JUTICS : Journal of Electrical Engineering and Informatics
Vol. 1, No. 1 (2026)

Implementation of ChaCha20-Poly1305 and BLAKE3 Algorithms for Data Security of Reporters on the Satpol PP Complaint Portal

Muhammad Zainul Hasan [1], Abdul Fatah [2], Fanny Brawijaya [3],
[1] Informatics Engineering, Faculty of Engineering, Nurul Jadid University, Probolinggo, Indonesia (mzhasan1404@gmail.com)
[2] Informatics Engineering, Faculty of Engineering, Nurul Jadid University, Probolinggo, Indonesia (af434613@gmail.com)
[3] Magister Informatics, Sains and Technology, Universitas Islam Negeri Maulana Malik Ibrahim Malang, Indonesia (240605210008@student.uin-malang.ac.id)

Article Info

Article history:

Submission 12 Desember 2025
Accepted 15 January 2026
Published 16 January 2026

Keywords:

Data Security;
Cryptography;
ChaCha20-Poly1305;
BLAKE3;
Complaint System;
Whistleblowing System.

ABSTRACT

The Civil Service Police Unit (Satpol PP) requires active community participation in enforcing Local Regulations (Perda). However, the effectiveness of the complaint system is often hampered by public fears of identity leaks and the slow process of uploading digital evidence on low-end mobile devices. This study implements the ChaCha20-Poly1305 *stream cipher* encryption algorithm and the BLAKE3 *hash* function on a web-based complaint portal. ChaCha20-Poly1305 was chosen for its efficiency on devices without cryptographic hardware acceleration, while BLAKE3 was used to verify the integrity of digital evidence at a speed far exceeding that of SHA-256. The implementation results demonstrate that the system can secure whistleblower data using a client-side encryption mechanism with minimal computational overhead, thereby ensuring the confidentiality of the whistleblower's identity and the authenticity of the evidence, without compromising the user experience.

Corresponding Author:
Muhammad Zainul Hasan,
Informatics Engineering, Faculty of Engineering, Nurul Jadid University,
Paiton, Probolinggo 67291, Indonesia
Email: mzhasan1404@gmail.com

1. INTRODUCTION

In the era of digital government transformation, data security guarantees are a crucial foundation for building public trust, particularly in e-government services that handle sensitive public information. The application of modern cryptographic algorithms in government systems is not only intended to maintain confidentiality, but also to ensure data integrity, thereby ensuring its validity is not questioned [1]. This is crucial given the demand for compliance with application security standards at the regional level to support the adaptation of new habits [2]. Specifically, the Civil Service Police Unit (Satpol PP), as the enforcer of regional regulations, plays a strategic role in implementing Smart Cities [3], which is highly dependent on active community participation through a web-based public complaint system [4].

However, the biggest challenge in whistleblowing systems is the public's fear of identity leaks. Legal protection for whistleblowers is necessary to ensure their safety from threats from the parties being reported [5]. In addition to legal aspects, technical constraints are also a significant obstacle. Conventional encryption standards, such as the Advanced Encryption Standard (AES) established by NIST [6]Although very secure, they require high computational resources. In scenarios involving the use of Internet of Things (IoT) devices or low-end mobile phones with limited power, heavy algorithms such as AES often burden device performance [7]. This finding aligns with the research of Crowley and Biggers, who emphasize the necessity for length-preserving encryption in entry-level processors lacking specialized hardware acceleration [8].

Previous research by Arifin has shown that on low-end mobile devices, AES-based encryption performs significantly worse than modern stream cipher algorithms [9]. As a solution, Bernstein introduced ChaCha as a variant of Salsa20 designed for high speed without compromising security [10]. The reliability of ChaCha20 has been recognized globally and standardized in the Internet Engineering Task Force (IETF) RFC 8439,

alongside Poly1305, as a message authenticator[11]. In Indonesia, the effectiveness of this algorithm has also been demonstrated in the design of a secure e-voting system [12], which proves its ability to handle sensitive data with low latency.

Identity confidentiality and evidence integrity, especially for photos or digital files, are essential. To efficiently secure digital images without slowing uploads [13]. This study uses BLAKE3, a fast cryptographic hash function that employs a parallel Merkle Tree architecture [14], rather than traditional, slower hash functions. This study implements ChaCha20-Poly1305 and BLAKE3 on the Satpol PP complaint portal. The aim is high cryptographic performance in web environments (Web Assembly) [15]. By protecting data and ensuring usability [16]. The system creates a secure, responsive reporting tool for the community across devices.

