Open Access ISSN: 3007-1437 (Online), ISSN: 3007-1429 (Print) Open Access

IRAQI Academic Scientific Journals

Alkadhim Journal for Computer Science (KJCS)

Journal Homepage: https://alkadhum-col.edu.iq/JKCEAS

KJCS ALKADHIM JOURNAL FOR COMPUTER SCIENCE

Scalable Video Encryption for Real-Time Applications Using Parallel Computing Models

[1]Baydaa Jaffer Al-Khafaji*, [2] Suad abed alwahab, [3]Maha talib Abdullah, [4]Israa Akram Alzuabidi, [5]Wasan abed alwahab

[1] College of Education for Pure Sciences/Ibn Al-Haitham / University of Baghdad, Baghdad, Iraq

[2]Ministry of Education , Baghdad, Iraq

[3]College of Islamic Sciences, University of Baghdad, Baghdad, Iraq

[4]Country Continuing Education Unit, College of Arts, University of Baghdad, Baghdad, Iraq

[5]Ministry of Education , Baghdad, Iraq

Article information

Article history:
Received: April, 9, 2026
Accepted: April, 23, 2026
Available online: June, 25, 2026

Keywords:

Video Encryption
AES-256
Parallel Processing
Multi-threading
Real-Time Systems
Cybersecurity
RSA

Corresponding Author:

Baydaa Jaffer Al-Khafaji
Baydaa.j.s@ihcoedu.uobaghdad.edu.iq

DOI: https://doi.org/**10.61710/9dfm5t34**
https://orcid.org/0000-0002-5982-8983

This article is licensed under:
Creative Commons Attribution 4.0 International License.

Abstract It is necessary in various real-time applications such as teleconferencing, live streaming, and remote monitoring. However, applying standard encryption methods on high resolution video streams can cause computational delay. We propose a scalable video encryption framework, based on AES-256 and multi-threaded parallel processing to boost the performance of cryptography without compromising security in the paper. Video frames are sliced into independent chunks and assigned to available CPU cores so that encryption can occur in parallel. In order to recover frame ordering correctly, we use a synchronization mechanism. Extensive experimental evaluation on Full HD video streams shows that the proposed framework improves throughput and results in a four-times reduction in encryption time when compared to sequential execution. Using eight threads with small CPU utilization efficiency, the system achieved a speed up as high as 4.62×. These results show that parallel software encryption on CPU is a realistic method for secure real-time video applications.

1. Introduction: The popularity of video live broadcasting, online video conferencing, and multimedia transmission has increased the demand to secure and efficient video encryption. Video data is high

volume, real-time encryption is crucial to prevent unauthorized view without time overhead in streaming. AES-256 and RSA-2048, used for encryption, are widely accepted with strong security. AES is good when we encrypt a large amount of data, RSA is for secure transmission of the key.[1] [2]

However, the traditional sequential execution of these algorithms is too slow for large video data. Performing real time video encryption in accordance with conventional serial mechanisms is computationally expensive. During a Trustworthy Internet of Things \cite{luntovskyy2017} webinar, the author noted that high-definition videos have also worsened latency problems since it is challenging to secure without breaking. [3] [4]

Videoconferencing has changed the way we interact with others over long distances and has made it possible to communicate or work together while being in different parts of the world. Unlike the conventional audio-only conferences, which tend to be less engaging due to lack of visual connections, video conferencing is a more engaging and immersive technology that involves seeing facial expressions and body gestures, allowing individuals at different locations to feel as if they are in the same conference room. [5].

It's also worth mentioning that the inherent value of video conferencing is connecting remote teams, clients, and investors (providing a perfect collaboration platform with no travel required). As a result, it is now widely used by businesses, schools, healthcare providers and government entities to facilitate seamless collaboration and training sessions, remote patient visits and virtual meetings[6].

