information

Article

Secure Sensitive Data Sharing Using RSA and ElGamal Cryptographic Algorithms with Hash Functions

Emmanuel A. Adeniyi [1], Peace Busola Falola [1], Mashael S. Maashi [2], Mohammed Aljebreen [3] and Salil Bharany [4,*]

[1] Department of Computer Sciences, Precious Cornerstone University, Ibadan 200223, Nigeria
[2] Software Engineering Department, College of Computer and Information Sciences, King Saud University, Riyadh 11451, Saudi Arabia
[3] Department of Computer Science, Community College, King Saud University, P.O. Box 28095, Riyadh 11437, Saudi Arabia
[4] Department of Computer Engineering & Technology, Guru Nanak Dev University, Punjab 143005, India
* Correspondence: salil.bharany@gmail.com

Abstract: With the explosion of connected devices linked to one another, the amount of transmitted data grows day by day, posing new problems in terms of information security, such as unauthorized access to users' credentials and sensitive information. Therefore, this study employed RSA and ElGamal cryptographic algorithms with the application of SHA-256 for digital signature formulation to enhance security and validate the sharing of sensitive information. Security is increasingly becoming a complex task to achieve. The goal of this study is to be able to authenticate shared data with the application of the SHA-256 function to the cryptographic algorithms. The methodology employed involved the use of C# programming language for the implementation of the RSA and ElGamal cryptographic algorithms using the SHA-256 hash function for digital signature. The experimental result shows that the RSA algorithm performs better than the ElGamal during the encryption and signature verification processes, while ElGamal performs better than RSA during the decryption and signature generation process.

Keywords: data sharing; cryptographic algorithm; RSA and ElGamal; communication; digital signature

Citation: Adeniyi, E.A.; Falola, P.B.; Maashi, M.S.; Aljebreen, M.; Bharany, S. Secure Sensitive Data Sharing Using RSA and ElGamal Cryptographic Algorithms with Hash Functions. *Information* **2022**, *13*, 442. https://doi.org/10.3390/info13100442

Academic Editor: Maanak Gupta

Received: 24 July 2022
Accepted: 16 September 2022
Published: 20 September 2022

1. Introduction

With the rapid development of information digitization, security and privacy concerns are among the most pressing problems confronting the emerging smart grid [1]. These issues include, among many others, a lack of shared authentication across communicating parties, the possibility of multiple cyber-attacks, illegitimate access to services, and the disclosure of computer and network confidential information to the interacting party. Before granting any individual access to a network and its associated services, it is necessary to validate the individual, which may be a computer or a person, and then validate the permission and control policies based on the individual's identification. A digital signature validates the user's identity, whereas authorization validates whether the person has the necessary authority to access the shared resource [2].

Encryption is always required for data transmission and communication [3]. Information security utilizing encryption and decryption is crucial since data transmission and reception are susceptible to outside assault. To increase security, data are transformed into a coded message (encryption) and then recovered into data (decryption) [4]. To offer secure transmission of data and information, several cryptographic algorithms have been proposed, which can be classified as symmetric and asymmetric cryptographic techniques [5]. Figure 1 displays the process the plaintext passed through before turning into ciphertext and then back into plaintext. The plaintext passes through the encryption process to pro-

duce a ciphertext, while the cipher text passes through the decryption process to produce the plaintext.

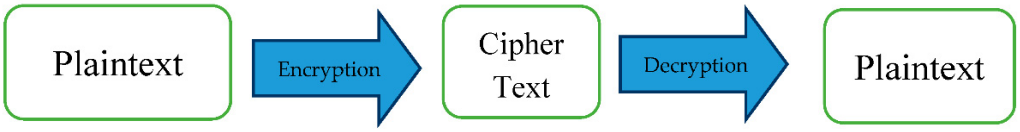

Figure 1. A basic illustration of the encryption and decryption process.

A digital signature is a message's authenticity and legality generated via a cryptographic process (a contrast to a digital certificate), device, or electronic record [6]. A digital signature is a digital equivalent to a signed signature or engraved seal, but it has much more essential protection. It is meant to address the issue of interference and spoofing in communications networks. Digital signatures can provide additional guarantees about the source, presence, and position of an electronic document, activity, or communication, as well as acknowledge the signer's permission. Digital signatures are a segment of digital signature technologies that sign documents using keys and encryption algorithms [7]. The digitally signed algorithm scheme is one of the most well-known digital signature systems, e.g., the RSA digital signing scheme, the ElGamal digital signing scheme, and many others based on public key cryptosystems. This study, therefore, aims at implementing the RSA and ElGamal cryptographic algorithms using the hash function to ensure data security with integrity. In addition, this study attempts to establish the data integrity of RSA and ElGamal cryptographic procedures that use the creation and validation of signatures. This study will be beneficial for controlling cryptographic operations using the sender's and receiver's private and public keys.

This study consists of four sections. We first describe the literature reviews. The materials and methods used are described in Section 2. Sections 3 and 4 present the results and discussion. Section 5 concludes the study.

Review of Literature

Zhang et al. [8] demonstrated an improved scheme using a modern main agreement protocol over the Chang and Chang [9] system, which does not use a one-way hashing algorithm or replication padding. Digital signature systems dependent on public-key cryptosystems are susceptible to existential identity fraud attacks, which can be avoided by using a one-way hash feature. The authors of this paper suggest a fraudulent assault on the digital signature system proposed by Chang and Chang in 2004.

Burr [10] studied the possibilities of cryptographic hash functions in his article. He emphasized that the cryptography tools include the SHA-1 and SHA-2 functions. Apart from Dobbertin's work after the MD5 near-break in 1996, hash function assessment saw little development until the middle of 2004. Since then, some academics have focused on almost all of the original hash functions, including SHA-1. These attacks shook cryptographers' long-term faith in almost all hash functions because SHA-2 functions are, even until now, related to the earlier broken functions built. Although cryptologists have discovered a lot over the past few years concerning hash functions and how to attack them, cryptanalysts widely concluded that realistic threats to SHA-2 hash functions remain impossible in the next decades.

