SOFTWARE REQUIREMENTS SPECIFICATION

for

Chaos-based-AES-cryptosystem

Version 1.0

Prepared by :
1. Dhruv Choksi (202001049)
2. Kandarp Devmurari (202001052)
3. Pranav Patil (202001055)
4. Krupal Shah (202001057)
5. Vihar Shah (202001110)

Submitted to :
Prof. Manish Gupta

May 1, 2023

Contents

1 Introduction

1.1 Purpose

The present era of information and data gives us an enormous amount of opportunities for data transfer, data storage, and analysis of sensitive data. This also exposes us to many cyber-attacks and issues related to confidentiality, data privacy, authentication, and other such risky events.
To address these issues, data is encrypted before storing or communicating to reduce data-related security issues. Currently, data storage can be done by encrypting the data with an AES encryption system. There are other methods too.
In the system, we aim to focus on the problem of encryption for data that is supposed to be stored and read by only authorized personnel. The current process of generating the master key in the AES encryption system is pseudo-random. The method we aim to build should increase the randomness of the master key or round key generation, which the AES uses. This will increase the cryptanalysis time by a considerable amount for already difficult AES systems.

1.2 Intended Audience and Reading Suggestions

This SRS doc is intended for all the people who are interested in working out the complexity of the system proposed by us. It is a vital piece of document for anyone working on building the system out of it, and everyone tried to perform a cryptanalysis of the system or verify the performance of the proposed system. This document is the most important piece of document for developers developing the system.

- **Introduction:** The purpose of this project is simple. Increase the difficulty and randomness of the AES cryptosystem and its key generation process. This is to be achieved by using a double pendulum. This is a perfect example of a pseudo-random process that can not be replicated unless all the initial conditions are known to at least the 6th decimal point accuracy.
- **Scope:** This project's scope lies in encrypting the image or text to be stored in a local machine or cloud storage. This process heavily improves security against data theft.
- **Functional and non-functional requirements:** Clear requirements are encrypting the data before storing, key generation and its randomization, and downloading of a key to the local machine.

- **Constraints and assumptions:** User must have a powerful enough computer that could process the data in considerable amount of time.
- **Acceptance criteria:** Successful encryption and decryption of text and image.

1.3 Project Scope

The scope of the proposed cryptosystem that uses a double pendulum system to generate random binary sequences and create keys for the AES cryptosystem is to provide a secure and efficient means of generating random binary sequences and creating keys for encryption and decryption of sensitive data. The product addresses the need for a reliable and unbiased method of generating keys for encryption and decryption.

The product primarily focuses on providing a key generation module that uses the double pendulum system to generate random binary sequences accurately and reliably. The generated binary sequences should be of high quality and should not be biased.

Additionally, the product comes with an encryption module that utilizes the binary sequences generated to produce keys that can be employed to encrypt and decrypt data using the AES cryptography algorithm. The encryption module should be able to handle large data sets and should be efficient and secure.

The product is designed to be user-friendly, with a clear and intuitive interface that allows users to generate keys easily and encrypt and decrypt data. The system should have robust security measures to ensure user data's confidentiality and integrity. The system should use secure protocols and encryption algorithms and have mechanisms to prevent unauthorized access to user data.

2 Overall Description

2.1 Product Perspective

The proposed cryptosystem is a software application that generates a random binary sequence using the chaotic motion of a double pendulum system. The generated sequence is then used to create keys for the AES cryptosystem. The product is designed to provide a highly secure and robust encryption technique that can be used for various applications where data security is a critical concern.

The product will be developed using state-of-the-art programming languages and libraries to ensure compatibility with different operating systems and devices. The application will have an intuitive user interface, making it easy for users to generate and use keys for encryption and decryption.

The product will be a standalone application that can be used independently or integrated with other software systems. The product will be tested extensively to ensure its accuracy and reliability, and it will be continuously updated to stay current with the latest security standards.

2.2 Product Functions

The proposed cryptosystem will have the following functions:

1. **Double Pendulum System:**
   The application will have a double pendulum system that generates a random binary sequence. The system will be designed to ensure that the generated line is highly random and unpredictable, making it impossible to break the encryption.

2. **Key Generation:**
   The application will generate keys using the random binary sequence generated by the double pendulum system. The keys will be used for AES encryption, ensuring high-level security for the data.

3. **Encryption:**
   The application will provide AES encryption functionality to encrypt the user's data using the generated keys. The encryption process will be fast and efficient, ensuring that the user's data is secure from prying eyes.

