2024



**Contact Information:**

* **Email:** [ [beefine@quelti.tech](mailto:beefine@quelti.tech) ]
* **Website:** [ [github.com](https://github.com/BeeFine/QueltiLock) ]
* **Phone:** [ +7 999 613-44-02 ]

**CAKE Encryption: Principles and Practices**

A Comprehensive Guide to Implementing and Understanding CAKE

**Daniil Kyznetsov**

**TABLE OF CONTENTS**

[INTRODUCTION 2](#_Toc156379621)

[**SECURITY** 2](#_Toc156379622)

[**PERFORMANCE** 2](#_Toc156379623)

[**COMPATIBILITY** 3](#_Toc156379624)

[**DOCUMENTATION PURPOSE** 3](#_Toc156379625)

[THEORETICAL FOUNDATIONS 3](#_Toc156379626)

[**ALGORITHM DESCRIPTION** 3](#_Toc156379627)

[**CRYPTOGRAPHIC SECURITY** 4](#_Toc156379628)

[IMPLEMENTATION 5](#_Toc156379629)

[**IMPLEMENTATION OF THE ENCRYPTION METHOD** 5](#_Toc156379630)

[**IMPLEMENTATION OF THE DECRYPTION METHOD** 8](#_Toc156379631)

[**CONFIGURATION AND SETUP** 9](#_Toc156379632)

[**INTEGRATION WITH SYSTEMS** 9](#_Toc156379633)

[COMPREHENSIVE SECURITY ASSESSMENT 12](#_Toc156379634)

[**Brute Force Attacks** 12](#_Toc156379635)

[**Side-Channel Attacks** 12](#_Toc156379636)

[**Man-in-the-Middle Attacks** 12](#_Toc156379637)

[**Dictionary Attacks** 12](#_Toc156379638)

[**Physical Attacks** 12](#_Toc156379639)

[**Password Cracking Attacks** 12](#_Toc156379640)

[**Social Engineering Attacks** 12](#_Toc156379641)

[**Software Subversion Attacks** 12](#_Toc156379642)

[**Implementation Flaw Attacks** 12](#_Toc156379643)

[**Cryptanalytic Attacks** 12](#_Toc156379644)

[Known-plaintext Attacks 12](#_Toc156379645)

[Chosen-plaintext Attacks 12](#_Toc156379646)

[Chosen-ciphertext Attacks 12](#_Toc156379647)

# **INTRODUCTION**

This document provides a detailed description of the CAKE encryption method, designed to ensure data security. The method employs four independent pseudorandom number generators, ensuring a high degree of protection and reliability. The CAKE method is applicable in various scenarios, including the protection of confidential corporate information, encrypting users' personal data, or securing important documents. Additionally, CAKE can also be applied to encrypt files, partitions, and disks, making it a versatile solution for a wide range of information security tasks.

It is important to note that, despite having analogs in other encryption systems, the CAKE method possesses its unique characteristics in terms of application and implementation. This provides it with advantages in precision, reliability, and versatility of use.

## **SECURITY**

The CAKE encryption method has been developed with the ability to set a time delay for the decryption process. This can be useful in scenarios where temporary restriction of access to data is required. For instance, it is possible to delay the decryption of a file for a certain number of minutes or hours, preventing immediate access to confidential information. This feature adds an additional layer of security, particularly important for protecting information from unauthorized access.

Additionally, the use of four independent pseudorandom number generators significantly increases the level of protection. To break into an encrypted message, it is necessary to simultaneously crack all four generators, which poses a complex challenge for potential attackers. This gives the CAKE method high resistance against various methods of cryptanalysis.

Furthermore, it is important to consider that the actual time for encryption and decryption depends on the computing power of the computer on which the process is carried out. With the use of quantum computing systems, the speed of encryption could be significantly increased. However, implementing the CAKE method for quantum systems is an extremely complex task, especially considering the need to adapt the four pseudorandom number generators for quantum computing.

