Analysis of Multiple Cryptographic Ciphers and Their Effectiveness

Brandon Parker, Nahor Yirgaalem

CP460 - Applied Cryptography

Lilatul Ferdouse

December 6, 2023

Abstract

Scope

The scope of this paper details an analysis of the three cryptographic ciphers: Shift, Stream, and Block ciphers. It discusses the high level details of the encryption and decryption of the ciphers. Additionally it explores the practical implementation of the ciphers using Python by utilizing the language's simplicity and ease of handling bitwise operations. The paper also talks about common attacks on these ciphers such as brute force attacks, character frequency analysis, and header attacks. It talks about the effectiveness of each attack and modern technology can make them more effective. It also touches on how the ciphers are used today and how important modern encryption standards we have today are with safeguarding people's information against different attacks.

Implementation

For this report, we have developed three separate applications, each for an individual cipher. The program contains encrypting and decrypting functions, a brute force attack as well as an example, to demonstrate the process of the cipher, for each cipher. For the Block and Stream ciphers, we have also incorporated a key generation function, which generates a keystream of requested length and updates the key.

Articles

We have referenced four different scholarly articles to assist writing the report. Each article was written by experts in the field, allowing us to use facts.

Limitations

One main limitation is that the analysis focuses on only three ciphers while there are other algorithms and ciphers that exist. Another limitation is that the implementation of the ciphers was done with Python for simplicity's sake, however, it does not discuss other options such as other programming languages or hardware implementation. Furthermore when discussing the different types of attacks only three are talked about in depth while other attacks exist. This is also prevalent in the code implementation as only the brute force attack was considered.

Cipher Introductions

Shift

The shift cipher was famously used by Julius Caesar with a 3 key shift. Now the term shift cipher represents a family of ciphers where a character of the plaintext is shifted by an arbitrary number. (Kotas, 2000) For example/simplification purposes, the shift cipher we will be discussing will use the 26 characters in the English alphabet (Note, the same process can be applied to any group of order characters of any length; there would just be a slight modification to the functions displayed in the diagrams below). The cipher works on one character of the message at a time. The encryption process requires a key which is the amount the character will be shifted to the right. If the shift makes it past the last character then it wraps around back to the beginning of the alphabet, this can be represented by the modulus operation (Figure 1). The decryption function works virtually the same just in the opposite direction by shifting to the left where if you were to make it past the first character you would wrap around to the end (Figure 2).

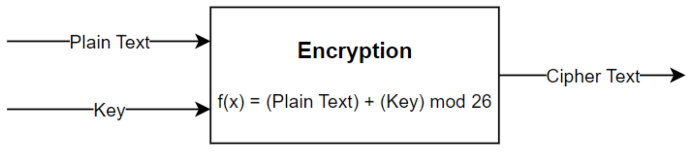

Figure 1: Shift Cipher Encryption

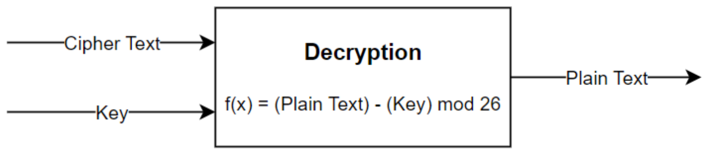

Figure 2: Shift Cipher Decryption

Stream

Used in modern encryption today the stream cipher is simple at its core. From a high-level view, the stream cipher works by XORing the bit stream output from the keystream generator with the bit stream of the plain text. The keystream generator is given an initial key to start its bit stream generation form (Figure 3). The more complex part of the stream cipher is the keystream generator. This part can be many different algorithms. You would think that a generator that always gives out a truly random key bit stream would be perfect, but that would require storing the entire key bit stream to decrypt the message. This has the potential to be a very large file which is why that method is not used. Instead, the keystream generator is typically an algorithm that, given the same initial key, an identical key bit stream can be generated. Thus making it easier and using fewer resources to decrypt the message. This allows the decryption function to work similarly to the encryption function with the only difference is the key bit stream is XORed with the cipher text to output the plain text (Figure 4).