4. **Decryption:**
   The application will provide AES decryption functionality to decrypt the encrypted data using the generated keys. The decryption process will be seamless, ensuring that the user can access their data quickly and easily.

5. **User Interface:**
   The application will have an intuitive user interface that makes it easy for users to generate keys, encrypt and decrypt data, and access other application features.

6. **Compatibility:**
   The application will be designed to be compatible with different operating systems and devices, ensuring that users can use it on their preferred devices.

7. **Security:**
   The application will be designed to ensure that the user's data is secure and protected from unauthorized access. The generated keys will be highly random and unpredictable, making it impossible to break the encryption

8. **Continuous Improvement:**
   The application will be continually updated to ensure that it stays current with the latest security standards and remains reliable and accurate.

2.3 User Classes and Characteristics

The proposed cryptosystem is designed for use by various user classes, each with their own characteristics and requirements. The user classes and their characteristics are described below:

- **Individuals:** This user class includes individuals who need to secure their personal data, such as financial records, health records, or personal correspondence. These users may not have extensive technical knowledge, so the application should have an easy-to-use interface and provide clear instructions for key generation and encryption/decryption.

- **Small Businesses:** Small businesses may use the application to secure their sensitive data, such as financial records, client information, or proprietary data. These users may have some technical knowledge but may not have a dedicated IT department, so the application should be easy to use and require minimal configuration.

- **Large Corporations:** Large corporations may use the application to secure their confidential data, such as financial records, customer data, or trade secrets. These users may have a dedicated IT department and require advanced features like integration with existing security systems or custom key generation algorithms

- **Government Agencies:** Government agencies may use the application to secure sensitive information such as classified documents, intelligence reports, or diplomatic communications. These users may require the highest level of security and may have specialized technical knowledge, so the application should be customizable and provide advanced security features.
- **Security Professionals:** Security professionals, such as cybersecurity experts or penetration testers, may use the application to test the strength of existing security systems or to analyze the effectiveness of encryption algorithms. These users may require advanced features such as custom key generation or analysis tools.

2.4 Operating Environment

The proposed cryptosystem is a software application that generates random binary sequences using the chaotic motion of a double pendulum system and uses these sequences to create keys for the AES cryptosystem. The application's operating environment includes hardware, software, and network requirements, which are described below:

1. **Hardware Requirements:**
   The hardware requirements for the application include a computer or mobile device with a processor capable of running the software efficiently. The system should have sufficient RAM and storage capacity to handle the application's data and processes.
   The application requires a double pendulum system to generate random binary sequences, which can be implemented using a physical double pendulum apparatus connected to the computer. Alternatively, the application can simulate a double pendulum system using mathematical models and algorithms.

2. **Software Requirements:**
   The software requirements for the application include an operating system that is compatible with the software. The application can be developed using various programming languages and libraries, such as Python, C++, or Java, and the interface could be created using any web or app-making technologies
   The application may also require additional software components, such as libraries for numerical calculations (i.e., for running the double pendulum simulation), visualization tools (to view the generated simulation), and cryptography algorithms. These components must be compatible with the operating system and the programming language used to develop the application.

3. **Network Requirements:**
   The application may require network connectivity to communicate with other systems or devices. For example, the application may need to access a remote server to store or retrieve encrypted data.
   To ensure the security of the communication, the application should use secure protocols, such as Design and Implementation Constraints

2.5 User Documentation

The proposed cryptosystem is a software application that uses a double pendulum system to generate random binary sequences and creates keys for the AES cryptosystem. User documentation is essential to ensure users can easily understand and use the application. The user documentation should include the following information:

1. **Installation/Working Guide:**
   The installation guide should provide step-by-step instructions for running the application on the user's computer or mobile device. This guide should include information on system requirements, software dependencies, and any other application installation prerequisites.

2. **User Manual:**
   The user manual should provide a detailed overview of the application's features and functionality. It should explain the process of generating keys using the double pendulum system, how to use the keys to encrypt and decrypt data, and how to customize the application's settings.
   The user manual should include screenshots or videos illustrating key features and a troubleshooting guide to help users resolve issues.

3. **FAQ Section:**
   The FAQ section should address common questions, and issues users may have when using the application. This section should include answers to questions such as:
   - How secure is the generated key?
   - How to retrieve your encrypted file and get your data back?

4. **Security Guide:**
   The security guide should explain how the application ensures the confidentiality and integrity of user data. This guide should include information on the encryption algorithms used, how the key is protected, and any other security measures implemented in the application.

