information

MDPI

Article

Enhancing Organizational Data Security on Employee-Connected Devices Using BYOD Policy

Manal Rajeh AlShalaan * and Suliman Mohamed Fati

College of Computer and Information Sciences, Prince Sultan University, Riyadh 11586, Saudi Arabia; sgaber@psu.edu.sa
* Correspondence: 221421246@psu.edu.sa

Abstract: To address a business need, most organizations allow employees to use their own devices to enhance productivity and job satisfaction. For this purpose, the Bring Your Own Device (BYOD) policy provides controllable access for employees to organize data through their personal devices. Although the BYOD practice implies plenty of advantages, this also opens the door to a variety of security risks. This study investigates these security risks and proposes a complementary encryption approach with a digital signature that uses symmetric and asymmetric algorithms, depending on the organization's digital certificate, to secure sensitive information stored in employees' devices within the framework of BYOD policies. The method uses Advanced Encryption System (AES), Blowfish, RSA and ElGamal with a digital signature to achieve strong encryption and address critical security considerations such as user authentication, confidentiality and data integrity. The proposed encryption approach offers a robust and effective cryptographic solution for securing sensitive information in organizational settings that involve BYOD policies. The study includes experimental results demonstrating the proposed approach's efficiency and performance, with reasonable encryption and decryption times for different key and file sizes. The results of the study revealed that AES and Blowfish have the best execution time. AES has a good balance of security and performance. RSA performs better than ElGamal in encryption and signature verification, while RSA is slower than ElGamal in decryption. The study also provides a comparative analysis with previous studies of the four encryption algorithms, highlighting the strengths and weaknesses of each approach.

Keywords: BYOD; AES; RSA; ElGamal; blowfish; digital signature; encryption; security

check for updates

Citation: AlShalaan, M.R.; Fati, S.M. Enhancing Organizational Data Security on Employee-Connected Devices Using BYOD Policy. *Information* **2023**, *14*, 275. https://doi.org/10.3390/info14050275

Academic Editors: Moutaz Alazab and Ammar Alazab

Received: 30 March 2023
Revised: 30 April 2023
Accepted: 3 May 2023
Published: 5 May 2023

1. Introduction

Organizations widely deploy rapidly evolving technologies that provide significant benefits but expose them to cyber-attacks due to employees connecting their devices to the information system. A key concern for modern organizations is protecting their assets against such attacks, particularly when safeguarding corporate data that are critical to their functions. Over time, the insecure storage of such data can negatively affect confidentiality, leading to financial losses and reputational damage for the organizations [1]. Organizations invest heavily in acquiring the latest hardware and software technologies with high-security standards to prevent such security breaches. Still, some underestimate the true nature of cybersecurity attacks, which limits the adoption of advanced security measures [1]. Common devastating cyber-attacks involve employee negligence and limited information on the best cybersecurity practices [2]. In particular, the traversal of organizational data to employees' devices is the primary source of data breaches, compromising the entire system's security. Therefore, securing organizational data by addressing this issue is very critical. Most organizations adopt Bring Your Own Device (BYOD) policies, and they should prioritize the security of devices connected to their systems.

BYOD is a widespread practice where employees use their personally owned devices, including laptops and smartphones, for work purposes [3,4]. It offers several benefits, such as increased mobility, flexibility, productivity and employee satisfaction. Most organizations have implemented information security policies to address the underlying security risks, but employee compliance can be an issue [3]. According to a survey by Bitglass, 69% of companies permit their employees to work with their own devices to complete their business work [5]. The security risks associated with BYOD are consistent with a BYOD policy compliance report by Palanisamy et al., which revealed that around 21% of organizations suffered a security breach due to mobile devices connecting to malicious Wi-Fi hotspots [3]. In 2021, LinkedIn experienced a data leak affecting approximately 500 to 700 million user accounts [6]. Although BYOD implies plenty of advantages, this also opens the door to various security risks, including data contamination and leakage, which can be costly financially and reputation-wise [3]. One of the leading security risks is that BYOD does not consider the security of the downloaded data in employees' devices [7], making the policy vulnerable to attacks and data breaches. Given that the storage space of these devices has a mixture of personal information and sensitive data and documents stored during the employees' work on the system, such stored data might be vulnerable to many threats that endanger the safety of user data storage, such as physical and malware attacks [8]. Therefore, the need to secure the stored data on employees' devices becomes a critical issue, which is the focus of this paper.

This study aims to enhance an organizational system's security by implementing additional security measures on employees' devices that are authorized to access and store sensitive corporate data. This security approach involves encrypting the download of company documents using the encryption algorithm in the organization's digital certificate. This encryption process must be applied as a mandatory requirement to enforce BYOD policies. The proliferation of BYOD adoption in organizations has not adequately considered the associated security risks of downloaded data. The study focuses on reducing the risk of corporate data breaches by securing downloaded data on employees' devices through encryption, thus protecting organizations' privacy. It emphasizes the vulnerability of organizational data stored on employees' devices, which can be exposed to attacks and data breaches. The proposed complementary encryption algorithms, such as Advanced Encryption System (AES), RSA, ElGamal and Blowfish, enhance BYOD security and protect corporate data on employee-connected devices. Additionally, this paper highlights the need for organizations to comply with the Essential Cybersecurity Controls established by the National Cybersecurity Authority to ensure the confidentiality, integrity and availability of an organization's information and technology assets. The following objectives are instrumental in achieving the goal of this study:

1. Propose an encryption algorithm that runs on the user's device upon downloading.
2. Build an encryption key generator that is centrally under the organization's control and will manage the encryption keys for the devices connected to the system.
3. Evaluate the proposed model to ensure security according to predefined criteria (key length, file size).

The remainder of the present study is structured as follows: Section 2 comprises a comprehensive review of the relevant literature about the topic under investigation. Section 3 expounds on the research methodology adopted and the proposed approach for the complementary encryption algorithm. Section 4 presents experimental results. Section 5 elaborates on the discussion and findings, encompassing a comparative analysis. Section 6 summarizes the paper, culminating in a conclusive statement that encapsulates its critical discoveries.

2. Related Work

The prevalence of technology adoption by enterprises in adherence to achieving optimal productivity, flexibility and end-user satisfaction has influenced the need to embrace BYOD, which allows employees to use their personal smart devices to access an organization's data and applications. The BYOD model has positively impacted creativity, communication and cohesion, increasing productivity and performance.

Despite its numerous benefits, the BYOD model has left enterprises susceptible to many cyber threats, which include data breaches, data manipulation and a lack of user authentication. As a result, multiple security concerns may emerge, such as malware, viruses, Trojans, data leaks and data manipulation [9,10]. All of these issues lead to the high cost of maintaining BYOD devices across respective network architectures [11,12]. Experienced and privileged user employees pose immense internal threats and risks [3]. They possess the ability to maliciously sabotage digitized transactions, and in other instances, they become victims of phishing and related social engineering attacks. About 75% of respondents indicated a vacuum in organizational requirements and mandates requiring the need to conduct security protocols and measures [3]. Although the BYOD model has multiple benefits, it remains sensitive to data privacy and confidentiality caused by cyber-attacks [8]. Consequently, commercial institutions must protect their digital systems from threat actors and related consequences. Apart from implementing basic training and programs to eliminate ignorance among unsuspecting end-users, this paper emphasizes that corporations should focus on technical countermeasures to mitigate and avoid cyber-attacks.

Cryptography is the most viable tool for providing network security based on its ability to uphold privacy, authenticity and integrity. Encryption algorithms facilitate data transformation through mathematical formulas and prevent unauthorized users from accessing private and sensitive data [13]. There are three types of cryptosystems: symmetric (private key), asymmetric (public key) and hash functions [12,14]. The commonly used symmetric algorithms are AES, DES, 3DES and chaos cryptology, whereas RSA is the most-used asymmetric algorithm [15]. In symmetric algorithms, the sender and receiver share a common key, whereas asymmetric systems entail two keys. One key is publicly known (aka public key), and only the receiver knows the second one (aka private key). Given that symmetric algorithms use similar keys, they are less complex than asymmetric cryptography and are a hundred times faster [16]. Symmetric keys are suitable for ensuring the safety of organizational devices and stored data [17]. They have the primary objective of upholding the privacy and confidentiality of the communication channels. However, asymmetric keys are instrumental in implementing encrypted data transfers even when both parties cannot acquire a symmetric key in a private algorithm [17]. The length of asymmetric keys corresponds directly to their security strength and performance; the higher the key length, the more difficult discovering the key becomes, and implicitly, the performance reduces [18]. Thus, the level of security strength needed will be based on the sensitivity of the encrypted data [19].

