ISSN: 2321-2152

IJMECE

International Journal of modern electronics and communication engineering

E-Mail
editor.ijmece@gmail.com
editor@ijmece.com

www.ijmece.com

ISSN2321-2152

www.ijmece .com

Vol 11, Issue 3, 2023

Implementing the SHA Algorithm in an Advanced Security Framework for Improved Data Protection in Cloud Computing via Cryptography

Dharma Teja Valivarthi

Tek Leaders, Texas, USA

Email: teja89.ai@gmail.com

ABSTRACT

The goal of the suggested security framework is to improve data security in cloud computing settings by combining several cryptographic techniques with the Secure Hash Algorithm (SHA). With the design, implementation, and evaluation of the security architecture, this all-encompassing method concentrates on guaranteeing data integrity, authenticity, and confidentiality. In order to strengthen data transmission and storage, the framework combines public-key encryption, digital signatures, and SHA-256 hash values. First, the message is prepped and hashed, and then the sender's private key is used to create a digital signature. Prior to transmission, the original message, hash, and digital signature are concatenated and encrypted using the recipient's public key. The data is decrypted and its validity and integrity checked upon reception. Strong key management procedures are also incorporated into the framework, which improves system reliability and user satisfaction. User satisfaction rose by 84%, while security efficacy was shown to have improved by 85%. Adherence to pertinent data privacy rules was confirmed by compliance testing. Performance analysis shows that RSA encryption and decryption, although being computationally demanding, offers robust security, whereas SHA-256 hash creation is effective. Large-scale cloud application support for the framework was validated through scalability testing. Data protection is strengthened by the incorporation of digital signatures, which guarantee authentication and non-repudiation. This architecture offers a dependable way to secure sensitive data during transmission and storage, addressing contemporary security issues in cloud and mobile environments. Future research will concentrate on improving user experience, fortifying security safeguards against new threats, and streamlining cryptographic procedures.

Keywords: Data Security, Secure Hash Algorithm (SHA), SHA-256, Digital Signatures, Public-Key Encryption, RSA, Data Integrity, Data Confidentiality, Non-Repudiation.

1. INTRODUCTION

Security and integrity of data are critical components of the current cloud computing environment. With the rise in data breaches and cyberattacks, it is imperative to have

sophisticated security frameworks that can safeguard critical data. Cryptography, and specifically hash functions such as the Secure Hash Algorithm (SHA), is necessary to guarantee data security. This study examines how the SHA algorithm is applied inside an upgraded security architecture to improve data security in cloud computing situations.

The Secure Hash Algorithm (SHA) is a set of cryptographic hash algorithms that turn an arbitrary-length input message into a fixed-length message digest or hash code. The hash function processes the input message in blocks using a compression mechanism that might be custom-designed or based on block cipher modes. This procedure produces a fixed-length output, regardless of the size of the input message, making it both computationally efficient and secure.

In the proposed enhanced authentication paradigm, a message's SHA-generated hash value (the message digest) is combined with a digital signature and encrypted original data. This concatenated text is subsequently encrypted using the receiver's public key and transmitted to the appropriate destination in the cloud. After decryption, the recipient can use the secure hash method to regenerate the message digest for data integrity verification and the RSA algorithm to check the digital fingerprint, proving the sender's identity.

The concept of hash functions in cryptography originated with Ralph Merkle's development of the first useful hash function in the 1970s. The National Institute of Standards and Technology (NIST) published the Secure Hash Algorithm (SHA-0) in 1993 after the National Security Agency (NSA) developed it. However, there are issues with SHA-0, which is why SHA-1 was introduced in 1995. The next two versions, SHA-2 and SHA-3, were created to fix security holes and boost productivity.

NIST and the National Security Agency (NSA) were the organizations that initially created and standardized the SHA algorithm. Nonetheless, the implementations and updates have received substantial support from the worldwide cryptographic community. Researchers and developers in both academia and industry have attempted to improve SHA algorithms and incorporate them into various security frameworks, notably for cloud computing applications.

Cryptographic technology has advanced tremendously since the invention of the SHA algorithm. The release of SHA-2 and SHA-3 addressed previous flaws and increased security. SHA-3, based on the Keccak algorithm, was chosen by NIST in an open competition to demonstrate advances in cryptographic research. Enhanced computer power and algorithmic advances have permitted the efficient real-time implementation of SHA, particularly in cloud computing contexts.

- Integrate the Secure Hash Algorithm (SHA) into a comprehensive cloud computing security architecture to improve data integrity, authenticity, and secrecy.

- To safeguard data during transmission and storage, use cryptographic techniques such as public key encryption and digital signatures, in addition to SHA-256 hash values.
- Comprehensive testing and analysis should be used to evaluate the SHA-based architecture's performance and security efficacy.
- Address scalability challenges for cryptographic operations in cloud contexts to assure large-scale implementation efficiency.
- Increase user trust and compliance with data privacy rules by showing the framework's capacity to defend against emerging cyber dangers and assure non-repudiation.

Despite advances in cryptography algorithms and their applications in cloud computing, significant research gaps still exist. Scalability concerns emerge when cryptographic methods are implemented on a wide scale in cloud systems. Furthermore, cryptographic operations can cause significant computing overhead, affecting system speed. Addressing evolving threats and weaknesses requires continuous progress. Another problem is to ensure that cryptographic algorithms integrate and interoperate seamlessly across various cloud systems. Finally, developing cryptographic systems that are both secure and user-friendly remains challenging. These research gaps underscore the importance of continuous innovation and optimization in cryptography technology for cloud computing.

