..

VII. REFERENCES

[1] Y. Fouzar, A. Lakhssassi and M. Ramakrishna, "A Novel Hybrid Multikey Cryptography Technique for Video Communication," in IEEE Access, vol. 11, pp. 15693-15700, 2023, doi: 10.1109/ACCESS.2023.3242616.

[2] M. Iavich, S. Gnatyuk, E. Jintcharadze, Y. Polishchuk and R. Odarchenko, "Hybrid Encryption Model of AES and ElGamal Cryptosystems for Flight Control Systems," 2018 IEEE 5th International Conference on Methods and Systems of Navigation and Motion Control (MSNMC), Kiev, Ukraine, 2018, pp. 229-233, doi: 10.1109/MSNMC.2018.8576289.

[3] Y. Hu, L. Gong, J. Zhang and X. Luo, "An Efficient Hybrid Encryption Scheme for Encrypting Smart Grid Business Data," 2023 IEEE 11th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), Chongqing, China, 2023, pp. 1490-1494, doi: 10.1109/ITAIC58329.2023.10408778.

[4] C. Prashanth, M. Mohamed, K. Latha, S. Hemavathi and D. Venkatesh, "Enhanced Hybrid Encryption Through Slicing and Merging of Data with Randomization of Algorithms," 2021 4th International Conference on

Computing and Communications Technologies (ICCCT), Chennai, India, 2021, pp. 376-380, doi: 10.1109/ICCCT53315.2021.9711883.

[5] T. Yue, C. Wang and Z. -x. Zhu, "Hybrid Encryption Algorithm Based on Wireless Sensor Networks," 2019 IEEE International Conference on Mechatronics and Automation (ICMA), Tianjin, China, 2019, pp. 690-694, doi: 10.1109/ICMA.2019.8816451.

[6] Huahong Ma, Bowen Ji, Honghai Wu, Ling Xing, "Video data offloading techniques in Mobile Edge Computing: A survey", Physical Communication, Volume 62, 2024, 102261, ISSN 1874-4907, doi: 10.1016/j.phycom.2023.102261

[7] M. A. El-Mowafy, S. M. Gharghory, M. A. Abo-Elsoud, M. Obayya and M. I. Fath Allah, "Chaos Based Encryption Technique for Compressed H264/AVC Videos," in IEEE Access, vol. 10, pp. 124002-124016, 2022, doi: 10.1109/ACCESS.2022.3223355

[8] Q. Zhang, "An Overview and Analysis of Hybrid Encryption: The Combination of Symmetric Encryption and Asymmetric Encryption," 2021 2nd International Conference on Computing and Data Science (CDS), Stanford, CA, USA, 2021, pp. 616-622, doi: 10.1109/CDS52072.2021.00111.

[9] X. Zhou, H. Wang, K. Li, L. Tang, N. Mo and Y. Jin, "A Video Streaming Encryption Method and Experimental System Based on Reconfigurable Quaternary Logic Operators," in IEEE Access, vol. 12, pp. 25034-25051, 2024, doi: 10.1109/ACCESS.2024.3365523

[10] T. Shanableh, "HEVC Video Encryption With High Capacity Message Embedding by Altering Picture Reference Indices and Motion Vectors," in IEEE Access, vol. 10, pp. 22320-22329, 2022, doi: 10.1109/ACCESS.2022.3152548.

[11] B. Jiang, Q. He, P. Liu, S. Maharjan and Y. Zhang, "Blockchain Empowered Secure Video Sharing With Access Control for Vehicular Edge Computing," in IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 9, pp. 9041-9054, Sept. 2023, doi: 10.1109/TITS.2023.3269058.

[12] J. Arif et al., "A Novel Chaotic Permutation-Substitution Image Encryption Scheme Based on Logistic Map and Random Substitution," in IEEE Access, vol. 10, pp. 12966-12982, 2022, doi: 10.1109/ACCESS.2022.3146792.

[13] M. Yu, H. Yao, C. Qin and X. Zhang, "A Comprehensive Analysis Method for Reversible Data Hiding in Stream-Cipher-Encrypted Images," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 10, pp. 7241-7254, Oct. 2022, doi: 10.1109/TCSVT.2022.3172226.

[14] A. Al-Hyari, C. Obimbo, M. M. Abu-Faraj and I. Al-Taharwa, "Generating Powerful Encryption Keys for Image Cryptography With Chaotic Maps by Incorporating Collatz Conjecture," in IEEE Access, vol. 12, pp. 4825-4844, 2024, doi: 10.1109/ACCESS.2024.334

[15] Lateef, M. A., & Kavitha, A. (2025). Blockchain smart contract fortification using bytecode analysis to address vulnerabilities. In S. Sethi, B. Sahoo, D. Tosh, S. K. Jayasingh, & S. K. Bhoi (Eds.), *Computing, Communication and Intelligence* (1st ed., pp. 4). CRC Press. Doi: 10.1201/9781003581215

[16] Lateef, M. A., & Amri Nesa Sultani (2023). Enhancing Security In Online Learning Environments: A Holistic Approach via MFA and OAuth, IJNRD - International Journal Of Novel Research And Development , ISSN:2456-4184, Vol.8, Issue 8, page no.d122-d129, August-2023.

[17] Lateef, M.A., Atheeq, C., Rahman, M.A., Faizan, M.A. (2023). Data Aegis Using Chebyshev Chaotic Map-Based Key Authentication Protocol. In: Manchuri, A.R., Marla, D., Rao, V.V. (eds) Intelligent Manufacturing and Energy Sustainability. Smart Innovation, Systems and Technologies, vol 334. Springer, Singapore. https://doi.org/10.1007/978-981-19-8497-6_19