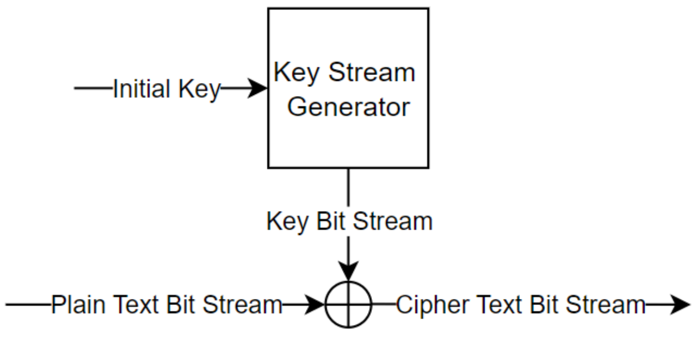

Figure 3: Stream Cipher Encryption

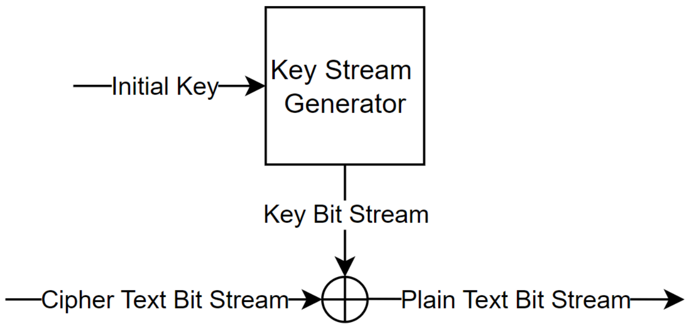

Figure 4: Stream Cipher Decryption

Block

The block cipher works similarly to the stream cipher with the major difference being that it works on a block of bits instead of one bit at a time. The encryption works by XORing the key block with a block of plain text to output a block of ciphertext (Figure 5). Then, thanks to the key block generator working the same way as the key stream generator, we are easily able to decrypt given the same initial key. With the same initial key, we can generate the same key block and then all we need to do is XOR that with cipher text to decrypt the message and output the original plaintext (Figure 6).