In the future, additional layers of security are planned to be developed, specifically designed to protect against potential threats associated with quantum computers. This will allow the CAKE method to maintain a high level of protection in the era of quantum technology development, providing a reliable solution for data encryption in the long term.

## **PERFORMANCE**

The CAKE encryption method is optimized for reliability and protection rather than for high-speed processing of large volumes of data. This makes it an ideal choice for tasks where security and data confidentiality are priorities. Despite this, thanks to the use of efficient algorithms and the capability of multithreading, the CAKE method is able to provide an acceptable speed of encryption and decryption, meeting the requirements of most practical applications.

Special attention is given to the balance between speed and security. Although the method may not be as fast as someother encryption systems designed for handling large data, it offers enhanced protection and resistance to cryptographic attacks. This makes the CAKE method particularly suitable for scenarios where data protection and prevention of unauthorized access are critically important.

Thus, CAKE is an effective solution for organizations and users who seek maximum protection of their information, even if it requires some compromise in terms of data processing speed.

## **COMPATIBILITY**

The method is file format neutral, as encryption occurs at the level of individual bytes. This allows it to be applied to various types of data, from text documents to multimedia content, without the need to make changes to the algorithm itself.

## **DOCUMENTATION PURPOSE**

This documentation is intended for technical specialists, developers, and cryptographers who are seeking a deep understanding of the CAKE encryption method. It covers both the theoretical foundations and practical aspects of applying the method, providing readers with all the necessary information for its effective use. This document thoroughly examines each element of the method, including its configuration, setup, and integration, and offers recommendations for optimal application in various conditions and usage scenarios. The primary goal of the documentation is to ensure a complete understanding of all aspects of the CAKE method, enabling specialists to effectively and safely apply it in their projects and systems.

# **THEORETICAL FOUNDATIONS**

## **ALGORITHM DESCRIPTION**

The **CAKE** *("Customizable Adaptive Key Environment")* encryption method is based on generating unique keys and sequences for data encryption. It employs four independent pseudorandom number generators, each initialized based on the user's input password and the number of sequences (by default, 16384 keys are generated). These generators create unique sequences that are applied to encrypt each individual byte of data. This approach ensures a high degree of protection and individualization of the encryption process, making each process unique and complicating the task for potential attackers attempting to decrypt. The effectiveness of the CAKE method lies in the combination of the uniqueness of the generated keys and the complexity of the algorithms used for their creation. This provides robust data protection against various types of attacks and makes the CAKE method a significant tool in modern cryptography.

## **CRYPTOGRAPHIC SECURITY**

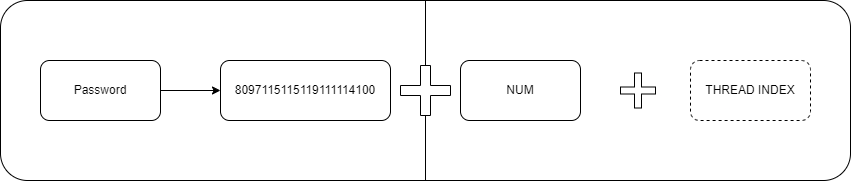
The cryptographic security of the CAKE method heavily relies on the use of CryptMT as the pseudorandom number generator. CryptMT provides high resistance to standard cryptanalytic attacks due to its advanced characteristics. It offers a reliable mechanism for generating random numbers with increased resistance to predictability and guessing, which is critical for data encryption.

However, like any cryptographic system, CAKE requires regular updates and adaptation to new threats and technologies to maintain its reliability and security.

In CAKE, it is possible to consider using hashing to process the password before using it as a seed for key generation. It is recommended to use the SHA-3 algorithm for password hashing, which provides an additional layer of security. However, this approach should only be applied if there is no need to store the hash itself for subsequent operations to avoid potential security threats associated with hash storage.

