**Comparative Analysis of the Speeds of AES, ChaCha20, and Blowfish Encryption/Decryption Algorithms**

**Maninder Kaur1, Nicholas Fishel1**

1 Hamilton Southeastern High School, Fishers, Indiana

**Student Authors**

Maninder Kaur - High School

**SUMMARY**

With the increasing reliance on digital data and sensitive information, the need for robust data protection methods became paramount. The research commenced with a thorough examination of the pressing problem of data breaches and unauthorized access that continued to plague modern information systems. This research paper delved into the realm of encryption algorithms and their effectiveness in safeguarding data in a fast-paced manner. The study's core objective was to compare and contrast the speed of three prominent encryption algorithms: Advanced Encryption Standard (AES), ChaCha20, and Blowfish. This study's value lay in its contribution towards understanding the most suitable encryption algorithm that could address this challenge effectively. To address this question, the main hypothesis tested that the ChaCha20 encryption algorithm would outperform the others in terms of speed. To obtain results, we employed a comprehensive set of experiments using lorem ipsum text files of various lengths. Each algorithm was subjected to rigorous testing to assess its encryption and decryption times. While each algorithm demonstrated unique strengths, a clear winner, ChaCha20, emerged concerning speed. These results offered valuable context to decision-makers seeking an optimal encryption solution for the specific need of speedy encryption.

**INTRODUCTION**

In an era where our online data was so very important, ensuring its security against unauthorized users became a major purpose. There were 1,802 large-scale reported data compromises last year (1), leading to the unfortunate exposure of numerous individuals' most valuable pieces of information to unauthorized hands. One such security that could be implemented to protect data was encryption and decryption algorithms. Defined as a method of converting data from plaintext (unencrypted) to ciphertext (encrypted); These algorithms played a pivotal role in safeguarding sensitive information from malicious intent (2). The significance of encryption and decryption algorithms lay in their ability to render plaintext data incomprehensible to anyone without the appropriate decryption key. Such encryption algorithms play a crucial role in safeguarding sensitive information across various types of data files (text documents, images, videos, audio files, executables, and more). While the plaintext content of a PDF file may differ from that of an MP4 file, encryption algorithms transform these files into ciphertext, solidifying a wall of security between attackers with intent to harm. This transformation ensures that even if attackers gain access to encrypted data, they cannot discern its original meaning without the appropriate decryption keys. Thus, encryption techniques create a robust security barrier against malicious intent, irrespective of the type of data being protected (2,3). This transformation ensured that even if unauthorized individuals gained access to encrypted data, they would not be able to discern its original meaning without the proper cryptographic keys. As data volumes grew exponentially, the demand for efficient encryption techniques intensified, making the speed of encryption and decryption a critical factor in evaluating their suitability for real-world applications. The primary objective of this study was to conduct a comprehensive comparative analysis of three widely adopted encryption and decryption algorithms: Advanced Encryption Standard (AES), ChaCha20, and Blowfish. We aimed to answer the hypothesis: Did the ChaCha20 encryption and decryption algorithm outperform AES and Blowfish in terms of speed? Our investigation aimed to shed light on the significance of speed in data protection, ultimately unveiling which algorithm proved most adept at safeguarding data from unauthorized access without compromising on efficiency.

Three encryption algorithms were selected for this study - AES (Advanced Encryption Standard), ChaCha20, and Blowfish, all of which were symmetric encryption algorithms, instead of asymmetric. Symmetric and Asymmetric encryption were two different forms of encryption. Symmetric encryption used the same key to both encrypt and decrypt a file, while an asymmetric encryption algorithm utilized different keys for the encryption and decryption process. The reason we only tested symmetric encryption was that they were normally faster in encryption and decryption, as the computer did not have to take too much time accessing two different keys; the computer simply used one (3). The bit sizes varied due to different levels of performance for each bit. An AES 128-bit worked differently than a ChaCha20 128-bit. This could give one encryption algorithm a clear advantage. Bits were the “words” of the computer world. Computers understood commands through a collection of 1s and 0s. For example, the 4-bit sequence of 1000 would mean the number 8. A key, or what allowed encrypted data to be reversed to the original plain text, consisted of a sequence of bits. Without the key, an encrypted piece of data could not be decrypted to its original form. So, in our study when we stated that we would be utilizing AES 128-bit, that meant that the key consisted of 128 bits that allowed encrypting/decrypting the chosen file. The bigger the bit, normally the stronger and more complex the encryption (4).