2. METHOD

This research uses an experimental approach, employing a system development method focused on implementing modern cryptography. The algorithms applied were selected based on efficiency on commodity hardware without special instructions. The following are the details of the methods used:

2.1. ChaCha20-Poly1305 Algorithm

ChaCha20 is a stream cipher that operates by manipulating a 4×4 state matrix. Unlike conventional block ciphers, which process data in fixed blocks, ChaCha20 generates a keystream that is then combined with the plaintext to produce the ciphertext. The core of this algorithm's diffusion strength lies in the Quarter Round function [10]. This function performs a series of simple modular arithmetic operations: addition, bit rotation, and Exclusive OR (XOR). These operations are known as ARX (Add-Rotate-Xor).

In each encryption round, the ARX operation is applied repeatedly to the matrix elements, thoroughly scrambling the data. The main advantage of the ARX approach is its swift execution on general-purpose processors (CPUs) because it uses basic instructions available in all computer architectures without requiring special cryptographic hardware [9]. To ensure message authentication, ChaCha20 is paired with Poly1305. Poly1305 generates a unique authentication tag (Message Authentication Code) for each encrypted message. This tag serves to verify that a third party has not manipulated the data during the transmission process from the reporting device to the server [11].

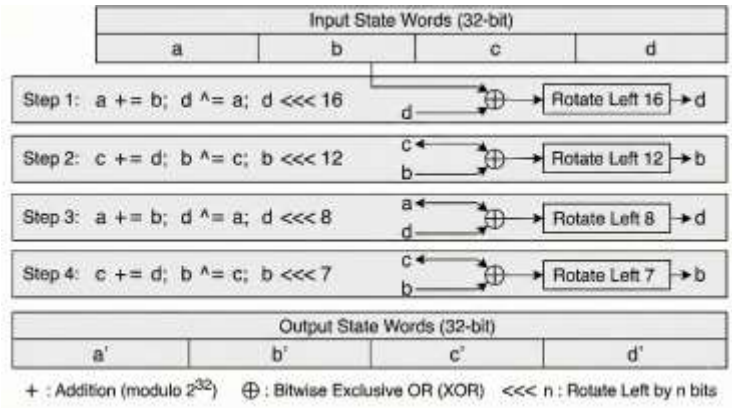

Figure 1. Mathematical Operation Diagram of *Quarter Round* ChaCha20

2.2. BLAKE3 Hashing Algorithm

To verify the integrity of digital evidence in the form of photos, this study applies the BLAKE3 hashing algorithm. Unlike the SHA-2 family (such as SHA-256), which processes data serially, BLAKE3 uses a binary tree structure known as a Merkle Tree [14]. This concept allows large amounts of data to be broken down into smaller pieces (chunks). Each piece of data can be processed independently and simultaneously (parallelism) using multithreading technology on modern processors. The hash results from each piece are then combined in stages to produce a single root hash value. This parallelization approach solves the speed bottleneck that typically occurs in traditional hash algorithms, enabling data integrity verification with very low latency even on mobile devices without compromising cryptographic security.

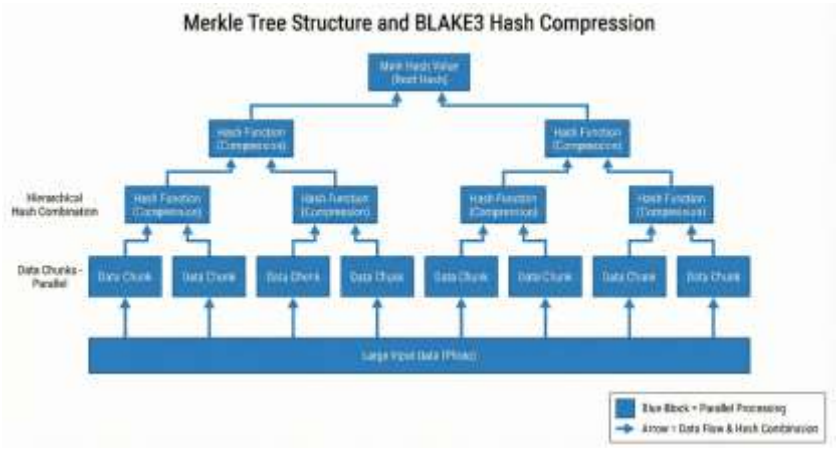

Figure 2. *Merkle Tree* Structure and BLAKE3 *Hash* Compression

2.3. System Design

The security implementation applies the principle of client-side encryption, starting from the reporter's browser to the server side (Satpol PP operator). The process begins when the reporter fills out the complaint form and uploads supporting evidence. The system automatically hashes the evidence file using BLAKE3 to obtain its digital fingerprint. Next, the system generates a random 256-bit session key. The generation of session keys and nonces utilizes the window. crypto. The getRandomValues() API is available in modern browsers to ensure cryptographically secure random entropy (CSPRNG) and prevent value repetition (collisions).