# **IMPLEMENTATION**

## **IMPLEMENTATION OF THE ENCRYPTION METHOD**

1. **Initial Setup and Seed Generation for PRNGs**:

*Figure 1: Seed Generation Scheme*

The first step in the CAKE encryption algorithm initialization process is the generation of an initial seed for four pseudorandom number generators (PRNGs). This seed is created by combining the user's password with the number of sequences specified by the user, thus ensuring uniqueness and enhancing the security of the generated key.

The transformation of the password into a numeric seed involves converting each character of the password into its numerical representation according to the ASCII table or any other character table. These numbers are then added to a vector that creates the numeric seed. A similar process is applied to the string representing the number of sequences as specified by the user, further populating the seed vector with additional numbers.

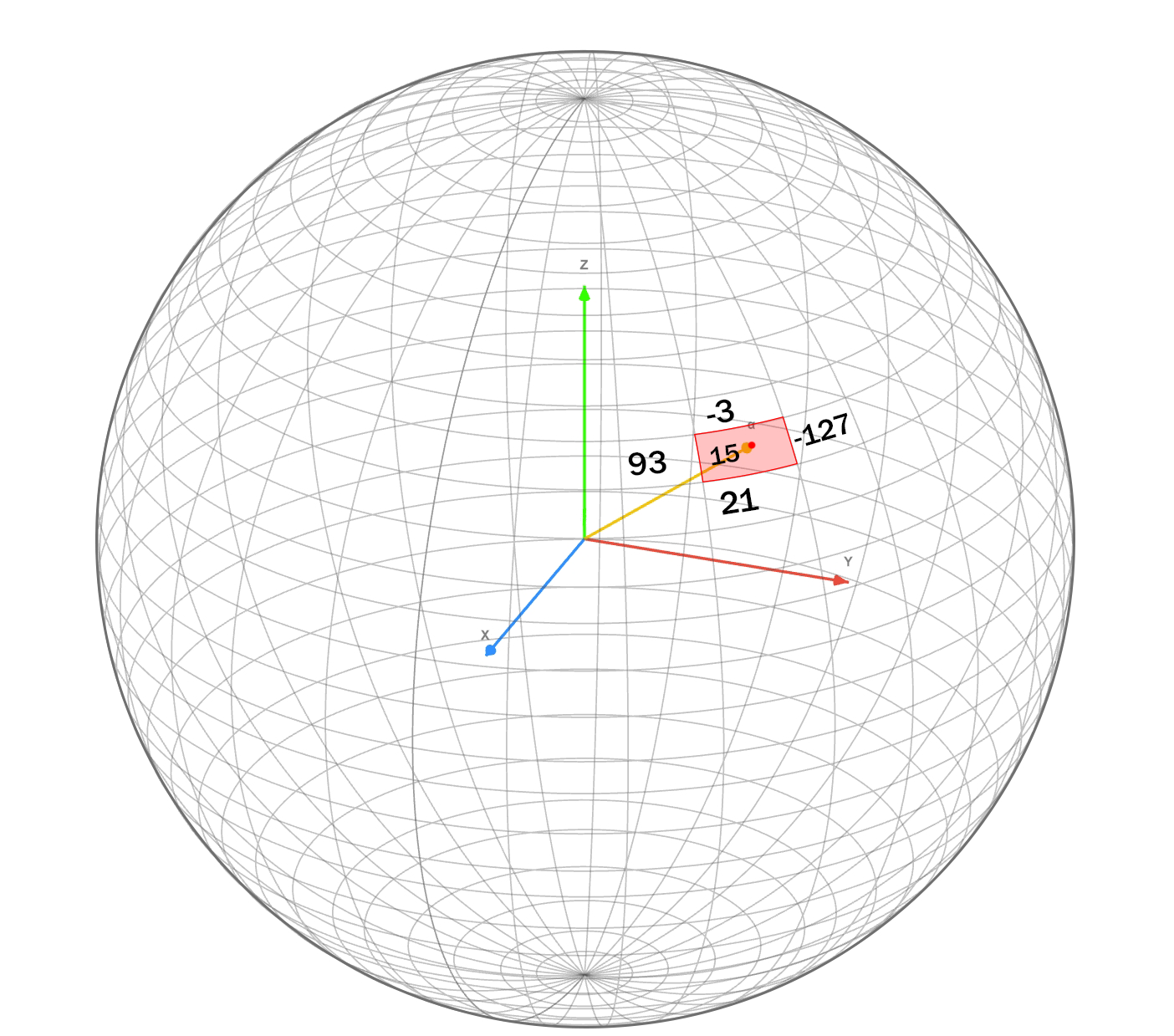
To further increase the level of protection and uniqueness of the seed, the password may be transformed using the cryptographic hash function SHA-3. In this case, the hash of the password obtained by using SHA-3 is concatenated with the string representation of the number of sequences to form the final numeric seed.

