algorithms

MDPI

Article

Computational Complexity of Modified Blowfish Cryptographic Algorithm on Video Data

Abidemi Emmanuel Adeniyi [1], Sanjay Misra [2,*], Eniola Daniel [3] and Anthony Bokolo, Jr. [4]

[1] Department of Computer Sciences, Precious Cornerstone University, Ibadan 200223, Nigeria
[2] Department of Computer Science and Communication, Østfold University College, 1757 Halden, Norway
[3] Department of Computer Science, Landmark University, Omu-Aran 251103, Nigeria
[4] Department of Applied Data Sciences, Institute for Energy Technology, 1777 Halden, Norway
* Correspondence: sanjay.misra@hiof.no

Abstract: Background: The technological revolution has allowed users to exchange data and information in various fields, and this is one of the most prevalent uses of computer technologies. However, in a world where third parties are capable of collecting, stealing, and destroying information without authorization, cryptography remains the primary tool that assists users in keeping their information secure using various techniques. Blowfish is an encryption process that is modest, protected, and proficient, with the size of the message and the key size affecting its performance. Aim: the goal of this study is to design a modified Blowfish algorithm by changing the structure of the F function to encrypt and decrypt video data. After which, the performance of the normal and modified Blowfish algorithm will be obtained in terms of time complexity and the avalanche effect. Methods: To compare the encryption time and security, the modified Blowfish algorithm will use only two S-boxes in the F function instead of the four used in Blowfish. Encryption and decryption times were calculated to compare Blowfish to the modified Blowfish algorithm, with the findings indicating that the modified Blowfish algorithm performs better. Results: The Avalanche Effect results reveal that normal Blowfish has a higher security level for all categories of video file size than the modified Blowfish algorithm, with 50.7176% for normal Blowfish and 43.3398% for the modified Blowfish algorithm of 187 kb; hence, it is preferable to secure data and programs that demand a high level of security with Blowfish. Conclusions: From the experimental results, the modified Blowfish algorithm performs faster than normal Blowfish in terms of time complexity with an average execution time of 250.0 ms for normal Blowfish and 248.4 ms for the modified Blowfish algorithm. Therefore, it can be concluded that the modified Blowfish algorithm using the F-structure is time-efficient while normal Blowfish is better in terms of security.

Keywords: time complexity; cryptographic algorithms; modified blowfish algorithm; blowfish algorithm; security

Citation: Adeniyi, A.E.; Misra, S.; Daniel, E.; Bokolo, A., Jr. Computational Complexity of Modified Blowfish Cryptographic Algorithm on Video Data. *Algorithms* **2022**, *15*, 373. https://doi.org/10.3390/a15100373

Academic Editor: Vangelis Th. Paschos

Received: 22 August 2022
Accepted: 7 October 2022
Published: 10 October 2022

1. Introduction

Data security has become increasingly important in today's world, prompting a variety of methods to circumvent it [1]. With the introduction of the internet, security became a key concern, and a better knowledge of the growth of security technologies may be gained by studying the history of security. The very nature of the internet has resulted in the emergence of various security threats. When the internet's mechanism is changed, it can minimize the number of possible attacks that can be sent across the network. Many modern systems use decryption and encryption technologies to protect themselves from the internet. The internet is used to transfer and store the majority of data in the modern world. As a result, it is critical to protect data from unwanted access. New types of security mechanisms are being created to protect data as the old ones are destroyed by various types of unauthorized attacks. The key pillars of data and information security are confidentiality,

integrity, and availability. This is a security paradigm and guides companies to keep their secret data secure from illegal access and data exfiltration, often known as the CIA trio [2]. Privacy protection prevents unauthorized employees from gaining access to data; integrity ensures that the information is correct, complete, and dependable; and availability ensures that data are both available and accessible to meet specified requirements.

Due to the obvious rapid development of diverse multimedia technology, a large number of audiovisual data are created and conveyed in the medical, advertisement, and military domains, which may contain sensitive information that should not be accessed by normal users or should only be partially disclosed to them [3–5]. Privacy and security have grown in importance while several encryption techniques have been employed to enable secure video transmission [6]. While a great number of multimedia encryption systems have been presented in the literature, and some have been implemented in real products, cryptanalytic work has revealed that most of the proposed multimedia encryption schemes have security issues and other flaws.

Cryptography is the science of securing data, used to solve important security issues concerning confidentiality, integrity, and authentication [7,8]. Its goal is to safely send sensitive information via vulnerable networks such as the internet [9]. To prevent others from accessing an encrypted message, the sender only discloses the decoding process to the intended recipients. Modern cryptography is primarily reliant on mathematical theory and computer science skills; cryptographic methods are based on computational hardness assumptions, making it difficult for any adversary to crack them in practice [10,11]. Although breaking into such a well-designed system is theoretically feasible, it is difficult to do so in actual practice. Cryptography is widely acknowledged as one of the most critical components of any organization's security policy, and it is widely accepted as the industry standard for information security, trust, resource management, and electronic financial transactions [12,13]. In essence, cryptographic algorithms/techniques can be divided into symmetric key cryptographic (using a single key for encoding and decoding) and asymmetric key cryptographic (using a pair of keys for both encoding and decoding of the message) algorithms/techniques.

Video encryption is a method of digitally disguising videos to prevent them from being intercepted and seen by unauthorized parties [14]. This method requires the encryption of videos using software and hardware encoding to protect their content. Without first decoding them, no one can access the encrypted videos. In December 1993, Bruce Schneier designed Blowfish, a symmetric cryptosystem, as a high-capacity algorithm that can be freely used as an alternative to prior encryption schemes. It is one of the most extensively used symmetric cryptographic algorithms for data security. Blowfish is regarded as one of the quickest and easiest symmetric algorithms since it is a generally accessible and license-free cryptography technique [15]. Therefore, this study intends to explore the secured method of the Blowfish algorithm by modifying the structure of the F-function to produce a modified version of the Blowfish algorithm. Both the normal Blowfish and the modified Blowfish algorithm will be used to encrypt video data and compare their performance in terms of time complexity, throughput, and avalanche effect to determine which of the algorithms performs better in terms of time and security usage. This study will spur researchers to design high-impact and more secure computing-encryption algorithms.