All form data and hash values are encrypted into a single data packet using ChaCha20-Poly1305 with the session key. To ensure secure key delivery, the session key is sent to the server via an SSL/TLS (HTTPS) encrypted channel that guarantees key confidentiality during transit, in accordance with standard secure protocols [11]. Thus, data crossing the internet is encrypted and can only be decrypted by servers with valid sessions.

On the server side, report data is stored in an encrypted database table. Session keys received from clients are managed separately, with strict access controls: either they are temporary in the server's memory during the verification process, or they are stored in a separate table, isolated from the primary data, to minimize the risk of a single point of failure. Decryption is only performed when an authorized operator accesses the report for verification. During decryption, the system also validates the Poly1305 tag and re-matches the BLAKE3 hash to ensure that the evidence is not corrupt or manipulated.

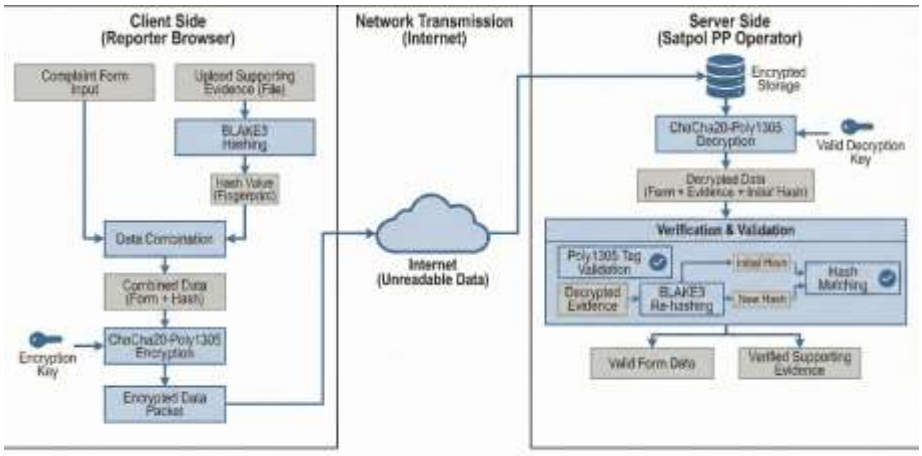

Figure 3. Cryptography Flow in the Complaint System

3. RESULTS AND DISCUSSION

3.1. Interface Implementation

The *user interface* implementation was designed with a *mobile-first* approach to facilitate citizen reporting, as the majority of users access the service via smartphones. The system was built using a responsive web framework that automatically adjusts the page layout based on the user's screen size, from low-resolution mobile devices to *desktop* monitors. The primary focus of interface development is minimalism and ease of navigation (*usability*) to reduce the cognitive load on reporters in urgent situations. On the *backend*, ChaCha20 encryption and BLAKE3 hashing run seamlessly when users press the send button, without interfering with the user experience or causing excessive loading *time*. The following is a visualization of the system interface implementation:

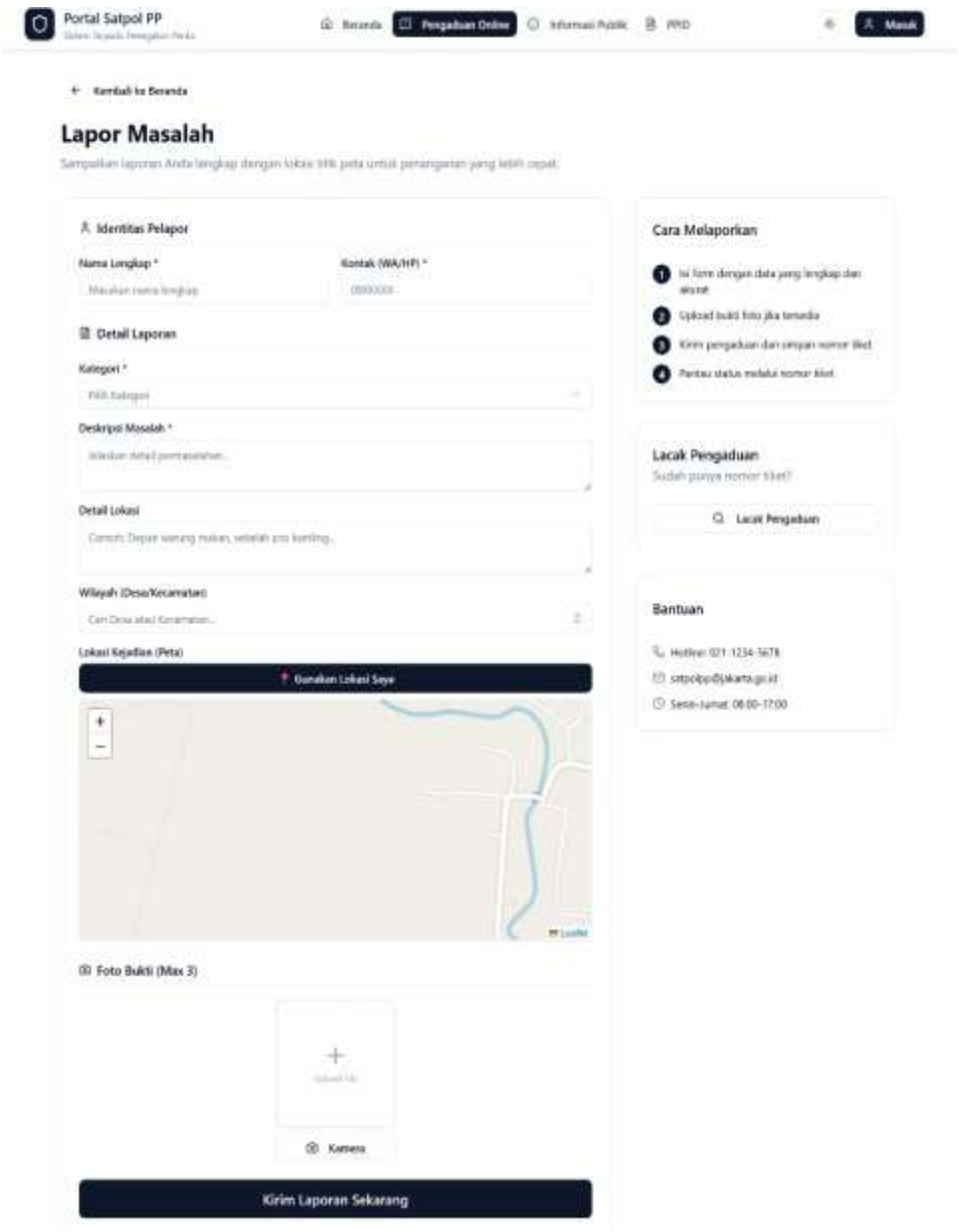

Figure 4. Implementation of the Complaint *Form* with Photo *Input* Validation

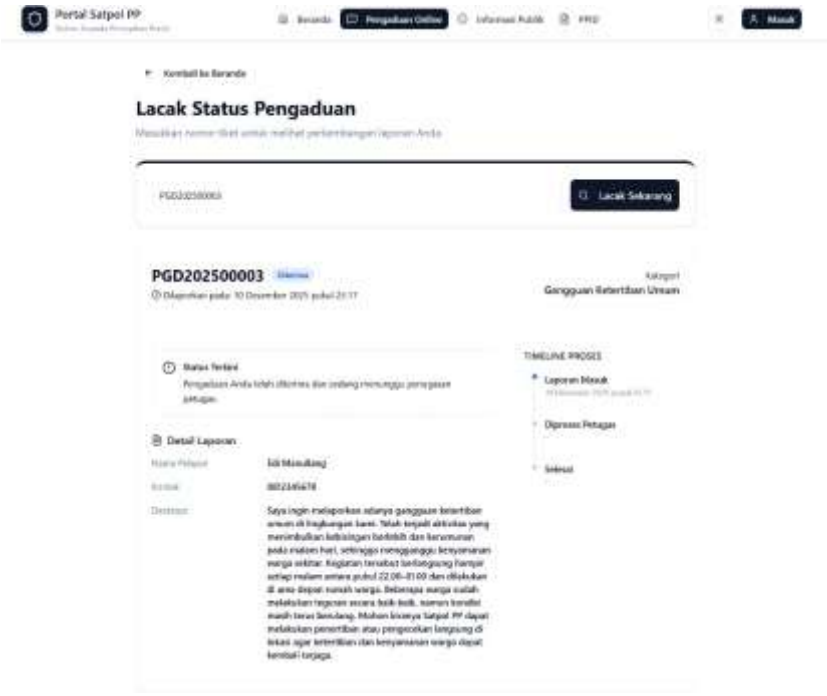

Figure 5. Report Status Tracking Page

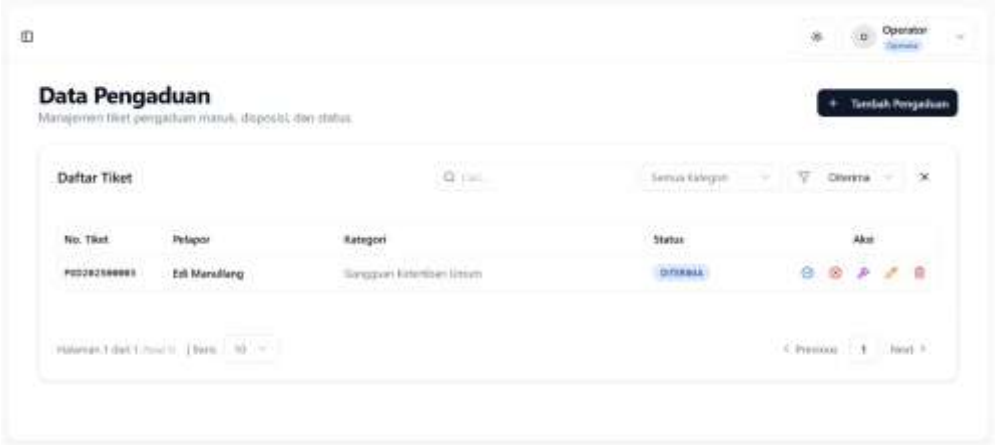

Figure 6. Operator *Dashboard* for Follow-up Management

3.1.1 Implementation of Encrypted Data Storage.

Table 1. Transformation of *Plaintext* Data to *Ciphertext* (*Database*)

Data Attributes	Input Data (Plaintext)	Data Stored in Database (Base64 Format)	Component Description
Reporter Name	Edi Manullang	7zKq3+L92/m8A2b1C9d4E5f6G7...	Nonce (12 bytes) + Ciphertext + Poly1305 Tag (16 bytes)
Contact	0812345678	X9s2/kL1+n7M3bP4qR5...	Unique column-by-column encryption (different nonce)

Photo Evidence	photo_proof.jpg	a45f891d2c3e... (Hex)	BLAKE3 Hash Digest (For integrity verification)

Table 1 visualizes the implementation of data storage on the server database. In accordance with the system design, identity data is not stored in plaintext; instead, it is encrypted using the ChaCha20-Poly1305 algorithm and converted to a Base64 string.

The data format stored in the database column is a *concatenation* of three crucial elements:

1. **Nonce (12 Bytes):** A unique random number generated using **CSPRNG** each time encryption is performed. This is crucial to prevent the same *ciphertext* pattern (*a reuse-key attack) even if the reporter's input data* is identical.
2. **Ciphertext:** The result of scrambling the primary data using the ARX (Add-Rotate-Xor) operation.
3. **Poly1305 MAC (16 Bytes):** *An* authentication *tag* at the end of the string that serves to validate that the data has not been illegally manipulated at the database level.

Meanwhile, for digital evidence, the system stores *a digest* or digital fingerprint generated by the BLAKE3 algorithm. This *hash* value is used to verify the file's integrity when the operator opens it, ensuring that the photo evidence matches that uploaded by the reporter.