Acharya et al. [11], in their paper, discussed and analyzed some well-known cryptographic algorithms to show the fundamental variations between current data encryption methods. Despite the computational philosophy behind such an algorithm, the effective techniques are well known and well documented since they have been thoroughly reviewed and analyzed. They noted that the power of cryptography is in the key selection; longer keys resist assault more easily than shorter keys. Nobody can guarantee complete defense.

Saleh and Meinel's [12] HPISecure is a suggested HTTP client that is in charge of encrypting or decrypting information. It must be mounted on the client's computer. It

also transmits HTTP request/response items and encrypts data before sending it to the network or decrypts the information sent back from the network. They were in favor of using public-key encryption. Besides that, to make it harder for unauthorized users to use a collection of secret keys, each record can be encoded with a different key. On the other hand, they recommend using a coordinator for key management, which may be a third-party cloud service or a USB that stores the credentials and associated material [13,14]. Conversely, one of the drawbacks of this research is that the client must install the program on each computer where it will be used. They also restricted information sharing and coordination among groups of individuals.

Hwang et al. [15] suggested a cloud infrastructure business strategy built on the principle of having two independent service providers, one for cryptography and another for processing. The database system retains encoded user information and keys while the cryptographic service model requires ciphering activities and then erases the information. The key idea behind their strategy is to divide the procedure among multiple service providers to reduce the operating cost of revealing user information. There is no certainty, though, that the cryptographic service system fully erases the information and does not preserve or use it. Moreover, Chandra et al.'s [16] Silverline is a technique that has been implemented to facilitate improved data protection in the cloud. Unlike the preceding methods, these authors concentrated on data and computation-intensive software. Their primary aim was to encrypt as much useful information as possible without interfering with the application's features. As a result, although the cloud program cannot compute any data it cannot control in plaintext, they proposed decoding only the information that is not used in the computation.

Haque et al.'s [17] study provided a comprehensive performance analysis in which common symmetrical and asymmetrical key encryption methods were compared to choose the one that worked best for handheld phones and resource-constrained environments. Various factors, including key size, data blocks, data type, and CPU time, were used to compare the AES, RC4, Blowfish, CAST, 3DES, Twofish, DSA, and ElGamal algorithms. The experiments show the utility of several cryptographic algorithms for use in practical applications in which quick execution and little memory usage are essential.

Dijesh et al. [18] worked on an asymmetric key scheme for enhancing e-commerce protection. The study explains asymmetrical techniques to make use of electronic commerce payments and other supportive cryptographic techniques that are crucial to the operation of electronic business. The paper also outlines the main security issues with online shopping. Based on security, the RSA encryption algorithm and the Fernet cipher encryption algorithm were proposed as multilayer encryption algorithms. A comprehensive and intricate technique for encryption was built using a multilayer encryption method. The study concluded that the proposed multilayer encryption discussed was the main method for making online transactions secure. A more advanced encryption technique can quickly and efficiently reduce fraudulent operations.

Hamza and Al-Alak [19] analyzed several asymmetric key generators in wireless sensor networks. Although the asymmetric key encryption algorithm provides a higher level of security than symmetric key encryption, it requires more sensors than symmetric key encryption. The twelve algorithm trials' chain keys were generated using the KCMA method (ECC, RSA, ElGamal). These chains were then combined using the SHA-2 and XOR hashing algorithms. The diehard test was used in all tests to assess the secret key's unpredictability and demonstrate its increased security. When compared to XOR, SHA-2 performed the best. Table 1 gives a summary of all the literature reviewed with the results they achieved.

Table 1. Summary of literature.

S/N	Author	Methods	Result	Limitations
1	Zhang et al. [8]	Digital signature algorithm	The authors proposed DSA to mitigate fraudulent assault.	Only digital signature was used.
2	Burr [10]	SHA-1 and SHA-2	The study concluded that realistic threats to SHA-2 hash functions remain impossible in the next decades.	The study only protects the integrity of data but does not properly secure the data.
3	Acharya et al. [11]	Analyzed some well-known cryptographic algorithms	The study noted that the power of cryptography is in the key selection.	The study lacks a proper way to ensure complete data security.
4	Saleh and Meinel [12]	HPISecure was used to secure the HTTP client.	The study recommends using a coordinator for key management.	The drawback of this research is that the client must install the program on each computer where it will be used.
5	Haque et al. [17]	AES, RC4, Blowfish, CAST, 3DES, Twofish, DSA, and ElGamal	The effectiveness of an algorithm depends on execution time and lower memory usage requirement.	The study only compares the computational time of the selected algorithms.
6	Dijesh et al. [18]	Multilayer encryption algorithm RSA and Fernet cipher encryption algorithms	The method used to decrease fraudulent activities easily and effectively over the internet.	The study recommends a more efficient algorithm to secure online transactions.
7	Hamza and Al-Alak [19]	KCMA for key generation (ECC, RSA, ElGamal) with SHA-1 and SHA-2	SHA-2 was the best as compared with XOR.	The study only compares the key generation of encryption algorithms with the hashing function.

From the summary of pieces of literature showing various limitations of the reviewed work, it is expedient to proffer a solution that will enhance the security of data as well as increase the integrity of the message. Therefore, this study embraced the use of RSA and ElGamal algorithms with SHA-256 to enhance the integrity of data.

2. Materials and Methods

This study uses asymmetric cryptography (the RSA and ElGamal) and the SHA-256 hash function for both the encryption and sharing of sensitive information and using a digitally signed system; security features including message authentication, data integrity, non-repudiation, and confidentiality are also provided. For any specified ciphertext regardless of length, the SHA-256 hash technique is employed to produce a fixed, singular value (referred to as a message digest). It is this message digest that is subsequently encrypted/signed to produce the signatures for the message. The system flow diagram of the system is displayed in Figure 2, which displays the flow of information from user A to user B.