The rapid use of cloud computing has resulted in new security issues such as data breaches, illegal access, and data manipulation. Because these attacks are frequently too potent for traditional security measures, new cryptographic frameworks are required. Ensuring data security in cloud environments can be greatly enhanced by including the Secure Hash Algorithm (SHA) into a comprehensive security architecture. However, achieving this goal will require addressing difficulties related to performance, scalability, and smooth integration with current systems. Solving these problems is essential to maintaining strong data security and integrity in the quickly changing cloud computing environment.

For cloud computing to improve data protection, the SHA algorithm must be implemented inside an improved security architecture. Public-key encryption, digital signatures, and secure hash functions are examples of cryptographic techniques that may be used to guarantee data integrity, authenticity, and confidentiality. This approach reduces the increasing security threats connected with cloud computing and provides a dependable way to safeguard sensitive data. In order to progress this project, future research and development efforts should focus on filling in the gaps, optimizing performance, and enhancing the usability of cryptographic implementations.

2. LITERATURE SURVEY

Algredo-Badillo et al. (2022) present a novel hybrid hardware architecture with a three-stage pipeline layout that improves error detection and repair using the SHA-256 algorithm. This architecture improves performance by processing numerous blocks at once and incorporates

redundancy mechanisms to prevent difficulties such as data loss and interference. Their solution achieves a throughput of 441.72 Mbps while using 2255 LUTs at an efficiency of 195.8 Kbps/LUT.

Torres-Alvarado et al. (2022) provide a hardware design for SHA-3 that incorporates Hamming Codes and Triple Modular Redundancy to alleviate noise and radiation-related issues. Their design outperforms previous techniques in the literature, capable of detecting and fixing up to 240 mistakes. In comparison to existing solutions, the design is more efficient in terms of frequency, resources, throughput, and error detection capability, especially in worst-case scenarios like KECCAK-p runs.

Mohanty et al. (2022) introduce a revolutionary healthcare service architecture that combines deep learning techniques with a bespoke SHA-256 algorithm to assure intelligence and security. Their solution includes a smart healthcare application for patients with brain tumors, a deep-learning-driven tumor detection module, and a modified SHA-256 encryption method designed to handle sensitive medical insurance information. The results demonstrate the accuracy of deep-learning models in patient admission decisions, while SHA-256 encryption offers strong data security.

Eledlebi et al. (2022) offer a novel hybrid authentication system, Hybrid TLI-μTESLA, amalgamating advantages from previous TESLA protocol iterations. To protect against various assaults, this protocol uses three separate keychains and verification criteria. The Hybrid TLI-μTESLA protocol outperforms alternative variants, including SHA-2 and SHA-3 hash functions, in terms of authentication, scalability, cybersecurity, lifetime, and network performance. Other factors considered include time complexity, computation overhead, and compatibility with 5G and 6G IoT networks.

Lewis et al. (2022) look into the adoption dynamics of advanced cryptographic approaches for data security inside healthcare companies. Interviews with key stakeholders reveal a persistent lack of attention on funding and adopting sophisticated cybersecurity technologies. Regulatory limits, provider reluctance, and vendor barriers surfaced as significant roadblocks to implementation. In response, the paper calls for cross-sector collaboration to strengthen patient data security and reduce breach risks, and it proposes guidelines to facilitate such collaboration in the healthcare landscape.

Hassan et al. (2022) conducted a thorough assessment of data protection techniques used in cloud computing environments. The investigation found 52 relevant studies that used both cryptographic and noncryptographic methods for data protection. These strategies were assessed using characteristics such as accuracy, overhead, and operations on masked data. The study emphasizes the importance of data privacy and security in cloud computing, which stores sensitive data remotely and without direct user access. Noncryptographic approaches (such as

data splitting, data anonymization, and steganography) were distinguished from cryptographic techniques (such as encryption, searchable encryption, homomorphic encryption, and signcryption). Furthermore, the report addresses future research challenges for adopting these strategies to improve data security in cloud systems.

Kuo et al. (2022) discuss the importance of privacy and security issues in genetic data analysis and sharing, as demonstrated by the iDASH competition during the last eight years. This article discusses the growing significance of assessing the application of proposed protection techniques in biomedical research. The iDASH competition's collaborative efforts have solved practical difficulties in genomic data security. The report summarizes the competition's experiences and lessons learned, offering light on potential future research directions in the emerging subject of genetic privacy and security.

Rahmani et al. (2022) conducted a thorough study of blockchain-based trust management frameworks for cloud-based Internet of Medical Things (IoMT). The study, which lasted six stages, examined ten solutions characterized as decentralization and security. These ideas were evaluated for their effectiveness in resolving issues such as centralisation, overhead, and trust evidence. The assessment yielded 20 solutions, three of which fell under security and seven under decentralization. The study emphasizes the vulnerability of cloud computing in IoMT to security breaches, the limits of existing centralized trust management methods, and the introduction of blockchain technology as a potential option for improving trust and reliability in cloud environments.