There are different types of symmetric and asymmetric cryptosystems, and they have unique contributions to securing digital systems through encryption. In the following subsections, we will explain some of the relevant cryptography algorithms, which are the focus of this study.

2.1. Advanced Encryption System

AES is a private key encryption algorithm that secures data and communication channels. AES attains encryption through permutations, substitutions, mixing and key-adding [17]. It is the fastest encryption algorithm, capable of encrypting 1500 KB files in less than a second while consuming the lowest energy levels [13]. AES was suggested to have better performance than other algorithms such as RSA, DES and hashing, where AES has the lowest time complexity [20,21]. AES could be used to mitigate DoS attacks on IoT or other devices [22]. AES operates on fixed data block lengths of 128 bits and incorporates the substitution and permutation structure. The AES protects network systems against differential, linear, statistical and brute-force attacks [16].

AES encryption–decryption processes are mainly done in rounds, and each one contains four basic stages as follows:

I. ShiftRows 'permutation stage'
II. Byte Substitute 'substitution stages'
III. MixColumn 'substitution stages'
IV. AddRundKey 'substitution stages'

Figure 1 represents the AES algorithm flow chart [23].

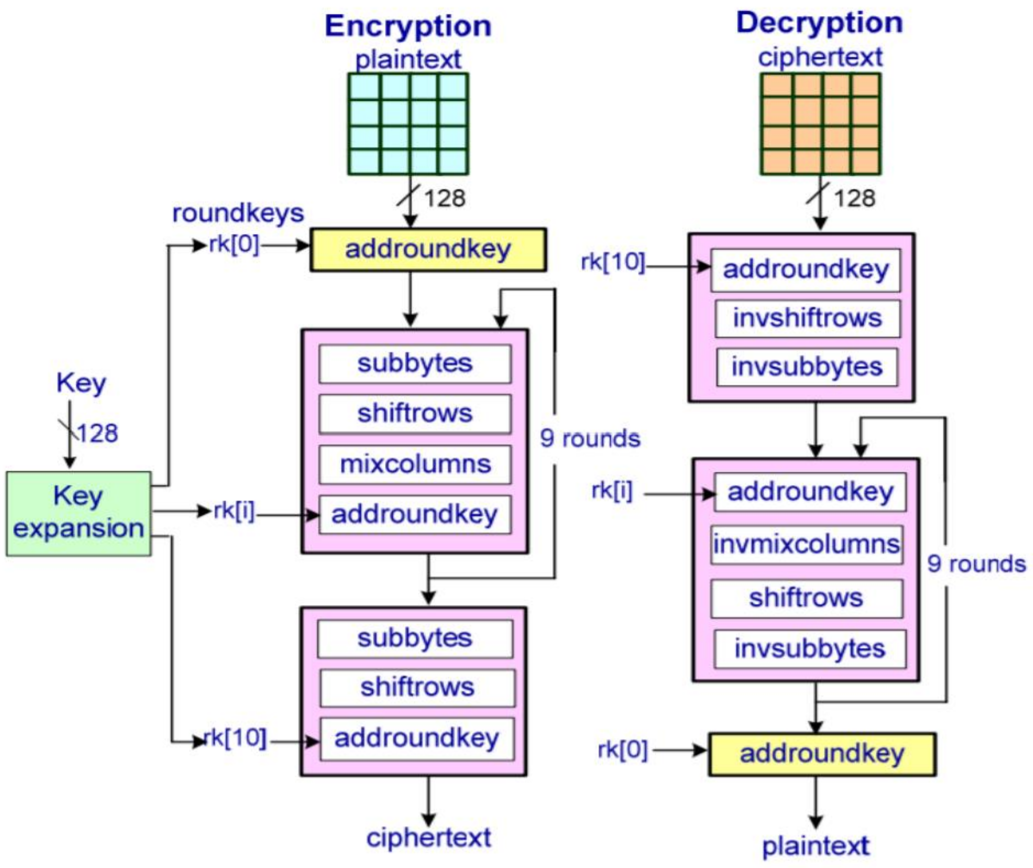


Figure 1. AES algorithm flow chart [23].

2.2. *Blowfish Algorithm*

The Blowfish algorithm is a symmetric key synonymous with good performance and outstanding optimization of the hardware and software applications. Blowfish is highly competent in encrypting large data files [16]. The Blowfish algorithm is faster than AES when handling image, audio, video and other data types in terms of processing time but uses similar or less memory [16,24]. Consequently, Blowfish uses less processing power, making its performance optimal. However, the blowfish algorithm is less secure than other cryptosystems [16].

The Blowfish algorithm works with varying lengths not greater than 449 bits and a 64-bit block with 16 rounds, as presented in Figure 2 [13].

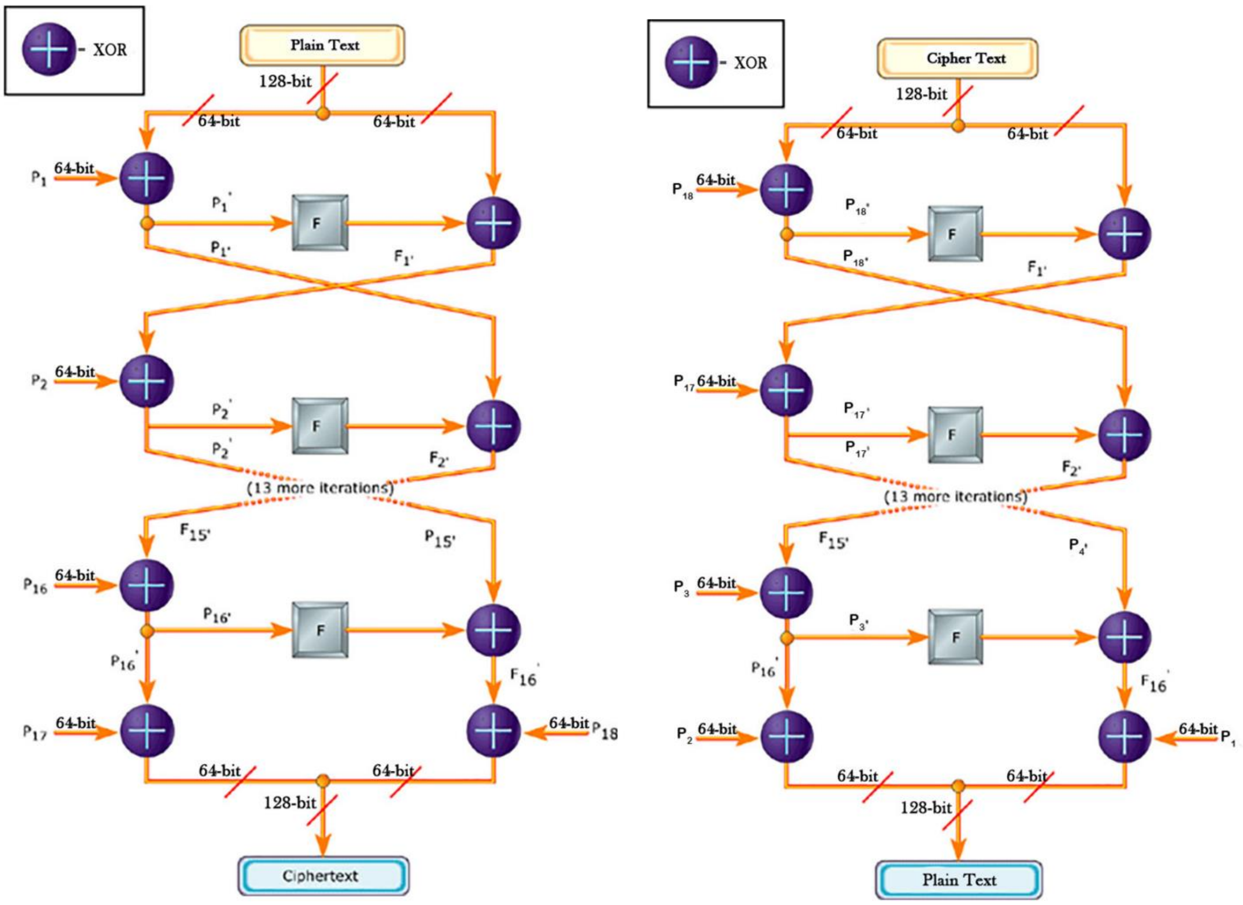


Figure 2. Blowfish algorithm flow chart [25].