One of the encryption algorithms we tested was the Advanced Encryption Standard (AES), a symmetric-key encryption algorithm with key sizes of 128, 192, or 256 bits. AES had been developed as a replacement for the aging Data Encryption Standard (DES). It was adopted by the U.S. National Institute of Standards and Technology (NIST) in 2001 as a federal government standard. AES employed block cipher encryption, dividing plaintext into fixed-size blocks and applying multiple rounds of transformation to produce ciphertext. Its selection had been based on its robustness, speed, and suitability for a wide range of applications (5).

ChaCha20 is a stream cipher and symmetric encryption algorithm designed to provide high security and fast performance with a fixed key size of 256 bits. A stream cipher is a process of encrypting a file by pushing every binary digit in the data through the algorithm via a cryptographic key (6). It was created by Daniel J. Bernstein in 2008 as part of the eSTREAM project. ChaCha20 is known for its simplicity and ease of implementation, while still offering a strong level of security against various cryptographic attacks (7). Its design aimed to achieve a good balance between security, speed, and resistance to side-channel attacks. ChaCha20 is widely used in various applications, including secure communication protocols, disk encryption, and digital signature schemes (7,8).

Blowfish is a symmetric-key block cipher encryption algorithm designed by Bruce Schneier in 1993 that uses variable key sizes, with the recommended key size being 128 bits. Block ciphers differ from stream ciphers as instead of encrypting every bit of data, the cipher puts data into blocks and encrypts them as groups (9). It was developed as an alternative to the aging Data Encryption Standard (DES) and provides a more secure and efficient solution for encrypting sensitive data. Blowfish operated on 64-bit blocks of plaintext and supported variable key lengths, making it adaptable to different security requirements. One of the key strengths of Blowfish is its simplicity, which allows for a relatively easy implementation and fast encryption and decryption processes (10). Additionally, Blowfish has undergone extensive cryptanalysis and has proven to be resilient against various attacks, further enhancing its reputation as a reliable encryption algorithm.

While newer algorithms like AES and ChaCha20 gained popularity in recent years, Blowfish remained a noteworthy historical milestone in the field of cryptography and continued to be used in various applications where speed and security were essential (11). The three encryption algorithms were chosen specifically since they shared multiple characteristics. They were all symmetric encryption algorithms, were widely used, and had different key sizes to choose from.

Several research studies examined the security and speed aspects of different encryption algorithms. A research experiment conducted a comparative analysis of multiple encryption algorithms, AES and Blowfish being two of them. In the study, the team assessed the algorithms' performance in terms of multiple factors (architecture, flexibility, reliability, security, and limitation) to known cryptographic attacks (12). Their findings shed light on the strengths and weaknesses of each algorithm, contributing to the understanding of their real-world applicability. Similarly, another paper delved into an analysis of the ChaCha20 cipher and its impact on the development of modern cryptographic techniques. Their exploration of the algorithm's mathematical foundations provided valuable context for understanding its strengths in asymmetric encryption (13). These sources collectively contributed to the body of knowledge surrounding encryption algorithms and formed a foundation for our research.

Our hypothesis theorized that the ChaCha20 algorithm would have a faster encryption and decryption time compared to the AES and Blowfish. In this paper, we focused on finding which encryption and decryption algorithm provided fast service: AES, ChaCha20, or Blowfish? By running multiple different text files of varying lengths multiple times through both algorithms, and recording the average of each trial, we were able to conclude which algorithm upheld the fastest encryption/decryption sequence. The efficiency of an encryption/decryption algorithm majorly lay within how fast the data would run through. The purpose of this paper was to allow individuals to find which encryption algorithm better suited their situation relating to text file encryption.

