**Comparative Analysis over the Speeds of Different Encryption/Decryption Algorithms on Text Files of Various Lengths**

**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. 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 didn’t have to take too much time accessing two different keys; the computer simply used one (INSERT). 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.(INSERT).

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 (3).

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 (INSERT). 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 (4). 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 (4,5).

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 (INSERT). 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 (6). 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 (7). 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 (8). 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 (9). 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, when applied to text files of varying lengths: sentence, paragraph, page, chapter, section, and book.

Each algorithm was tested for encryption and decryption speed using ten different trials for each text file length. (How much each length; Regarding the lengths of each file, here they are in total number of characters: sentence: , paragraph: , page: , chapter: , section: , and book: .)

The average encryption and decryption times for each algorithm, measured in seconds, are summarized in Figure 7. AES consistently showed a higher average encryption time compared to ChaCha20 and Blowfish for all text file lengths. However, its decryption time was generally faster than ChaCha20 and Blowfish. Notably, AES decryption was significantly faster than AES encryption, highlighting the asymmetric nature of the algorithm's encryption and decryption operations.

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 exhibited consistent performance across the different text file lengths.

Blowfish generally showed comparable performance to AES in terms of encryption time, but consistently had higher decryption times. It was notably slower than ChaCha20 for both encryption and decryption. Key details about each encryption algorithm:

- ChaCha20 consistently outperformed both AES and Blowfish in terms of encryption and decryption speed for all text file lengths. When adding the average encryption and decryption times together for each algorithm, ChaCha20 had the shortest average time (0.00372 seconds) compared to AES(0.0973 Seconds) and Blowfish(0.00892 Seconds).

- AES performed better in decryption compared to its encryption times, while ChaCha20 maintained a more balanced performance in both operations.

- Blowfish, an older encryption algorithm, demonstrated slower performance compared to the other two modern algorithms in all cases.

Based on the results, ChaCha20 emerged as the fastest encryption algorithm among the tested options for various text file lengths. AES, although widely used and efficient, showed slightly slower performance than ChaCha20. Meanwhile, Blowfish, while still providing acceptable security, proved to be the slowest of the three algorithms. Therefore, researchers and developers seeking a fast and secure encryption solution for text files should consider ChaCha20 as a promising option.

**DISCUSSION**

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, but had relatively faster decryption times compared to its encryption speed. Its widespread usage and high security make it a suitable choice for applications where symmetric encryption with faster decryption is needed. Lastly, while still providing acceptable security, Blowfish exhibited slower performance in both encryption and decryption when compared to the other two algorithms. This might limit its use in scenarios where speed is a crucial factor.

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. 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. 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. 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. There is a possibility that video, audio and other such files may have different encryption/decryption times compared to text 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.

**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 aims to analyze the encryption and decryption performance of each algorithm to gain insights into their respective strengths and weaknesses.

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.

Code and pseudocode was developed for each encryption algorithm to simulate the encryption and decryption processes (10). 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 (11), 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.

For each text file, the following steps were performed for each encryption algorithm:

a. Key Generation: A random key was generated for each algorithm, adhering to the specific key size requirements for AES (128 bits), ChaCha20 (256 bits), and Blowfish (128 bits).

b. Encryption: The text file was encrypted and decrypted using the respective algorithm. The total time was measured by recording the time taken to encrypt and decrypt the entire file.

c. Decryption: The encrypted file was decrypted using the same algorithm and key. The decryption time was measured by recording the time taken to decrypt the entire file.

The study measured the encryption and decryption times for each algorithm and recorded the results in seconds. Each encryption and decryption process was repeated ten times to calculate the average time for better accuracy.

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.

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.

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 experts 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.