It is crucial to note that when using the SHA-3 hash, the hash of the password must not be stored anywhere, as possession of the hash by a malicious actor could lead to easy decryption of the file.Figure 1 illustrates the seed generation scheme.

1. **Generation of Unique Keys**:

Within the CAKE encryption method, key generation is conducted using four pseudorandom number generators, significantly enhancing the cryptographic robustness of the system. Three of these generators determine the coordinates of points on the surface of a sphere, with each point representing a potential encryption key. The fourth generator is used to create an array of numbers, each corresponding to different sectors on the sphere.

The coordinates generated by the first three generators point to a specific sector on the sphere, where the number chosen serves as the key. This number, corresponding to a cell in the spherical grid, is directly employed in the encryption algorithm. The selection process includes converting coordinates into an index within the array of numbers, requiring precise mathematical correlation between spherical coordinates and the linear arrangement of numbers.

The use of four independent pseudorandom number generators significantly complicates the predictability or compromise of the key generation sequence, thus increasing encryption security. The combination of spherical geometry and multiple pseudo randomness establishes a multilayered defense structure resilient to attacks based on statistical analysis or other known methods of cryptanalysis.

*Figure 2: Scheme of vectors*

The accompanying illustration (figure 2) visually represents this methodology, where each number on the sphere, denoted by a unique point, may be used as a key, and each vector originating from the center of the sphere points to one of these numbers.

1. **Multithreaded Sequence Generation**:

To increase data processing speed, the CAKE encryption algorithm utilizes multiple threads that perform sequence generation in parallel. A unique seed is generated for each thread, ensuring diversity in the sequences produced. The seed for each thread is determined as follows:

The string representation of the common seed used for the entire encryption process is concatenated with the string representation of the thread index.

The unique seed for the i-th thread () is formed by combining the string representation of the second seed () and the string representation of the thread index ().

For distributing the sequence generation tasks across threads, the following formula is used:

Each thread () is assigned a portion of work equal to the ceiling of the division of the total number of keys (), specified in the previous stage, by the total number of threads ().

When using this formula, it is necessary to ensure that the rounding process is performed correctly, so that all sequences are generated and evenly distributed among the threads. This guarantees the complete generation of the required number of keys without any omissions or overlaps.

1. **Encryption Process**:

Each byte of the file is encrypted by replacing it with the corresponding symbol from the selected sequence of keys. The procedure for selecting a sequence is as follows:

**Where:**

- is the i-th encrypted byte of data.

- is the sequence corresponding to the key generated for that particular byte.

- is the global set of all bytes.

- is the i-th original byte of data.

- is the function to search within the array of bytes, returning the position of the original byte, which is used to determine the corresponding encrypted byte in the sequence.

The i-th key is selected from the overall set of generated keys.

This key is associated with the generated sequence corresponding to it. For ease of association, it is recommended to use data containers such as std::map or similar structures.