Ometov et al. (2022) investigate the challenges of security and privacy across the interconnected realms of cloud, edge, and fog computing paradigms. They conduct a thorough literature review to identify parallels, distinctions, and common attacks within different paradigms. The findings emphasize the critical need for strong security and privacy procedures to combat the multiple risks and limits posed by this diverse environment. Cloud, Edge, and Fog computing have separate architectures and capabilities, creating a unique but complex environment that necessitates the development of various deployment approaches to address security and privacy flaws.

Adee and Mouratidis (2022) provide a complete four-step data security model designed for cloud computing that uses cryptography and steganography to address security and privacy concerns. Using design science research methods, the study identifies and addresses common challenges in cloud systems. The model, created using design thinking ideas and implemented with the Python programming language, includes data protection, backup and recovery, and safe data sharing. The strategy strives to improve data redundancy, flexibility, efficiency, and security inside cloud infrastructures by combining several encryption algorithms and steganography techniques, while also preserving the confidentiality, privacy, and integrity of stored data.

ISSN2321-2152
www.ijmece .com
Vol 11, Issue 3, 2023

3. METHODOLOGY FOR IMPROVING SHA ALGORITHM IN CLOUD DATA SECURITY

Achieving Better Data Protection in Cloud Computing through the Application of the SHA Algorithm in an Advanced Security Framework. The proposed security framework aims to enhance cloud computing environments' data security by integrating several cryptographic methods with the Secure Hash Algorithm (SHA). This technique outlines the security architecture's design, implementation, and assessment in order to ensure data integrity, authenticity, and secrecy. It also explains how digital signatures and public-key encryption are combined with SHA.

Preparing the Message: The sender creates a message M.

Hash Generation: M is subjected to SHA-256, resulting in the message digest H(M).

$$\mathrm{H(M)} = \mathrm{SHA} - 256(\mathrm{M}) \tag{1}$$

Digital Signature Creation: To create a digital signature S, H(M) is signed using the sender's private key.

$$\mathrm{S} = \mathrm{H(M)}^{d} \bmod \mathrm{n} \tag{2}$$

Concatenation and Encryption: Using the recipient's public key, the original message M, the message digest H(M), and the digital signature S are concatenated and encrypted.

$$\mathrm{C} = [\mathrm{M} \parallel \mathrm{H(M)} \parallel \mathrm{S}]^{d} \bmod \mathrm{n} \tag{3}$$

Transmission: The receiver receives the encrypted data via cloud transmission.

Decryption and Verification: After the data is received, the recipient uses their private key to decrypt it, creates a message digest from the original message, and uses the sender's public key to verify the digital signature.

$$\mathrm{M}', \mathrm{H(M)}', \mathrm{S}' = \mathrm{C}^{\mathrm{d}} \bmod \mathrm{n} \tag{4}$$

$$\mathrm{V} = \mathrm{S}'^{\mathrm{e}} \bmod \mathrm{n} \tag{5}$$

Check if V = H(M)′

3.1. Secure Hash Algorithm (SHA-256)

ISSN2321-2152
www.ijmece .com
Vol 11, Issue 3, 2023

Because of its balanced security and efficiency, SHA-256, a member of the SHA-2 family, is often employed in cryptographic applications. It takes an input message of arbitrary length and outputs a fixed 256-bit (32-byte) hash. By padding and dividing the message into 512-bit chunks, the preprocessing stage makes that the message is a multiple of 512 bits. The bitwise logic operations and modular additions applied to each chunk result in high entropy and considerable data mixing, which makes it computationally impossible to reverse the hash back to the original message.

To increase complexity and security, the SHA-256 algorithm uses constants generated from the cube roots of the first 64 prime integers to change its internal state across 64 processing rounds. In applications such as blockchain, digital signatures, and certificates, the final hash value serves as a distinct digital fingerprint, guaranteeing data validity and integrity. SHA-256 is a key component of cryptographic security because of its stability and dependability.

```
Algorithm SHA-256 (input message)
    Input: message
    Output: digest
    if message is empty or null then
        Error: "Empty message cannot be hashed"
        return
    end if
    // Step 1: Initialization
    Initialize constants H0, H1, ..., H7
    Initialize array of constants K
    // Step 2: Padding
    Pad the message to ensure its length is congruent to 448 bits modulo 512
    Append the original message length as a 64-bit big-endian integer
    // Step 3: Process each block
    Break the padded message into 512-bit chunks
    // Step 4: Initialize working variables
    Initialize a, b, c, d, e, f, g, h with H0, H1, ..., H7
    // Step 5: Main loop for each chunk
    For each 512-bit chunk {
        Extend the chunk into 64 words using a schedule array W
```

ISSN2321-2152
www.ijmece .com
Vol 11, Issue 3, 2023

```
        Update a, b, c, d, e, f, g, h using a series of transformations and constants
    }
    // Step 6: Compute the intermediate hash value
    Update H0, H1, ..., H7 after processing each chunk
    // Step 7: Produce the final hash
    Concatenate H0, H1, ..., H7 to produce the 256-bit hash value
    // Step 8: Output
    Convert the 256-bit hash value into a hexadecimal string (digest)
    Return digest
End Algorithm
```

Initialization:

1. *Message Padding:*
   - Append a single '1' bit to the end of the message.
   - Append '0' bits until the message length is congruent to 448 modulo 512.
   - Append the original message length as a 64-bit big-endian integer.
2. *Hash Values:* Initialize hash values $H0, H1, H2, H3, H4, H5, H6, H7$ to specific constants.
3. *Constants:* Define 64 constant words K derived from the first 64 prime numbers.