**RESULTS**

The study compared the performance of three encryption algorithms—AES, ChaCha20, and Blowfish—across various text file lengths: sentence, paragraph, page, chapter, section, and book. Each algorithm underwent testing for encryption and decryption speed through ten trials for each text file length. The lengths of each file were as follows (total number of characters, approximated): sentence: 15-20, paragraph: 150-200, page: 300-700, chapter: 3000-5000, section: 10000-20000, and book: 100000.

The average encryption and decryption times, measured in seconds, are summarized in Table 1, derived from the time taken for each encryption and decryption process for each text file length over 10 trials.

AES consistently exhibited a higher average encryption time than ChaCha20 and Blowfish across all text file lengths. However, its decryption time was generally faster than ChaCha20 and Blowfish. Notably, AES decryption was significantly faster than AES encryption, emphasizing the asymmetric nature of the algorithm.

ChaCha20 consistently demonstrated faster encryption and decryption times compared to AES and Blowfish for all text file lengths. The algorithm's encryption and decryption times were relatively balanced and showed consistent performance across different lengths.

Blowfish generally performed comparably to AES in terms of encryption time but consistently had higher decryption times. It was notably slower than ChaCha20 for both encryption and decryption.

Here are some observations from our data:

* ChaCha20 consistently outperformed AES and Blowfish in terms of encryption and decryption speed for all text file lengths. When averaging both operations, ChaCha20 had the shortest average time (0.00372 seconds) compared to AES (0.0973 seconds) and Blowfish (0.00892 seconds).
* AES exhibited better performance in decryption compared to its encryption times, while ChaCha20 maintained a more balanced performance in both operations.
* Blowfish, being an older encryption algorithm, demonstrated slower performance compared to the other two modern algorithms in all cases.
* Despite the varying sizes of the text files, there was no discernible pattern indicating an increase or decrease in encryption/decryption times. All three algorithms consistently completed their processes in approximately the same amount of time for each of the data files.

**DISCUSSION**

From the experiment, we aimed to compare the speed of three encryption algorithms—AES, ChaCha20, and Blowfish—when applied to text files of various lengths. Our findings reveal that ChaCha20 emerged as the fastest encryption algorithm, showcasing consistently faster encryption and decryption times compared to AES and Blowfish. While AES demonstrated competitive decryption times, it generally lagged behind in encryption speed, whereas Blowfish proved slower overall. These results offer valuable insights for researchers and developers seeking optimal encryption solutions for different applications.

A quick clarification would go a long way. The encryption process includes both encryption and decryption, yet encryption and decryption are two different things.

Based on the results, ChaCha20 emerged as the fastest encryption algorithm among the tested options for various text file lengths. Although AES is widely used and efficient, it showed slightly slower performance than ChaCha20. Meanwhile, Blowfish, while still providing acceptable security, proved to be the slowest of the three algorithms. Researchers and developers seeking a fast and secure encryption solution for text files should consider ChaCha20 as a promising option. The findings of this experiment provide valuable insights into the performance of three encryption algorithms, namely AES, ChaCha20, and Blowfish, when applied to text files of varying lengths. The research question aimed to compare the speed performance of these algorithms for encryption and decryption, and the results shed light on their relative efficiency in different scenarios. The significance of this study lies in assisting researchers, developers, and practitioners in making informed decisions regarding the selection of encryption algorithms based on their specific use cases and performance requirements. Comparing each of the results we observe, the standout performer in this study was ChaCha20, consistently demonstrating faster encryption and decryption times compared to the other two algorithms. Its balanced performance in both operations makes it an attractive choice for applications that require speed without compromising security. ChaCha20's resistance to various cryptographic attacks, like differential and linear cryptanalysis, adds to its appeal.