Nie, Song, and Zhi [16] investigated the security, DES, and Blowfish cryptography methods' strength and speed requirements, which are extensively used for network data encryption. According to their experiments, the Blowfish algorithm is quicker than DES while consuming about the same amount of electricity. They demonstrated that the Blowfish encryption technique is better suited to the security of wireless network applications.

Tahseen and Habeeb [17] proposed a novel method of generating random numbers from images. Read the image pixel by pixel, then choose any two colors at random for the precise spot. To mix specific colors, use XOR; then, choose the key sizes. This key is employed in the decryption of plaintext. The Blowfish scheme utilizes this key generation

process. Finally, they state that under the symmetrical scheme, this kind of key production is sufficient for shorter keys.

Agrawal and Mishra [18] improved the safety of the Blowfish method while reducing the time it takes to encrypt and decrypt data. Choose a number between 0 and 65,535 at random. Set the value of the signal to zero. Transform an arbitrary integer to 16-bit code and look for places with 0 entries; set the flag value to one and zero otherwise. If indeed the signal is present as 1, the F-function will not execute, but it will work if the signal is set to 0. In every round, a randomly chosen number is generated, resulting in a distinction in the implementation of the F function. They discovered that when compared to the original Blowfish algorithm, the encryption and decryption times are reduced.

Geethavani, Prasad, and Roopa [19] proposed employing a discrete wavelet transform to secure the data transfer of audio signals. They developed a new hybrid technique for sending messages in a highly secure manner by combining cryptography and steganography. The information is encrypted using the Blowfish technique, and the secret message is then embedded in an audio file using the discrete wavelet transform. The authors assert that their method is efficient at hiding information in audio files such that data may be sent to their intended location without being tampered with.

Dulla, Gerardo, and Medina [20] enhanced data security by modifying the Blowfish algorithm. They created a software program to encrypt files. The encryption algorithm is implemented when a file is separated into several sections based on the user's specifications. They changed the F-function in the Blowfish algorithm to improve the software's performance. The F-function is made up of four S-boxes (S1, S2, S3, and S4). F(X) = ((S1 + S2 mod 232) XOR S3 + S4 mod 232) XOR S3 + S4 mod 232). They changed F(X) = ((S1 XOR S2 mod 232) + (S3 XOR S4 mod 232) to F(X) = ((S1 XOR S2 mod 232) + (S3 XOR S4 mod 232)). They demonstrated that the modified Blowfish algorithm takes 14 percent less time to execute than the original Blowfish method in an experiment.

The Blowfish technique was updated by Christina and Joe Irudayaraj [21] in such a way that the larger the key lengths, the stronger the key; however, the encryption process periods are considerable. To address this issue in the proposed technique, decreasing two S-boxes will increase performance and information security. When compared to the original techniques, the primary advantage of the modified Blowfish algorithm is that the processing time is reduced to 0.2 ms and the throughput is increased to 0.24 bytes/ms. The optimized Blowfish encryption technique's cryptanalysis was explored, and the algorithm was tested with several data types such as text files, audio files, and video files.

Prasetyo, Purwanto, and Darlis [22] show the Blowfish method's effectiveness by utilizing the overall time complexity, avalanche impact, and throughput as factors in various testing situations. The Blowfish algorithm was written in VHDL and implemented on an FPGA. The results show that lowering the round of Feistel ciphers reduces the total encryption time, increases throughput, and has no major impact on the avalanche effect.

Prasad, Anusha, Jyothi, and Dileepkumar [23] proposed a novel data encryption method based on the Blowfish algorithm. They designed and implemented a new strategy based on the benefits of the Blowfish algorithm to improve the previous algorithm's performance in terms of factors such as the throughput and computational cost. The most noteworthy feature of this improved Blowfish cryptographic algorithm is that the encrypted message generated each time is unique. This is because each time it is run, a new random variable is generated, resulting in a difference in the F function's application over each round. The security element of the Blowfish method will be considerably improved as a result of the different cipher text produced for the same input.

Manju and Neema [24] investigated IoT security issues and mechanisms. According to an analysis of various IoT security issues, the majority of them arise in the insecure passage that links distinct IoT networks, as well as IoT and WSN gateways. Cryptographic techniques can be used at the network level, where data communication protection is provided. The time complexity, memory utilization, throughput, energy usage, and privacy are all

factors to consider; accordingly, Blowfish was determined to be the best cryptographic algorithm, making it suitable for IoT.

Ali and Abead [25] suggested an enhanced Blowfish technique based on five S-boxes for picture scrambling. The Blowfish algorithm's security level was enhanced by raising the difficulty of cracking the original message, resulting in protection from unauthorized assault. This approach uses grayscale images of various sizes to implement a 64-bit block encrypted with a symmetrical variable-length key. In the proposed technique, both encryption and decryption occurred in the Feistel function in round one; an additional key (KEY2) of a one-byte length was utilized rather than a single key in the encryption operation. Furthermore, the suggested modified Blowfish method uses five Sboxes instead of four; the additional key (KEY2) is chosen at random from the additional Sbox5, while the fifth Sbox is constructed in GF(28), and it is variable to increase the proposed algorithm's complexity.

Reyes, Festijo, and Medina [26] proposed a revised Blowfish technique that can handle 128-bit blocks. Although being recognized as an unbreakable method, the Blowfish algorithm has been unsuitable for some applications due to limited block length compatibility. A unique revised version of the Blowfish encryption system is developed in this work to accommodate a 128-bit block size input utilizing a flexible choice encryption system and decreased encrypted function operations through randomly selected rounds.

Corpuz, Gerardo, and Medina [27] used a revised Blowfish algorithm technique for information security in cloud computing. Cloud computing is a common issue associated with data and information security. The computational complexity was tested using a modified Blowfish Technique employing the Shuffle Strategy.

Shetty, Anusha, and Hegde [28] improved and compared the Blowfish method in terms of encryption quality, correlation coefficients, key sensitivity testing, and output file size. By combining the XOR and addition utilized in the original technique, the 'f' function was updated. Four different scenarios were generated and assessed. The findings of all the tests conducted on these scenarios all pointed to the same conclusion: the updated algorithm's security in various cases makes the original Blowfish method more compact and secure than before.