3.2. Performance Testing and Analysis

Performance testing was conducted to measure the computational efficiency of the proposed algorithm compared to current industry standards. The test scenario simulated a mobile device environment with limited resources (*low-resource devices*), where CPU and memory efficiency are crucial.

3.2.1. Testing Device Specifications

The validity of performance testing is ensured by using physical devices (*real devices*) that represent the minimum standards of devices still widely used by the lower-middle class. The **Xiaomi Redmi 7** was chosen as the testbed because it has minimal specifications (3 GB RAM and an older 600-series processor), making it ideal for testing algorithm efficiency claims under resource-constrained conditions.

Table 2. Client Device Specifications (*Low-End* Testing Environment)

Component	Technical Specifications
Device	Xiaomi Redmi 7
System on Chip (SoC)	Qualcomm Snapdragon 632 (14 nm)
CPU Architecture	Octa-core (4x1.8 GHz Kryo 250 Gold & 4x1.8 GHz Kryo 250 Silver)
Memory (RAM)	3 GB LPDDR3 933 MHz (Single Channel)
Internal Storage	64 GB eMMC 5.1
Operating System	Android 10 (Q) - MIUI 12
Browser	Google Chrome Mobile v143.0 (Latest Stable)
Network Connectivity	Wi-Fi (Download: 54.2 Mbps, Upload: 57.1 Mbps)

The selection of the **Snapdragon 632** chipset with **3 GB** of LPDDR3 RAM is crucial. This device has a much lower memory *bandwidth* than modern devices. If ChaCha20-Poly1305 and BLAKE3 run smoothly on this device, then they will run very well on almost all devices on the market.

3.2.2. Testing Scenarios

To ensure the validity of performance measurements on resource-constrained devices, a series of tests was designed in three isolated scenarios. All tests were run on a Xiaomi Redmi 7 test device, with no other background processes running to maintain CPU and memory stability. The overall testing methodology flow is shown in Figure 7.