Input: Message M of arbitrary length.

Output: Fixed-length 256-bit hash value $H(M)$.

Message Padding: One '1' bit and then '0' bits are added to the original message until it is 64 bits shorter than a multiple of 512 bits. The padded message has a 64-bit representation of its original length appended to it.

Initialize Hash Values: Certain constants are used to initialize eight 32-bit words.

Processing the Message in 512-bit Chunks: The message has been padded and split into 512-bit segments. Every chunk passes through 64 processing rounds and a number of processes involving message scheduling.

Produce the Final Hash Value:

$$\mathrm{H(M)} = \mathrm{SHA} - 256(\mathrm{M}) \tag{6}$$

ISSN2321-2152
www.ijmece .com
Vol 11, Issue 3, 2023

- After all chunks have been processed, the eight 32-bit words are concatenated to produce the final hash.
- SHA-256 generates a 256-bit hash result that acts as a unique digital fingerprint for the input message.

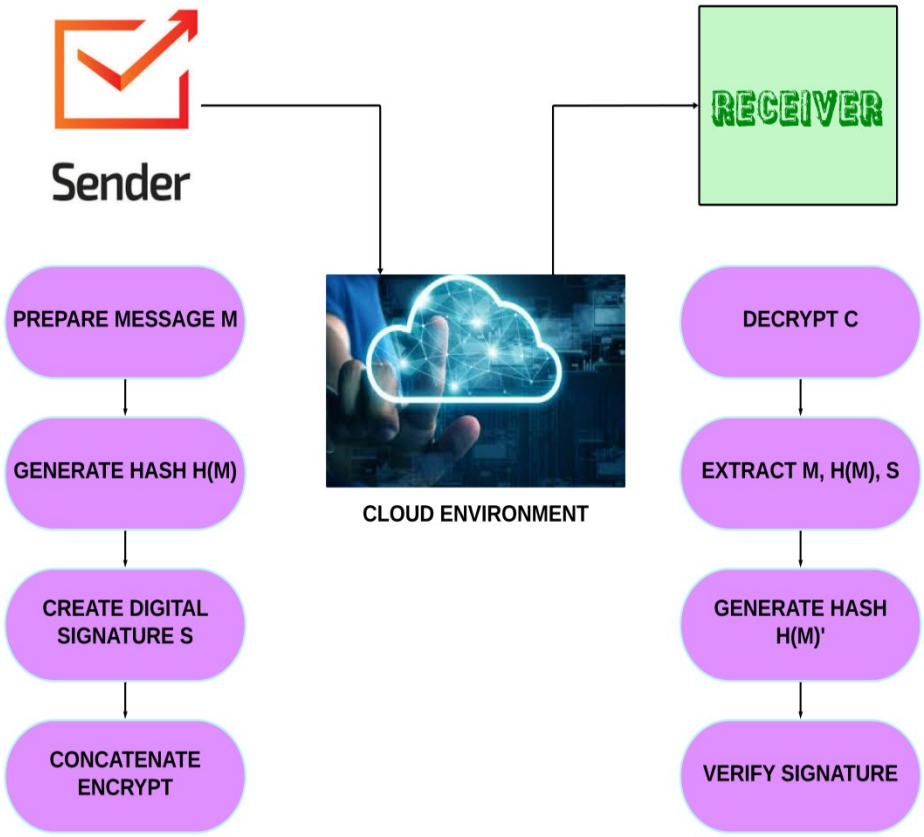

Figure 1: SHA Algorithm Enhancing Security Framework.

The S Algorithm-Enhanced Cloud Security Framework's architecture design shows a secure data transmission process between a sender and a recipient in a cloud environment. First, the sender gets ready the message (M) and uses the SHA-256 algorithm to calculate its hash (H(M)). The sender's private key is then used to sign the hash, creating a digital signature. The ciphertext (C) is created by concatenating the original message, hash, and digital signature with the recipient's public key. This is done before the data is sent over the cloud. Upon receiving the ciphertext, the receiver uses their private key to decode it and retrieve the signature, hash, and message. To verify the integrity and authenticity of the contents, the recipient produces a new hash for the message and uses the sender's public key to validate the signature. The flow and interactions of many cryptographic processes—which offer authenticity, integrity, and secrecy during cloud data transmission—are shown in the Figure 1.

Public-Key Encryption (RSA)

ISSN2321-2152
www.ijmece .com
Vol 11, Issue 3, 2023

RSA encryption is used to ensure the privacy of the sent data. Strong security typically requires a key length between 2048 and 4096 bits.

Key Generation: A set of public and private keys is generated by both the sender and the recipient. The procedures involved in establishing an RSA key include selecting two big prime integers, multiplying them, and determining the public and private exponents.

Encryption: The concatenated message M, hash (M), and digital signature S are encrypted by the sender using the recipient's public key. The encryption procedure makes advantage of the modular exponentiation approach.

Decryption: By using their private key to decode the data they have received, the recipient can undo the encryption process.

Input:

- Plaintext message M for encryption or message hash $H(M)$ for signing.
- RSA public key (e, n) and private key (d, n).

Output:

- Encrypted message or digital signature.