Kumar and Karthikeyan [29] investigated the efficacy of Blowfish and AES algorithms. Their studies were carried out on a Pentium P4 2.4 GHz processor with 2 GB RAM. The trials were repeated numerous times to ensure that the findings were consistent and valid for comparing the various methods. To evaluate the performance of the encryption methods, the study employed the encryption time, decryption time, CPU process time, CPU clock cycles, and battery metrics. In virtually all the test situations, the results reveal that Blowfish outperforms AES. There was no discernible difference between base64 and hexadecimal encoding schemes. It was discovered that Blowfish is good for text-based encryption, whilst AES is better for picture encryption. The study also discovered that changing the key size of the AES algorithm affects its performance. Overall, it was determined that AES can be utilized in situations requiring high security, whereas Blowfish can be utilized in situations requiring high-performance. A summary of the important works is given in Table 1.

This study consists of five sections. The next section describes the materials and methods used in the study. Section 3 presents the results. Section 4 presents a discussion, while Section 5 concludes the study.

Table 1. Summary of Literature reviews.

S/N	Authors	Algorithms	Parameter	Outcome	Gap
1	Nie, Song, and Zhi [16]	DES and Blowfish	Wireless Sensor Network Application (WSN)	Blowfish outperforms DES in terms of speed	The algorithms were tested on small WSN data.
2	Tahseen and Habeeb [17]	Blowfish using Random Key Generator	Image data	The Random Key Generator was used to generate Blowfish algorithm encryption key	The study only tests the enhanced Blowfish on image data

Table 1. *Cont.*

S/N	Authors	Algorithms	Parameter	Outcome	Gap
3	Agrawal and Mishra [18]	Modified Blowfish	Text Data	The study captures the runtime of encrypting the plaintext.	The study did not specify how the algorithm was modified
4	Geethavani, Prasad, and Roopa [19]	Blowfish and Steganography	Text Data and Audio File	The study encrypts text data using Blowfish and embeds the cipher text in an audio file using discrete wavelet transform	The method used is secure; however, it is not time efficient.
5	Manju and Neema [20]	Blowfish Algorithm	Text Data on IoT devices	The algorithm seems to be better for IoT devices in terms of execution time, memory usage, throughput, power consumption, and security	The algorithm block size and key length were reduced because it was used on devices with limited resources.
6	Ali and Abead [21]	Modified Blowfish	Image Data	The five S-Boxes were modified with multi keys applied to encrypt the image.	The complexity of the modified algorithm was greatly increased.
7	Dulla, Gerardo, and Medina [22]	Blowfish-128 Modified	Text Data	The modifications improved performance and execution time	The complexity and diffusion of the algorithm were increased.
8	Shetty, Anusha, and Hegde [23]	Improved Blowfish	Encryption quality, correlation coefficients, key sensitivity testing, and output file size	The study used XOR to update the F function so as to improve the algorithm.	The study did not specify the type of parameter used for either text, image, or audio data.
9	Kumar and Karthikeyan [29]	Blowfish and AES	Text and Image	Blowfish is better for text data while AES is better for image data.	The experiment was simulated on a system with limited memory space.

2. Materials and Methods

This research compares the Blowfish algorithm with the modified Blowfish algorithm to assess the security, effectiveness, and overall performance of both cryptographic algorithms. The S-Box preparation, sub-key creation, and encryption are the three primary aspects of Blowfish. This research adjusted the structure of the F-function by utilizing fewer S-Boxes to improve the existing Blowfish method. The modified Blowfish encryption algorithm was implemented using the Blowfish library in Python 3.8 version, Flask micro web framework, and JavaScript Programming language. All development, testing, and design processes were implemented on a windows 10 operating system of intel core i5 (7th generation) with processing power of 2.7 GHz CPU with 8 GB RAM.

The study was evaluated in three stages: the time complexity of the Blowfish with respect to video data, the time complexity of the modified Blowfish algorithm, the security performance of the Blowfish algorithm using avalanche effects, and the performance evaluation of Blowfish and the modified Blowfish algorithm when applied to video data. Throughput is another metric that may be used to evaluate Blowfish and the modified Blowfish algorithm's performances.

$$\text{Throughput} = \text{Data (in kb)}/(\text{Process end time} - \text{Process start time}) \quad (1)$$

The final parameter used for the performance evaluation is the avalanche effect of the encryption algorithms. This parameter will be used to test the security levels of Blowfish and modified Blowfish algorithms. The behavior of mathematical functions employed in encryption is described by the "avalanche effect." One of the desirable elements of any encryption technique is the avalanche effect. The cipher text should change drastically if

the plain text or key is changed slightly. This characteristic is termed the avalanche effect. In simple words, it quantifies the impact of a slight change in plain text or the key on the ciphertext. The formula used to obtain the avalanche effect is as follows:

$$\text{Avalanche effect} = (\textit{Average number of flipped bits in chiphertext/Number of the bits in chiphertext}) * 100\% \quad (2)$$

The modified Blowfish encryption algorithm was implemented using the Blowfish library in Python programming language, Flask micro web framework, and JavaScript Programming language. The framework of the proposed modified Blowfish algorithm is displayed in Figure 1.