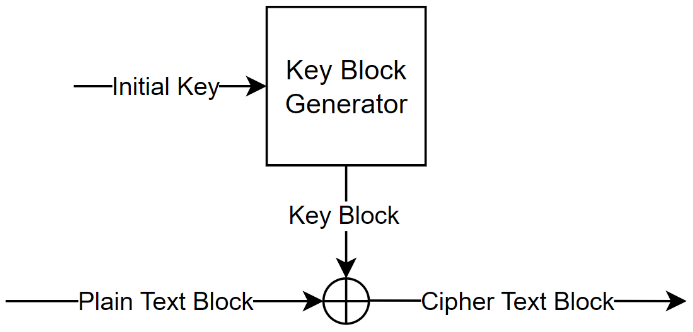

Figure 5: Block Cipher Encryption

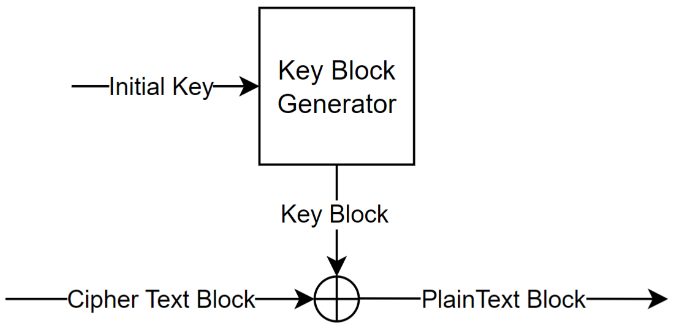

Figure 6: Block Cipher Decryption

Literature Review

Text Encryption using Extended Bit Circular Shift Cipher

The article addresses the escalating vulnerability of digital data to cyber-attacks in an era where internet-based data transmission is at its peak. It covers the significance of data security using cryptographic techniques, in specific, symmetric and asymmetric keys. Symmetric keys, although faster, are considered easier to attack because they rely on a single key. The article introduces a more secure encryption technique for the symmetric key Shift Cipher named ‘Extended Bit Circular Shift Cipher’. The proposed technique consists of three steps; encryption, decryption and evaluation. The encryption step involved bit shifting and character shifting based on a key. The decryption step inverts the previous operation to recover the plaintext. When comparing the Extended Bit Circular Shift Cipher to existing methods, the article states that the results showed a high avalanche effect value, low character error rate and minimal file size changes, indicating an effective decryption, out performing the existing techniques.

Stream cipher designs - a review

The article discusses the importance of efficient and secure information processing in various fields, including, e-commerce, e-government, cloud computing, and big data, as well as the role cryptography holds in meeting these demands. The article stresses the use of stream ciphers as cryptographic algorithms to build protocols for information confidentiality, highlighting their hardware and software implementation, speed, and limited error propagation advantages.

The article discusses the historical development of stream ciphers, referring back to Shannon’s one-time pad (OTP). Stream ciphers are presented as practical alternatives to OTP due to their easier key management and high-security margins. Stream ciphers are categorized into two parts, synchronous and self-synchronous, where synchronous is preferred for their known security properties. It goes on to address the phases of stream cipher execution, including initialization and keystream generation, and various stream cipher structures designed to meet specific security and implementation needs, such as LSFR-based, NSFR-based, FCSR-based, and others.

The article talks about how lightweight stream ciphers are suitable for resource-constrained applications, homomorphic encryption, and resistance to side-channel attacks, categorizing them based on their structures and analyzing their advantages and weaknesses.

The article introduces the basic components of stream ciphers, focusing on the role of cryptographic functions, particularly Boolean functions, in stream cipher design. Feedback shift registers (FSRs) are discussed, including linear and nonlinear FSRs, and basic operations used in stream cipher design are highlighted.

The article provides an overview of different structures of keystream generators in stream ciphers and which are considered outdated and the reason why (i.e., security vulnerabilities). It then goes on to discuss the design of specific stream ciphers such as LILI-128, A5/1, and MICKEY 2.0, including their structures and vulnerabilities.

Overall, the article provides insights into various stream cipher design approaches, their security, and potential vulnerabilities. It emphasizes the importance of finding a balance between security and efficiency.

The block cipher Square

In this research paper, the authors propose the SQUARE block cipher, characterized by a block length and key length of 128 bits. Notably, the modular design approach employed allows for seamless extensions to higher block lengths. The cipher exhibits a novel self-reciprocal structure, akin to that found in THREEWAY and SHARK [2, 15].

The chosen structure of the cipher, encompassing the types of building blocks and their interactions, is thoughtfully curated to enable highly efficient implementations on a diverse range of processors. The specific selection of building blocks is guided by the cipher's resilience against both differential and linear cryptanalysis.

Following the exploration of the cipher's structure and its implications for implementations, the authors elucidate the strategies employed to counteract linear and differential cryptanalysis. This is succeeded by a detailed exposition of an efficient attack that capitalizes on the unique properties embedded in the cipher's structure.

The paper makes it evident that it does not encourage anyone to use the Square today in sensitive applications.

Code Implementation

We chose to write the implementation of each cipher in Python for a couple of reasons. First off, Python is easy to understand for the average reader which will increase comprehension of the code itself. Secondly, which builds off the first point, is the bitwise operations that are typically used in cipher, Python has a very simple and easy-to-understand implementation of all the operations needed such as XOR.