2.3. RSA Algorithms

RSA is an asymmetric cryptosystem that utilizes two separate keys for decrypting and encrypting data files. The keys entail private and public keys, whereby the former decrypts the data while the latter encrypts files. Based on its capitalization on factorizing large prime numbers in key generation, it has formidable security systems compared to other asymmetric algorithms [17]. The generation of keys based on the factorization of big prime numbers culminates in a larger encryption time for available datasets. Furthermore, a larger time is also needed to decrypt the files, making RSA the slowest among the algorithms [16]. Its complexity requires the double memory used by other symmetric algorithms such as AES and Blowfish. However, RSA is efficient in offering a high degree of confidentiality and beats ElGamal in most performance metrics [26].

RSA key generation can be used to produce public and private key pairs by following the steps below [27]:

I. p and q are prime numbers. Then, we calculate the modulus n = pq.
II. Choose a substantial prime number (r) to produce (p 1) (q1); r will be the public exponent.
III. Use the result of $(rs - 1)/((p - 1)(q - 1))$ to find a number s. s will be the private exponent.
IV. The public key will be (n, r). Although n and r are public, knowing r from n and s is computationally not possible except if p and q are not large enough.
V. For encrypting message m, the cipher text C will be generated using the equation $C = m^r \bmod (n)$
VI. To decrypt the cipher text, the following equation will be used: $m = c^5 \bmod(n)$.

Figure 3 illustrates the flow chart of the RSA algorithm.

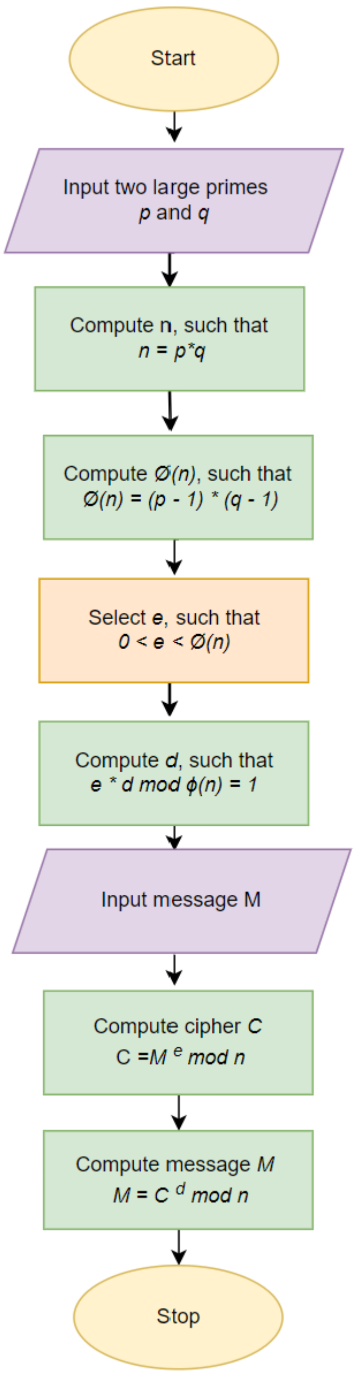

Start
Input two large primes p and q
Compute n, such that n = p*q
Compute Ø(n), such that Ø(n) = (p - 1) * (q - 1)
Select e, such that 0 < e < Ø(n)
Compute d, such that e * d mod ϕ(n) = 1
Input message M
Compute cipher C C =M e mod n
Compute message M M = C d mod n
Stop

Figure 3. RSA algorithm flow chart [13].

Digital Signature

A digital signature is a method of signing files electronically and achieving authenticity and legitimacy using keys and encryption [26]. The signature will be different for each file based on the file information, which guarantees that the electronic document is reliable, the signature was created by a known source, and the document has not been altered; selected verifiers only can recover and verify the file from the digital signature [28]. The digital signature can be done using one of the public key cryptosystems, RSA or ElGamal and many other systems [29,30]. Kritsanapong et al. [31] showed that RSA is 100% accurate in both signing and checking procedures, and it completed these processes with a good performance rate.

2.4. ElGamal

ElGamal is central to upholding and maintaining information security. It is a public algorithm cryptosystem vital in encrypting large data files. ElGamal is a robust and effective algorithm for maintaining the security and quality of encrypted and decrypted, and it shows performance efficiency comparable to state-of-the-art methods [32,33].

Although RSA is faster than ElGamal in the encryption and signature verification process, the latter decrypts and generates signatures faster than the former [12]. Although RSA and ElGamal are slower than symmetric algorithms such as AES and Blowfish, they have better security. The symmetric algorithms are faster and demand less computational power but are less secure than asymmetric algorithms. Public and private keys indicate that every user possesses a unique key inaccessible to other parties, leading to increased security compared with other encryption algorithms.

ElGamal key generation will be processed as follows [12,34]:

I. Pick p, which is a prime number.
II. Select g as a generator number.
III. Choose x as a random integer between 0 and p-2, where 0 < x < p-2, and x will be the secret value.
IV. Generate y using this equation: y = gx mod p. (p, x) is the private key, and (p, g, y) is the public key

After generating y, the key encryption and decryption process will be as follows [12,34]:

I. To encrypt a message M, a public key will be used along with a random secret integer k, where 1 < k < p-2.
II. A message bit will be transformed into a cipher by calculating C1, C2 . . .
III. To calculate C1, this equation will be used: a = gk mod p.
IV. To calculate C2, this equation will be used: b = (yk ∗ M) mod p.
V. The cipher file will be C = (C1, C2).
VI. To decrypt C, the private key (p, x) will be used.
VII. Find the cipher C = (a, b).
VIII. Find a, where a = (C1x) p-2 mod p, and calculate M, where M = (a ∗ C2) mod p.

Figure 4 shows the flow chart of ElGamal.

2.5. Discussion of Related Work

Various researchers have examined the encryption capabilities of different algorithms and identified their capacity to ensure adequate encryption. Tiwari et al. [21] examined the suitability of AES and RSA algorithms and noticed that they offered a high level of encryption capacity because they depend on public and private keys shared securely to reduce the risk of unauthorized viewing or modification of documents and messages. Buhari et al. [24] investigated the capabilities of AES and Blowfish and found that the two algorithms offer benefits such as low execution times that can enhance user experience. Yousif [26] argued that the implementation of a public and private key helps encryption by reducing the risk of individuals having both keys and modifying messages and documents. Commey et al. [16] compared the efficiency of the Triple DES, AES, Blowfish and RSA algorithms. However, they mentioned that a browser and a word processor affected the result accuracy. Emmanuel et al. [12] evaluated the performance and space complexities of RSA and ElGamal cryptographic algorithms, but the concluded results were based on one data type only. Adeniyi et al. [29] assessed the execution times of RSA, ElGamal, RSA digital signature and ElGamal digital signature, though the performance was measured based on one file type: text files. Their work was applied to the transmitted data, and they suggested a future implementation that secures stored sensitive data as well. Finally, Kritsanapong et al. [31] contended that digital signatures are created using public keys that are unique to the sender and verified using corresponding public keys. Therefore, they have a high level of security and allow data encryption, reducing the risk of unauthorized viewing and modification.

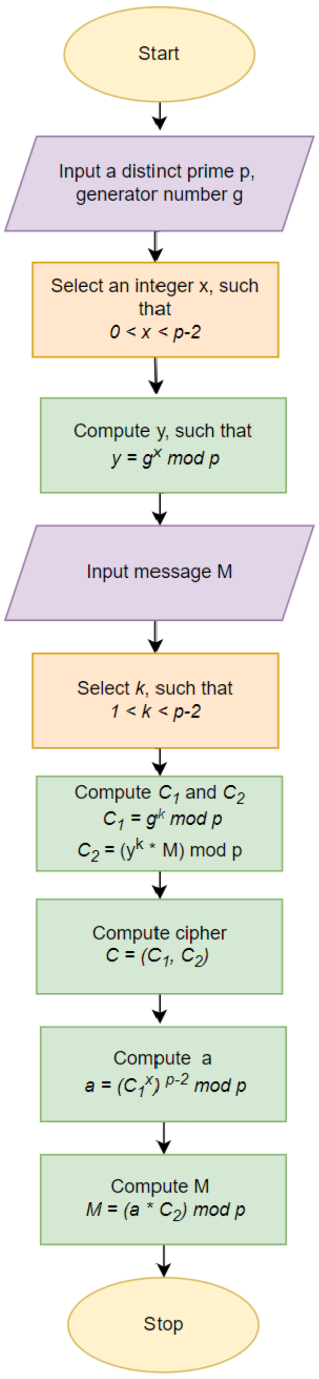

Start
Input a distinct prime p, generator number g
Select an integer x, such that $0 < x < p-2$
Compute y, such that $y = g^x \mod p$
Input message M
Select k, such that $1 < k < p-2$
Compute C_1 and C_2 $C_1 = g^k \mod p$ $C_2 = (y^k * M) \mod p$
Compute cipher $C = (C_1, C_2)$
Compute a $a = (C_1^x)^{p-2} \mod p$
Compute M $M = (a * C_2) \mod p$
Stop

Figure 4. ElGamal algorithm flow chart [12].

