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
malateefz2101@gmail.com

Dr. P. Swetha
Computer Science and Engineering
Jawaharlal Nehru Technological University Hyderabad
Hyderabad, India
drpswetha@jntuh.ac.in

***Abstract*—The proliferation of digital content consumption, particularly in the form of text and video files, underscores the paramount importance of securing these assets against unauthorized access and copyright infringement. In response to this imperative, this research presents a comprehensive approach to file encryption, tailoring the cryptographic strategy based on the nature of the content. For text-based inputs, hybrid of RSA and AES encryption model is employed, combining key exchange strength with efficient encryption. Video based data is emphasized to utilize a blend between RSA and Elliptic Curve Cryptography (ECC) approach, adapting to the dynamic nature of video data through multiple key generation. The implemented system is realized through dynamic software featuring dedicated modules for text and video encryption and decryption in this innovative approach, real-time key generation, based on unique identifiers, dynamically encrypts and decodes file chunks, adapting to evolving data. To assess the effectiveness of the system, rigorous evaluations of its security and performance were conducted. The empirical results underscore the proposed approach's superiority, showcasing advancements in both performance and security metrics. The research stands at the forefront of addressing contemporary challenges in the realm of file security, offering a resilient and efficient solution to combat copyright infringement and piracy risks through cryptographic implementation.**

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

I. INTRODUCTION

The proliferation of digital video content in today's information age necessitates robust encryption methods to ensure the security and privacy of video data. Video encryption is critical for protecting sensitive content from unauthorized access, piracy, and tampering. Unlike text or static images, video files present unique challenges for encryption due to their large size and the real-time transmission requirements. Traditional encryption techniques often fall short in addressing these specific needs, highlighting the importance of developing specialized encryption algorithms tailored to video data.

Encrypting video content involves complexities that stem from the nature of video data. The substantial size of video files requires encryption algorithms that can handle large data volumes efficiently without introducing significant latency. Moreover, the need for real-time processing in applications such as video streaming adds another layer of complexity. Standard encryption algorithms like AES and RSA, while secure, are computationally intensive and may not be optimized for the continuous, high-throughput nature of video streams. These challenges have led to the exploration of hybrid encryption schemes that combine the strengths of both symmetric and asymmetric cryptography to meet the specific demands of video encryption.

One promising approach to video encryption is the hybrid multi-key cryptography technique. 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 secure key exchange and initial key encryption. 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 custom algorithm developed for video encryption addresses the specific challenges associated with video data while enhancing both security and performance. By employing dynamic key generation, the algorithm ensures that each video chunk is encrypted with a unique key, significantly increasing the difficulty of unauthorized decryption. The algorithm is also optimized to minimize latency, making it suitable for real-time video streaming applications. Additionally, the flexibility to save or discard dynamic keys based on specific requirements provides a balanced approach to security and storage management. This combination of robust security measures and efficient processing positions the custom algorithm as a highly effective solution for securing video content in various applications.

Hence a custom algorithm for video-based encryption introduces a sophisticated approach to protecting video data, addressing both the security and performance needs of modern applications. By leveraging 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

Iavich and et. Al. 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.

Y. Hu in 2023. proposed an efficient symmetric cryptographic algorithm based on garbled coding, and combine 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.

C. Prashanth and et. al. in their work on 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 divided into three segments and encrypted with AES, Salsa20, and Fernet algorithms. Then the private keys are encrypted with the RSA (Rivest-Shamir-Adleman) algorithm. The RSA cipher file is stored in phone storage. For decryption, the RSA cipher file is obtained and the responding keys are obtained. With the keys, the AES, Salsa20, Fernet algorithms decrypt the responding encrypted segments. The decrypted segments are combined and the original file is obtained. Then the file is stored in the device.

T. Yue and et. Al. in 2019 proposed a hybrid encryption algorithm based on wireless sensor networks that provides the analysis of existing wireless sensor networks (WSNs) security vulnerability, combining the characteristics of high encryption efficiency of the symmetric encryption algorithm and high encryption intensity of asymmetric encryption algorithm. Firstly, by grouping plaintext messages, this algorithm uses advanced encryption standard (AES) of symmetric encryption algorithm and elliptic curve encryption (ECC) of asymmetric encryption algorithm to encrypt plaintext blocks, then uses data compression technology to get cipher blocks, and finally connects MAC address and AES key encrypted by ECC to form a complete ciphertext 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.