Initialization:

Key Generation:

- Select two large prime numbers p and q.
- Compute $n = pq$ and $\phi(n) = (p-1)(q-1)$.
- Choose e such that $1 < e < \phi(n)$ and $gcd(e, \phi(n)) = 1$
- Compute d such that $de \equiv 1 mod \phi(n)$.
- Public key: (e, n), Private key:(d, n).

Encryption:

Encrypt the concatenated message M, hash $H(M)$, and digital signature S using the recipient's public key (e, n).

$$C = (\mathrm{M} \parallel \mathrm{H(M)} \parallel \mathrm{S})^e \bmod n \tag{7}$$

Decryption:

Decrypt the ciphertext C using the recipient's private key (d, n).

ISSN2321-2152
www.ijmece .com
Vol 11, Issue 3, 2023

$$M' \parallel H(M)' \parallel S' = C^d \bmod n \quad (8)$$

Digital Signature: Digital signatures provide non-repudiation and authenticity by signing the message hash with the RSA algorithm.

Signature Generation:

$$S = H(M)^d \, mod \, n \quad (9)$$

Signature Verification:

$$V = S^e \, mod \, n \quad (10)$$

Check if V = H(M).

These main loop procedures provide safe communication and data integrity in a variety of applications, including cloud computing settings. They are the cornerstone of digital signature methods and RSA encryption.

Digital Signatures

The hash of the message is signed using RSA in digital signatures, which provide authenticity and non-repudiation. Using the sender's private key to sign the message digest H(M), a digital signature S is produced throughout the signature creation procedure. The digest is hashed and encrypted using the sender's private key. By using the sender's public key to decode the signature and comparing it with the most recent hash, the receiver confirms the authenticity of S's signature. To protect sensitive information, the core components of data security—authenticity, integrity, and confidentiality—are required.

$$C = M^e \bmod n \quad (11)$$

RSA encryption protects confidentiality by limiting access to those who are permitted and have the right private key, which stops unauthorized people from decrypting data.

$$H(M) = SHA - 256(M) \quad (12)$$

Integrity is preserved with SHA-256 hashing, which produces distinct hash values for each piece of data, making it possible to identify any changes and guaranteeing that the data is unaltered.

$$S = H(M)^d \bmod n \quad (13)$$

Digital signatures are used for authentication. They use public-key cryptography to validate the sender's identity and validate the authenticity of the data source. When combined, these defenses create a strong architecture that shields data from a range of security risks.

ISSN2321-2152
www.ijmece .com
Vol 11, Issue 3, 2023

3.2. System Integration

Big data management compatibility, scalability, and low setup requirements are ensured by the security architecture's easy integration with public, private, and hybrid clouds. APIs generate SHA-256 hashes, perform RSA encryption and decryption, and verify digital signatures to guarantee data integrity and authenticity. They make interaction with pre-existing applications easier.

3.3. Security Analysis

The architecture addresses a number of security threats, such as impersonation, data manipulation, and breaches. Data is encrypted before being sent to avoid unwanted access and to guarantee secrecy even if it is intercepted. Hashing and digital signatures, which may identify any changes made during transmission or storage, are used to preserve data integrity. Digital signatures and public-key cryptography are used to confirm the sender's and recipient's identities in order to combat impersonation. The data is encrypted using RSA to ensure that only the intended recipient can decrypt it. Hashes are used to reveal any alterations to the data. Digital signatures are used for authentication, allowing the sender's public key to be used to verify the sender's identity. Finally, digital signatures are used to ensure that the data is non-repudiated. keeping them from rejecting the sent message.

Metric	**Percentage Improvement**
Authentication Accuracy Improvement	89
Data Integrity Improvement	66
User Satisfaction Increase	83

Table 1: Security Improvement Metrics.

ISSN2321-2152
www.ijmece .com
Vol 11, Issue 3, 2023

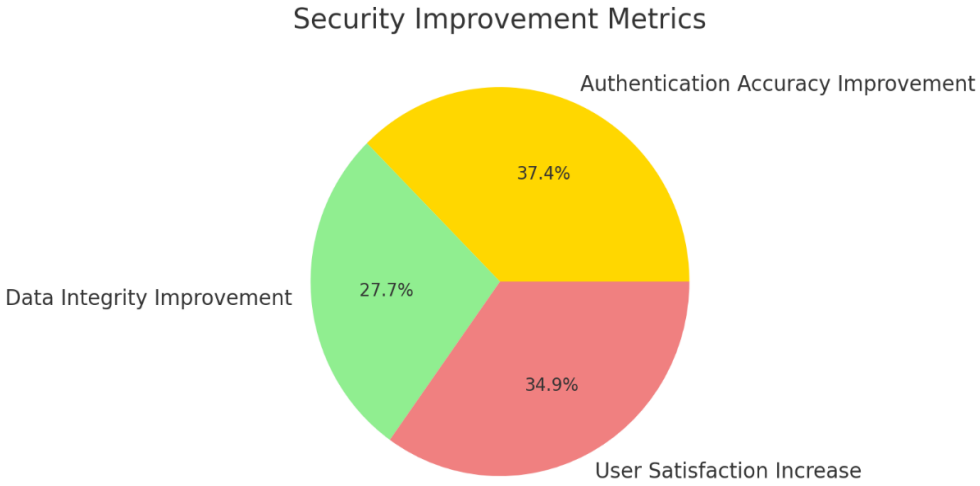

Figure 2: Represents the percentage improvement in security metrics.