Although the researchers recognize the usefulness of these algorithms in data encryption, they do not address their relevance in a BYOD policy. Accordingly, the present study intends to address these research gaps by implementing an algorithm to secure stored organizational data and examining the performance based on different types of documents, as well as ensuring that the accuracy will not be affected by any external factors.

Table 1 shows a summary of all the reviewed related work with their results.

Table 1. Summary of related work.

S/N	Author	Methods	Result
1	Commey et al. [16]	3DES, AES, Blowfish and RSA	Blowfish is the fastest followed by AES, while RSA uses about twice the memory used by the symmetric algorithms
2	Abay [14]	Blowfish, IDEA and AES	AES uses less power for encrypting the text than Blowfish, and AES can be used in situations where high security is needed. In case of performance aspects, Blowfish can be used
3	Emmanuel et al. [12]	RSA and ElGamal	RSA performs better than the ElGamal during encryption, where the ElGamal algorithm performs better in terms of decryption, and RSA operates better in terms of space usage during decryption.
4	Rouaf and Yousif [13]	DES, AES, RSA, PRESENT and REA	AES and DES have the greatest execution time with good performance on all devices. REA performs well on all mobile devices. PRESENT is slow as a lightweight algorithm. RSA is the slowest one.
5	Oleiwi et al. [17]	DES, AES and RSA	AES is considered the best algorithm among symmetric key encryption algorithms. Among asymmetric encryption algorithms, RSA provides greater security.
6	Adeniyi et al. [29]	SHA-256, RSA, ElGamal and digital signature	RSA is better than ElGamal during the encryption and signature verification, while ElGamal performs better than RSA during the decryption and signature generation process.
7	Yousif [26]	RSA and ElGamal	The analysis reveals that the El-Gamal and RSA methods are efficient and sufficient for offering a high degree of confidentiality, security and reliability. However, RSA outperforms ElGamal in most performance metrics
8	Ali et al. [20]	AES, RSA, DES and 3DES	The results of the experimental analysis suggest that AES algorithm outperforms other algorithms, such as RSA and DES, in encrypting and decrypting files of different sizes. Therefore, the algorithm offers less time complexity to achieve better operation.
9	Tiwari et al. [21]	AES, RSA and hashing	AES algorithm takes less time to encrypt and decrypt files compared with hashing and RSA algorithms.
10	Buhari et al. [24]	AES and Blowfish	Blowfish is more efficient compared with AES when handling most file types. Therefore, Blowfish outperforms AES in most of the test cases.

The reviewed articles show that using encryption and digital signatures is a reliable solution for data security and integrity. As mentioned by BYOD research papers, there are several limitations while applying BYOD, which lacks protection for the downloaded data on the device. Encryption and digital signature were not applied by previous studies along with BYOD policies. This motivated us to design a complimentary encryption module with key generation and digital signatures for organizations that apply BYOD policies to add a level of security for organizations' confidential data because BYOD devices are the most vulnerable security link in most organizations.

3. Materials and Methodology

This study aims to enhance the protection of organizational data by adding another protection layer to employee-connected devices according to the BYOD policy. The proposed methodology assumes the organization has high standard of security on the organizational side, with an effective BOYD policy that is acceptable to all the employees. However, the download of confidential business documents and data into the employees' devices is one of the severe security issues that violate data confidentiality. Such confidentiality violations occur when the employees use the devices in a less secure environment.

Thus, there is an urgent need to add extra security to employees' devices, which are authorized to store sensitive organizational data securely. This extra feature involves encrypting the download of organizational documents; the encryption process will be a part of BYOD policy enforcement requirements. This study will evaluate the effectiveness of various encryption algorithms along with digital signatures and a hash function (SHA-256) for securing organizational data that are accessible by employees through their personal devices for work purposes.

The evaluation involves symmetric encryption algorithms (AES and Blowfish) and asymmetric cryptography algorithms (RSA and ElGamal), both with RSA digital signatures. The proposed system, built using Java as a programming language and Swagger UI as GUI for the created API, focuses on fundamental security considerations, such as confidentiality, non-repudiation and data integrity. Encryption was used for confidentiality, while digital signatures were used to ensure the integrity of the data, with any disparity between the signatures implying that the data had been altered, providing a measure of data integrity.

The methodology involves assessing the performance of implemented algorithms in terms of speed and complexity and comparing results with other previous studies. The experiments include evaluating the encryption and decryption times of the algorithms with digital signatures and the impact of different key lengths and file sizes on the performance.

The system diagram is displayed in Figures 5 and 6. Figure 5 displays the flow of downloading organizational documents to employees' devices.

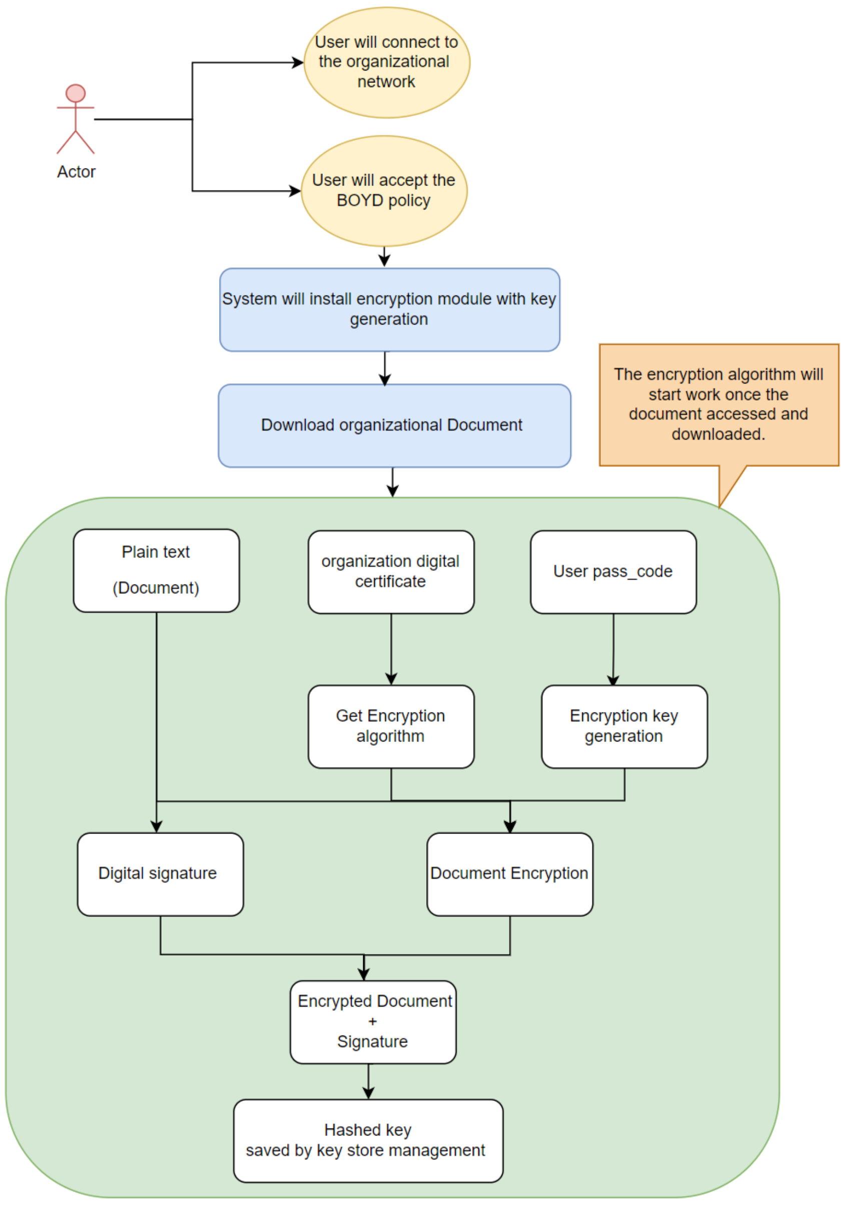

Actor
User will connect to the organizational network
User will accept the BOYD policy
System will install encryption module with key generation
Download organizational Document
The encryption algorithm will start work once the document accessed and downloaded.
Plain text (Document)
organization digital certificate
User pass_code
Get Encryption algorithm
Encryption key generation
Digital signature
Document Encryption
Encrypted Document + Signature
Hashed key saved by key store management

Figure 5. System flow of downloading organizational documents.