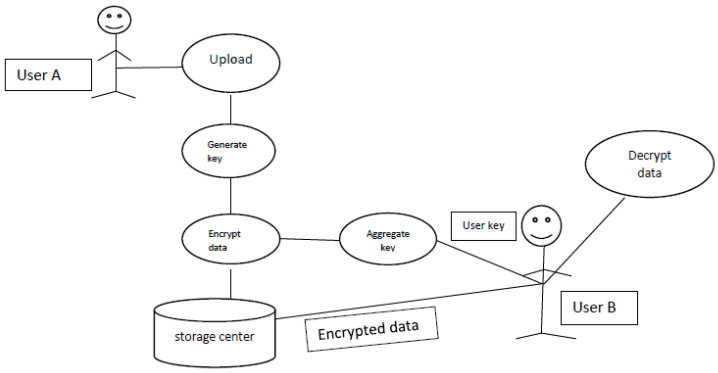

Figure 2. System flow diagram.

The system is developed in such a way that the recipient also recomputes the digital signature to ensure its integrity after the sender produces it using SHA-256. The authenticity of the content is determined if the two signatures from the originator and the recipient are equal; if not, the data have been changed during transit or transmission.

2.1. The RSA Algorithm

The RSA's reliability is dependent on how challenging it is to factor huge prime numbers. The encryption and decryption stages of the RSA algorithm involve modular exponentiation.

2.1.1. Key Generation

i Randomly choose two huge, unique primes p and q.
ii Compute the modulus n, $n = p * q$ and the phi function $Ø(n) = (p - 1) * (q - 1)$.
iii Choose a random integer e, such that $0 < e < Ø(n)$.
iv Compute $d = e^{-1} \bmod Ø(n)$.
v The private key is given as (d, n) and the public key as (e, n).

2.1.2. Encryption and Decryption

Given the message to be M and the cipher C,

i Encryption is carried out with the aid of the public key (e, n).
ii $C = M^e \bmod n$.
iii The secret key is used for decryption (d, n).
iv $M = C^d \bmod n$.

2.2. Signing and Verification

The communicator must carry out the following to create the signatures for document M:

i Calculate the hash h = H(M) of the message M.
ii The signature S is given as $S = H^d \bmod n$.

To verify the signature,

i Calculate the hash H of the message M.
ii Compute $H' = S^e \bmod n$.
iii If $H == H'$, then the signature is valid.

Any modification to the document would provide a changed hash code, which would not correlate with the signature.

2.3. The ElGamal Algorithm

Dr. Taher Elgamal developed the ElGamal algorithm, which is a public-key method of encryption. It is based on the one-way feature, which ensures that encryption schemes are performed separately [20–24].

2.3.1. Key Generation

i Generate a large random prime number (p).
ii Choose a generator number (a).
iii Choose an integer (x) less than $(p - 2)$, as the secret number.
iv Compute (d), where $d = a^x \bmod p$.
v The private key is given as (x) and the public key as (p, a, d).

2.3.2. Encryption and Decryption

Represent the plaintext as an integer m, where $0 < m < p - 1$.
Encryption is achieved using the public key (p, a, d).

i Choose an integer k such that $1 < k < p - 2$.
ii Compute y, $y = a^k \bmod p$.

iii Compute z, $z = (d^k * m) \bmod p$.
iv The ciphertext is given as C = (y, z).

Decryption is achieved using the private key (x).

i The receiver obtains the ciphertext C = (y, z).
ii Next, r is computed as follows: $r = y^{p-1-x} \bmod p$.

The plain text is recovered as follows: $m = (r * z) \bmod p$.

2.3.3. Signature Generation

This is accomplished first by generating the hash m of the message M, with the private key given as (x).

The signer should then perform the following:

i Choose a random integer K with $1 \leq K \leq (p-1)$ and $\gcd(K, p-1) = 1$.
ii Compute the temporary key: $h = a^k \bmod p$.
iii Compute $K-1$ the inverse of K mod $(p-1)$.
iv Compute the value $s = K^{-1}(m - xh) \bmod (p-1)$.
v The signature is (h, s).

Any other user who receives the message M and signature (h, s) can carry out verification using the public key (p, a, d) by computing the following:

i The hash m for the message M;
ii $V_1 = a^m \bmod p$;
iii $V_2 = d^h h^S \bmod p$;
]iv] The signature is valid if $V_1 = = V_2$.

2.4. The SHA-256 Hash Function

SHA-256 (secure hash algorithm, FIPS 182-2) is a cryptographic hash function that processes input blocks of 512 bits with a digest length of 256 bits. It is a keyless hash function. The SHA-256 follows the same model as SHA-1 and begins by defining several constants [25–29]. Several operating systems frequently use hash methods to secure passwords. Figure 3 illustrates how hashing assesses a file's authenticity. Figure 4 shows the hashing algorithms involving rounds of the hash function such as a block cipher [30–33].

Figure 3. A basic illustration of the hashing process.

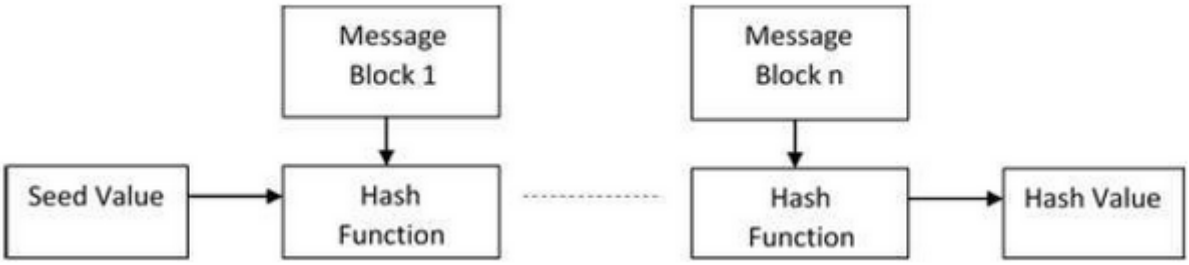

Figure 4. Schematic illustration of hashing algorithms.

The SHA-256 Algorithm

The algorithm for the SHA-256 hash function is given below:

1. Append a single bit, whose value is set to 1, to the input x.

2. Compute the smallest r such that (b + r) mod 512 = 448. Append r-1 bits, whose values are set to 0, to the result of step 1.
3. Compute the 64-bit value b mod 2^64 and append this value to the result of step 2.
4. This yields a string of length that must be a multiple, m, of 512 bits and, thus, may be represented as 16*m 32-bit blocks.