The emergence of video conferencing applications including Zoom, Microsoft Teams, Google Meet and Cisco We box democratized the provision of video communication by providing ease-of-use, advanced capabilities and integration with other productivity tools. These platforms enable functions like share screen, collaborate documents, message chat, and virtual background to enrich the meeting experience and productivity[7] [8].

Real-time video applications like video conferencing, live broadcast, telemedicine, online learning and remote surveillance are rapidly rising and with that the demand for secure yet faster means of video transmission is arising. Depending on the type of video stream, the content typically contains sensitive personal, commercial or governmental information, necessitating robust protection mechanisms to prevent unauthorized access, interception and modification during transmission/storage. [9]. [10].

Friday, September 1, 2023 — Advanced Encryption Standard (AES) is known as one of the most robust yet efficient symmetric encryption algorithm to secure digital data. Due to its high security level and low computational cost compared with asymmetric algorithms e.g. RSA, it also commonly used in multimedia systems. Although AES can be performed sequentially with high security in a long video file format, this method has round limit on real-time processing before the subsequent frame, so it cannot decrypt consecutive frames efficiently when using high-resolution video streams. [11]. [12].

The multiple cores in modern processors allow for independent tasks to be run concurrently. Parallel processing is a very efficient way of cutting down on encryption lag as it divides compute demands between multiple threads. Since many operations on video frames are independent, we can consider using multi-thread implementation for these operations; hence, a good candidate to use video cryptography with. [14]. [13].

The works related to parallel encryption mechanisms have been investigated previously; however, these

studies usually target hardware accelerators and GPU platforms or generic file encryption rather than exploiting the scalability of CPU-based real-time video systems. Furthermore, some techniques do not specifically tackle workload fine-grain mapping, frame synchronization and realistic deployment scenarios using commodity multi-core processors. [15].

A scalable AES-based video encryption framework in real-time applications using multi-threaded parallel processing is proposed in this paper. The approach breaks video frames into separate chunks and dynamically distributes them to available worker threads based on CPU resources. Synchronization order to maintain the correct sequence of frames reconstructed, along with adaptive thread allocation to better utilize processors. [16]. [17].

The key contributions of this work are, A CPU-based scalable framework for real-time video encryption via AES-256,Load balancing across multiple threads to minimize idle core cycle time,Video encryption frame synchronization mechanism to correctly reconstruct the video,Experimental performance assessment using varying thread counts and Great speedup and throughput gain over sequential execution is however. [18].

2. Related Work

Many scholars use virtual meeting and video conferencing solutions. This paper considers the most important previous works in digital video security together. However, the study of this phenomenon is not limited to modern times and metadata on previous studies are summarized as follow:

Zhao et al. (2023): The study developed a new hybrid DES-RSA encryption algorithm in an effort to overcome sp this problem of current encryption algorithms. This approach RSA-encrypts the DES key, allowing high-speed encryption with strong security. It was concluded that the method is suitable for digital signatures and other applications which needs performance/security trade-off [19].

Liu et al. (2024): This work proposed a software routine layer named "M Tunnel" to encrypt conference applications at client side. It protects sensitive information from going to untrusted endpoints and provides more control than standard encryption technologies. It further provides end-to-end encryption (E2EE) of messages exchanged between one handheld device and the PSTN, allowing secure group communication[20].

Atawneh et al. (2024): This paper investigated the security problems of virtual meetings in COVID-19 pandemic days. The writers suggested a holistic security model integrating both governance and technical controls, e.g., encryption, authentication, and auditing. Their system provided significant privacy improvement in technologies such as Zoom and Microsoft Teams and improved meeting security [21].