AES showed competitive encryption times with Blowfish on average (Table 1), but stood in the middle regarding decryption times compared to the other two algorithms. Of all the algorithms, AES on average with its total time was the slowest. This might limit its use in scenarios where speed is a crucial factor. Lastly, Blowfish exhibited slower performance in both encryption when compared to ChaCha20, yet did surpass AES. On the topic of decryption, however, the algorithm did a phenomenal job, beating the other two algorithms by a margin on average (Table 1). Its widespread usage and high security make it a suitable choice for applications where symmetric encryption with faster decryption is needed.

Yet, there are possible limitations and drawbacks in each encryption algorithm and with our experiment. Due to ChaCha20 being a relatively newer algorithm, it might not have undergone the same level of scrutiny as AES (14). On the topic of AES, Its weaknesses include potentially slower performance in certain applications and susceptibility to timing attacks. Blowfish is known for its slower performance compared to modern algorithms, and the absence of widespread adoption in critical applications limit its use in scenarios where speed is a priority. The experiment's limitations include the use of pseudocode instead of actual code execution, which might not account for all implementation details and optimizations in specific programming languages and libraries. Additionally, the experiment focused solely on speed performance and did not assess the algorithms' resistance to advanced attacks. Real-world implementations also consider other factors, such as memory usage and platform support, which were not addressed in this study.

Our research could have other aspects taken into consideration while also assisting in other research opportunities. For example, future research should consider evaluating the algorithms' security strengths and weaknesses, including resistance to more advanced attacks like side-channel attacks and quantum computing-related vulnerabilities (15). Another consideration is investigating the algorithms' performance in resource-constrained environments, such as Internet of Things (IoT) devices or embedded systems, could provide valuable insights for practical applications (16). Alongside this, one could also explore hybrid encryption approaches, combining the strengths of different algorithms, may offer improved security and performance for specific use cases. Lastly, extending the experimentation to include real-world implementations in various programming languages and platforms can validate the findings and identify potential optimization opportunities. If an individual or group wants to improve or add onto our research, they should consider testing different types of data files. A video file is far more complex than a text file, thus there is a possibility that a certain algorithm may work better for such file types unlike others.

It is essential to acknowledge some limitations of the study. First, the experiments were conducted on a single machine, and results may vary on different hardware configurations. A computer operating on a Windows OS or Linux may produce different results.

Second, the study focused on text files with Lorem Ipsum content, and the results might differ when encrypting real-world data. While Lorem Ipsum is meant to imitate real-world data, the results may appear differently if text excerpts from a news article or book were taken for the experiment. Furthermore, the language used was English, thus not completely representative of all languages that these encryption algorithms can encrypt.

Also, the length of each text file is corresponding with a certain average length (such as a paragraph composed of 6 sentences, a page of 3 paragraphs, etc.) which may not be totally factual. One book may have 700 pages while another has 50. A paragraph can have 15 sentences, or 3 sentences. In an attempt to rectify this error in our experiment, we generated text files with average lengths of their corresponding writing elements.

Lastly, the code and pseudocode was used for simulation purposes, and actual implementation may have subtle differences.

This study sheds light on the critical role of encryption algorithms in safeguarding digital data and highlights the importance of considering speed in algorithm selection. By identifying ChaCha20 as a standout performer in terms of speed and efficiency, this research offers a promising avenue for enhancing data security speed in various domains. Moving forward, continued exploration of encryption techniques and their real-world applications will undoubtedly drive advancements in cybersecurity, ensuring a safer digital landscape for individuals and organizations alike.

**MATERIALS AND METHODS**

The primary objective of this study is to compare the speed of three encryption algorithms - AES, ChaCha20, and Blowfish - when applied to text files of varying lengths. The study aimed to analyze the encryption and decryption performance of each algorithm to gain insights into their respective strengths and weaknesses.

The primary objective of this study is to compare the speed of three encryption algorithms - AES, ChaCha20, and Blowfish - when applied to text files of varying lengths. The study aimed to analyze the encryption and decryption performance of each algorithm to gain insights into their respective strengths and weaknesses.