3. Results

The proposed secure sensitive data sharing system possesses the following features:

1. Encryption of files using RSA and ElGamal algorithms;
2. Signature generation and verification for text files;
3. Decryption of information using the RSA and ElGamal algorithms;
4. Generation of message digest for information/data;
5. GUI interface for easy interaction with the system;
6. Auto-generation of private and public keys for encryption, signing, and decryption;
7. Provision of interface for the selection of files or documents to be signed or encrypted. See Figure 5.

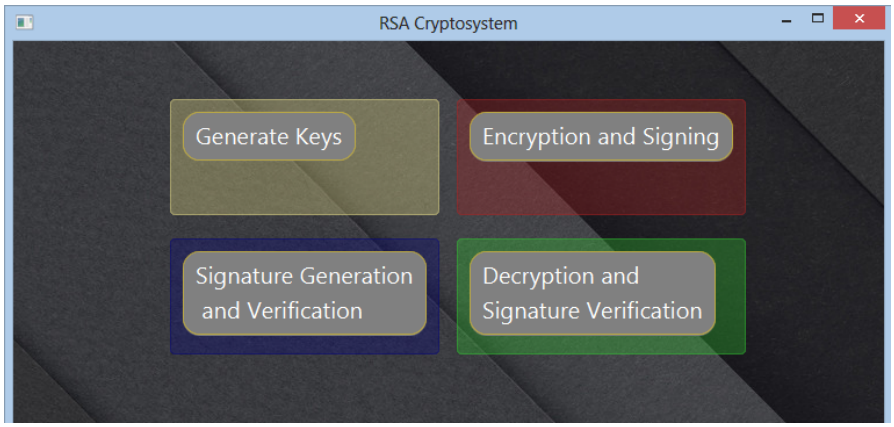

Figure 5. Application homepage.

Figure 5 displays the interface that provides the user with various functionalities to encrypt and sign, decrypt and verify, or generate or verify the signature of a file after generating or loading the appropriate keys needed. See Figure 6.

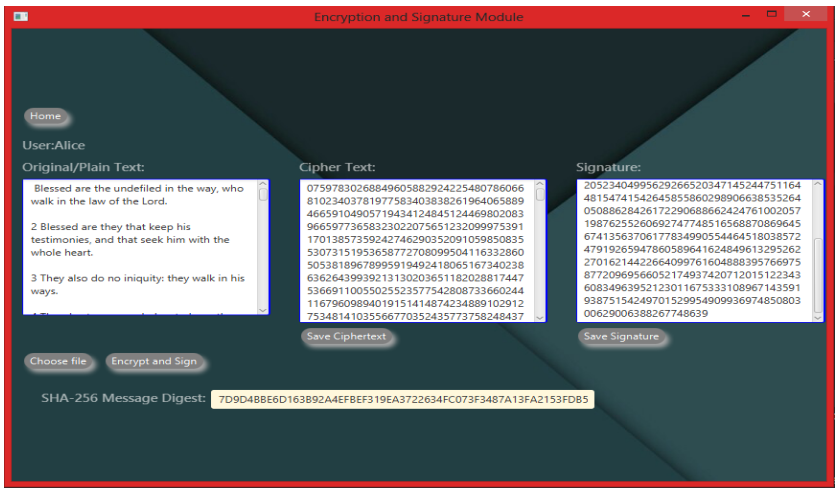

Figure 6. Encryption and signature generation to secure sensitive information.

In Figure 6, the user inputs their text to be encrypted and then clicks on the 'Encrypt and sign' button to generate the cipher text and digital signature for that text input. Figures 7 and 8 illustrate the decryption and signature verification of the file encrypted with the instance of Figure 7 returning a valid signature, while that of Figure 8 returns a message dialog for an invalid signature, which proves that either the signature does not correspond to that file or the file has been altered in some way [34,35].

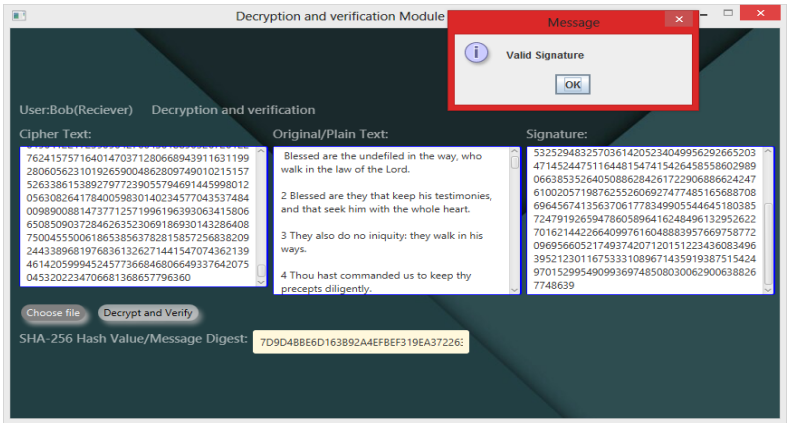

Figure 7. Decryption and signature verification returning a valid signature.

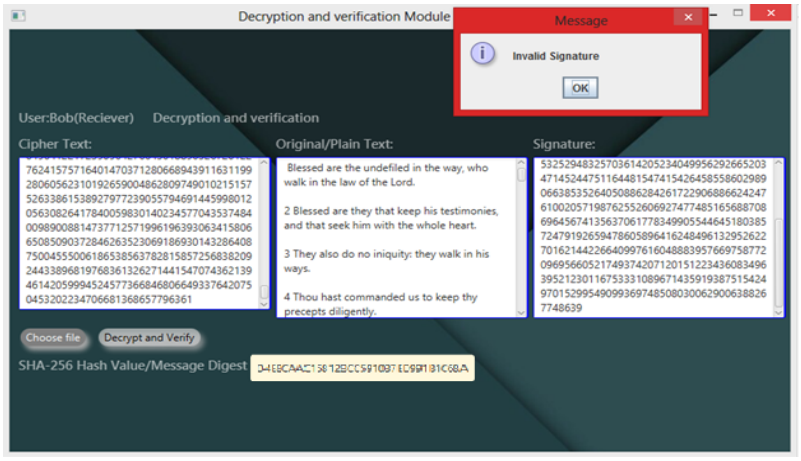

Figure 8. Decryption and signature verification returning an invalid signature.