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. “Identity Theft Resource Center’s 2022 Annual Data Breach Report Reveals Near-Record Number of Compromises - ITRC.” ITRC, 19 Jan. 2023,
2. What Is Encryption? Data Encryption Defined | IBM.
3. Dray, Morris J. Dworkin Elaine B. Barker, James R. Nechvatal, James Foti, Lawrence E. Bassham, E. Roback, James F., Jr. “Advanced Encryption Standard (AES) | NIST.” NIST, Mar. 2021,
4. Computerphile. “Chacha Cipher - Computerphile.” YouTube, 19 Feb. 2021,
5. Nagaraj, Karthikeyan. “Understanding ChaCha20 Encryption: A Secure and Fast Algorithm for Data Protection | 2023.” Medium, 8 Mar. 2023.
6. GeeksforGeeks. “Blowfish Algorithm With Examples.” GeeksforGeeks, July 2023.
7. Awati, Rahul. “Blowfish.” Security, Jan. 2022.
8. Kumari, Shailja, and Jyoti Chawla. Comparative Analysis on Different Parameters of Encryption Algorithms for Information Security. May 2015.
9. “Study and Analysis of eSTREAM Cipher Salsa and ChaCha.” IEEE Conference Publication | IEEE Xplore, 1 Mar. 2016.
10. Themaninderkaur. “GitHub - Themaninderkaur/Research-paper-sources: Code, Pseudocode and Text Files Used in Research Paper.” GitHub.
11. “Lorem Ipsum – Generator, Origins and Meaning.” Lorem Ipsum.

**Tables with Captions**

Figure 1: Sentence encryption. Time in seconds of encryption algorithms to encrypt/decrypt a text file containing a sentence of Lorem Ipsum. Ran the sentence text file 10 times for each encryption algorithm code and recorded the time.

|  | Time (in seconds) | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Trial (Sentence) | AES Encryption | AES Decryption | ChaCha20 Encryption | ChaCha20 Decryption | Blowfish Encryption | Blowfish Decryption |
| 1 | 0.157829762 | 0.001281023 | 0.004184961 | 0.000613928 | 0.161406994 | 0.001870155 |
| 2 | 0.091483116 | 0.00058794 | 0.002929926 | 0.000460863 | 0.096866846 | 0.000597 |
| 3 | 0.098189116 | 0.000602245 | 0.002563 | 0.000564098 | 0.088738203 | 0.001410961 |
| 4 | 0.092118979 | 0.00043416 | 0.002677917 | 0.001266003 | 0.089432955 | 0.00079298 |
| 5 | 0.087681055 | 0.000607014 | 0.002701998 | 0.00050807 | 0.089328289 | 0.000684738 |
| 6 | 0.087242126 | 0.000415087 | 0.002652884 | 0.001274824 | 0.092391968 | 0.000680923 |
| 7 | 0.088404894 | 0.000412226 | 0.002662897 | 0.000756025 | 0.08743906 | 0.001280069 |
| 8 | 0.088419914 | 0.000635147 | 0.002636671 | 0.001400232 | 0.088032961 | 0.001218796 |
| 9 | 0.093755245 | 0.000652075 | 0.002574682 | 0.001172066 | 0.093288898 | 0.000631809 |
| 10 | 0.088280201 | 0.00045085 | 0.002670288 | 0.000586033 | 0.091717958 | 0.000911713 |

Figure 2: Paragraph encryption. Time in seconds of encryption algorithms to encrypt/decrypt a text file containing a paragraph of Lorem Ipsum. Ran the paragraph text file 10 times for each encryption algorithm code and recorded the time.

|  | Time (in seconds) | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Trial (Paragraph) | AES Encryption | AES Decryption | ChaCha20 Encryption | ChaCha20 Decryption | Blowfish Encryption | Blowfish Decryption |
| 1 | 0.185265064 | 0.00210619 | 0.003730059 | 0.000470877 | 0.160014868 | 0.00151515 |
| 2 | 0.093444109 | 0.000481129 | 0.002621174 | 0.001269817 | 0.088263035 | 0.001267195 |
| 3 | 0.089739084 | 0.000412226 | 0.002569914 | 0.001158237 | 0.092304945 | 0.00093317 |
| 4 | 0.089798927 | 0.000574827 | 0.002799988 | 0.000510931 | 0.089615822 | 0.000986814 |
| 5 | 0.088904858 | 0.000503063 | 0.002617836 | 0.000602245 | 0.088310003 | 0.00128293 |
| 6 | 0.099683046 | 0.000459909 | 0.002665997 | 0.00123477 | 0.096293926 | 0.00165987 |
| 7 | 0.087789774 | 0.000613928 | 0.002640009 | 0.001121998 | 0.089673996 | 0.000751019 |
| 8 | 0.087352753 | 0.000431299 | 0.002722025 | 0.001204967 | 0.088140011 | 0.001560211 |
| 9 | 0.087594032 | 0.000559092 | 0.002608776 | 0.001039743 | 0.092224836 | 0.001347065 |
| 10 | 0.089903116 | 0.00060606 | 0.002630234 | 0.00114274 | 0.090899229 | 0.001303196 |