5. **Support Contact Information:**
   The user documentation should provide contact information for users to get support, such as email addresses or phone numbers for technical support or customer service.

2.6 Assumptions and Dependencies

The proposed cryptosystem that uses a double pendulum system to generate random binary sequences and create keys for the AES cryptosystem depends on several assumptions and dependencies. These are outlined below:

1. **Hardware Dependence:**
   The system relies on a physical double pendulum apparatus or a simulation of a double pendulum system using mathematical models and algorithms. Therefore, the system's functionality depends on the hardware or simulation's accuracy and reliability.

2. **Mathematical Model Dependence:**
   The system's ability to generate random binary sequences using a double pendulum system depends on the accuracy of the mathematical models and algorithms used. Any inaccuracies or errors in the models can lead to inaccurate or biased binary sequences.

3. **Cryptography Algorithm Dependence:**
   The system relies on the AES cryptography algorithm to encrypt and decrypt data. The system's security and effectiveness depend on the strength and reliability of the AES algorithm.

4. **Operating System Dependence:**
   The system's functionality depends on the operating system's compatibility with the software used to develop the application. The application must be compatible with the user's operating system to function correctly.

5. **Network Dependence:**
   The system may require network connectivity to communicate with other systems or devices. The system's ability to communicate securely depends on the network's availability, reliability, and security protocols.

6. **User Skill Level Assumption:**
   The system assumes users understand cryptography, random number generation, and computer systems. Users with little experience in these areas may require additional training or support to use the system effectively.

7. **Security Assumptions:**
   The system assumes that the user's computer or mobile device is secure and free of malware or malicious software. Any compromises to the user's device can lead to security breaches and compromise the system's effectiveness.

3 External Interface Requirements

3.1 User Interfaces

The user interface for the proposed cryptosystem should be intuitive and user-friendly. The interface should allow users to generate keys easily and encrypt and decrypt data. The following are the user interface requirements:

- The interface should be easy to navigate.
- The interface should be responsive and have low latency.
- The interface should provide clear and concise instructions for generating keys and encrypting and decrypting data.
- The interface should display the progress of key generation and encryption/decryption processes.
- The interface should allow users to customize key generation parameters if desired.
- The interface should have error-handling mechanisms in place to alert users of any errors.

3.2 Hardware Interfaces

- The software product requires a computer with a minimum of 4GB RAM and a processor with a clock speed of at least 1GHz.
- The software product requires a monitor with a minimum resolution of 1024x768 pixels.
- The software product requires a keyboard and a mouse for input.
- The software product does not require any special hardware components such as GPUs or ASICs.
- The software product will require an active internet connection for downloading encrypted files and keys and uploading the files to be encrypted.

3.3 Software Interface

- The software will be compatible with multiple web browsers, including Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge.
- The software will be compatible with various operating systems, such as Windows, MacOS, and Linux.
- HTTP and HTTPS will be the standard communication protocols for data transfer.

3.4 Communications Interfaces

The system will use secure communication protocols when transmitting data i.e., text file or image file over the internet or other networks.

The system is such that when the user opens the interface, a session token is generated, valid for 2 minutes. Suppose the user doesn't upload the file. In that case, the session expires, and the user has to create a new session again, thus, providing a mechanism to prevent unauthorized access to user data during transmission.

4 System Features

This proposed system should serve to protect the data of users from getting breached while it is on the cloud or in any of the storage. Image and text are the supported data type for encryption. Data security is at the forefront of requirements.

4.1 Description and Priority

The proposed system should have the following features:
The features with priority up to down -

1. **Text Encryption:** This is the system's most important feature. The system should allow users to encrypt their text data in the form of images for protection. This image can then be stored at any convenient place as its size is quite small and encrypted.

2. **Image Encryption:** This is also one of the two types supported. The image should be encrypted into another image and become unidentifiable. The encrypted image can be later put on cloud storage which are generally not encrypted.

3. **Allow users to upload keys:** In the default mode, new keys are generated for every encryption request. We also allow users to upload the pre-created key of this system only to encrypt another round of data with the same key for convenience. Although, this brings down the security factor. Also, the key is checked for its correctness.

4. **Local storage:** The keys and encrypted images generated by the system are automatically downloaded into the user's computer. The path to it is shown on the result screen. This improves security as no data is stored on the server, which provides an additional security layer.

5. **Tokenization:** The tokenization system generates a new key for each user for every encryption request. This token stays alive for 2 minutes or some limited time to prevent any automated attacks and malicious use of the key by someone other than the legitimate data owner.