3.1. Result Analysis

The RSA and the ElGamal algorithms were tested using 2048-bit keys. The time taken for the encryption, decryption, signature generation, and verification modules is given in milliseconds.

3.1.1. Encryption

Various files of different sizes were encrypted using RSA and ElGamal cryptographic algorithms. The encryption time of both algorithms was obtained and placed in a tabular form. See Table 2.

Table 2. Data analysis for encryption process for RSA and ElGamal algorithms.

S/N	File Size (Kb)	RSA Encryption Time (ms)	ElGamal Encryption Time (ms)
1	10	95	3520
2	15	256	4340
3	20	312	6689
4	25	476	7311
5	30	499	7834
6	35	561	8372
7	40	606	9161
8	50	1094	13,215
9	100	2136	19,359
10	200	4229	44,689

Figure 9 displays the encryption time of the RSA and ElGamal process, and its shows that the ElGamal algorithm consumes more time during decryption for various file sizes.

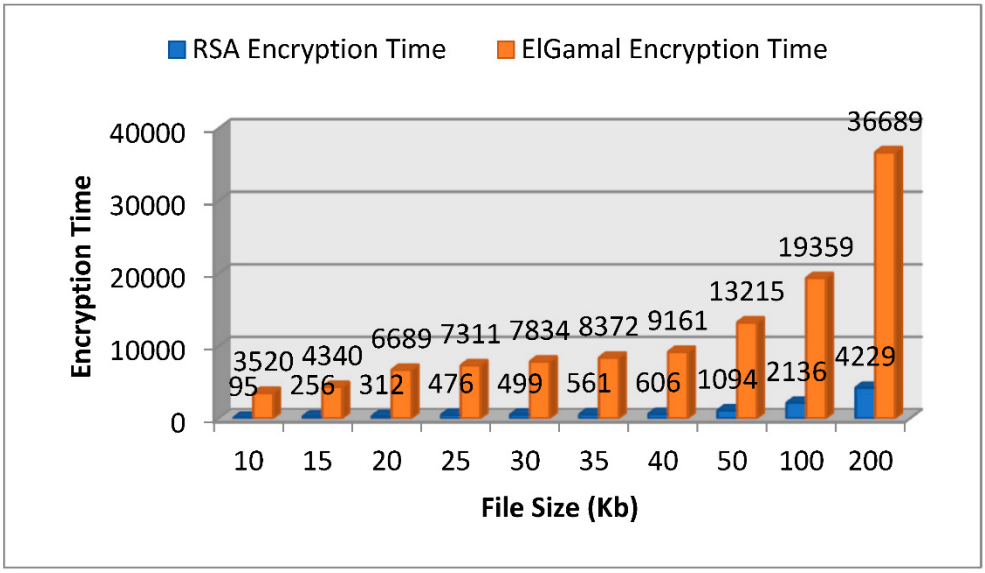

Figure 9. Graphical representation of RSA and ElGamal encryption time.

3.1.2. Decryption

The same file sizes encrypted in Table 2 were decrypted, and their various decryption times during the decryption process were obtained and placed in a tabular form. See Table 3.

Table 3. Data analysis for the decryption process for RSA and ElGamal algorithms.

	Size (Kb)	RSA Decryption Time (ms)	ElGamal Decryption Time (ms)
1	10	3428	637
2	15	5207	975
3	20	7809	1233
4	25	9832	1807
5	30	12,692	2645
6	35	16,325	3293
7	40	18,593	3990
8	50	23,986	4525
9	100	35,479	6829
10	200	42,708	9968

Figure 10 displays the graphical analysis of the RSA and ElGamal decryption process for different file sizes, and the analysis shows that the ElGamal algorithm consumes lesser time during the decryption of file sizes compared to the RSA algorithm.

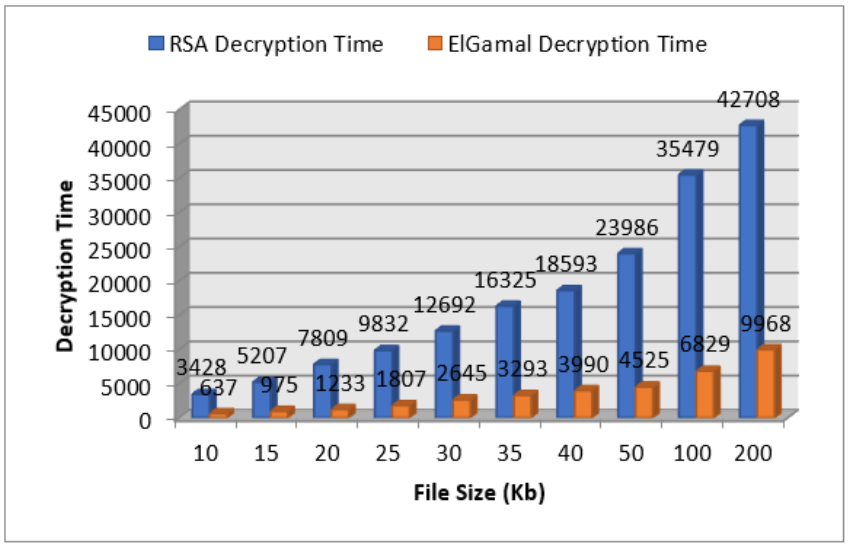

Figure 10. Graphical analysis of RSA and ElGamal decryption time (ms).

3.1.3. Signature Generation

The time taken for both RSA and ElGamal to generate a signature was captured and recorded. Moreover, the time taken for RSA and ElGamal without SHA-256 was obtained and recorded in a tabular form. See Table 4.

Table 4. Data analysis of signature generation process for RSA and ElGamal algorithms.