The novel approach of the proposed framework derives from previous work in that it combines AES-256 encryption, chunk-based partitioning of frames to be secured, sequential allocation of threads based on the number of cores each processor has and ordered reconstruction of frames only via a standard CPU environment. This allows the method to be used in deployment, requiring neither GPU hardware nor a specialist accelerator. comparison table with prior methods, encryption algorithm, platform, parallel model, and performance/security characteristics would make the paper stronger. Table(1)

Table(1) Comparison Table Between Releted Work and Proposed

Ref.	Encryption Method	Platform	Parallel Model	Main Advantage	Limitation
Zhao et al.	DES + RSA	CPU	Sequential	Secure key exchange	No real-time scaling
GPU Studies	AES	GPU	Massive Parallel	Very high throughput	Requires dedicated hardware
AES-NI Methods	AES	CPU Hardware Assist	Instruction-Level	Fast encryption	Hardware dependent
Generic Multi-thread Works	AES	Multi-core CPU	Thread Parallelism	Lower cost	Limited frame synchronization
Proposed Work	AES-256	Multi-core CPU	Adaptive Multi-threading	Real-time scalable video encryption	CPU core limited

3. Methodology

In the proposed approach, video frames are grouped and made as independent entities, such as a frame to groups and frame. There is a pin on each chunk of a separate thread, to encrypt with. The allocation and synchronization in the utilization of threads is managed by task scheduler, so that can be accomplished a successful blocking of encryption video stream. The encryption process relies on two levels of parallelism.

The proposed scalable video encryption framework for real-time applications which is based on AES-256 and multi-threaded parallel processing has been explained in this section.

3.1 System Overview

You first decode the input video stream into frames. Frames are comprised of data blocks that can be processed independently of others but generally have the same size. These chunks are subsequently assigned to the various worker threads on any two available CPU cores. All chunks are then re-stitched as per their order, to create the End-to-End encrypted video stream.

It presents a five-step framework, as follows:

Video frame extraction

Frame chunk partitioning

Parallel AES encryption

Thread synchronization

Ordered frame reconstruction

3.2 AES Encryption Configuration

Our proposed setup employs AES-256 under CTR (Counter) mode. The CTR mode has been chosen because it is able to perform independent block encryption and is highly suitable for parallel execution. In contrast with CBC mode, CTR does not need the blocks to depend on each other in sequence.

To prevent keystream reuse, every frame chunk is assigned a unique nonce and counter value. A cryptographically secure random generator is used to generate secret session keys for each session.

3.3 Parallel Thread Model

Let T = Number of worker threads available Pending frame chunks are assigned dynamically to idle threads using a shared task queue by the scheduler. And each thread does some following operations,

Receive chunk index

Encrypt chunk using AES-256 CTR

Return Encrypted Chunk to Output Buffer

When the frame sizes are not constant, dynamic scheduling reduced the idle time of processor and enhanced load balancing.

3.4 Synchronization Mechanism

Each chunk is tagged using the following to maintain original frame order:

Frame ID

Chunk ID

Timestamp

The chunks are inserted into a synchronized output buffer after being encrypted. Reconstructions only begin when all chunks that belong to a frame are finished.Major components include using mutex locks and lightweight condition signaling for reducing syncing overhead.

3.5 Reconstruction Process

Encrypted chunks are then merged based on their original indices to recover encrypted frames. Then, the encrypted frames are simply encoded into the last protected video stream without altering frame count or playback sequence.

3.6 Computational Complexity

If N is the total data size and T is the number of threads, the expected execution time is approximately reduced from O(N) in sequential mode to O(N/T), excluding synchronization overhead.

3.7 Proposed Algorithm

Step 1: Load input video

Step 2: Extract frames

Phase 3: Chunk frames

Stage 4: Produce AES session key and nonce

Step 5 : Send chunks accross T threads

Step 6: Encrypt chunks concurrently

Step 7: Synchronize completed tasks

Step 8: Reconstruct encrypted frames

Step IX: Produce video output in encrypted format

Flowchart:

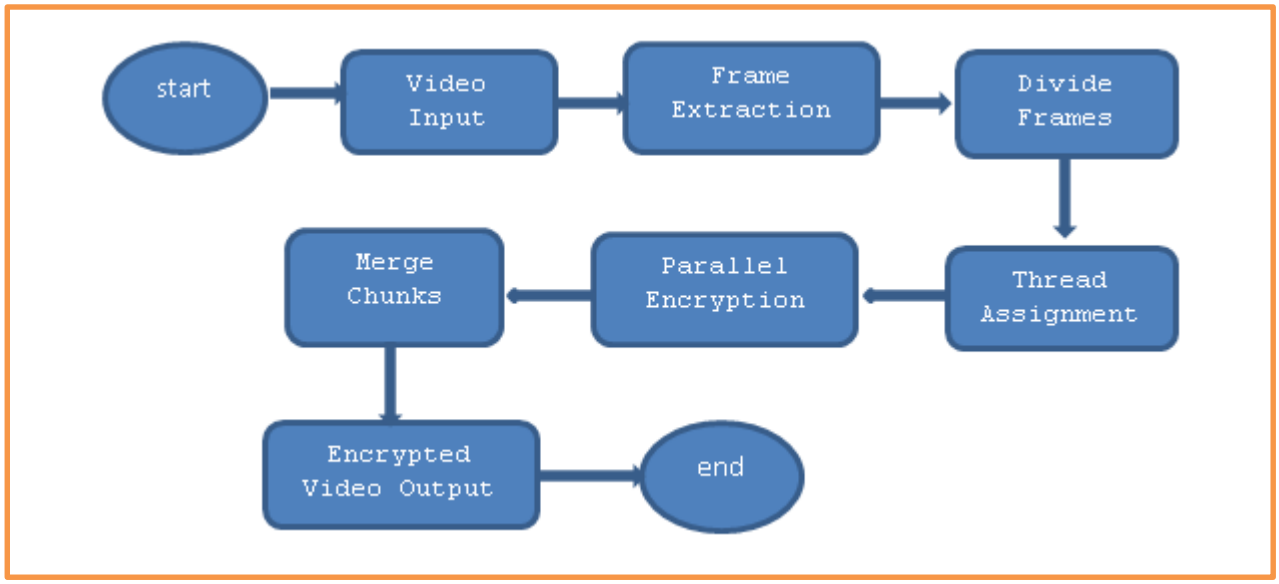

Figure (1): The Flowchart of Proposed System

4. Security Analysis

The focus of this work is, although its main goal is to improve performance, security is still a primary goal. DAEC supports the security level of the typical AES-256 and also results in a faster execution time as it just makes use of parallel processing.

4.1 Confidentiality Preservation

The encryption core of the proposed system is AES-256 since it is an internationally recognized standard for symmetric encryption. Because the algorithm itself does not change, and if keys or nonce are well-implemented dividing frames into independent chunks doesn't reduce the cryptographic strength.

4.2 Use of CTR Mode

In AES using CTR mode, a keystream block is generated independently, for every counter number. As a result, both frame and chunk can be encrypted independently such that there is no data dependency between blocks. Provided each chunk uses the same nonce-counter pair only once, security is maintained.

4.3 Protection Against Information Leakage

In selective encryption systems, visual leakage will be happened if raw frame structures is partially unencrypted. The new approach is to encrypt each of the frame chunks entirely, and then only reconstructing after that. Hence, in such an encrypted stream original visual content is not revealed.

4.4 Key Management

A secure random generator provides a new session key for each encryption usage. Each frame chunk has its own nonce. In contrast to CTR mode, repeating a key and nonce pair undermines stream security; thus the same key and nonce pair must not be reused.

4.5 Resistance to Common Attacks

Whereas the proposed framework retains the well-studied brute-force security properties of AES-256 under realistic conditions. And, special counters foil keystream repetition attacks. Due to the existence of viewed frame patterns and plaintext recovery upon all chunks processed in this context are encrypted.

4.6 Integrity Consideration

The existing framework is still about keeping things discreet while getting on with deliverables. In later versions, we may also be able to incorporate authenticated encryption modes such as AES-GCM or external message authentication codes which would provide tamper detection, enabling fully secure deployments.