6. **Flexibility:** Users are allowed to change the name or path of the file according to their convenience, which makes it easier for them to manage the keys and corresponding encrypted images.

4.2 Functional Requirements

The functional requirements of the system are as follows:

1. **Random binary sequence generation:** The system should generate random binary sequences using the double pendulum system. The double pendulum system should be designed to produce chaotic and unpredictable motion to generate truly random binary sequences. The generated binary sequences should have equal probability of 0 and 1.

2. **Key generation:** The system should generate cryptographic keys using the random binary sequences generated by the double pendulum system. The system should be designed to generate keys that are sufficiently long, and the strength of the generated keys should be tested using standard cryptographic tests. The system should also provide an option for the user to input their own keys.

3. **Encryption:** The system should encrypt plaintext using the generated keys and AES algorithm. The system should input plaintext and apply the AES encryption algorithm with the generated key to produce ciphertext. The system should also provide options for the user to choose the AES encryption mode and padding scheme.

4. **Decryption:** The system should decrypt ciphertext using the generated keys and the AES algorithm. The system should input ciphertext and apply the AES decryption algorithm with the generated key to produce plaintext. The system should also handle any errors that may occur during the decryption process.

5. **User interface:** The system should provide a user interface for the user to input plaintext and receive the corresponding ciphertext. The user interface should be easy to use and should provide options for the user to choose the encryption mode, padding scheme, and key length. The system should also provide feedback to the user about the encryption and decryption process.

   The system is built using Node.js and Express Template Engine for the web and views part, while the encryption and decryption algorithms are worked out by the Python scripts.

6. **Hardware compatibility:** The system should be compatible with the hardware used for the double pendulum system. The hardware should be able to capture the motion of the double pendulum system accurately and provide input to the method for generating random binary sequences.

7. **Software compatibility:** The system should be compatible with the software used for the double pendulum system. The software should be able to process the input from the hardware and generate random binary sequences according to the designed algorithm.

8. **Error handling:** The system should handle any errors that may occur during the encryption or decryption process. The system should provide error messages to the user in case of any error and provide suggestions for resolving the error.

9. **Key management:** The system should manage the generated keys securely and allow the user to change the keys. The system should also store the keys securely and prevent unauthorized access to the keys.

10. **Security:** The system should ensure the data's confidentiality and integrity. The system should use a secure encryption algorithm with strong keys to ensure that unauthorized parties cannot intercept or modify the data. The system should also use secure key management practices to prevent unauthorized access to the keys.

5 Other Nonfunctional Requirements

5.1 Performance Requirements

Time taken by the system is one of the main performance requirements of the system. It is also the bottleneck as the cryptography processes take time to perform computations even with dedicated GPUs. It is generally the case that cryptosystems of higher quality take more time due to the complex and lengthy calculations involved.

5.2 Security Requirements

The data of users should be safe. The encrypted image and the corresponding key. And what place is safer than the user's own hardware? The encryption standard should be high enough, which could make an attack almost impossible over the system. The native AES system is already providing quality cryptography and data security but there's never an end. The proposed system should be an even better cryptosystem if it were to replace the current AES.

5.3 Software Quality Attributes

Exhaustive testing is the most important part of any encryption-related algorithm. The complexity analysis of crypt analysis is important to prove the security of any system and what it has to offer.
Certain UI, ease, and logical ability of navigation are important too.

5.4 Business Rules

We clearly propose to store none of the user data on server machines. All the required data are taken from the user as one-time input, processed, and returned to the users. The data is downloaded into the user's computer, and every bit of data is deleted from the server.
Save time, improve efficiency, and increase the expected time to attack the system.

5.5 Other Requirements

A user-friendly website that can be used as an interface to upload and see files and data.
A faster server that can increase the processing speed.

6 References

1. Stachowiak, Tomasz, and Toshio Okada. "A numerical analysis of chaos in the double pendulum." Chaos, Solitons & Fractals 29.2 (2006): 417-422.

2. Shruthi, K. M., S. Sheela, and S. V. Sathyanarayana. "Image encryption scheme with key sequences based on chaotic functions." 2014 International Conference on Contemporary Computing and Informatics (IC3I). IEEE, 2014.

3. Liu, Shubo, Jing Sun, and Zhengquan Xu. "An Improved Image Encryption Algorithm based on Chaotic System." J. Comput. 4.11 (2009): 1091-1100.

4. `https://link.springer.com/article/10.1007/s11277-020-07052-4`