The experiments were conducted on a computer with a standard configuration, including an Intel Core i5 processor, 8 GB RAM, and a solid-state drive (SSD). The operating system used was a macOS Monterey Version 12.6. The experiments were performed using Python 3.x with the `cryptography` library. All Python libraries were updated and ran on the latest versions (All compatible with Python 3.11, thus the cryptography library was running on version 41.0.1 at the time). Code and pseudocode were developed for each encryption algorithm to simulate the encryption and decryption processes (17). The pseudocode for AES, ChaCha20, and Blowfish was adapted from standard Python implementations using the `cryptography` library. Each algorithm was tested ten times on each of the text files, with the size of the file ranging from a few bytes to a few kilobytes.

The experiment utilized five distinct text files with diverse lengths of Lorem Ipsum content as the input data for encryption and decryption. The content was created using a Lorem Ipsum generator (18), and was used as Lorem Ipsum can mimic real life data well. The text files were of varying lengths, comprising a sentence, a paragraph, a page, a chapter, a section, and the length of a complete book.The text files were preprocessed to ensure consistent formatting and content.

We used the following key sizes to ensure equivalency between all algorithms: AES 128, ChaCha20 256, and Blowfish 128 bits. The reason the ChaCha20 had a larger bit size was in part to differences in the design and structure of the algorithms. While AES 256 and Blowfish 256 have a larger key size, the security of an encryption algorithm depends on various factors, including key schedule, block size, and the algorithm's resistance to known attacks.Thus, ChaCha20 256 was chosen to be equal in comparison with AES and Blowfish in terms of performance ability. Thus, no encryption algorithm has an advantage over another,

Performance: ChaCha20 is known for its efficiency in software implementations, especially on devices without hardware support for AES instructions. In some cases, ChaCha20 with a 256-bit key can provide comparable or better performance than AES128, making it an attractive choice for applications where computational efficiency is critical

The collected data was analyzed to compare the average encryption and decryption times for each algorithm. The results were tabulated and graphically represented to visualize the relative performance of AES, ChaCha20, and Blowfish across different text file lengths.

The methodology presented in this research allowed for a comparative analysis of AES, ChaCha20, and Blowfish encryption algorithms in terms of speed and efficiency when encrypting and decrypting text files of varying lengths. The results of the study provide valuable insights into the performance of each algorithm, which can be used to make informed decisions regarding their application in specific scenarios.