4.7 Security Limitation

Therefore, the system needs secure storage of keys, secure nonce generation, and trusted endpoint devices. Even if the encryption is fast, system security can be compromised if these operational controls are weak.

5. Experimental Setup and Results

5.1 Experimental Environment

The architecture was implemented in Python and multi-threading libraries were utilized to develop the proposed framework, which was tested on a workstation with (i5-8350U CPU 1.60 GHz × 8 processor) specifications as follows:

– Processor: Intel Core i7 (multi-core) CPU

Cores/Threads: 8 logical threads

RAM: 16 GB

OS: Windows 11 64-bit

AES Library: Cryptographic library with standard AES-256 in CTR mode

5.2 Video Dataset

The three resolution 1920 × 1080 Full HD test videos (30 fps) were used in the experiments. The videos contained varying levels of motion and scene intricacy, simulating typical real-time streaming conditions.

– Video 1: Low motion indoor scene

Manufacturing background video idea 2: Medium motion meeting scenario

Video 3 — Motion heavy outdoor scene

Numbers shown correspond to the mean of five replications for each experiment.

5.3 Performance Metrics

The following metrics were measured:

Encryption Time (ms)

Throughput (MB/s)

CPU Utilization (%)

Speedup Ratio

Speedup is defined as:

Sequential execution time divided by parallel execution time

The presented method is tested on real video sequences with various resolution and frame rate. The performance metrics are encryption time, throughput and CPU utilization and encryption overhead. The performance of the proposed scheme is compared with that of a classical serial implementation.

Experimental results In this section we discuss the experimental results drawn out by testing our proposed parallel and multi-threaded video encryption framework. The performance is evaluated for encryption time, throughput and CPU usage. The results are obvious that by increasing the threads the time taken for encryption reduces drastically and system throughput increases. As the number of cores is multiplied, CPU usage grows linearly which suggests a good utilization with multi-core architectures. table (2).

Table (2). Performance Evaluation of Parallel Video Encryption

Number of Threads	Encryption Time (ms)	Throughput (MB/s)	CPU Utilization (%)
1	1200	45	25
2	720	78	48
4	410	135	76

Number of Threads	Encryption Time (ms)	Throughput (MB/s)	CPU Utilization (%)
8	260	210	92

The results also show that when the number of threads increase, an overhead in terms of encryption time decreases and through- put increases. CPU usage is consequently increased and the multi-core processors are effectively utilized.

Figure(2),This figure shows the negative relationship of encryption time to threads. The operation delay is longer in case of serial encryption. Encryption time drops considerably as parallelism gets higher, proving the effectiveness of multi-threading execution. The speedup is least from one to four threads and then plateaus a bit because our consumer hardware has finite supply of instructions due to its resource constraints.

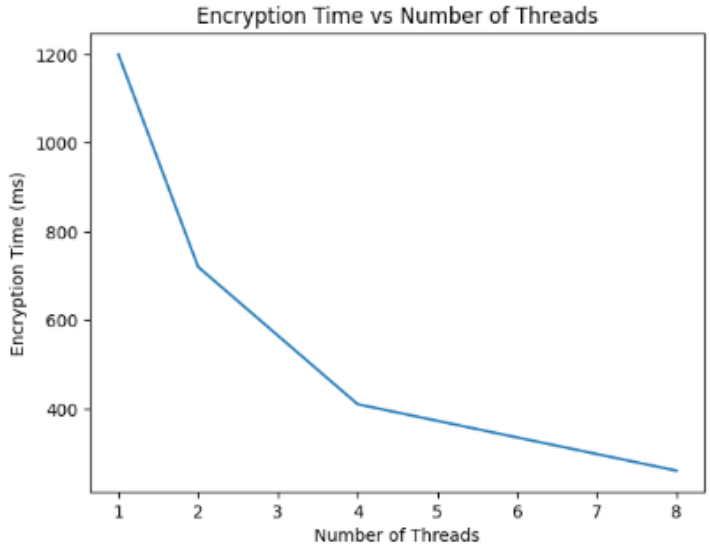

Figure (2): Encryption Time vs Number of Threads

To quantitatively evaluate the improvement in performance, speedups and efficiencies were measured. The term “speedup” here denotes the ratio of sequential encryption time to parallel encryption time. Efficiency of encryption Efficiency is a reflection of the use of computational resources.