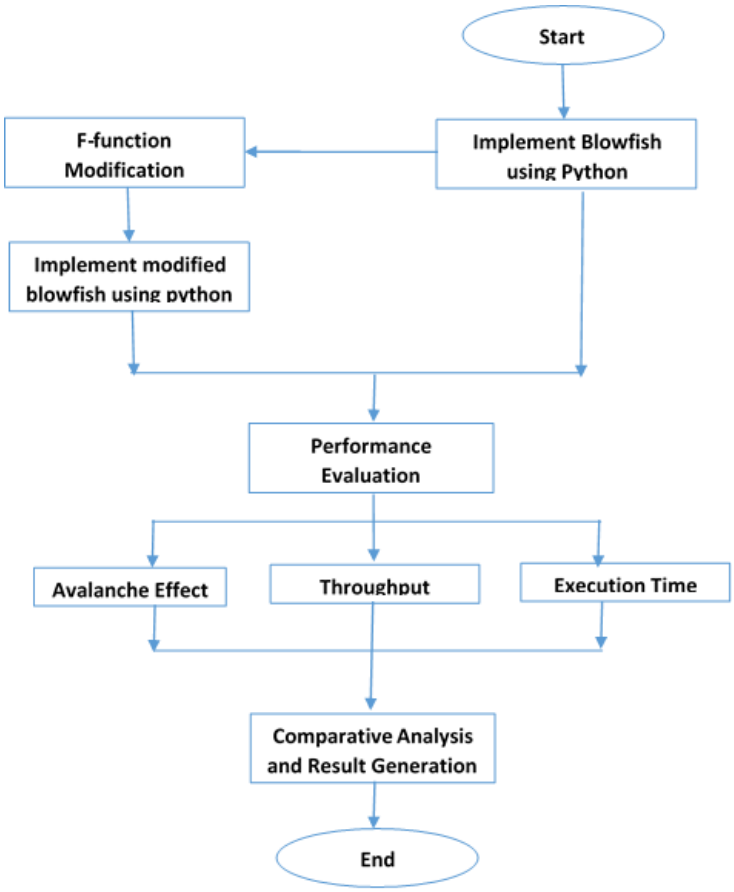

Figure 1. Methodology workflow diagram of the proposed modified algorithm.

2.1. Blowfish Encryption

With the Blowfish encryption algorithm, a 16-round Feistel arrangement is used to encrypt information. A key-subordinate replacement and an information-subordinate replacement occur in each cycle. All of the tasks are 32-bit XORs and augmentations. Four filed exhibit information queries every cycle are the other major activities. Blowfish employs a wide range of techniques. Before any encryption techniques or decrypting, these keys must be pre-registered. The key clusters, often known as the P-exhibit, are made up of 18 32-bit sub-keys: P1, P2...P18.

There are four 32-bit S-boxes with 256 entries each: S1, 0, S1, 1...S1, 255; S2, 0, S2, 1... S2, 255; S3, 0, S3, 1...S3, 255; S4, 0, S4, 1 . . . S4, 255.

The encryption requires a function that iterates the network 16 times (see Algorithm 1). Each round includes a key and data-dependent permutation as well as a key and data-dependent substitution. For 32-bit words, all operations are XORs and additions. For each cycle, four indexing array data retrieval banks are the only additional procedures. The x is

a 64-bit communication-instrumental variable's data. Gap x is split up into two 32-bit parts: xL and xR. The steps involved in the encryption process are as follows:

Algorithm 1. Blowfish F function.

```
Divide x into two 32-bit halves: xL, xR
        For i = 1 to 16:
          xL = XL XOR Pi
                xR = F(XL) XOR xR
                Swap XL and xR
                Swap XL and xR (Undo the last swap.)
                 xR = xR XOR P17
                      xL = xL XOR P18
          Recombine xL and xR
```

For Function F: partition xL into four eight-piece quarters: a, b, c, and dF(xL) = ((SI, a + S2, b mod 232) XOR S3, c) + S4, d mod 232.

Decryption is identical to encryption, with the exception that PI, P2, and P 18 are used as part of the switch configuration. Blowfish executions that require the fastest speeds should unroll the circle and ensure that all subkeys are stored in the cache-store.

2.2. Modified Blowfish Encryption

Blowfish is optimized by changing the structure of the F-function, while the Feistel structure of the Blowfish algorithm remains unchanged. The optimized Blowfish uses two S-boxes instead of the four S-boxes used in Blowfish's F-function.

Pseudo-Code
A. Pseudo-code for F-Function with four S-Boxes (S0, S1, S2, and S3)
1: Divide xL into four eight-bit quarters: a, b, c, and d 2: F(xL) = ((S0,a + S1,b mod 232)^S2,c) + S3,d mod 232
B. Pseudo-code for optimized F function with two S-boxes
1: Divide xL into two sixteen-bit quarters: a, and b. 2: F(xR) = (S0,a^ S1,b)
C. Pseudo-code for Encryption
1: Divide the 64-bit input data into two 32-bit halves (left and right): xL and xR 2: for i = 0 to16 xL is XORed with P[i]. Find F(xL) F(xL) is XORed with xR. Interchange xL and xR. 3: Interchange xL and xR. 4: xR is XORed with P [16]. 5: xL is XORed with P [17]. 6: Combine xL and xR.
D. Pseudo-code for Decryption
1: Divide the 64-bit input data into two 32-bit halves (left and right): xL and xR 2: for i = 17 to1 xL is XORed with P[i]. Find F(xL); F(xL) is XORed with xR. Interchange xL and xR. 3: Interchange xL and xR. 4: xR is XORed with P [1]. 5: xL is XORed with P [0]. 6: Combine xL and xR

3. Results

The system flowchart depicts the system's process flow across various stages. The flowchart essentially introduces the application system as well as the analysis system. Figure 2 shows the flowchart for the proposed modified Blowfish algorithm.