Huahong Ma and et. Al. offered a comprehensive review of the state-of-the-art approaches in computational offloading techniques applied to video data, innovatively revisiting existing task offloading technologies from the perspective of mobility of edge server nodes, including solutions based on both fixed and dynamic MEC servers. Additionally, they explored the differences in video offloading techniques between scenarios involving a single MEC server and those involving multiple MEC servers.

M. A. El-Mowafy and et. Al. proposed two new algorithms for compressed videos by the advanced H.264/AVC video coding are presented. First algorithm approach was implemented by robust video encryption algorithm based on the chaos maps with random key to be tested under different attacks. The second algorithm approach has been implemented by using the hybrid of both steganography and cryptography based on chaotic maps for video frames. The proposed algorithms are conducted on luminance component Y of a set of different YUV video sequences with different resolution using MATLAB software. The simulation results of the proposed algorithms are evaluated using various performance metrics comparing to that with the state-of-art techniques.

Q. Zhang, 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.

III. PROPOSED SYSTEM

The exponential growth of video data transmission over the internet has heightened the need for robust encryption techniques capable of securing large volumes of data without compromising performance. Traditional encryption methods, such as AES and RSA, have inherent limitations when applied to video data. AES, although efficient, relies on a single key for both encryption and decryption, posing a significant security risk if the key is compromised. On the other hand, RSA, while providing secure key exchange, is computationally intensive and not suitable for encrypting large data volumes due to its slow processing speed.

Moreover, the increasing sophistication of cyber-attacks necessitates the development of more advanced encryption techniques. Attack vectors such as brute force attacks, man-in-the-middle attacks, and key compromise scenarios demand encryption methods that can withstand these threats. The challenge lies in developing a framework that not only secures data but also maintains the performance required for real-time applications, such as video streaming and secure file storage. Cipher Shield addresses these challenges by introducing a dynamic, multi-key encryption framework that enhances security and performance.

In the era of digital communication, ensuring the security of multimedia content, especially videos, is of paramount importance. The developed system addresses this need by implementing a robust encryption and decryption algorithm named Dynamic Key Video Encryption Algorithm (DVKEA). The DVKEA is designed to secure video files through a combination of RSA for initial key encryption and AES for chunk-wise dynamic key encryption. This hybrid approach leverages the strengths of both cryptographic techniques to provide a high level of security and 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.

The DVKEA employs a two-step process for encryption and decryption. Initially, RSA is used to encrypt a master key, ensuring secure key exchange. Subsequently, AES is utilized to encrypt each chunk of the video file with dynamically generated keys. These keys are derived from a predefined equation and are unique for each chunk, enhancing security.

The Dynamic Hybrid Multi-Key Video Encryption Algorithm (DHVEA) addresses the challenges of encrypting video content by combining the efficiency of symmetric cryptography with the security of asymmetric cryptography. The method involves dynamically generating unique keys for

each video chunk, encrypting these chunks with symmetric keys, and further securing the symmetric keys using asymmetric encryption. This hybrid approach ensures robust security while maintaining the performance required for real-time video applications. The proposed method optimizes the encryption process to handle large video files efficiently, making it suitable for streaming and other high-throughput applications.

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. These challenges necessitate a new approach that combines the strengths of both symmetric and asymmetric encryption to secure video content effectively without compromising performance.

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.

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

Where x is derived as below for first chunk,

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

5. Encrypt the first chunk using the 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 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 $i >= 1$.

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

Where, x is obtained as,

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

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

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

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

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

TABLE I. PSEUDOCODE FOR ENCRYPTION PROCESS USING DVEA 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).

$$E_0, K_{ENC\_master} = E_0[:256], K_{ENC\_master}[256:]$$

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

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

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

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

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

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

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

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

TABLE II. PSEUDOCODE FOR DECRYPTION PROCESS USING DVEA 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.

TABLE CAPTION

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.

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. 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 Figure 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.

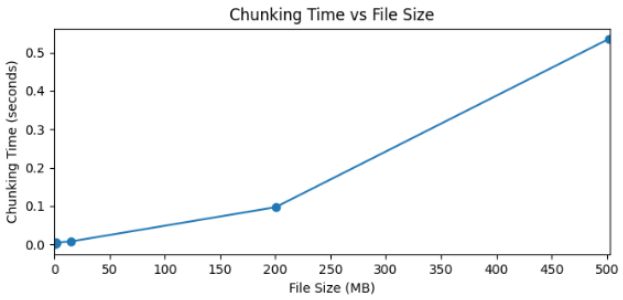

Fig. 1. Time taken to Chunk the video

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. Figure 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.