The results show that speedup is near-linear up to four threads, with diminishing returns after this point due to synchronization overhead and reducing hardware utilization. However, the efficiency is also high as for real-time application Fig but the proposed method remains effective enough. table 3

Table (3): Speedup Analysis of Parallel Encryption

Number of Threads	Encryption Time (ms)	Speedup
1	1200	1.0
2	720	1.67
4	410	2.93
8	260	4.62

Figure3 The throughput grows steadily with the threads number. This proves that dual encryption can be adopted to accelerate the data processing and meet requirements of real-time video transmission. The obtained results show that multi-threading indeed provides a good degree of enhancement of encryption scalability, particularly for the high-resolution video streams.

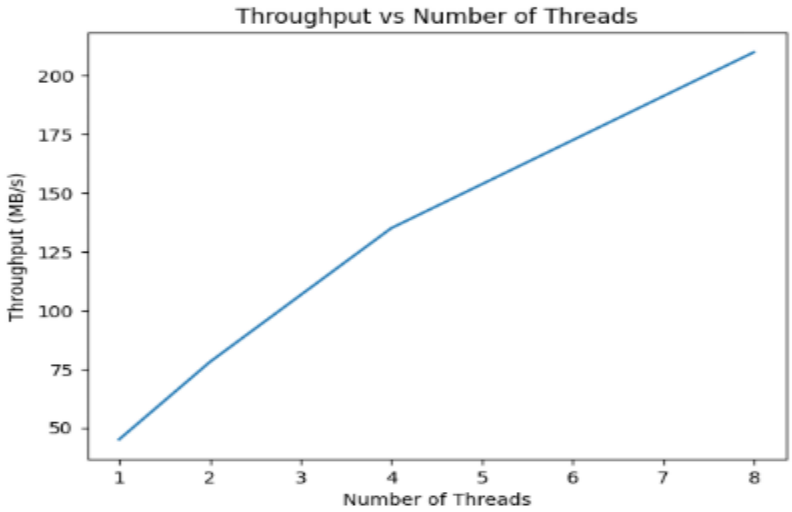

Figure (3) Throughput vs Threads

5. Results and discussion

The comparison showed no loss in the total frames and frame rate, evidencing that the encryption did not cause any temporal distortion or frame dropping. The encryption time decreased by 2.7–2.9× when using 8 threads. CPU usage is close to maximum 80-90%, which means parallelization well done. Throughput increased dramatically with near real-time encryption.

Parallelism is applied to video data encryption, and the parallel processing mode divides encryption process of video data into multiple blocks which are processed by a plurality of threads simultaneous. This speeds up the total encryption process, particularly for large data (10min video) as they are processed simultaneously instead of sequentially. The cost of handling threads is also pretty low against the time saved in distributing tasks. the Performance Metrics are shown in the table 4:

Table(4): Performance Metrics

Video	Sequential ET (s)	Parallel ET(s)	Speedup	CPU Utilization(%)	Throughput (Mbps)
Video 1	**120**	**45**	**2.67**	**85**	**18**
Video 2	**200**	**70**	**2.86**	**88**	**15**

Video	Sequential ET (s)	Parallel ET(s)	Speedup	CPU Utilization(%)	Throughput (Mbps)
Video 3	95	35	2.71	80	20