	File Size (Kb)	RSA Signature Generation Time Taken (ms)	RSA without SHA-256 Time Taken (ms)	ElGamal Signature Generation Time Taken (ms)	ElGamal without SHA-256 Time Taken (ms)
1	10	485	2223	136	381
2	15	469	3405	139	602
3	20	484	4448	145	823
4	25	493	5683	138	1057
5	30	464	6944	147	1346
6	35	473	8073	134	1871
7	40	486	9299	136	2018
8	50	493	10,601	146	2667
9	100	496	18,886	131	3243
10	200	481	23,981	136	4036

Figure 11 displays the graphical analysis of the signature generation. It shows that ElGamal outperforms RSA in signature generation.

3.1.4. Signature Verification

RSA's and ElGamal's time taken for the signature verification process was obtained and recorded. The time taken for both algorithms without SHA-256 was obtained as well in milliseconds and displayed in tabular form. See Table 5.

Figure 12 displays the graphical analysis of RSA and ElGamal signature verification. The analysis shows that RSA performs better than ElGamal in the signature verification process.

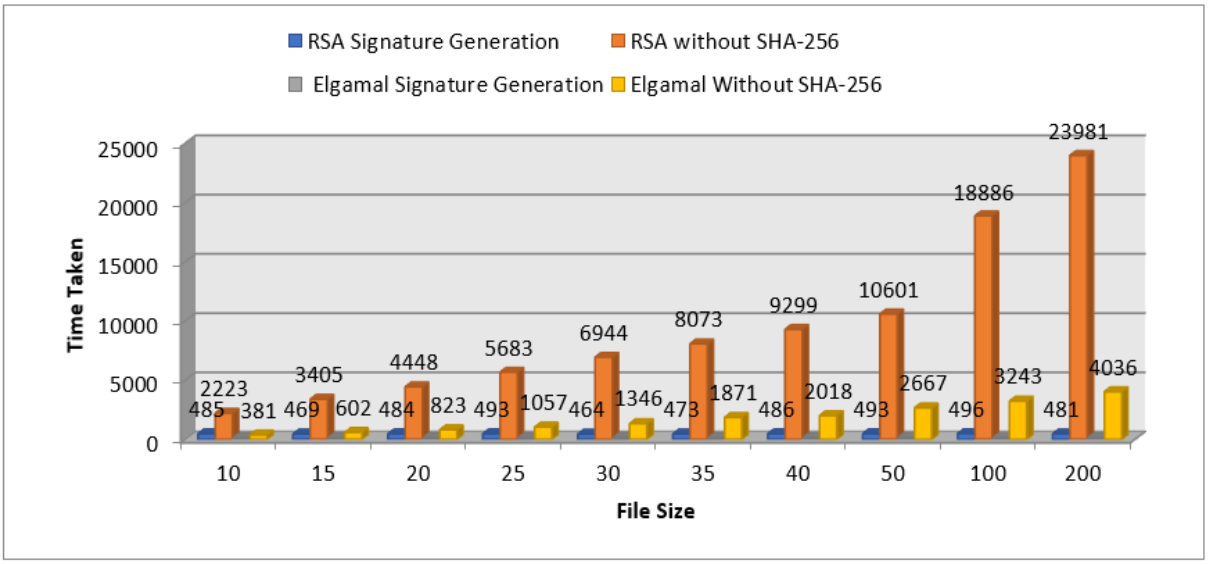

Figure 11. Graphical analysis of RSA and ElGamal signature generation process (ms).

Table 5. Data analysis for signature verification process for RSA and ElGamal algorithms.

	File Size (KB)	RSA Signature Verification Time Taken (ms)	RSA without SHA-256 (ms)	ElGamal Signature Verification (ms)	ElGamal without SHA-256 (ms)
1	10	15	63	177	827
2	15	15	66	189	1281
3	20	12	69	189	1630
4	25	14	76	194	2057
5	30	15	77	165	2718
6	35	15	82	167	3152
7	40	19	87	188	3770
8	50	15	98	190	4234
9	100	21	103	179	5141
10	200	25	122	199	6089

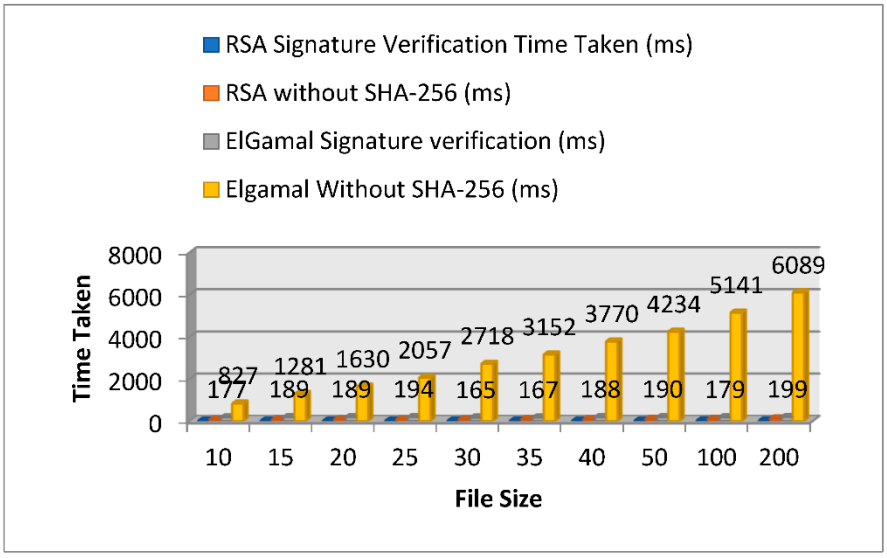

Figure 12. Graphical analysis of the signature verification process of RSA and ElGamal algorithms.

4. Discussion

This study examined the RSA and ElGamal cryptographic algorithms to improve information security. The application of the SHA-256 hash function to the digital signatures of the RSA and ElGamal asymmetric cryptographic algorithms was implemented. From the various experimental results displayed in tables and figures, it can be seen that the RSA algorithm performs better than the ElGamal during the encryption and signature verification processes, while ElGamal performs better than RSA during the decryption and signature generation process. Therefore, it can be deduced that each of the algorithms performs better than the other in some processes; however, there is no obvious superiority of one cryptosystem over the other in all the processes of encryption, decryption, signature generation, and signature verification.