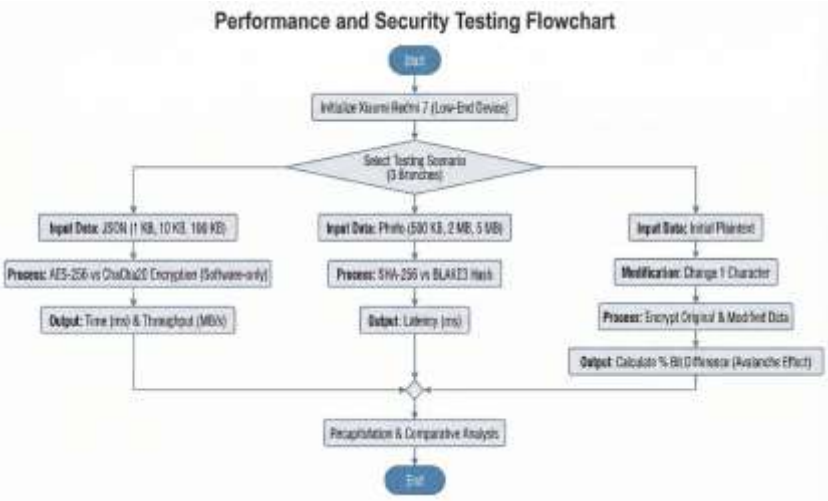

Figure 7. System Performance and Security Testing Flow

Figure 7 illustrates the testing flow, which is divided into three main scenarios. Testing was conducted in isolation on a Xiaomi Redmi 7 device to ensure consistent performance measurements for **ChaCha20-Poly1305** and **BLAKE3** against the comparison algorithms (**AES-256** and **SHA-256**). Details of the three scenarios are as follows:

1. **Scenario 1: Encryption Performance Testing (Text Data)**This scenario aims to measure the efficiency of the **ChaCha20-Poly1305** algorithm compared to the industry standard **AES-256-GCM**. Testing was conducted using simulated JSON data representing reporters' identity attributes, with data packet sizes of **1 KB, 10 KB, and 100 KB**. Both algorithms are run in *software-only* mode to simulate real-world conditions on *low-end* Devices that lack cryptographic hardware acceleration. The measured parameters are *execution time* in milliseconds (ms) and data *throughput* (MB/s).

2. **Scenario 2: Evidence Integrity Testing (Photo Hashing)** The second scenario focuses on the speed of creating digital fingerprints (*message digests*) for report evidence attachments. The test compares the conventional **SHA-256** *hash* function (which operates serially) with BLAKE3 (which uses Merkle trees for parallel processing). The test samples are photo files with sizes **of 500 KB, 2 MB, and 5 MB**. The main parameter recorded is the processing latency to the generation of the hash value.

3. Sc**enario 3: Security Testing (Avalanche Effect)** The third scenario aims to test the diffusion quality of the **ChaCha20** algorithm. The test is conducted by applying the *Avalanche Effect* principle, where one character in *the* initial *plaintext* (e.g., "Violation Report") is changed, then re-encrypted using the same key. The goal is to verify whether a slight change in the input can yield a significant change in the ciphertext, indicating resistance to statistical analysis.

3.2.3. Text Data Encryption Analysis (Identity)

The first test compares the encryption speeds of AES-256-GCM and ChaCha20-Poly1305 on JSON text containing reporter identity attributes. This test simulates a *software-only implementation* scenario, where the algorithm runs entirely on the CPU without the aid of a dedicated hardware acceleration module (AES-NI), reflecting real-world conditions on *low-end* devices such as the Xiaomi Redmi 7.

Table 3. Encryption Time Comparison (AES-256-GCM vs ChaCha20-Poly1305)

Data Size	Algorithm	Average Time (ms)	Throughput (MB/s)
1 KB	AES-256-GCM	0.946 ms	1.03 MB/s
	ChaCha20-Poly1305	0.436 ms	2.24 MB/s
10 KB	AES-256-GCM	1.766 ms	5.53 MB/s
	ChaCha20-Poly1305	1.222 ms	7.99 MB/s
100 KB	AES-256-GCM	13.202 ms	7.40 MB/s
	ChaCha20-Poly1305	9.666 ms	10.10 MB/s

Based on the results in **Table 3**, a consistent performance pattern is seen:

- **AES-256-GCM** shows slower performance as the data size increases. At a load of 100 KB, this algorithm requires an average time **of 13,202 ms** with *a throughput of* **7.40 MB/s**. This latency occurs because the *MixColumns* matrix operation in AES intensively burdens the CPU cycle when executed in software.
- **ChaCha20-Poly1305** demonstrates superior efficiency with a processing time of only **9.666 ms** and *a throughput* of **10.10 MB/s** for 100 KB data sizes. This advantage is achieved because ChaCha20 uses simple ARX (*Add-Rotate-Xor*) operations that are friendly to CPU registers on the Snapdragon 632 architecture. These results confirm that ChaCha20 is more suitable for public reporting applications targeting devices with limited specifications.