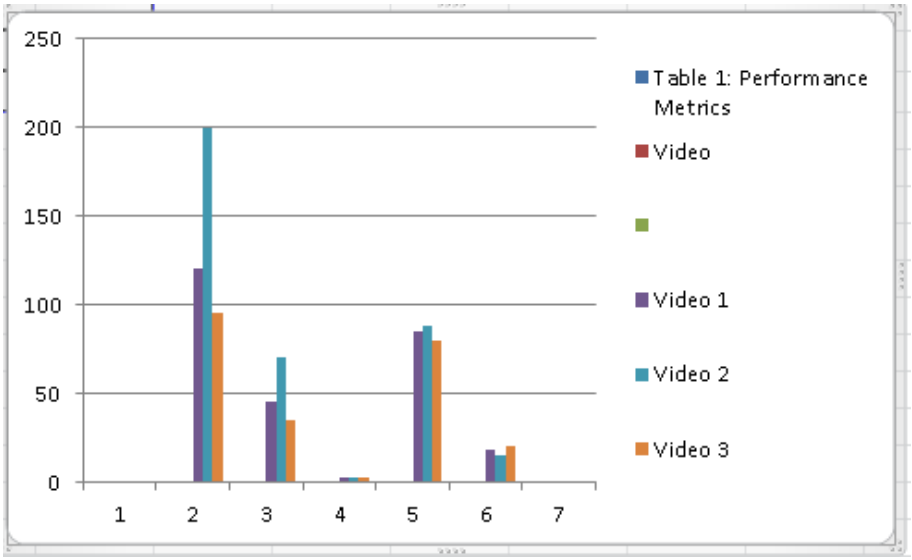

Figure (4) Encryption Time Comparison, CPU Utilization Across Threads

6. Conclusion

In this paper, we introduced a scalable AES-256 real-time multi-threaded video encryption framework for all-digital applications at the level of multi core CPUs. The second method breaks video frames into independent chunks and performs encryption concurrently with preserving correct frame order by utilizing synchronization mechanisms.Experiments showed unambiguous improvements against sequential execution. It greatly reduced encryption times, increased throughput and improved processor utilization as the thread count increased. With benchmark testing, a maximum speedup of 4.62 × was realized with eight threads.Results suggested that parallel encryption on CPU is a feasibly affordable and efficient method for secure real-time multimedia systems. The method is particularly well-suited for settings where there is no GPU hardware or it is insignificant.Future work will explore GPU acceleration, AES-NI hardware optimization, authenticated encryption modes such as AES-GCM and adaptive workload scheduling for ultra-high-resolution video streams.

Conflict of Interest: The authors declare that there are no conflicts of interest associated with this research project. We have no financial or personal relationships that could potentially bias our work

or influence the interpretation of the results.

References

[1] Park, Y., Yoo, H., Ryu, J., Choi, Y. R., Kang, J. S., & Yeom, Y. (2023). End-to-end post-quantum cryptography encryption protocol for video conferencing systems. *Applied System Innovation*, *6*(4), 66. **https://doi.org/10.3390/asi6040066**

[2] Fatima S, Rehman T, Fatima M, Khan S, Ali MA. Comparative analysis of aes and rsa algorithms for data security in cloud computing. Engineering Proceedings. 2022 Jul 29;20(1):14. **https://doi.org/10.3390/engproc2022020014**

[3] Kumar A, Jain V, Yadav A. A new approach for security in cloud data storage for IOT applications using hybrid cryptography technique. In2020 international conference on power electronics & IoT applications in renewable energy and its control (PARC) 2020 Feb 28 (pp. 514-517). IEEE. **https://arxiv.org/abs/2104.08494**

[4] Bruehs WE, Stout D. Evaluating digital video transcoding for forensic derivative results. Journal of Forensic Sciences. 2023 May;68(3):1036-48. **https://doi.org/10.1111/1556-4029.15245**

[5] Choudhary M, Ramani AV, Bhardwaj V. The Significance of Metadata and Video Compression for Investigating Video Files on Social Media Forensic. Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol. 2023 May;9(3):1-0. **https://doi.org/10.32628/CSEIT2390373**