Figure 3: Page encryption. Time in seconds of encryption algorithms to encrypt/decrypt a text file containing a page of Lorem Ipsum. Ran the page text file 10 times for each encryption algorithm code and recorded the time.

|  | Time (in seconds) | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Trial (Page) | AES Encryption | AES Decryption | ChaCha20 Encryption | ChaCha20 Decryption | Blowfish Encryption | Blowfish Decryption |
| 1 | 0.175486088 | 0.002148151 | 0.002671242 | 0.00071311 | 0.092159033 | 0.00144124 |
| 2 | 0.092870235 | 0.00056982 | 0.002690077 | 0.00078702 | 0.089770079 | 0.000763178 |
| 3 | 0.09651494 | 0.000626802 | 0.002619982 | 0.001207113 | 0.087939024 | 0.001305819 |
| 4 | 0.088747978 | 0.000451088 | 0.002547026 | 0.00125289 | 0.088756323 | 0.001158237 |
| 5 | 0.091102839 | 0.000435352 | 0.002619982 | 0.000748158 | 0.090011835 | 0.001302004 |
| 6 | 0.089423895 | 0.000645876 | 0.002558947 | 0.000419855 | 0.088992119 | 0.001319885 |
| 7 | 0.089033127 | 0.000705957 | 0.002695084 | 0.000751734 | 0.091644764 | 0.00089097 |
| 8 | 0.090562105 | 0.000614882 | 0.002566814 | 0.001127958 | 0.08930397 | 0.001363993 |
| 9 | 0.089470863 | 0.000616312 | 0.002836943 | 0.001145124 | 0.088298798 | 0.001341105 |
| 10 | 0.089745045 | 0.000609159 | 0.002573013 | 0.000404835 | 0.091223001 | 0.000858307 |

Figure 4: Chapter encryption. Time in seconds of encryption algorithms to encrypt/decrypt a text file containing a chapter length of Lorem Ipsum. Ran the chapter text file 10 times for each encryption algorithm code and recorded the time.

|  | Time (in seconds) | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Trial (Chapter) | AES Encryption | AES Decryption | ChaCha20 Encryption | ChaCha20 Decryption | Blowfish Encryption | Blowfish Decryption |
| 1 | 0.091027975 | 0.001761913 | 0.004687786 | 0.00067687 | 0.094291925 | 0.00265789 |
| 2 | 0.089843035 | 0.000701904 | 0.002738953 | 0.001167059 | 0.092753172 | 0.001893044 |
| 3 | 0.089301109 | 0.000674009 | 0.002682686 | 0.001229048 | 0.088912964 | 0.001673937 |
| 4 | 0.088361979 | 0.000651121 | 0.002706051 | 0.001225948 | 0.090374947 | 0.001679182 |
| 5 | 0.08893609 | 0.000690937 | 0.00267005 | 0.001213789 | 0.089257956 | 0.001776934 |
| 6 | 0.089712858 | 0.000462055 | 0.002643108 | 0.000468969 | 0.088304043 | 0.001243114 |
| 7 | 0.088606119 | 0.000729799 | 0.002723932 | 0.00129509 | 0.089350939 | 0.001315832 |
| 8 | 0.093358755 | 0.000658989 | 0.002722025 | 0.001322985 | 0.089164019 | 0.001856089 |
| 9 | 0.088025093 | 0.000559807 | 0.002894163 | 0.000819921 | 0.09680891 | 0.001791 |
| 10 | 0.088316917 | 0.000751019 | 0.002681732 | 0.00126195 | 0.001791 | 0.001263857 |