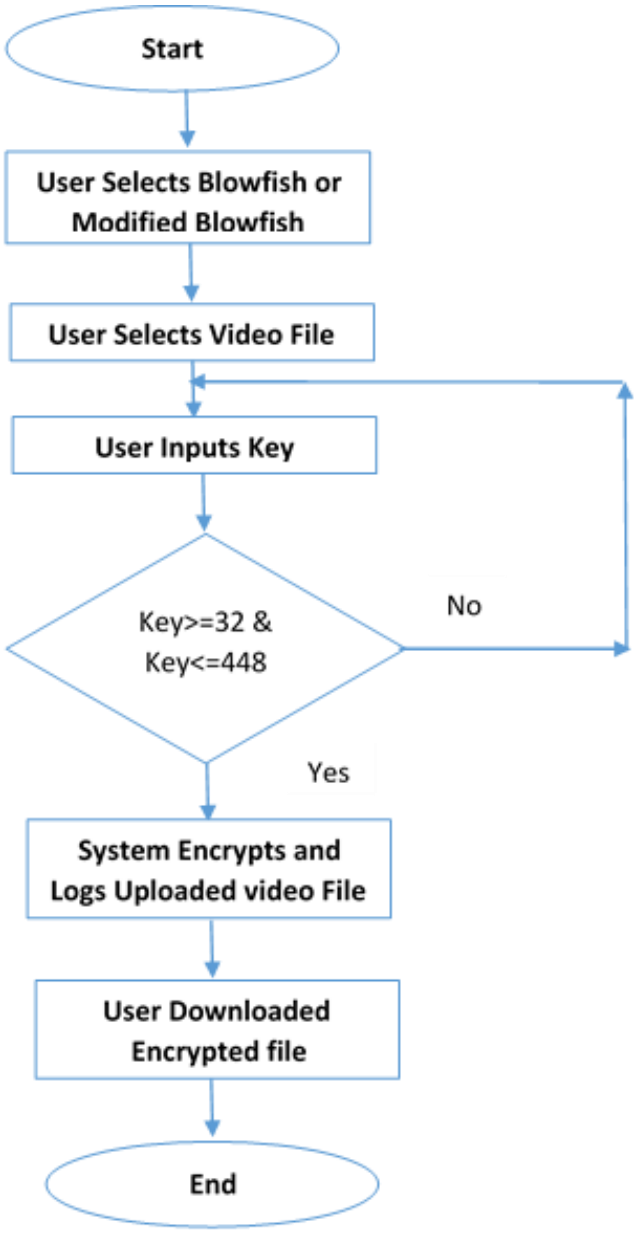

Figure 2. System Flowchart for the proposed modified algorithm.

Application Terminal

This section shows the code and terminal implementation of the Blowfish algorithm and modified Blowfish encryption algorithm, and their encryption/decryption processes are shown in Figure 3. It features an interactive field that requires input and response from users. It also includes the video file selection process for encryption and decryption.

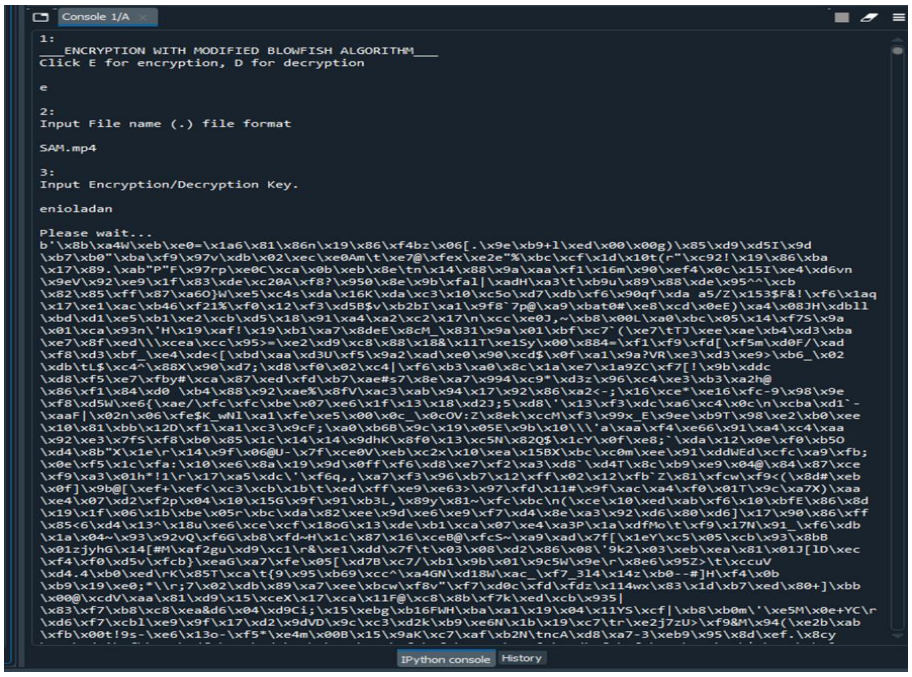

Figure 3. Video Selection Process on Application Terminal.

Figure 4 shows the selection process for video encryption. When the "Upload" button is clicked, it initially checks if a video file has been selected and whether a suitable key has been entered into the password field. If affirmative, it takes the video and passes it through the algorithm for processing and encryption, and then it returns the results (encrypted video file) to the application log/register, making the resultant file available for download by the user.

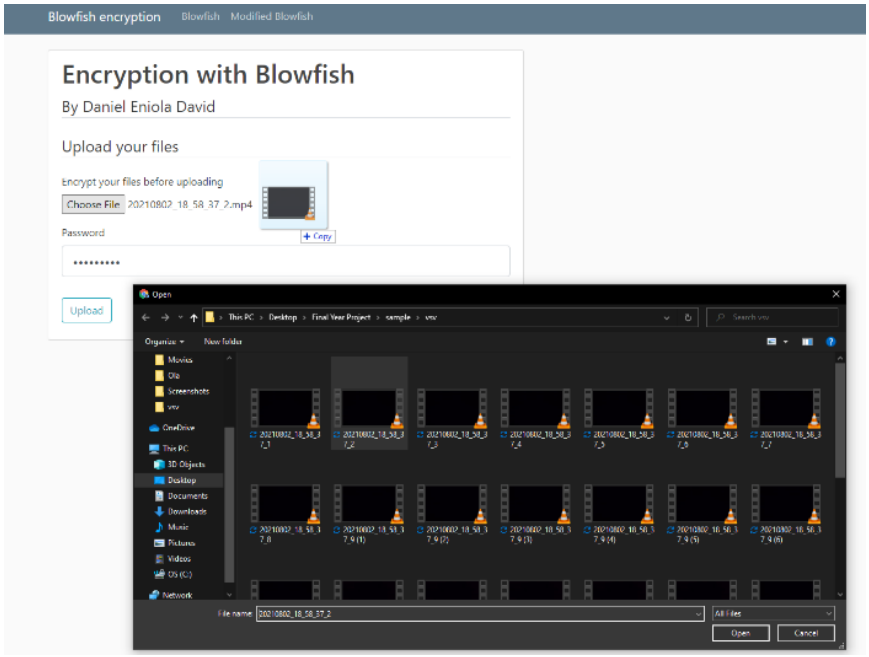

Figure 4. The selection process of video encryption app.

Figure 5 shows that the file register is a log that is located on the home page of the web application. This section of the application stores files that have been encrypted by the two algorithms and enables users to download the files at any time. The file format used for the encryption is the mp4 format.