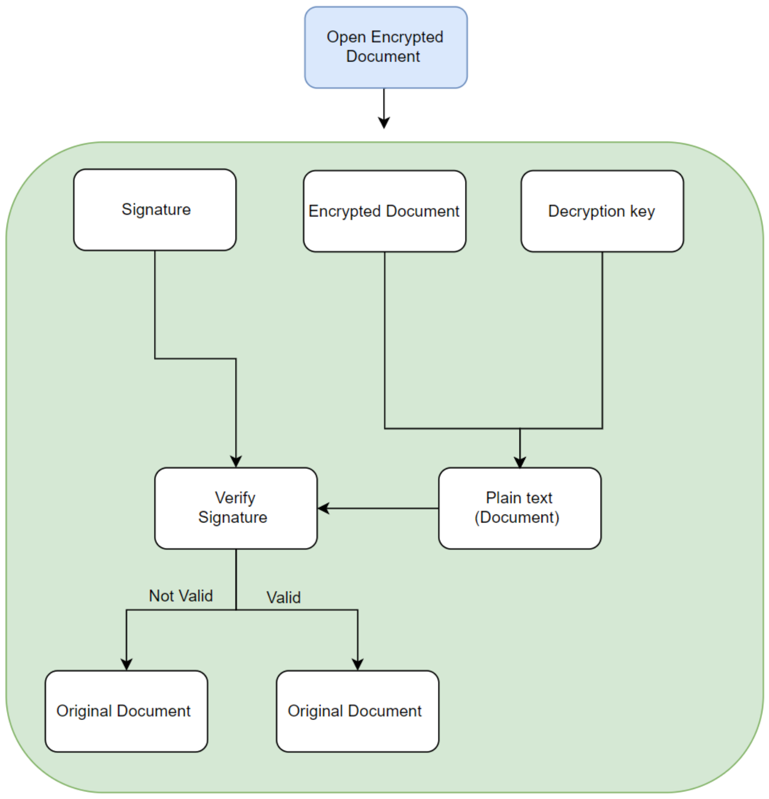


Figure 6. System flow of opening encrypted documents.

The diagram in Figure 6 shows the flow of opening encrypted documents.

3.1. Description of the Technical Details

3.1.1. Java Programming Language

Java Spring boot with spring framework used to implement multiple classes along with Maven dependency management to run the application.

3.1.2. Java Cryptography

We focus on the most used cryptographic library, namely the Java Cryptography Architecture (JCA), which offers an extensive variety of cryptographic services, including symmetric and asymmetric encryption, digital signatures and key management.

3.1.3. Swagger User Interface

User interface framework was used to generate an interactive documentation website to read open API to visualize the interface.

3.1.4. Hardware Specifications

The following hardware specifications were chosen carefully to achieve good performance:

1. Processor: Intel (R) Core (TM) i7-8565U CPU @ 1.80–1.99 GHz
2. RAM: 32.0 GB.

3.2. Key Generation

In our proposed system, key generation follows these steps:

I. Initialize the algorithm key with user passcode and salt using a key generator
II. Initialize the key size and iteration count if needed.
III. Generate the secret key or key pairs depending on the algorithm type.
IV. Calculate the generation time.

For generating a key, one of the classes, KeyGenerator or KeyPairGenerator, can be used, relying on the algorithm. KeyGenerator has the ability of a symmetric key generation, whereas KeyPairGenerator has the ability of an asymmetric key generation. To ensure appropriate entropy in the secret key, the hash of this data is changed into a secret. The secret is then used to generate the key by using the methods SecretKeyFactory, PBEKeySpec and SecretKeySpec from the javax.crypto library. Part of the code for key generation is shown below.

```
SecretKeyFactory factory = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA256");
KeySpec spec = new PBEKeySpec(userInput.toCharArray(), salt , 65536,keySize);
SecretKey key = new Secret-KeySpec(factory.geneateSecret(spec).getEncoded(), "AES");
```

3.3. Encryption and Decryption

The following steps are used for encryption and decryption of implemented algorithms in our proposed system (RSA, ElGamal, AES, Blowfish):

I. Declare the cipher by specifying the algorithm name.
II. Initialize the cipher for encryption or decryption by specifying the mode and key.
III. Encrypt or decrypt the file using the doFinal method.
IV. Calculate the encryption or decryption time.

Here, the file is encrypted and decrypted in a single process, as shown below. The transformation and process used in this project were recommended by a Google team [35].

```
Cipher encryptionCipher = Cipher.getInstance("AES/GCM/NoPadding");
encryptionCipher.init(Cipher.ENCRYPT_MODE, key,
new GCMParameterSpec(T_LEN,"AES/GCM/NoPadding".getBytes(), 0 , 12));
byte[] encryptedBytes = encryptionCi-pher.doFinal(messageInBytes);
```

3.4. Digital Signature

Our proposed system will compute the digital signature and verification to guarantee the integrity of the file using SHA-256. The authenticity of the file is revealed if the recomputed signature and the signature with the encrypted file are equal; otherwise, the file has been altered.

The following steps are used for digital signature generation and verification:

I. Produce a key pair, generated by the KeyPairGenerator class. In our case, an RSA key pair was generated with a 2048-bit length.
II. Initialize signature using hash function SHA256.
III. Initiate signature or verify it using the file and key corresponding to the operation.
IV. Return generated signature or verification response.

The file signature generation process is shown below.

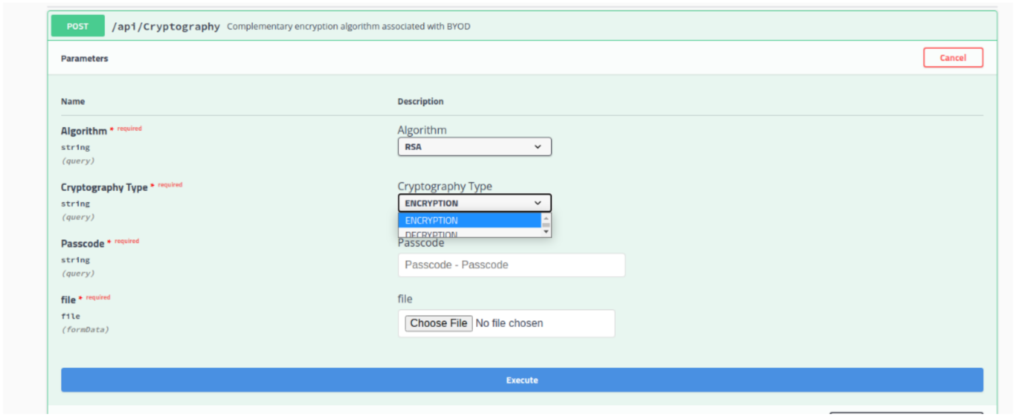
```
Signature privateSignature = Signature.getInstance("SHA256withRSA");
        privateSignature.initSign(privateKey);
        privateSignature.update(plainText.getBytes(UTF_8));
        byte[] signature = privateSignature.sign();
        return Base64.getEncoder().encodeToString(signature);
```

4. Results

The proposed BYOD-based complementary encryption system provides the following features:

1. Encrypting and decrypting files using one of the algorithms (RSA, ElGamal, AES and Blowfish) based on the company digital certificate
2. Generating digital signatures and verifying signatures by using the RSA algorithm to ensure the data's integrity.
3. Including user passcodes in the generation of keys for symmetric and asymmetric algorithms.
4. Using the Swagger UI interface for easy interaction with the API for the selection of documents, cryptography processes, algorithms and user passcodes to be encrypted/decrypted and signed/verified.

Figure 7 shows the display API in the Swagger UI that provides the organization's users with various selections to upload their files to be encrypted after selecting encryption. The algorithm then clicks on the 'Execute' button to generate the file that includes the cipher text and the digital signature for that given file.



POST /api/Cryptography Complementary encryption algorithm associated with BYOD

Parameters

Name	Description
Algorithm * required string (query)	Algorithm RSA
Cryptography Type * required string (query)	Cryptography Type ENCRYPTION ENCRYPTION DECRYPTION
Passcode * required string (query)	Passcode Passcode - Passcode
file * required file (formData)	file Choose File No file chosen

Figure 7. Complementary encryption algorithm Swagger UI.