We used one library called langdetect. This was only used to help reduce the amount of plausible outputs. This is done by detecting which strings are similar to the English language as there would be a large list of strings to go through when using a brute force attack. Thus increasing the user-friendliness of the output so the user wouldn’t have to look through them all.

For each cipher, we wrote three functions with the same intentions for each but unique to the different ciphers. Those three functions consist of one to encrypt, and decrypt and one to brute force attack the encryption to determine the plain text. The encryption functions take the plain text and the key, where the key value depends on the type of cipher being used, and then return the encrypted text. The decryption functions take the encrypted text along with the same key used for encryption and return the original plain text. Lastly, we have the brute force attack which goes through all the key options available within the keyspace and returns a list of strings that were detected to be English.

Conclusions

Cipher Attacks

Brute Force

One attack that works on the shift, stream, and block ciphers is a brute force attack. This involves going through all the available key options and deciphering the message with each key. Then the deciphered messages must be sorted through to determine which one is the message you are looking for. This method is not efficient especially on large sets of keys as the computation can take an exponential amount of time.

Character Frequency Analysis

One method that is simple and effective for shift ciphers is character frequency analysis. First, you go through the encrypted message and count the frequency of each character. You then take that data and match the encrypted messages' character frequencies to the known frequency in the real world and substitute them character for character. You could also try to determine the shift key value by comparing the two most frequent characters by finding the difference between them. You can then use that value as the key value for the decryption function. Therefore, once you have either substituted the characters or run the decryption function with the potential key value you will likely be left with a slightly or fully decrypted message.

Header Attack

Another method that can be effective is the header attack. It takes advantage of how packets/files can have the same information in certain sections. Therefore if you know the common part you can use that to try and compute the key used to encrypt the entire message. This would then allow you to decrypt the entire message.

Code Assessment and its Significance

With modern technology, it was simple to implement these different ciphers on the software level. Using all built-in functions and operations in Python we were easily able to convert values to binary and compute bitwise operations on them. This allowed us to quickly get a working implementation of the ciphers and common attacks. We were then also able to take advantage of modern packages which allowed us to make the brute force attack more practical. The package we used was able to sort through all the messages and determine which messages were most likely English. This made sorting through all the brute-force messages more practical. With this implementation and modern computing power, those attacks were able to be computed quickly. For example, the average time for the brute force attack on small messages with middle-tier hardware averaged around 1 second. That one-second average also included the time for the package to sort through all the messages and determine the more plausible responses.

After working through the implementation and seeing how easily the messages were able to be cracked, it shows how important the modern encryption we have today is. The computing power we have today would easily break some of the simpler ciphers that would typically take forever to do by hand. Whereas with the current encryption standards thankfully it would take many years for it to be cracked. We can then take advantage of modern computing power and techniques to increase the complexity and robustness of our ciphers.

Ciphers Today

Due to the nature of the shift cipher and its simplicity they are not used for much in the modern era other than to help people to learn and understand the basics of cryptography. Shift and block ciphers can still be used today but with modern standards. One encryption method where the block cipher is still used is in AES. AES is widely used across many fields and offers a strong encryption standard. Stream ciphers on their own are not typically used in modern encryption as they are more susceptible to attacks.

References

Daemen, J., Knudsen, L., & Rijmen, V. *The block cipher Square*. Fast Software Encryption: 4th International Workshop, 149-165, 2006.

Kostas, W. A. *A Brief History of Cryptography*, 2000.
H. N. Noor Muchsin, D. E. Sari, D. R. Ignatius Moses Setiadi and E. H.

Muchsin, H. N. N., Sari, D. E., Setiadi, D. R. I. M., & Rachmawanto, E. H. Text Encryption using Extended Bit Circular Shift Cipher. *2019 Fourth International Conference on Informatics and Computing (ICIC)*, 8138-8143, 2019.

Jiao, L., Hao, Y., & Feng, D. *Stream cipher designs: a review*. Science China Information Sciences, 1-25, 2020.