Findings and Comparison with Existing Work

The use of cryptographic hash functions in digital signature generation provides a mechanism such that the integrity check feature of the hash value guarantees a party of the integrity and originality of a document or data; the finding in this study corroborates that of Hamza and Al-Alak [19]. Signing the hash value of data with the use of hash functions, instead of signing the data directly provides a more efficient scheme for a digital signature because the hash of the data is a relatively smaller value compared to the original data, in accordance with Burr [10]. This finding in this study matches that of Haque et al.'s [17] study. However, Haque et al.'s [17] study was outperformed by implementing SHA-256 to achieve data integrity.

5. Conclusions

The need for information security in this present time has become non-negligible in our society due to the daily increasing emergence of cybercrimes, piracy, scam, and fraud cases. As it has been noticed that security and safety concerns are among the most pressing problems confronting potential distributed data, the sending and reception of data are considered vulnerable to external attacks. Therefore, data protection through encryption/decryption is essential. This study examined two asymmetric algorithms (RSA and ElGamal) developed in improving information security services. In addition, the application of the SHA-256 hash function to the digital signatures of the RSA and ElGamal cryptosystems was implemented to establish information integrity. The technique ensures the protection of the security of users' sensitive data and at the same time provides users with full control of their data. Various benefits associated with this study and the correctness of the implemented systems make it suitable for any secure sensitive data sharing system. Therefore, it is recommended that further implementation such as secure submission, storage, and extraction operations of the sensitive data sharing system should be implemented for full and maximum protection of sensitive data.

Author Contributions: Conceptualization, E.A.A., P.B.F. and S.B.; methodology, E.A.A., P.B.F. and S.B.; software, E.A.A., P.B.F. and S.B.; validation, E.A.A., P.B.F. and S.B.; formal analysis, M.S.M., M.A. and S.B.; investigation, E.A.A. and P.B.F.; resources, M.S.M., M.A. and S.B., data curation, S.B.; writing—original draft preparation, E.A.A. and S.B.; writing—review and editing, E.A.A. and S.B.; visualization, E.A.A. and S.B.; supervision, M.S.M., M.A. and S.B.; project administration, M.S.M., M.A. and S.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Research Supporting Project (number RSP2022R459), King Saud University, Riyadh, Saudi Arabia.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: Research Supporting Project (number RSP2022R459), King Saud University, Riyadh, Saudi Arabia.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gunduz, M.Z.; Das, R. Cyber-security on smart grid: Threats and potential solutions. *Comput. Netw.* **2020**, *169*, 107094. [CrossRef]
2. Saxena, N.; Choi, B.J.; Lu, R. Authentication and Authorization Scheme for Various User Roles and Devices in Smart Grid. *IEEE Trans. Inf. Forensics Secur.* **2015**, *11*, 907–921. [CrossRef]
3. Misbha, A.; Baswal, K.; Simha, M.N.; Abujam, R.; C.M., S. GUPTDOC AN ENTERPRISE PORTAL FOR CRYPTING WITH AES. *Int. Res. J. Eng. Technol.* **2017**, *4*, 1309–1311.
4. Emmanuel, A.A.; Okeyinka, A.E.; Adebiyi, M.O.; Asani, E.O. A Note on Time and Space Complexity of RSA and ElGamal Cryptographic Algorithms. *Int. J. Adv. Comput. Sci. Appl.* **2021**, *12*, 143–147. [CrossRef]
5. Chandra, S.; Paira, S.; Alam, S.S.; Sanyal, G. A comparative survey of Symmetric and Asymmetric Key Cryptography. In Proceedings of the 2014 International Conference on Electronics, Communication and Computational Engineering (ICECCE), Hosur, India, 17–18 November 2014; pp. 83–93. [CrossRef]
6. Sejfuli-Ramadani, N. The Role and the Impact of Digital Certificate and Digital Signature in Improving Security During Data Transmission. *Eur. J. Sustain. Dev. Res.* **2017**, *2*, 116–120.
7. Winter, C.; Berchtold, W.; Hollenbeck, J.N. Securing physical documents with digital signatures. In Proceedings of the 2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP), Kuala Lumpur, Malaysia, 18 November 2019; pp. 1–6.
8. Zhang, H.; Yuan, Z.; Wen, Q.Y. A Digital Signature Schemes Without Using One-way Hash and Message Redundancy and Its Application on Key Agreement. In Proceedings of the 2007 IFIP International Conference on Network and Parallel Computing Workshops (NPC 2007), Dalian, China, 18–21 September 2007; pp. 873–878. [CrossRef]
9. Chang, C.C.; Chang, Y.F. A novel three-party encrypted key exchange protocol. *Comput. Stand. Interfaces* **2004**, *26*, 471–476. [CrossRef]
10. Burr, W. Cryptographic hash standards: Where do we go from here? *IEEE Secur. Priv.* **2006**, *4*, 88–91. [CrossRef]
11. Acharya, K.; Sajwan, M.; Bhargava, S. Analysis of Cryptographic Algorithms for Network Security. *Int. J. Comput. Appl. Technol. Res.* **2014**, *3*, 130–135. [CrossRef]
12. Saleh, E.; Meinel, C. HPISecure: Towards Data Confidentiality in Cloud Applications. In Proceedings of the 2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, Delft, Netherlands, 13–16 May 2013; pp. 605–609. [CrossRef]
13. Mell, P.; Grance, T. The NIST definition of cloud computing. *NIST Spec. Publ.* **2011**, *800*, 145.
14. Abiodun, M.K.; Awotunde, J.B.; Ogundokun, R.O.; Misra, S.; Adeniyi, E.A.; Arowolo, M.O.; Jaglan, V. Cloud and Big Data: A Mutual Benefit for Organization Development. *J. Phy. Conf. Ser.* **2021**, *1767*, 012020. [CrossRef]
15. Hwang, J.J.; Chuang, H.K.; Hsu, Y.C.; Wu, C.H. A Business Model for Cloud Computing Based on a Separate Encryption and Decryption Service. In Proceedings of the 2011 International Conference on Information Science and Applications, Jeju, Korea, 26–29 April 2011; pp. 1–7. [CrossRef]
16. Chandra, D.G.; Prakash, R.; Lamdharia, S. A Study on Cloud Database. In Proceedings of the 2012 Fourth International Conference on Computational Intelligence and Communication Networks, Mathura, India, 3–5 November 2012; pp. 513–519.
17. Haque, E.; Zobaed, S.; Islam, M.U.; Areef, F.M. Performance Analysis of Cryptographic Algorithms for Selecting Better Utilization on Resource Constraint Devices. In Proceedings of the 2018 21st International Conference of Computer and Information Technology (ICCIT), Dhaka, Bangladesh, 21–23 December 2018; pp. 1–6. [CrossRef]
18. Dijesh, P.; Babu, S.; Vijayalakshmi, Y. Enhancement of e-commerce security through asymmetric key algorithm. *Comput. Commun.* **2020**, *153*, 125–134. [CrossRef]
19. Hamza, A.H.; Al-Alak, S.M.K. Evaluation key generator of multiple asymmetric methods in wireless sensor networks (WSNs). *J. Phys. Conf. Ser.* **2021**, *1804*, 012096. [CrossRef]
20. Bharany, S.; Sharma, S.; Badotra, S.; Khalaf, O.I.; Alotaibi, Y.; Alghamdi, S.; Alassery, F. Energy-Efficient Clustering Scheme for Flying Ad-Hoc Networks Using an Optimized LEACH Protocol. *Energies* **2021**, *14*, 6016. [CrossRef]
21. Kaur, K.; Bharany, S.; Badotra, S.; Aggarwal, K.; Nayyar, A.; Sharma, S. Energy-efficient polyglot persistence database live migration among heterogeneous clouds. *J. Supercomput.* **2022**, 1–30. [CrossRef]
22. Bharany, S.; Sharma, S.; Bhatia, S.; Rahmani, M.K.I.; Shuaib, M.; Lashari, S.A. Energy Efficient Clustering Protocol for FANETS Using Moth Flame Optimization. *Sustainability* **2022**, *14*, 6159. [CrossRef]
23. Steichen, M.; Fiz Pontiveros, B.; Norvill, R.; Shbair, W. Blockchain-Based, Decentralized Access Control for IPFS. In Proceedings of the 2018 IEEE International Conference on Blockchain (Blockchain-2018), Halifax, NS, Canada, 30 July–3 August 2018; pp. 1499–1506.
24. Gaby, G.; Chandra, L.; Enderson, T. Towards Secure Interoperability between Heterogeneous Blockchains using Smart Contracts. In Proceedings of the Future Technologies Conference (FTC), Vancouver, BC, Canada, 15–16 November 2017; pp. 73–81.
25. Bharany, S.; Sharma, S.; Khalaf, O.I.; Abdulsahib, G.M.; Al Humaimeedy, A.S.; Aldhyani, T.H.H.; Maashi, M.; Alkahtani, H. A Systematic Survey on Energy-Efficient Techniques in Sustainable Cloud Computing. *Sustainability* **2022**, *14*, 6256. [CrossRef]
26. Nizamuddin, N.; Salah, K.; Azad, M.A.; Arshad, J.; Rehman, M. Decentralized document version control using ethereum blockchain and IPFS. *Comput. Electr. Eng.* **2019**, *76*, 183–197. [CrossRef]
27. Bharany, S.; Kaur, K.; Badotra, S.; Rani, S.; Kavita; Wozniak, M.; Shafi, J.; Ijaz, M.F. Efficient Middleware for the Portability of PaaS Services Consuming Applications among Heterogeneous Clouds. *Sensors* **2022**, *22*, 5013. [CrossRef]