4.1. Result Analysis

AES and Blowfish were tested using different key sizes of 128, 192 and 256 bits, whereas 512, 1024, 2048 and 3072 bits were used for the RSA and the ElGamal algorithms for encryption and decryption. For signature generation and verification, 2048-bit RSA was used. Files of different types and sizes were tested, and the time taken for each operation was recorded in milliseconds.

4.1.1. Encryption

Different file sizes were encrypted using the four algorithms (RSA, ElGamal, AES and Blowfish) with different key sizes and RSA digital signatures. The encryption time for each algorithm was recorded and tabulated (Tables 2 and 3).

Table 2. Encryption data for RSA and ElGamal algorithms with signature.

Key Size	File Size (KB)	RSA with Signature Time (ms)	ElGamal with Signature Time (ms)
512	10	43	431
512	15	54	687
512	20	75	840
1024	10	64	895
1024	15	87	1967

Table 2. *Cont.*

Key Size	File Size (KB)	RSA with Signature Time (ms)	ElGamal with Signature Time (ms)
1024	20	108	3012
2048	10	89	2960
2048	15	192	3898
2048	20	254	4976
3072	10	132	4590
3072	15	221	6321
3072	20	310	8150

Table 3. Encryption data for Blowfish and AES algorithms with signature.

Key Size	File Size (KB)	Blowfish with Signature Time (ms)	AES with Signature Time (ms)
128	10	0.5	0.7
128	15	0.9	1
128	20	1	2
192	10	1	2
192	15	2	3
192	20	4	5
256	10	6	7
256	15	6.5	8
256	20	8	11

The two figures and tables illustrate the encryption execution times for different encryption algorithms.

Table 2 and Figure 8 show that the ElGamal algorithm execution time rate was constantly higher than RSA during encryption for different file sizes and key lengths. The ElGamal execution time was under 450 ms with a 512-bit key size and 10 KB file size, which then doubled to around 900 ms with a 1024-bit key size. With a 3072-bit key size, the time reached a high of almost 4600 ms, almost 10 times that of the execution time with a 512-bit key size. The RSA execution time rate showed a similar trend but was between 10 to 30 times less than that of ElGamal with all key sizes.

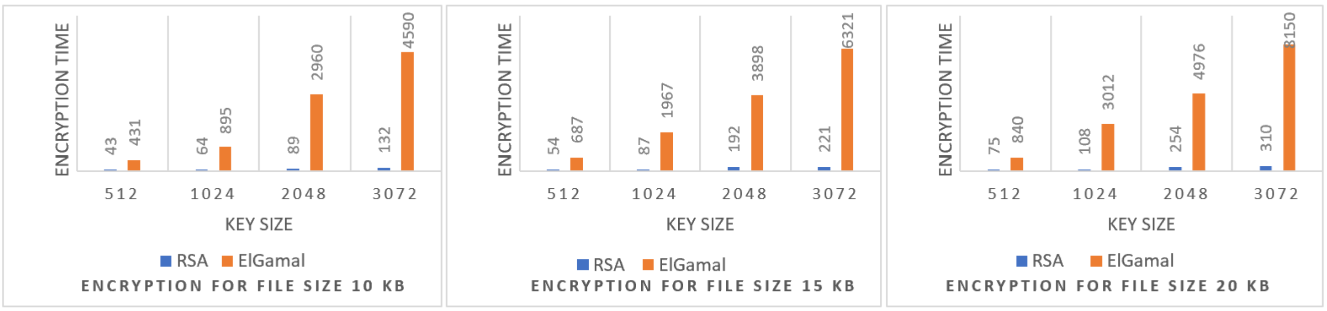


Figure 8. Graphical representation of different file sizes with RSA and ElGamal encryption time.

Table 3 and Figure 9 show a slight difference between Blowfish and AES encryption times, but Blowfish consumes less time than AES. There is a direct relationship between increasing key length and file size on performance. Blowfish execution time started at 0.5 ms with a 128-bit key size and 10 KB file size, which then doubled to 1 millisecond with a 192-bit key size. Then, it showed an increase and reached around 6 ms with a 256-bit key size. The AES execution time rate showed a similar trend but was between 1 to 1.3 times higher than that of Blowfish with all key sizes.

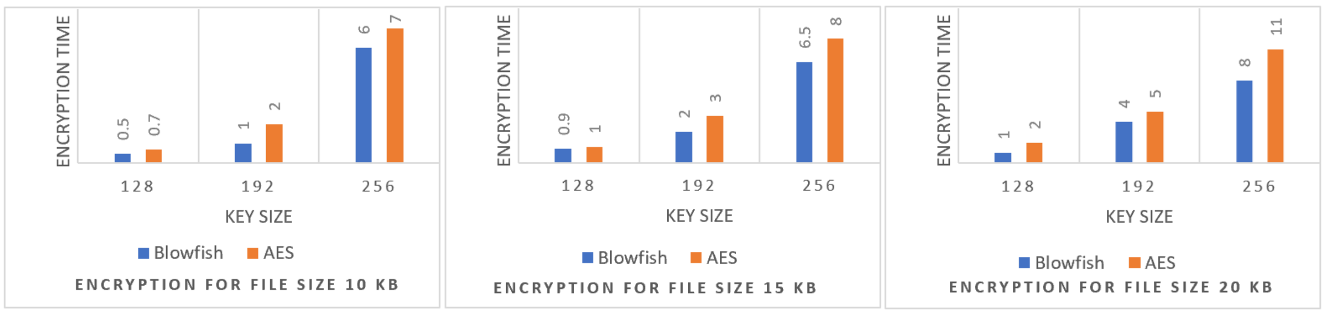


Figure 9. Graphical representation of different file sizes with Blowfish and AES encryption time.

4.1.2. Decryption

The exact file sizes encrypted in Tables 2 and 3 were decrypted. The decryption time for each algorithm was recorded and tabulated (Tables 4 and 5).

Table 4. Decryption data for RSA and ElGamal algorithms with verification.

Key Size	File Size (KB)	RSA with Verification Time (ms)	ElGamal with Verification Time (ms)
512	10	406	197
512	15	734	287
512	20	1269	360
1024	10	1184	357
1024	15	2207	461
1024	20	3032	593
2048	10	2711	549
2048	15	4870	821
2048	20	6589	1047
3072	10	5930	997
3072	15	7842	2130
3072	20	9771	3270

Table 5. Decryption data for Blowfish and AES algorithms with verification.

Key Size	File Size (KB)	Blowfish with Verification Time (ms)	AES with Verification Time (ms)
128	10	0.3	0.5
128	15	0.5	0.6
128	20	0.7	1
192	10	1	1
192	15	2	3
192	20	4	4
256	10	4	6
256	15	5	6
256	20	7	9

The two figures and tables illustrate the decryption execution times for different algorithms.

Table 4 and Figure 10 show that ElGamal consumes less time than RSA during decryption for different key lengths and file sizes. The RSA rate was around 400 ms with a 512-bit key size and a file size of 10 KB, which then increased to more than double to reach around 1200 ms with a 1024-bit key size. This then ended by reaching around 6000 ms with a 3072-bit key size. The ElGamal execution time rate showed a similar trend but was between 2 to 6 times less than the RSA rate with all key sizes.

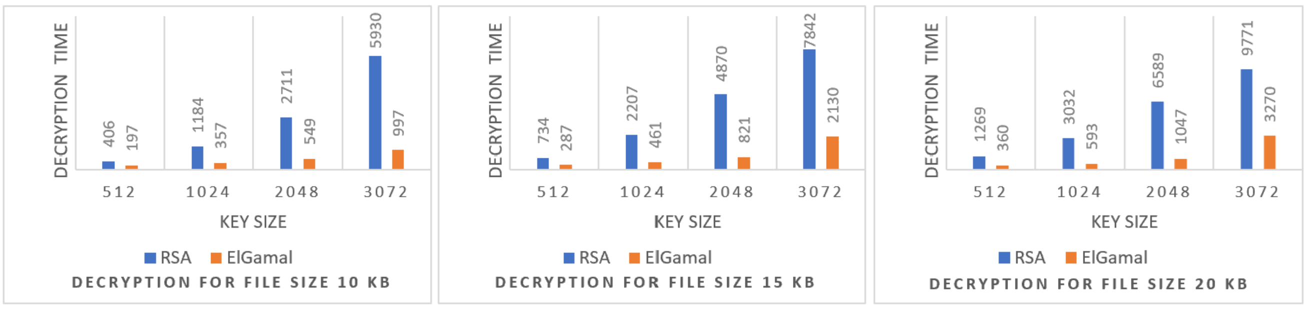


Figure 10. Graphical representation of different file sizes with RSA and ElGamal decryption time.