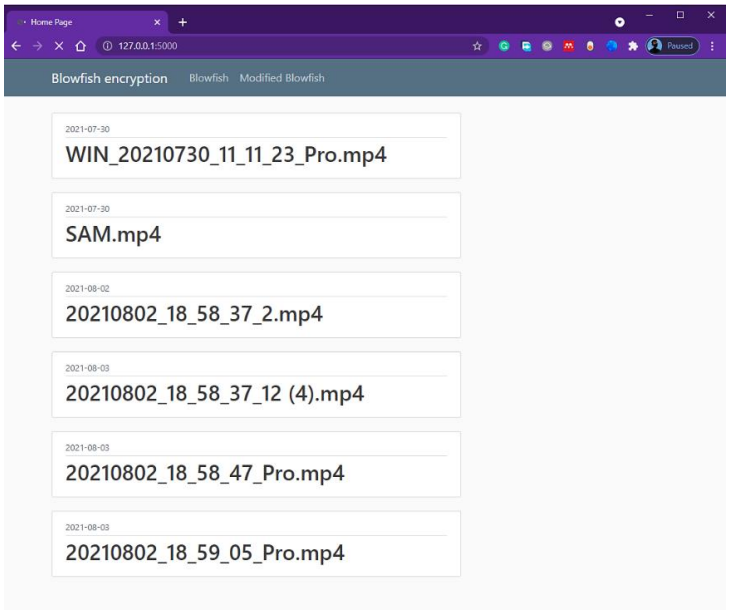

Figure 5. The home page shows the log file register with the file format.

4. Discussion

4.1. Performance Evaluation

Only the video data are compared in this comparison. To improve the accuracy of the timing measurement, the program has been run ten times. Milliseconds were used to measure the encryption and decryption times.

4.1.1. Performance Comparison Based on Execution Time

The encryption and decryption in Blowfish and the modified Blowfish algorithm were conducted for videos of varying sizes and formats. The time required for the encryption and decryption operations is computed for various video sizes (in kilobytes) and the key size also varies accordingly. The overall execution time includes both encryption and decryption time. The execution time for each video and the average execution time has been calculated as shown in Table 2.

Table 2. Analysis of computational time of Blowfish and modified Blowfish algorithms.

Video Size (Kilobytes)	Key Size (Bytes)	Blowfish Algorithm		Modified Blowfish Algorithm		Blowfish Algorithm	Modified Blowfish Algorithm
		Encryption Time (ms)	Decryption Time (ms)	Encryption Time (ms)	Decryption Time (ms)		
187.0	12	25.8	26.1	23.9	24.5	51.9	49.5
342.0	12	27.3	27.8	26.6	27.0	55.1	53.6
575.0	16	30.5	31.2	29.7	30.1	61.7	59.8
762.0	16	41.8	41.9	39.9	40.8	83.7	80.7
970.0	20	42.5	42.8	42.0	42.1	85.3	84.1
1045.0	24	49.8	49.8	49.2	49.4	99.6	98.6
1234.0	24	51.6	51.9	51.1	51.1	103.5	102.0
1445.0	28	67.6	66.9	66.8	67.1	134.5	133.9
1760.0	36	89.2	89.7	88.9	88.9	178.9	177.8
2500.0	40	124.7	125.3	124.1	124.3	250.0	248.4
		Average Execution Time				110.4	108.9

The performance analysis of the Blowfish and modified Blowfish techniques is shown in Table 2 and Figure 6. The results reveal that the modified Blowfish algorithm performs faster than the original Blowfish Algorithm based on the average execution time

of the two methods. (Average execution time is 250.0 ms for Blowfish and 248.4 ms for modified Blowfish.)

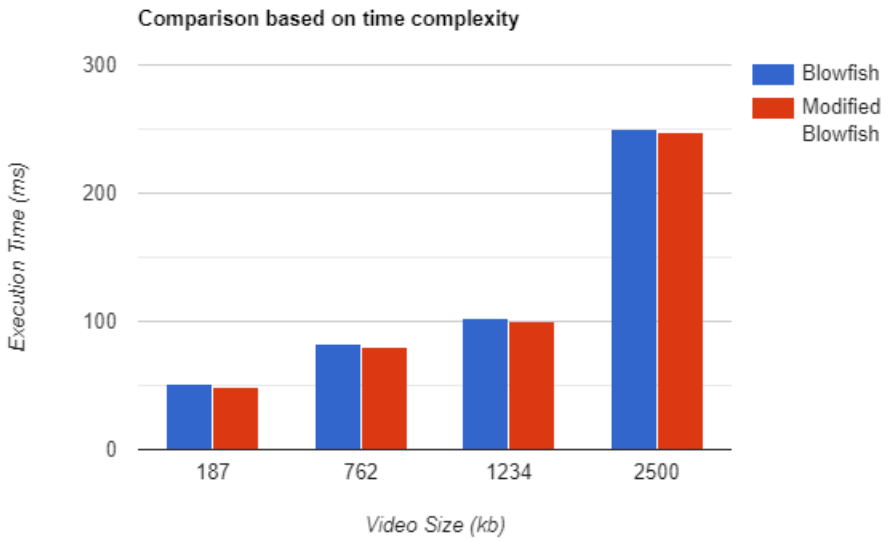

Figure 6. Experimental Results for video data types based on time complexity.

4.1.2. Performance Comparison Based on Throughput

The number of data successfully transported from one location to another in a certain time period is known as throughput. The throughput of an encryption algorithm can be calculated using:

$$\text{Throughput} = \text{data (in kilobytes)}/(\text{process end time} - \text{process start time}) \quad (3)$$

The graph in Figure 7 depicts the results of a comparison based on throughput with various sizes of video data. The modified Blowfish algorithm has a high throughput, as shown in the graph. A high throughput signifies that the encryption procedure takes less time.