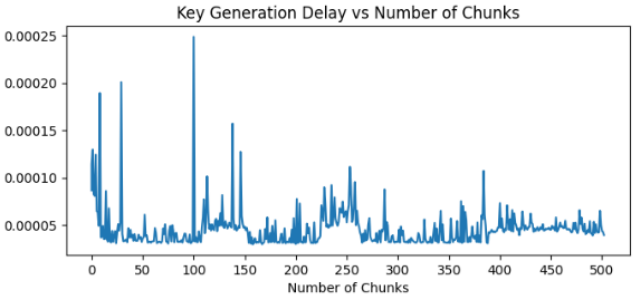

Fig. 2. Time taken generate dynamic keys for the chunks created

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. Figure 6 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.

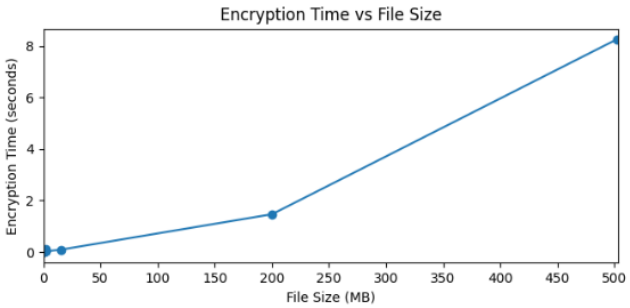

Fig. 3. Time taken to Encrypt the video file using dynamic keys generated.

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 Figure. The decryption delay generally mirrors the encryption delay, although transmission medium factors may introduce additional delays. However, the decryption process ensures that the video is ready for playback without significant lag.

Fig. 4. Time taken to Decrypt the video chunks

F. *Comparison of Results*

Fig. 5. 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.

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.

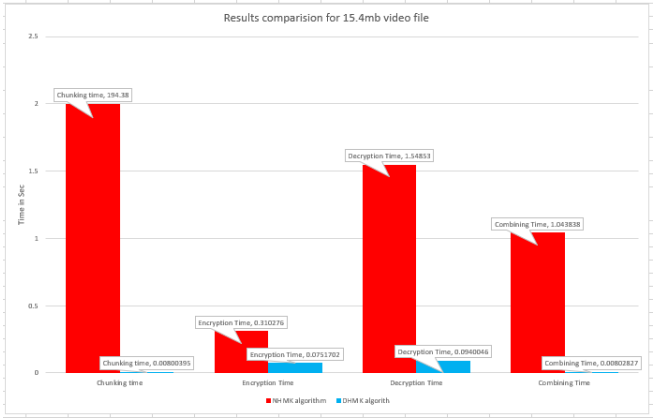

Fig. 6. Results comparison for 15.4mb video file

VI. CONCLUSION

This research presents a sophisticated and highly secure method for video encryption and decryption, designed to protect sensitive content while maintaining efficiency. The developed system leverages advanced cryptographic techniques, including dynamic key generation and chunk-wise encryption, to ensure that video data remains secure throughout the transmission and storage processes. By employing these innovative methods, the proposed system not only enhances the security of video files but also optimizes processing times, making it suitable for real-time applications.

The results of this study demonstrate significant improvements in both security and performance compared to existing solutions. The dynamic key generation mechanism, coupled with the chunk-based encryption approach, ensures that even if one segment of the video is compromised, the integrity of the remaining content is preserved. Furthermore, the system's ability to handle large video files efficiently without sacrificing security makes it a promising solution for a wide range of applications, from secure video conferencing to protected video-on-demand services.

In conclusion, this research contributes to the field of video encryption by offering a robust and efficient method that addresses the limitations of traditional encryption techniques. The system's ability to adapt to various scenarios, combined with its strong security guarantees, positions it as a valuable tool for protecting video content in an increasingly digital world. Future work could explore the integration of this system with other security protocols, further enhancing its applicability in different environments.

VII. REFERENCES

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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

[6] 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

[7] 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.

[8] 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

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.2024.3349470