Table 5 and Figure 11 reveal that AES requires more time than Blowfish. A direct relationship is found between increasing key size and file size on performance. AES execution time was around 0.5 ms with a 128-bit key size and a 10 KB file size, which then doubled to reach 1 millisecond with a 192-bit key size. With a 256-bit key size, the time had reached a high of almost 6 ms, almost 12 times that of the execution time with a 128-bit key size. The Blowfish execution time rate showed a similar trend but was between 1 to 1.6 times less than the AES rate with all key sizes.

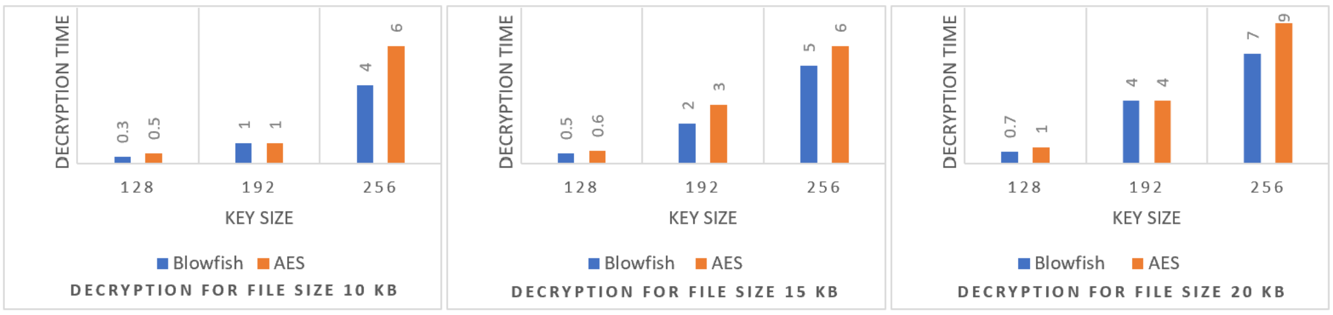


Figure 11. Graphical representation of different file sizes with Blowfish and AES decryption time.

4.1.3. Key Generation

The time taken for RSA, ElGamal, AES and Blowfish to generate a key was captured and recorded (Tables 6 and 7).

Table 6. Key generation data for RSA and ElGamal.

Key Size	User Input Size (Bytes)	RSA Time (ms)	ElGamal Time (ms)
512	10	56	43
512	15	67	51
512	20	88	76
1024	10	142	126
1024	15	180	167
1024	20	267	173
2048	10	1306	428
2048	15	1381	643
2048	20	1407	780
3072	10	2866	1983
3072	15	3124	2432
3072	20	3507	2974

Table 7. Key generation data for Blowfish and AES.

Key Size	User Input Size (Bytes)	AES Time (ms)	Blowfish Time (ms)
128	10	151	143
128	15	167	158
128	20	183	176
192	10	194	187
192	15	229	210
192	20	246	231
256	10	337	297
256	15	389	327
256	20	415	386

The two tables illustrate the key generation execution times for different algorithms.

Table 6 shows a slight difference between ElGamal and RSA algorithms in the key generation process. Here, ElGamal consumes less time than RSA. ElGamal spends around 45 ms in generating a 512-bit key size and 2000 ms to generate a 3072-bit key size, around 40 times that of the execution time with a 512-bit key size. The RSA execution time rate showed a similar trend but with a difference of around 30% greater than that of ElGamal with all key sizes.

Table 7 reveals that AES spends more time generating keys than Blowfish. There is a direct relationship between increasing key size and user passcode size on the key generation time. AES takes around 150 ms to generate a 128-bit key size, and this more than doubles when generating a 256-bit key size. Blowfish's execution time rate showed a similar trend but with a difference of around 8% less than that of AES with all key sizes.

5. Discussion

A comparative analysis was done of the four encryption algorithms (RSA, ElGamal, Blowfish and AES) based on the changeable variables of each algorithm to enhance organization security in BYOD policies. The experimental outcomes in the tables show from the analysis of asymmetric algorithms that encryption and decryption times depend on the key and file sizes, and key generation times depend on the key and user passcode sizes. As the key and file sizes or user passcode sizes increase, the RSA and ElGamal algorithms' encryption and decryption key generation times significantly increase. RSA with signatures has a better execution time than ElGamal with signatures in the encryption processes, while ElGamal with verification has a better execution time than RSA with verification during the decryption process and key generation.

By contrast, for symmetric algorithms, the encryption and decryption times for Blowfish and AES algorithms remain relatively constant or have small differences where Blowfish performs better than AES. Key generation time will depend on the key and user passcode sizes, the key generation times remarkably increase, when we process large key sizes and passcode.

The encryption and decryption times may increase with larger file sizes and key sizes. Therefore, the average encryption, decryption and key generation times appear reasonable and should not significantly impact the performance of employees' devices.

Findings and Comparison with Existing Work

In this section, a comparative analysis and a discussion of the implemented encryption algorithms with previous research have been demonstrated in terms of performance. The RSA, ElGamal, AES and Blowfish results were compared with those of Adeniyi et al. [29], Emmanuel et al. [12], Rouaf and Yousif [13], Abay [14] and Ali et al. [20].

Tables 8 and 9 show a time comparison of our suggested algorithm with previous studies implementing similar techniques, where the time taken in our proposed system seems to be better in terms of performance.

Table 8. RSA and ElGamal comparison.

Technique	File Size (KB)	Time Taken	
		Encryption	**Decryption**
RSA [29]	10	95 ms	3428 ms
RSA [12]		372 ms	3819 ms
RSA [13]		2747 ms	-
Complementary encryption algorithm (RSA)		89 ms	2711 ms
ElGamal [29]		3520 ms	637 ms
Complementary encryption algorithm (ElGamal)		2960 ms	549 ms

Table 9. AES and Blowfish comparison.

Technique	File Size (KB)	Time Taken	
		Encryption	**Decryption**
AES [14]	10	11 ms	12 ms
AES [20]		203 ms	203 ms
AES [13]		18 ms	-
Complementary encryption algorithm (AES)		7 ms	6 ms
Blowfish [14]		12 ms	13 ms
Complementary encryption algorithm (Blowfish)		6 ms	4 ms

Table 8 shows that our RSA and ElGamal execution time was between 0.9 to 30 times lower than previous studies' algorithm rates in the encryption and decryption process. Table 9 demonstrates that AES and Blowfish execution time was between 1.5 to 29 times lower than previous studies' algorithm rates in terms of encryption and decryption.

The analysis aligns with the literature findings, highlighting the performance and security of these algorithms. For example, in a comparative study of RSA and ElGamal algorithms, Emmanuel et al. [12] concluded that they are both secure cryptographic algorithms. Still, their efficiency may vary depending on the application and file format. RSA may be faster for audio files [12]. Adeniyi et al. [29] inferred that RSA is faster in encrypting messages and verifying digital signatures, while ElGamal is faster in decrypting messages and generating digital signatures. Blowfish is about four times faster than AES, which is faster than RSA (the slowest algorithm) [16]. Abay [14] also revealed that AES has better performance with higher throughput than Blowfish. When the throughput value increases, the power consumption of the encryption method decreases. Overall, the literature shows that AES has a good balance of security and performance. Abay's [14] findings revealed that AES has better performance, with high throughput and lower power consumption, compared with other algorithms, suggesting that AES is more efficient for encryption and decryption operations, especially in applications that require high throughput. These findings align with the results and reviewed work, which revealed that AES had a good balance of security and performance and could be the best choice for the organization's BYOD infrastructure. However, further analysis and testing are required to determine the extent to which AES ensures the security of organizational data based on the specific needs and requirements.

6. Conclusions

This paper proposes a BYOD-based data secure storage system with encryption and decryption features using AES, Blowfish, ElGamal and RSA algorithms with signature generation and verification, an API with a Swagger UI interface and the generation of private and public keys with user passcode, applied on downloaded files on employees' devices to ensure organizational data security. The paper compares four encryption algorithms and concludes that AES has the best balance of security and performance for high-throughput applications in an organization's BYOD infrastructure. Further testing is recommended to assess the system's security based on specific needs. This proposed system addresses BYOD challenges and provides an efficient and secure data storage solution for organizations. The study may serve as a helpful reference for organizations implementing secure data storage techniques.

Author Contributions: Conceptualization, methodology, validation, formal analysis, investigation, and visualization, M.R.A. and S.M.F. Software and writing—original draft preparation, M.R.A. Writing review and editing, S.M.F. All authors have read and agreed to the published version of the manuscript.