For each byte, its position in the overall set of bytes is determined, and then it is replaced with a symbol from the corresponding sequence:

The encrypted byte () is derived from the sequence () corresponding to the key () generated for that particular byte. The position of the original byte in the overall array of bytes () is determined by the search function (), which returns the index of the original byte. This index is used to determine the corresponding encrypted byte in the sequence.

A key aspect is maintaining a strict correlation between the position of a byte in the original data and the position of the corresponding symbol in the selected key sequence. This ensures the integrity of the encryption and allows for the original data to be restored during decryption.

To ensure the correctness of the encryption process, each byte of the original data must be precisely replaced with the corresponding byte from the selected sequence. Errors in determining byte positions or their replacement can lead to the inability to restore the original data and consequently, to information loss. Therefore, it is critically important to thoroughly check the logic of mapping and replacing bytes in the algorithm.

This encryption technique achieves a high level of security by using unique sequences for each byte, which significantly complicates the task of cryptanalysis and increases the algorithm's resistance to various types of attacks.

1. **Writing Encrypted Data**:

At the conclusion of the encryption process, the transformed bytes, now securely encrypted, are sequentially compiled. This stream of encrypted data is methodically written to the target output file, which is then marked with the ***.cake*** extension. This extension serves as a hallmark of the CAKE encryption method, signifying that the contents are encrypted and require the appropriate decryption procedure to revert to their original, accessible form.

## **IMPLEMENTATION OF THE DECRYPTION METHOD**

**1-3.** **Initialization, Key Generation, and Sequence Generation**:

The decryption process begins similarly to the encryption process: pseudorandom number generators are initialized, unique keys are generated, and multithreaded sequence generation is carried out.

1. **Decryption Process**:

Decryption is performed using the following formula:

In this formula, the positions of and are swapped compared to the encryption process, ensuring accurate restoration of the original data.

1. **Writing Decrypted Data**:

The decrypted bytes are sequentially written to a new output file. This ensures the restoration of the original file in its initial form, guaranteeing the integrity of the data.

It is important to note that successful decryption requires precisely replicating the key generation process used during encryption. This includes using the same initial values and parameters for the pseudorandom number generators. Any discrepancies in the key generation process can lead to the inability to correctly decrypt the data.

The decryption process requires accuracy and attention to detail, as any error in restoring key sequences or applying the decryption formula can lead to data distortion. Therefore, it is crucial to ensure that each stage of the decryption method is meticulously configured and verified to guarantee the error-free restoration of the original information.

## **CONFIGURATION AND SETUP**

The central element in the configuration of the CAKE encryption method is the use of four independent pseudorandom number generators. This approach ensures an enhanced level of security due to the diversity and complexity of the generated sequences. Proper setup and initialization of each of these generators are critically important for ensuring the cryptographic robustness of the method.

*(Addition: It is also vital to carefully select the generators themselves; they must be cryptographically secure and deterministic to ensure the necessary level of security and predictability in the encryption process. This is essential for guaranteeing accurate decryption. If the key generation algorithm cannot be replicated precisely, it could lead to decryption errors and the inability to recover the original message.)*

Special attention should be given to generating unique initial values (seeds) for each of the generators. These seeds must be chosen in such a way as to eliminate any predictability and to ensure maximum randomness in the generated keys. Using complex combinations of the user's password and other variable data to create seeds significantly enhances the security of the method.