3.2.4. Attachment Hashing Analysis (Photo Evidence)

The second test focused on the speed of generating digests, or digital fingerprints, for photo evidence. This test compared the conventional SHA-256 *hash* function with BLAKE3, which utilizes *WebAssembly* technology for parallel processing.

Table 4. Comparison of Photo File *Hashing* Speed (SHA-256 vs BLAKE3)

File Size Scenario	SHA-256 Time (ms)	BLAKE3 Time (ms)	Speed Improvement
500 KB (High Compression)	13.30 ms	4.00 ms	3.3x
2 MB (Average Photo)	56.40 ms	15.30 ms	3.7x
5 MB (Maximum Limit)	139.10 ms	38.00 ms	3.7x

Referring to the data **in Table 4**, there is a significant performance improvement in BLAKE3:

- **SHA-256** experiences a sharp decrease in speed with large files. To process a 5 MB file, SHA-256 requires **139.10 ms**. Latency above 100 ms can cause micro-stutter in the application interface.
- **BLAKE3** demonstrates high stability with a processing time of only **38.00 ms** for a 5 MB file, or about **3.7 times faster** than SHA-256. For smaller file sizes (500 KB), BLAKE3 also remains superior with a time **of 4.00 ms** compared to SHA-256's 13.30 ms. This speed is enabled by the Merkle Tree structure in BLAKE3, which splits data into chunks and processes them in parallel (multithreading), allowing the digital evidence validation process to run instantly without

disrupting the user experience.

3.2.5. Avalanche Effect Analysis (Security)

Security aspects are tested using the *Avalanche Effect* parameter to ensure the diffusion quality of the ChaCha20 algorithm. This principle requires that a slight change in the input (1 bit/character) must change approximately 50% of the *ciphertext* output.

Table 5. *Avalanche Effect* Test of the ChaCha20 Algorithm

Original Text Sample	Modified Text (Change 1 character)	Ciphertext Change (%)
"Violation Report"	"Violation Report"	51.97
"Jl. Ahmad Yani No. 45"	"Ahmad Yani Street No. 46"	52.38

The results in **Table 5** show excellent diffusion characteristics:

- In the first test, changing the string "Violation Report" to "Violation Report" resulted in a **51.97**% change in the *ciphertext* bits.
- In the second test, changing one digit of the address resulted in a change of **52.38**%.

These two values, which are slightly above 50%, indicate that the algorithm is susceptible to changes in input (*plaintext*). This proves that ChaCha20 produces random output and shows no linear pattern with respect to its input, making it highly resistant to statistical analysis and brute-force decryption attempts.

3.3. Research Limitations

Although this research successfully proves the efficiency of the algorithm on low-specification devices, several limitations need to be considered in the current implementation:

1. **Dependence on Web Platforms (Browser-Based)** The system is currently implemented as a responsive web application. This limits the system's ability to perform more secure *low-level* memory *management*, such as instantly removing traces of the encryption key from RAM, which is usually done more efficiently in *native* applications (Android/iOS).
2. **Static Key Exchange Mechanism** The process of sending *session keys* from the client to the server currently still relies entirely on the security of the standard SSL/TLS (HTTPS) channel. This research has not implemented more advanced dynamic key exchange protocols, such as *the Noise Protocol Framework*, to improve session security independently of transport layer security.
3. **File Size Limitations:** Performance testing and system operational limitations are restricted to a maximum file size **of 5 MB** per photo attachment. This study does not yet present performance data for handling larger files (e.g., high-resolution videos), which may require more complex chunking or stream-processing mechanisms.
4. **Network Testing Conditions** Performance testing was conducted on a stable Wi-Fi network with a download speed of 54.2 Mbps and an upload speed of 57.1 Mbps. This study did not include testing in poor or unstable network conditions, such as areas with high latency or frequent connectivity fluctuations, so analysis of system resilience and encryption retry mechanisms was not included.

4. CONCLUSION

The implementation of a hybrid security architecture using the ChaCha20-Poly1305 algorithm and BLAKE3 hash function on the Satpol PP complaint portal has proven successful in addressing the dual challenges of reporter privacy and performance efficiency on mobile devices. Based on the results of testing and analysis, several key conclusions can be drawn:

1. **Client-Side Encryption Efficiency:** The ChaCha20-Poly1305 algorithm demonstrated superior performance compared to AES-256-GCM in a simulated environment without hardware acceleration (software-only). With **an average speed efficiency increase of 40% to 2 times** across varying data packet sizes, the system can guarantee the confidentiality of reporter data without burdening computational resources.

2. **Data Integrity Speed:** The implementation of BLAKE3 has a significant impact on *user experience*, especially in processing ample digital evidence. BLAKE3's ability to process photo file hashing **3.7 times faster** than SHA-256 minimizes report delivery latency, ensuring evidence *integrity* is maintained even under unstable network conditions.