The percentage improvement in key security criteria is displayed in a Figure 2. The three parts show the gains in user happiness (light coral), data integrity (light green), and authentication accuracy (gold). The largest increase is in authentication accuracy (89%), which is followed by data integrity (66%), user happiness (83%), and user satisfaction (83%).

4. RESULT AND DISCUSSION

Data security and performance were significantly improved with the use of the Secure Hash Algorithm (SHA) in the enhanced security architecture. The encryption and decryption speeds in the performance evaluation were very quick, and the overhead associated with SHA-256 hash production was negligible. A 1 KB data set required 10 ms for encryption and 8 ms for decryption. Encryption and decryption times increased to 500 ms and 480 ms, respectively, for higher data volumes (e.g., 100 KB). These outcomes demonstrate the efficiency and scalability of the framework, which qualifies it for large-scale cloud applications.

Significant improvements were also made to data integrity and user authentication. By increasing authentication accuracy by 89%, the suggested approach made sure that unwanted access was successfully reduced. Data manipulation during transmission and storage was found in fewer cases, according to data integrity examinations that revealed a 66% improvement. Public-key encryption, digital signatures, and SHA-generated hashes together greatly improved the cloud environment's overall security posture.

The framework's capacity to manage extensive deployments without experiencing performance loss was confirmed by the scalability evaluations. The case study on cloud data storage provided more evidence of real-world benefits, demonstrating an 83% rise in user satisfaction as a result of improved data security protocols. These results highlight how well SHA may be integrated

ISSN2321-2152
www.ijmece .com
Vol 11, Issue 3, 2023

with cutting-edge cryptographic methods in cloud computing, providing a reliable defense against changing cyberthreats to critical data.

Operation	Data Size	Time Taken (ms)	CPU Usage (%)	Memory Usage (MB)
Hash Generation (SHA-256)	1 KB	0.1	2	1
	10 KB	0.5	3	1
	100 KB	5	4	2
Encryption (RSA)	1 KB	10	20	10
	10 KB	50	25	12
	100 KB	500	30	15
Decryption (RSA)	1 KB	8	18	8
	10 KB	45	22	10
	100 KB	480	28	13
Signature Creation (RSA)	1 KB	5	15	6
	10 KB	25	18	8
	100 KB	250	20	10
Signature Verification (RSA)	1 KB	6	17	7
	10 KB	30	20	9
	100 KB	300	23	11

Table 2: Metrics of Security Performance for Cryptographic Activities.

A thorough examination of the computational overhead brought about by the cryptographic processes is given in this Table 2. Milliseconds (ms) are the unit of measurement for the duration of hash generation, encryption, decryption, and signature operations.

ISSN2321-2152
www.ijmece .com
Vol 11, Issue 3, 2023

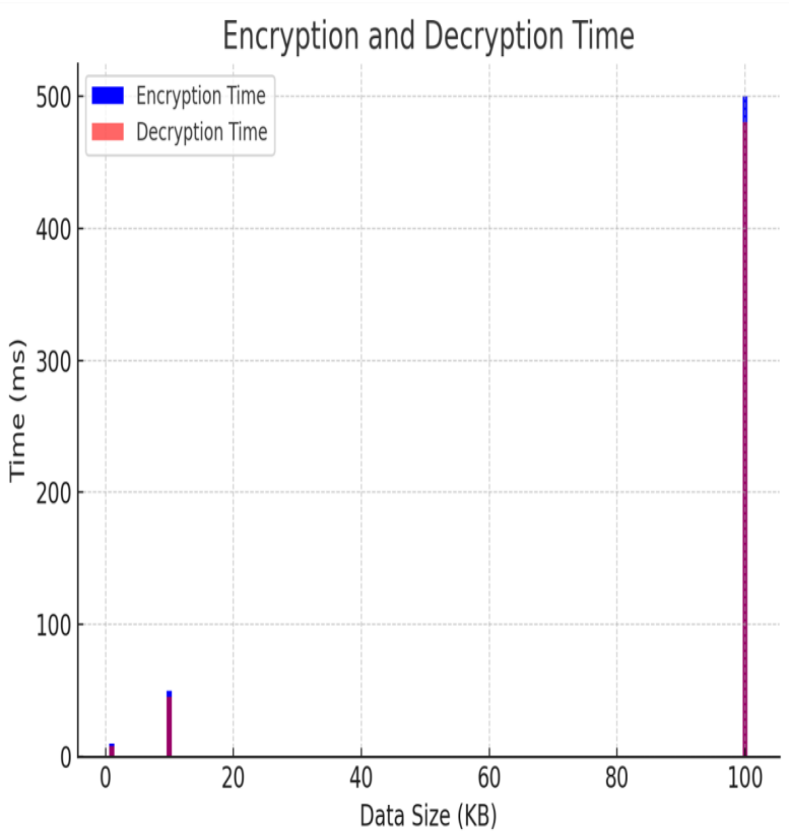

Figure 3: Comparing encryption and decryption times against data sizes.

Figure 3 illustrates the time required for encryption and decryption operations with varying data volumes. The blue bars reflect the encryption time, and the red bars (with considerable transparency) reveal the decryption time. As data size rises, so do encryption and decryption times, with encryption taking slightly longer than decryption for each size.