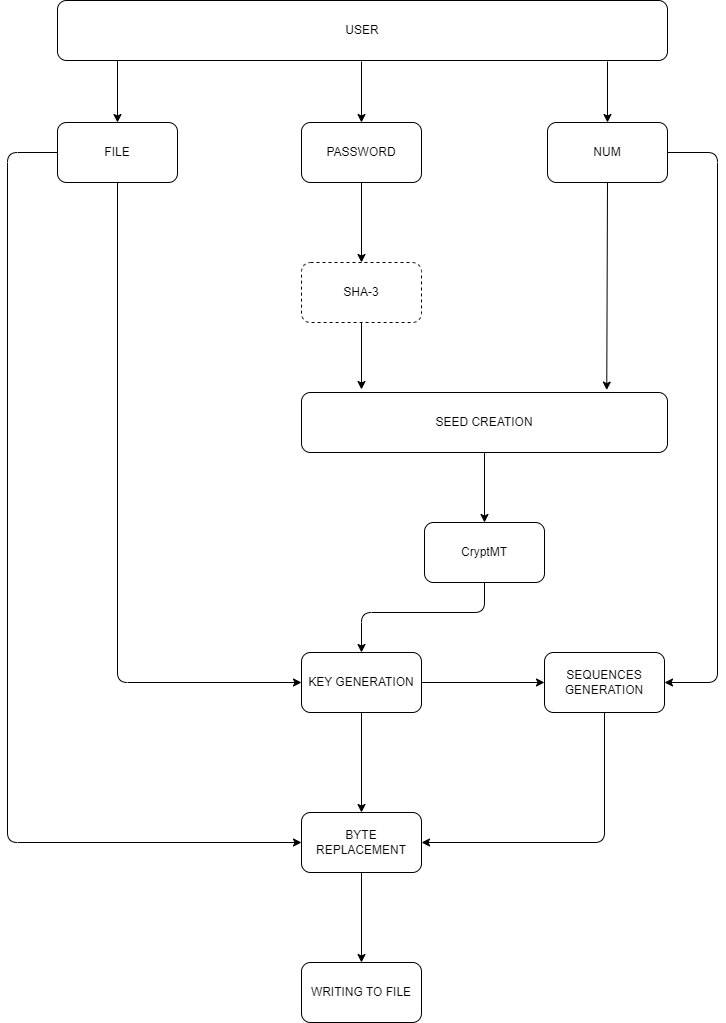
In addition, it is important to configure the key parameters, which determine how data will be encrypted and decrypted. This includes selecting the key length and the mechanism of its generation, which directly affects the complexity of cracking and the level of security.

Furthermore, the possibility of integrating the algorithm into various systems should be considered. The flexibility in setting encryption parameters allows for the adaptation of the CAKE method to specific requirements and conditions of use, making it suitable for both small mobile devices and large-scale corporate systems.

## **INTEGRATION WITH SYSTEMS**

The CAKE encryption method, with its robust and adaptable design, presents an ideal solution for local encryption needs. Its utility is particularly evident in scenarios requiring secure data storage and transfer, such as on USB flash drives. By encrypting data locally on these devices, the CAKE method significantly enhances security, safeguarding against unauthorized access and data breaches.

Furthermore, the inherent versatility of the CAKE method allows for seamless integration into a wide array of systems. Its fundamental principles and mechanisms can be readily adapted and rewritten to suit various platforms and environments. This flexibility makes it a valuable tool for developers looking to implement high-level encryption in diverse systems, ranging from small-scale applications to complex enterprise solutions. With the ability to tailor the method to specific requirements, CAKE offers a comprehensive encryption solution that combines reliability, efficiency, and adaptability.

Figure 3: General Scheme of the CAKE Method Encryption ProcessНачало формы

# **COMPREHENSIVE SECURITY ASSESSMENT**

This section describes the resilience of the CAKE method to major types of threats.

## **Brute Force Attacks**

## **Side-Channel Attacks**

## **Man-in-the-Middle Attacks**

## **Dictionary Attacks**

## **Physical Attacks**

## **Password Cracking Attacks**

## **Social Engineering Attacks**

## **Software Subversion Attacks**

## **Implementation Flaw Attacks**

## **Cryptanalytic Attacks**

### Known-plaintext Attacks

### Chosen-plaintext Attacks

### Chosen-ciphertext Attacks