Figure 5: Section encryption. Time in seconds of encryption algorithms to encrypt/decrypt a text file containing a section length of Lorem Ipsum. Ran section text file 10 times for each encryption algorithm code and recorded the time.

|  | Time (in seconds) | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Trial (Section) | AES Encryption | AES Decryption | ChaCha20 Encryption | ChaCha20 Decryption | Blowfish Encryption | Blowfish Decryption |
| 1 | 0.096870184 | 0.000304937 | 0.003117085 | 0.000393867 | 0.164867878 | 0.001132011 |
| 2 | 0.087626934 | 0.000203133 | 0.002571821 | 0.000315905 | 0.088931799 | 0.001351833 |
| 3 | 0.088101149 | 0.000186205 | 0.002604008 | 0.00109005 | 0.089232922 | 0.001025915 |
| 4 | 0.093580008 | 0.000196934 | 0.002606153 | 0.00037384 | 0.089479923 | 0.001029015 |
| 5 | 0.089359999 | 0.000206947 | 0.002742052 | 0.000560045 | 0.089491129 | 0.001180172 |
| 6 | 0.088580132 | 0.000196218 | 0.002507925 | 0.000986099 | 0.091023922 | 0.001150131 |
| 7 | 0.089049101 | 0.000192881 | 0.002528191 | 0.000829935 | 0.088914871 | 0.001146078 |
| 8 | 0.095077753 | 0.000210762 | 0.002475023 | 0.000977039 | 0.089835882 | 0.001013041 |
| 9 | 0.090180159 | 0.000190973 | 0.002663136 | 0.000288725 | 0.089296103 | 0.001026869 |
| 10 | 0.098166943 | 0.000197172 | 0.002708435 | 0.000585318 | 0.088784933 | 0.000540972 |

Figure 6: Book encryption. Time in seconds of encryption algorithms to encrypt/decrypt a text file containing a book length of Lorem Ipsum. Ran the book text file 10 times for each encryption algorithm code and recorded the time.

|  | Time (in seconds) | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Trial  (Book) | AES Encryption | AES Decryption | ChaCha20 Encryption | ChaCha20 Decryption | Blowfish Encryption | Blowfish Decryption |
| 1 | 0.089272976 | 0.000221014 | 0.002452135 | 0.000880957 | 0.084707975 | 0.001057863 |
| 2 | 0.085862875 | 0.00018096 | 0.002707958 | 0.000265837 | 0.085093975 | 0.001044989 |
| 3 | 0.084143162 | 0.000187159 | 0.002430916 | 0.001082897 | 0.087824821 | 0.001068115 |
| 4 | 0.086806059 | 0.00018096 | 0.002403021 | 0.000274897 | 0.085568905 | 0.001332998 |
| 5 | 0.085216045 | 0.000185013 | 0.002424955 | 0.000380039 | 0.085328817 | 0.001059055 |
| 6 | 0.084425926 | 0.000178099 | 0.002406836 | 0.00056386 | 0.088064909 | 0.001363039 |
| 7 | 0.085717916 | 0.00017786 | 0.002483845 | 0.000552893 | 0.088246107 | 0.001233101 |
| 8 | 0.084599733 | 0.000178099 | 0.002351046 | 0.000840902 | 0.085444927 | 0.000599146 |
| 9 | 0.084213018 | 0.000184059 | 0.002412081 | 0.000937223 | 0.086187124 | 0.001073122 |
| 10 | 0.085932016 | 0.000181913 | 0.002393007 | 0.001146793 | 0.085671186 | 0.001224041 |

Figure 7: Averages. Average time in seconds of each encryption algorithm to encrypt/decrypt a text file containing a certain length of Lorem Ipsum. Found mean using the numbers from Figures 1-6, divided by 10 and found mean; proceeded to be recorded.

| Length | AES Encryption | AES Decryption | ChaCha20 Encryption | ChaCha20 Decryption | Blowfish Encryption | Blowfish Decryption |
| --- | --- | --- | --- | --- | --- | --- |
| Sentence | 0.1024 | 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 |