Funding: The authors would like to acknowledge the support of Prince Sultan University for paying the Article Processing Charges (APC) for this publication.

Data Availability Statement: Not applicable.

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

References

1. Perera, S.; Jin, X.; Maurushat, A.; Opoku, D.-G.J. Factors affecting reputational damage to organisations due to cyberattacks. *Informatics* **2022**, *9*, 28. [CrossRef]
2. Bhusal, C.S. Systematic review on social engineering: Hacking by manipulating humans. *J. Inf. Secur.* **2021**, *12*, 104–114. [CrossRef]
3. Palanisamy, R.; Norman, A.A.; Mat Kiah, M.L. BYOD Policy Compliance: Risks and Strategies in Organizations. *J. Comput. Inf. Syst.* **2022**, *62*, 61–72. [CrossRef]
4. Hertel, G.; Stone, D.L.; Johnson, R.D. *The Wiley Blackwell Handbook of the Psychology of the Internet at Work*; John Wiley & Sons: Hoboken, NJ, USA, 2017.
5. Business Wire. Bitglass 2020 BYOD Report: Increased Remote Work Drives BYOD, But Security Is Not Keeping Pace. Available online: https://www.businesswire.com/news/home/20200708005267/en/Bitglass-2020-BYOD-Report-Increased-Remote-Work-Drives-BYOD-but-Security-is-Not-Keeping-Pace (accessed on 9 March 2023).
6. Scrubbed. LinkedIn Data Leak—What We Can Do about It. Available online: https://scrubbed.net/blog/linkedin-data-leak-what-we-can-do-about-it/ (accessed on 8 March 2023).
7. Turban, E.; Pollard, C.; Wood, G. *Information Technology for Management: On-Demand Strategies for Performance, Growth and Sustainability*; John Wiley & Sons: Hoboken, NJ, USA, 2018.
8. Bahaddad, A.A.; Almarhabi, K.A.; Alghamdi, A.M. Factors Affecting Information Security and the Implementation of Bring Your Own Device (BYOD) Programmes in the Kingdom of Saudi Arabia (KSA). *Appl. Sci.* **2022**, *12*, 2707. [CrossRef]
9. Ntwari, R.; Habinka, A.E.; Kaggwa, F. BYOD systematic literature review: A layered approach. *Eur. J. Technol.* **2022**, *6*, 69–85. [CrossRef]
10. Shrestha, P.; Thakur, R.N. Study on Security and Privacy Related Issues Associated with BYOD Policy in Organizations in Nepal. *LBEF Res. J. Sci. Technol. Manag.* **2019**, *1*, 41–62.
11. Maglaras, L.; Almomani, I. Digitization of healthcare sector: A study on privacy and security concerns. *Korean Inst. Commun. Inf. Sci.* 2023, *in press*. [CrossRef]
12. Emmanuel, A.A.; Marion, A.O.; Aderemi, O.; Olugbara, O.O. Computational complexity of RSA and ElGamal cryptographic algorithms on video data. *J. Theor. Appl. Inf. Technol.* **2022**, *100*, 5437–5445. [CrossRef]
13. Rouaf, M.T.; Yousif, A. Performance Evaluation of Encryption Algorithms in Mobile Devices. In Proceedings of the 2020 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE), Khartoum, Sudan, 26 February–1 March 2021. [CrossRef]
14. Abay, M.M. Performance Analysis of Blowfish, IDEA and AES Encryption Algorithms. *Int. J. Res. Anal. Rev.* **2020**, *7*, 668–678.
15. Alenezi, M.; Usama, M.; Almustafa, K.; Iqbal, W.; Raza, M.A.; Khan, T. An efficient, secure, and queryable encryption for nosql-based databases hosted on untrusted cloud environments. *Int. J. Inf. Secur. Priv.* **2019**, *13*, 14–31. [CrossRef]
16. Commey, D.; Griffith, S.; Dzisi, J. Performance comparison of 3DES, AES, Blowfish and RSA for Dataset Classification and Encryption in Cloud Data Storage. *Int. J. Comput. Appl.* **2020**, *177*, 17–22. [CrossRef]
17. Oleiwi, Z.C.; Alawsi, W.A.; Alisawi, W.C.; Alfoudi, A.S.; Alfarhani, L.H. Overview and Performance Analysis of Encryption Algorithms. *J. Phys. Conf. Ser.* **2020**, *1664*, 012051. [CrossRef]
18. Rasool, M.-U.; Iftikhar, S.; Saba, T.; Al-ghamdi, J.S. Ensuring authentication in cloud computing through homomorphic encryption. *J. Theor. Appl. Inf. Technol.* **2017**, *95*, 3032–3040.
19. Shrestha, P.; Thakur, R.N. Channel state information-based cryptographic key generation for Intelligent Transportation Systems. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 7496–7507. [CrossRef]
20. Ali, K.; Akhtar, F.; Memon, S.A.; Shakeel, A.; Ali, A.; Raheem, A. Performance of cryptographic algorithms based on time complexity. In Proceedings of the 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), Sukkur, Pakistan, 29–30 January 2020. [CrossRef]

21. Tiwari, D.; Singh, A.; Prabhakar, A. Performance Analysis of AES, RSA and Hashing Algorithm Using Web Technology. In *Computing Algorithms with Applications in Engineering*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 413–418. [CrossRef]
22. Javed, Y.; Khan, A.S.; Qahar, A.; Abdullah, J. Preventing Dos Attacks in IOT Using AES. *J. Telecommun. Electron. Comput. Eng.* **2017**, *9*, 55–60.
23. Ribouh, S.; Phan, K.; Malawade, A.V.; Elhillali, Y.; Rivenq, A.; Al Faruque, M.A. A novel secure artificial bee colony with advanced encryption standard technique for biomedical signal processing. *Period. Eng. Nat. Sci.* **2022**, *10*, 288. [CrossRef]
24. Buhari, B.A.; Obiniyi, A.A.; Sunday, K.; Shehu, S. Performance evaluation of symmetric data encryption algorithms: AES and Blowfish. *Saudi J. Eng. Technol.* **2019**, *4*, 407–414. [CrossRef]
25. Kothandan, A. Modified Blowfish Algorithm to Enhance Its Performance and Security. Ph.D. Thesis, National College of Ireland, Dublin, Ireland, 2020.
26. Yousif, S.F. Performance comparison between RSA and El-Gamal algorithms for Speech Data Encryption and decryption. *Diyala J. Eng. Sci.* **2023**, *16*, 123–137. [CrossRef]
27. Ahmed, B.K.A.; Mahdi, R.D.; Mohamed, T.I.; Jaleel, R.A.; Salih, M.A.; Zahra, M.M.A. Secure and efficient data storage operations by using intelligent classification technique and RSA algorithm in IOT-based cloud computing. *Sci. Program.* **2022**, *2022*, 2195646. [CrossRef]
28. Tahat, N.; Shaqboua, R.; Abdallah, E.E.; Bsoul, M.; Shatanawi, W. A New Digital Signature Scheme with Message Recovery Using Hybrid Problems. *Int. J. Electr. Comput. Eng.* **2019**, *9*, 3576–3583. [CrossRef]
29. 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. [CrossRef]
30. Kavin, B.; Ganapathy, S. A new digital signature algorithm for ensuring the data integrity in cloud using elliptic curves. *Int. Arab. J. Inf. Technol.* **2021**, *18*, 180–190. [CrossRef]
31. Somsuk, K.; Thakong, M. Authentication system for e-certificate by using RSA's digital signature. *Telecommun. Comput. Electron. Control* **2020**, *18*, 2948. [CrossRef]
32. Imran, O.A.; Yousif, S.F.; Hameed, I.S.; Abed, W.N.A.-D.; Hammid, A.T. Implementation of el-gamal algorithm for speech signals encryption and decryption. *Procedia Comput. Sci.* **2020**, *167*, 1028–1037. [CrossRef]
33. Babu, T.G.; Jayalakshmi, V. Conglomerate energy efficient Elgamal encryption based data aggregation cryptosystems in Wireless Sensor Network. *Int. J. Eng.* **2022**, *35*, 417–424. [CrossRef]
34. Kasodhan, R.; Gupta, N. A new approach of digital signature verification based on BioGamal algorithm. In Proceedings of the 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 27–29 March 2019. [CrossRef]
35. Oracle. Class Cipher. Available online: https://docs.oracle.com/javase/7/docs/api/javax/crypto/Cipher.html (accessed on 10 January 2023).