28. Guo, R.; Shi, H.; Zhao, Q.; Zheng, D. Secure Attribute-Based Signature Scheme With Multiple Authorities for Blockchain in Electronic Health Records Systems. *IEEE Access* **2018**, *6*, 11676–11686. [CrossRef]
29. Bharany, S.; Badotra, S.; Sharma, S.; Rani, S.; Alazab, M.; Jhaveri, R.H.; Gadekallu, T.R. Energy efficient fault tolerance techniques in green cloud computing: A systematic survey and taxonomy. *Sustain. Energy Technol. Assess.* **2022**, *53*, 102613. [CrossRef]
30. Dias, J.P.; Reis, L.; Ferreira, H.S.; Martins, Â. Blockchain for access control in e-health scenarios. *arXiv* **2018**, arXiv:1805.12267.
31. Bharany, S.; Sharma, S.; Frnda, J.; Shuaib, M.; Khalid, M.I.; Hussain, S.; Iqbal, J.; Ullah, S.S. Wildfire Monitoring Based on Energy Efficient Clustering Approach for FANETS. *Drones* **2022**, *6*, 193. [CrossRef]
32. Fukumitsu, M.; Hasegawa, S.; Iwazaki, J.; Sakai, M.; Takahashi, D. A proposal of a secure P2P-type storage scheme by using the secret sharing and the blockchain. In Proceedings of the 2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA), Taipei, Taiwan, 27–29 March 2017; pp. 803–810.
33. Pãnescu, A.T.; Manta, V. Smart Contracts for Research Data Rights Management over the Ethereum Blockchain Network. *Sci. Technol. Libr.* **2018**, *37*, 235–245. [CrossRef]
34. Dai, M.; Zhang, S.; Wang, H.; Jin, S. A Low Storage Room Requirement Framework for Distributed Ledger in Blockchain. *IEEE Access* **2018**, *6*, 22970–22975. [CrossRef]
35. Nizamuddin, N.; Hasan, H.; Salah, K.; Iqbal, R. Blockchain-Based Framework for Protecting Author Royalty of Digital Assets. *Arab. J. Sci. Eng.* **2019**, *44*, 3849–3866. [CrossRef]