3. **Proven Security:** *Avalanche Effect* testing resulted in an average *ciphertext* change **of around 50%**, indicating that the system has strong resistance to pattern analysis or basic cryptanalysis attacks. This provides security assurance for people who want to participate as *whistleblowers* without fear of their identities being exposed.

As a suggestion for further development, research can be expanded to include native mobile applications (Android/iOS) to leverage low-level access to device memory management and to explore the use of noise framework protocols to secure session key exchanges more dynamically.

REFERENCES

[1] R. W. Sudibyo and T. A. Purnomo, "Penerapan Algoritma Kriptografi Modern pada Aplikasi E-Government untuk Menjamin Integritas Data," *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, vol. 8, no. 6, pp. 1205–1214, 2021, doi: 10.25126/jtiik.202183574.

[2] S. A. Khairunisa, A. Mardiyah, E. Agustine, and N. A. Rakhmawati, "Analisis Kepatuhan Keamanan Aplikasi E-Government Tingkat Daerah sebagai Penunjang New Normal," *Explore: Jurnal Sistem Informasi dan Telematika*, vol. 11, no. 2, pp. 123–130, 2020, doi: 10.36448/jsit.v11i2.1563.

[3] S. Rahayu and M. I. Fikri, "Implementation of Smart City in Improving Public Order Services: A Case Study of Satpol PP," *Journal of Government and Public Policy*, vol. 9, no. 2, pp. 88–99, 2022.

[4] A. Pratama and B. Santoso, "Evaluation of the Security of Web-Based Public Complaint Systems in Local Government Agencies," *Jurnal Sistem Informasi dan Teknologi*, vol. 5, no. 2, pp. 112–120, 2023.

[5] B. H. Siregar, "Perlindungan Hukum Bagi Whistleblower dalam Pengungkapan Tindak Pidana Korupsi dan Pelanggaran Perda," *Jurnal Hukum dan Peradilan*, vol. 9, no. 3, pp. 301–315, 2020.

[6] National Institute of Standards and Technology, "Advanced Encryption Standard (AES)," 2001.

[7] D. Kurniawan, "Analysis of Lightweight Cryptography for Data Security in Low-Power IoT Devices," *Jurnal Nasional Teknik Elektro dan Teknologi Informasi (JNTETI)*, vol. 11, no. 1, pp. 45–52, 2022, doi: 10.22146/jnteti.v11i1.3321.

[8] P. Crowley and E. Biggers, "Adiantum: length-preserving encryption for entry-level processors," *IACR Transactions on Symmetric Cryptology*, vol. 2018, no. 4, pp. 39–61, 2018, doi: 10.13154/tosc.v2018.i4.39-61.

[9] M. S. Arifin, "Comparative Analysis of AES and ChaCha20 Encryption Algorithms on Low-End Mobile Devices," *International Journal of Cyber Security and Digital Forensics*, vol. 10, no. 4, pp. 34–42, 2021, doi: 10.17781/P002672.

[10] D. J. Bernstein, "ChaCha, a variant of Salsa20," *Workshop Record of SASC*, vol. 8, pp. 3–5, 2008.

[11] Y. Nir and A. Langley, "ChaCha20 and Poly1305 for IETF Protocols," Jun. 2018. [Online]. Available: https://www.rfc-editor.org/info/rfc8439

[12] A. F. Rochim, A. A. Zulfi, and K. T. Martono, "Design of secure e-voting system using ChaCha20-Poly1305 algorithm," in *2020 7th International Conference on Information Technology, Computer,*

and Electrical Engineering (ICITACEE), IEEE, 2020, pp. 1–6. doi: 10.1109/ICITACEE50144.2020.9239105.

[13] F. Nugroho, "Optimization of Digital Image Data Security Using the Hybrid Cryptosystem Method," *Scientific Journal of Informatics*, vol. 8, no. 1, pp. 30–38, 2021, doi: 10.15294/sji.v8i1.26660.

[14] J. O'Connor, J.-P. Aumasson, S. Neves, and Z. Wilcox-O'Hearn, "BLAKE3: One Function, Fast Everywhere," in *Security and Cryptography for Networks (SCN 2020)*, in Lecture Notes in Computer Science, vol. 12238. Springer, 2020, pp. 326–348. doi: 10.1007/978-3-030-57990-6_17.

[15] H. Setiawan and D. R. Sari, "Comparative Analysis of Symmetric Cryptography Algorithm Performance on the Web Assembly Platform," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 7, no. 1, pp. 15–22, 2023, doi: 10.29207/resti.v7i1.4673.

[16] Y. A. Singgalen, "Evaluation of E-Government Application Usability using System Usability Scale," *Journal of Information Systems and Informatics*, vol. 5, no. 1, pp. 102–115, 2023, doi: 10.51519/journalisi.v5i1.436.