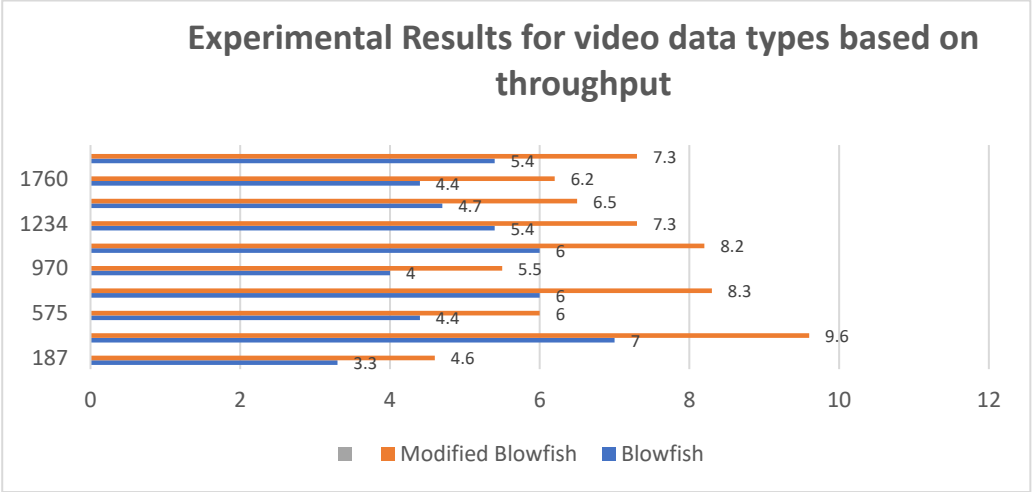

Figure 7. Comparison based on throughput.

4.1.3. Avalanche Effect

The avalanche effect occurs whenever a shift in one bit of the original message or one bit of the original key causes a change in many bits of the encrypted message. Any cryptographical approach ought to have a feature wherein a little alteration in the simple text or key results in a significant difference in the ciphertext. If the modifications are

minimal, the quantity of the simple text or key area to be examined may be reduced, making cryptanalysis considerably easier. To be safe, a cryptography method should have a significant avalanche effect. As a result, the greater the avalanche value, the greater the security. The outcome is examined using two keys that differ only by one bit location. In this scenario, the distance measure is used to calculate the number of bits that vary between the two ciphertexts. The avalanche impact is then computed as follows:

$$\text{Avalanche Effect} = (\text{Hamming Distance}/\text{Block Size}) \times 100\% \quad (4)$$

Table 3 shows the avalanche effect on the Blowfish and modified Blowfish algorithms when the key is "ABCDEFGH" and flipping one bit from the key to obtaining "CBCDEFGH" (upon flipping A (0100 0001) to C (0100 0011).

Table 3. Analysis of avalanche effect of Blowfish and modified Blowfish algorithms.

Video Size (kb)	Variance in Key (%)	Blowfish Algorithm Avalanche Effect (%)	Modified Blowfish Algorithm Avalanche Effect (%)
187	30	50.7176	43.3398
342	30	50.5176	43.1653
575	30	50.4782	42.9867
762	30	50.4486	42.8815
970	30	50.4597	42.6710
1045	30	50.3176	41.8910
1234	30	49.9974	41.7910
1445	30	49.9931	41.8910
1760	30	49.9813	41.4501
2500	30	49.8972	41.1252

The experimental results show that the avalanche effect exhibited by the Blowfish algorithm is very strong. Approximately 50% of the ciphertext bits differ after every round.

4.1.4. Threats to Validity

To begin with, the threats to the validity of this research are defined by the lack of experimentation with a large video dataset. We try to mitigate this threat by recommending the usage of a large dataset in future work. Another threat to the validity of this research regarding our experimental environment is the hardware requirements that were used to execute the developed program. Lastly, the final threat to the validity of this study is that the energy performance of the device used was not taken into consideration, which is also another performance metric for the encryption algorithm.

5. Conclusions

Although Blowfish is one of the best cryptographic algorithms, it performs poorly when encrypting huge files. Therefore, there is a need to modify the algorithm. The structure of the F function of the algorithm was altered in this research by using two S-boxes rather than four as in the classic Blowfish algorithm. Regarding the outcome of estimating the time required to encrypt and decode a video file, as well as the execution time and throughput, the modified Blowfish algorithm outperforms Blowfish.

The experimental results of our proposed modification of the time complexity and throughput are superior to the classic Blowfish algorithm, as the average execution time is 110.4 ms for the classic Blowfish while it is 108.9 ms for the modified Blowfish algorithm. Our approach achieves an improvement in terms of time complexity. Our approach achieves a throughput of 7.3% over the 5.4% of the classic Blowfish, which signifies that our proposed modified algorithm takes less time during the encryption of any video file. Based on the experimental results of this study, it is suggested to utilize the Blowfish encryption algorithm when encrypting sensitive data and applications, rather than the

modified Blowfish, for better security. Otherwise, if speed and system resources are priorities, the modified Blowfish encryption technique is recommended. For instance, banking applications require a higher level of security while gaming applications require efficient time and memory to work effectively. Therefore, this study contributes to the literature together with providing a state-of-the-art study demonstrating that Blowfish or the modified Blowfish algorithm can be used to encrypt and decrypt any form of data (text, audio, image, and video).

For further work, we plan to complete our approach in order to consider other performance metrics such as energy and a large video dataset of different file sizes to demonstrate the benefits of our approach.

Author Contributions: Conceptualization, A.E.A. and E.D.; methodology, E.D.; software, A.E.A.; validation, E.D., A.E.A., S.M. and A.B.J.; investigation, E.D. and A.E.A.; resources, A.E.A. and E.D.; data curation, E.D. and A.E.A.; writing-original draft preparation, E.D. and A.E.A.; writing-review and editing, S.M. and A.B.J.; supervision, S.M.; project administration, S.M. and A.B.J.; funding acquisition, S.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used in the study was randomly generated by the authors.

Conflicts of Interest: Authors do not have any conflict of interest.