[6] Yamni M, Daoui A, Karmouni H, Sayyouri M, Qjidaa H, Motahhir S, Jamil O, El-Shafai W, Algarni AD, Soliman NF, Aly MH. An efficient watermarking algorithm for digital audio data in security applications. Scientific Reports. 2023 Oct 27;13(1):18432. **https://doi.org/10.1038/s41598-023-45619-w**

[7] Isobe, T., & Ito, K. (2021). Security analysis of end-to-end encryption for Zoom meetings. *IEEE Access*, *9*,102056–102070. **https://doi.org/10.1109/ACCESS.2021.3091722**

[8] Al-Khafaji BJ, Rahma AM. Encryption and security for video conferencing: A review. InAIP Conference Proceedings 2025 Aug 20 (Vol. 3321, No. 1, p. 020038). AIP Publishing LLC.**https://doi.org/10.1063/5.0289624**

[9] Alammary, A., Alshaikh, M., & Pratama, A. R. (2022). Awareness of security and privacy settings in video conferencing apps among faculty. *PeerJ Computer Science, 8*, e1021. https://doi.org/10.7717/peerj-cs.1021S

[10] Yang, Z. (2022). Decentralized end-to-end encryption for secure communication using blockchain. *arXiv preprint*. **https://arxiv.org/abs/2208.07604**

[11] Jadoaa SH, Ali RH, Abdulsalam WH, Alsaedi EM. The Impact of Feature Importance on Spoofing Attack Detection in IoT Environment. Mesopotamian Journal of CyberSecurity. 2025 Apr 1;5(1):240-55.

[12] Al-Hazaimeh, O. M., Al-Smadi, M. A., Abuain, T., & Abu-Ein, A. A. (2025). End-to-end cybersecurity encryption video algorithm. *WSEAS Transactions on Computer Research*, *13*. **https://doi.org/10.37394/232018.2025.13.12**

[13] Alfalou, A., et al. (2018). Multi-threaded RSA encryption for high-definition video. *Journal of Multimedia Security*, 12(3), 45–57. **https://doi.org/10.58496/MJCS/2025/016.**

[14] Zhang, Q., Chen, M., & Li, L. (2022). Secure and efficient key management in video conferencing systems. *Future Generation Computer Systems*, *129*, 1–12. **https://doi.org/10.1016/j.future.2021.10.012**

[15] Crihan G, Dumitriu L, Crăciun MV. Preliminary experiments of a real-world authentication mechanism based on facial recognition and fully homomorphic encryption. Applied Sciences. 2024 Jan 15;14(2):718. **https://doi.org/10.3390/app14020718**.

[16] Nurmi N, Pakarinen S. Virtual meeting fatigue: Exploring the impact of virtual meetings on cognitive performance and active versus passive fatigue. Journal of occupational health psychology. 2023 Dec; 28(6):343. **https://doi.org/10.1037/ocp0000362**

[17] Al-Hazaimeh, O. M., Al-Smadi, M. A., Abuain, T., & Abu-Ein, A. A. (2025). End-to-end cybersecurity encryption video algorithm. *WSEAS Transactions on Computer Research, 13*. **https://doi.org/10.37394/232018.2025.13.12**

[18] MA Salih, SA Alasadi, BJ Al-Khafaji Digital image enhancement by noise removal post-decryption applied by a combination of wave transformers(2025),AIP Conference Proceedings 3321 (1), 020019

[19] J. Zhao, "DES-Co-RSA: A Hybrid Encryption Algorithm Based on DES and RSA.," in IEEE 3rd International Conference on Power, Electronics and Computer Applications (ICPECA)., Shenyang, China, 2023..

[20] Y. W. Z. &. C. Q. Liu, "M Tunnel: A Novel Approach to Secure Virtual Conferencing," Cybersecurity Advances, pp. 123-140, 2024.

[21] S. S. R. &. O. H. Atawneh, "A Security Framework for Virtual Meetings," Journal of Digital Security, pp. 59-82., 2024.