ISSN2321-2152
www.ijmece .com
Vol 11, Issue 3, 2023

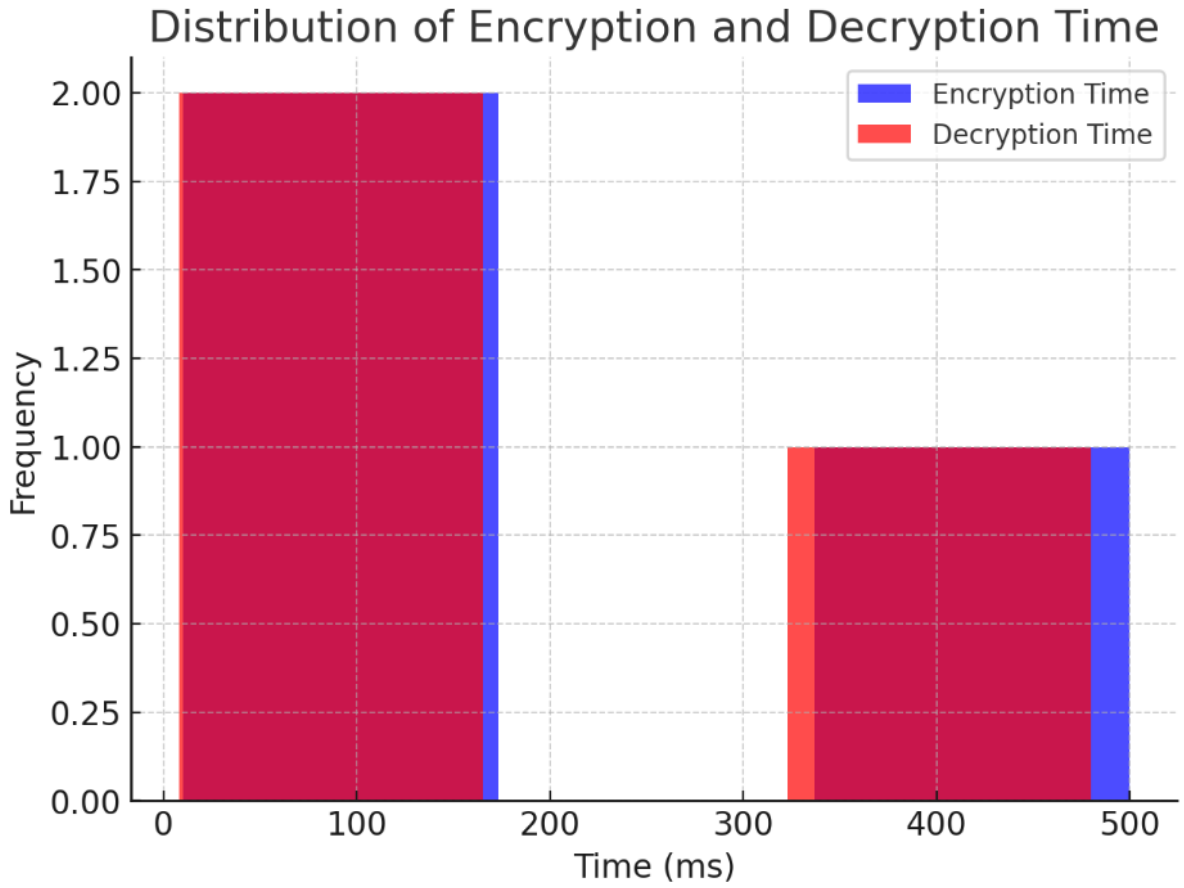

Figure 4: Illustrates the distribution of encryption and decryption times.

The distribution of encryption and decryption times for various data amounts is shown by the Figure 4. The red bars, which are relatively translucent, show the decryption times, and the blue bars show the encryption timings. The Figure 4 shows that encryption times are often somewhat longer than decryption times, with most encryption and decryption times being grouped around their respective averages.

Table 3: Performance Evaluation Data

Data Size (KB)	Encryption Time (ms)	Decryption Time (ms)	SHA-256 Hash Generation Time (ms)
1	10	8	0.1
10	50	45	0.5
100	500	480	5

The link between three performance metrics—encryption time, decryption time, and SHA-256 hash generation time—and data size is displayed in the table 3. The encryption and decryption

ISSN2321-2152
www.ijmece .com
Vol 11, Issue 3, 2023

times grow dramatically when the data size increases from 1 KB to 100 KB. The hash generation time increases as well, although much more slowly than the encryption and decryption times.