References

1. Lyon, D. Surveillance, Snowden, and big data: Capacities, consequences, critique. *Big Data Soc.* **2014**, *1*, 2053951714541861. [CrossRef]
2. Ryan, M. *Ransomware Revolution: The Rise of a Prodigious Cyber Threat*; Springer: Berlin/Heidelberg, Germany, 2021.
3. Ma, N. Distributed video coding scheme of multimedia data compression algorithm for wireless sensor networks. *EURASIP J. Wirel. Commun. Netw.* **2019**, *2019*, 254. [CrossRef]
4. Kandris, D.; Nakas, C.; Vomvas, D.; Koulouras, G. Applications of wireless sensor networks: An up-to-date survey. *Appl. Syst. Innov.* **2020**, *3*, 14. [CrossRef]
5. Ogundokun, R.O.; Awotunde, J.B.; Adeniyi, E.A.; Ayo, F.E. Crypto-Stegno based model for securing medical information on IOMT platform. *Multimed. Tools Appl.* **2021**, *80*, 31705–31727. [CrossRef]
6. Mat Kiah, M.L.; Al-Bakri, S.H.; Zaidan, A.A.; Zaidan, B.B.; Hussain, M. Design and develop a video conferencing framework for real-time telemedicine applications using secure group-based communication architecture. *J. Med. Syst.* **2014**, *38*, 133. [CrossRef] [PubMed]
7. 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]
8. Barona, R.; Anita, E.M. A survey on data breach challenges in cloud computing security: Issues and threats. In Proceedings of the 2017 International Conference on Circuit, Power and Computing Technologies (ICCPCT), Kollam, India, 20–21 April 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–8.
9. Sudha, M.; Monica, M. Enhanced security framework to ensure data security in cloud computing using cryptography. *Adv. Comput. Sci. Its Appl.* **2012**, *1*, 32–37.
10. Dilsha; Unni, S.; Jothi, L.M.; Nair, L.S.; Kumar, N.M. *Visual Pathognomy*; Visvesvaraya Technological University: Belgaum, India, 2016.
11. Aumasson, J.P. *Serious Cryptography: A Practical Introduction to Modern Encryption*; No Starch Press: San Francisco, CA, USA, 2017.
12. Bertino, E.; Khan, L.R.; Sandhu, R.; Thuraisingham, B. Secure knowledge management: Confidentiality, trust, and privacy. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **2006**, *36*, 429–438. [CrossRef]
13. Alanazi, H.; Zaidan, B.B.; Zaidan, A.A.; Jalab, H.A.; Shabbir, M.; Al-Nabhani, Y. New comparative study between DES, 3DES and AES within nine factors. *arXiv* **2010**, arXiv:1003.4085.
14. Fitwi, A.; Chen, Y.; Zhu, S.; Blasch, E.; Chen, G. Privacy-preserving surveillance as an edge service based on lightweight video protection schemes using face de-identification and window masking. *Electronics* **2021**, *10*, 236. [CrossRef]
15. 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*. [CrossRef]
16. Nie, T.; Song, C.; Zhi, X. Performance evaluation of DES and Blowfish algorithms. In Proceedings of the 2010 International Conference on Biomedical Engineering and Computer Science, Wuhan, China, 23–25 April 2010; IEEE: Piscataway, NJ, USA, 2010; pp. 1–4.

17. Tahseen, I.; Habeeb, S. Proposal new approach for blowfish algorithm by using random key generator. *J. Madenat Alelem Univ. Coll.* **2012**, *4*, 5–13.
18. Agrawal, M.; Mishra, P. A modified approach for symmetric key cryptography based on blowfish algorithm. *Int. J. Eng. Adv. Technol. IJEAT* **2012**, *1*, 79–83.
19. Geethavani, B.; Prasad, E.V.; Roopa, R. A new approach for secure data transfer in audio signals using DWT. In Proceedings of the 2013 15th International Conference on Advanced Computing Technologies (ICACT), Rajampet, India, 21–22 September 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 1–6.
20. Dulla, G.L.; Gerardo, B.D.; Medina, R.P. An Enhanced BlowFish (eBf) Algorithm for Securing x64FileMessage Content. In Proceedings of the 2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), Baguio City, Philippines, 29 November–2 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–6.
21. Christina, L.; Joe Irudayaraj, V.S. Optimized Blowfish encryption technique. *Int. J. Innov. Res. Comput. Commun. Eng.* **2014**, *2*, 5009–5015.
22. Prasetyo, K.N.; Purwanto, Y.; Darlis, D. An implementation of data encryption for Internet of Things using blowfish algorithm on FPGA. In Proceedings of the 2014 2nd International Conference on Information and Communication Technology (ICoICT), Bandung, Indonesia, 28–30 May 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 75–79.
23. Prasad, K.L.; Anusha, P.; Jyothi, G.; Dileepkumar, K. Design and Analysis of Secure and Efficient Image with Embedded Sensitive Information Transferring Technique using Blowfish Algorithm. *I-Manag. J. Inf. Technol.* **2016**, *5*, 1.
24. Abiodun, M.K.; Awotunde, J.B.; Ogundokun, R.O.; Adeniyi, E.A.; Arowolo, M.O. Security and information assurance for IoT-based big data. In *Artificial Intelligence for Cyber Security: Methods, Issues and Possible Horizons or Opportunities*; Springer: Cham, Switzerland, 2021; pp. 189–211.
25. Ali NH, M.; Abead, S.A. Modified Blowfish Algorithm for Image Encryption using Multi Keys based on five Sboxes. *Iraqi J. Sci.* **2016**, *57*, 2968–2978.
26. Reyes AR, L.; Festijo, E.D.; Medina, R.P. Blowfish-128: A modified blowfish algorithm that supports 128-bit block size. In Proceedings of the 8th International Workshop on Computer Science and Engineering, Bangkok, Thailand, 28–30 June 2018; pp. 578–584.
27. Corpuz, R.R.; Gerardo, B.D.; Medina, R.P. Using a modified approach of blowfish algorithm for data security in cloud computing. In Proceedings of the 6th International Conference on Information Technology: IoT and Smart City, Hong Kong, China, 29–30 December 2018; pp. 157–162.
28. Shetty, V.S.; Anusha, R.; MJ, D.K.; Hegde, P. A survey on performance analysis of block cipher algorithms. In Proceedings of the 2020 International Conference on Inventive Computation Technologies (ICICT), Lalitpur, Nepal, 20–22 July 2022; IEEE: Piscataway, NJ, USA, 2020; pp. 167–174.
29. Kumar, M.A.; Karthikeyan, S. Investigating the efficiency of Blowfish and Rejindael (AES) algorithms. *Int. J. Comput. Netw. Inf. Secur.* **2012**, *4*, 22.