**REFERENCES**

1. Identity Theft Resource Center. (2024, May 16). *Identity Theft Resource Center’s 2022 annual data Breach report reveals Near-Record number of compromises - ITRC*. ITRC. https://www.idtheftcenter.org/post/2022-annual-data-breach-report-reveals-near-record-number-compromises/
2. *What is encryption? | IBM*. (n.d.). https://www.ibm.com/topics/encryption
3. *Encryption choices: rsa vs. aes explained | Prey*. (2021, June 15). https://preyproject.com/blog/types-of-encryption-symmetric-or-asymmetric-rsa-or-aes#:~:text=AES%20vs.&text=RSA%20encryption%20methods%2C%20it%20is,key%20exchange%20and%20digital%20signatures.
4. *All about cryptographic bit lengths*. (n.d.). TCC - Technical Communications Corporation: Network Encryption, Secure Communications. https://www.tccsecure.com/NewsResources/CipherONEBlog/TabId/1222/ArtMID/1578/ArticleID/2/All-About-Cryptographic-Bit-Lengths.aspx
5. Dray, M. J. D. E. B. B. J. R. N. J. F. L. E. B. E. R. J. F., Jr. (2021, March 1). *Advanced Encryption Standard (AES)*. NIST. https://www.nist.gov/publications/advanced-encryption-standard-aes
6. Contributor, T. (2021, April 23). stream cipher. Security. https://www.techtarget.com/searchsecurity/definition/stream-cipher
7. Computerphile. (2021, February 19). *Chacha Cipher - computerphile* [Video]. YouTube. https://www.youtube.com/watch?v=UeIpq-C-GSA
8. Nagaraj, Karthikeyan. “Understanding ChaCha20 Encryption: A Secure and Fast Algorithm for Data Protection | 2023.” Medium, 8 Mar. 2023.
9. GeeksforGeeks. (2024, March 1). *Difference between Block Cipher and Stream Cipher*. GeeksforGeeks. https://www.geeksforgeeks.org/difference-between-block-cipher-and-stream-cipher/
10. GeeksforGeeks. (2024b, March 7). *Blowfish Algorithm with Examples*. GeeksforGeeks. https://www.geeksforgeeks.org/blowfish-algorithm-with-examples/
11. Awati, Rahul. “Blowfish.” Security, Jan. 2022.
12. *Kumari, Shailja, and Jyoti Chawla. Comparative Analysis on Different Parameters of Encryption Algorithms for Information Security. May 2015. - Google Search*. (n.d.). https://www.google.com/search?q=Kumari%2C+Shailja%2C+and+Jyoti+Chawla.+Comparative+Analysis+on+Different+Parameters+of+Encryption+Algorithms+for+Information+Security.+May+2015.&oq=Kumari%2C+Shailja%2C+and+Jyoti+Chawla.+Comparative+Analysis+on+Different+Parameters+of+Encryption+Algorithms+for+Information+Security.+May+2015.&gs\_lcrp=EgZjaHJvbWUyBggAEEUYOdIBBzQxNWowajeoAgCwAgA&sourceid=chrome&ie=UTF-8
13. Shi, Z., Zhang, B., Feng, D., & Wu, W. (2013). Improved key recovery attacks on Reduced-Round Salsa20 and ChaCha. In *Lecture notes in computer science* (pp. 337–351). https://doi.org/10.1007/978-3-642-37682-5\_24
14. *RFC 7539: CHACHA20 and POLY1305 for IETF protocols*. (n.d.). IETF Datatracker. https://datatracker.ietf.org/doc/html/rfc7539#:~:text=The%20ChaCha20%20Encryption%20Algorithm%20ChaCha20,successively%20increasing%20block%20counter%20parameters.
15. *Cybersecurity of quantum computing: a new frontier*. (2023, April 10). SEI Blog. https://insights.sei.cmu.edu/blog/cybersecurity-of-quantum-computing-a-new-frontier/
16. Intuz. (n.d.). *Securing the IoT data Landscape: IoT Encryption Algorithms - InTUZ*. https://www.intuz.com/blog/securing-the-iot-data-landscape-iot-encryption-algorithms
17. Themaninderkaur. (n.d.). *GitHub - themaninderkaur/research-paper-sources: Code, pseudocode and text files used in research paper*. GitHub. https://github.com/themaninderkaur/research-paper-sources
18. *Lorem ipsum – Generator, Origins and meaning*. (n.d.). Lorem Ipsum. https://loremipsum.io/

**Tables with Captions**

Table 1: Average time in seconds of each encryption algorithm to encrypt/decrypt a text file containing a certain length of Lorem Ipsum. The data was collected by recording the time taken for each encryption and decryption process for each text file length over 10 trials, and the mean was calculated resulting in these values.

| Length | AES Encryption | AES Decryption | ChaCha20 Encryption | ChaCha20 Decryption | Blowfish Encryption | Blowfish Decryption |
| --- | --- | --- | --- | --- | --- | --- |
| Sentence | 0.0097 | 0.0006 | 0.0028 | 0.0009 | 0.097 | 0.001 |
| Paragraph | 0.099 | 0.0007 | 0.0028 | 0.0011 | 0.098 | 0.0013 |
| Page | 0.099 | 0.0007 | 0.0026 | 0.0009 | 0.09 | 0.0012 |
| Chapter | 0.09 | 0.0008 | 0.0029 | 0.0011 | 0.083 | 0.0017 |
| Section | 0.092 | 0.0008 | 0.0027 | 0.0011 | 0.097 | 0.0011 |
| Book | 0.086 | 0.0002 | 0.0027 | 0.0007 | 0.086 | 0.0011 |