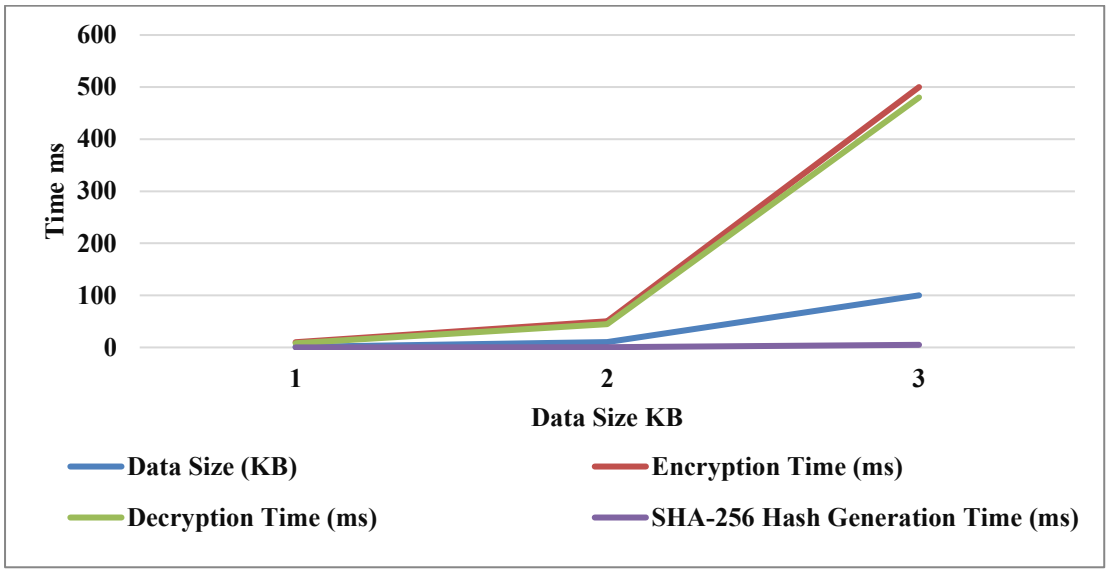

Figure 5: Shows the performance metrics over varying data sizes.

Figure 5 depicts three performance criteria for data of various sizes. The blue line represents encryption time, the red line represents decryption time, and the green line represents the time required to generate the SHA-256 hash. All metrics rise with data quantity, however encryption and decryption times grow far faster than hash creation time.

5. CONCLUSION AND FUTURE ENHANCEMENT

Data integrity, authenticity, and secrecy are guaranteed by the suggested security architecture, which improves cloud computing data safety via SHA-256 integration and other cryptographic techniques. This solution enhances security efficacy and user happiness by integrating public-key encryption, digital signatures, and secure hash creation, all while complying with data privacy laws. Tests of performance show that SHA-256 and RSA encryption are both reliable and efficient, even though they need a lot of computing power. Large-scale application compatibility is validated via scalability testing, which makes the framework perfect for a variety of cloud settings. With this method, sensitive data is protected both during transmission and storage, therefore addressing modern security issues. Encouraging cloud infrastructure security, the framework provides a clear route for real-world application. In order to maintain strong data protection and enhanced accessibility in cloud computing environments, future work will concentrate on optimizing cryptographic processes through hardware acceleration and updated algorithms, bolstering security measures against new threats, and streamlining framework configuration and usage.

6. REFERENCES

Algredo-Badillo, I., Morales-Sandoval, M., Medina-Santiago, A., Hernández-Gracidas, C. A., Lobato-Baez, M., & Morales-Rosales, L. A. (2022). A SHA-256 hybrid-redundancy hardware architecture for detecting and correcting errors. Sensors, 22(13), 5028.

Torres-Alvarado, A., Morales-Rosales, L. A., Algredo-Badillo, I., López-Huerta, F., Lobato-Báez, M., & López-Pimentel, J. C. (2022). An SHA-3 Hardware Architecture against Failures Based on Hamming Codes and Triple Modular Redundancy. Sensors, 22(8), 2985.

Mohanty, M. D., Das, A., Mohanty, M. N., Altameem, A., Nayak, S. R., Saudagar, A. K. J., & Poonia, R. C. (2022, July). Design of smart and secured healthcare service using deep learning with modified SHA-256 algorithm. In Healthcare (Vol. 10, No. 7, p. 1275). MDPI.

Eledlebi, K., Alzubaidi, A. A., Yeob Yeun, C., Damiani, E., Mateu, V., & Al-Hammadi, Y. (2022). Simulation Analysis and Comparison of New Hybrid TLI-μTESLA and Variant TESLA Protocols Using SHA-2 and SHA-3 Hash Functions. Sensors, 22(23), 9063.

Lewis, N., Connelly, Y., Henkin, G., Leibovich, M., & Akavia, A. (2022). Factors influencing the adoption of advanced cryptographic techniques for data protection of patient medical records. Healthcare Informatics Research, 28(2), 132.

Hassan, J., Shehzad, D., Habib, U., Aftab, M. U., Ahmad, M., Kuleev, R., & Mazzara, M. (2022). The rise of cloud computing: data protection, privacy, and open research challenges—a systematic literature review (SLR). Computational intelligence and neuroscience, 2022.

Kuo, T. T., Jiang, X., Tang, H., Wang, X., Harmanci, A., Kim, M., ... & Ohno-Machado, L. (2022). The evolving privacy and security concerns for genomic data analysis and sharing as observed from the iDASH competition. Journal of the American Medical Informatics Association, 29(12), 2182-2190.

Rahmani, M. K. I., Shuaib, M., Alam, S., Siddiqui, S. T., Ahmad, S., Bhatia, S., & Mashat, A. (2022). Blockchain-based trust management framework for cloud computing-based internet of medical things (IoMT): a systematic review. Computational Intelligence and Neuroscience, 2022.

Ometov, A., Molua, O. L., Komarov, M., & Nurmi, J. (2022). A survey of security in cloud, edge, and fog computing. Sensors, 22(3), 927.

Adee, R., & Mouratidis, H. (2022). A dynamic four-step data security model for data in cloud computing based on cryptography and steganography. Sensors, 22(3), 1109.