# Introduction

Our task is to implement an Encryption/Decryption algorithm that encrypts and decrypts the following file formats, images, text files, RAR or zip files, and PowerPoint using a password on an intuitive user interface. We have decided to make use of Python programming language to accomplish this task. Amongst plentiful of choices to accomplish the task, our team decided to make use of a password, SHA256 Hashing, cipher and AES encryption algorithm to both encrypt and decrypt files.

# Processes

# Encryption process

Our teams algorithm first requires a user to set their desired method of encryption. Set a password to input to encrypt and decrypt a file. The user opens a file directory to select a file path. By clicking Browse after locating the file to encrypt in the file directory, our algorithm obtains the file path as a location to save the decrypted file and later remove the original file. Now, the user can initiate the process of encryption by clicking the encrypt button. Before the process begins, the user is prompt to enter the initial password set. The algorithm then converts the password into bytes, as the hash library requires bytes as an input. The hash, is then used to construct the poly-alphabetic substitution for the cipher. The contents of the file are, first, ciphered using the poly-alphabetic substitution, then, encrypted using the Advanced Encryption Standard (AES). The algorithm saves the encrypted file, removes the original file, and also extends the file with the “.aes” file type - which is specific to our algorithm process. Lastly, the algorithm gives a report whether the encryption was a success or not with a pop up, and a status of encryption - if whether the process is done or not.

Process for encryption (.txt, .rar, .zip, .pptx and image files)

Encryption

Set method of

encryption

Set a password

File is selected

Encryption initiated

By the press of

Encrypt button

Enter the initial

password

Results from

encryption are

a save

(as .aes file)

SHA256 generated

From password

Cipher results

Encrypts with

AES

Original

unencrypted file

Is removed

# Decryption process

The algorithm requires a user to select a file path with the “.aes” extension. The password used to encrypt the file initially, and the decryption method to the selected in order to decrypt a file. The algorithm constructs the hash from the password and stores the hash in memory, decrypts the “.aes” file content using AES. Then, uses the hash to generate the poly-alphabetic substitution, which is used to decrypt the previously encrypted file back to its original contents. The algorithm removes the “.aes” and a new file is stored as the original file before encryption.

Process for decryption (.txt, .rar, .zip, .pptx, and image files that are encrypted as “.aes”)

Decryption

Method of

decryption is set

Password is

set

Process is

Initiated by

Pressing the

decryption button

.aes file is

selected

Decryption result

is saved as

Original file

extension

.aes file content

decrypted with

AES

SHA256 Hash

generated from

password

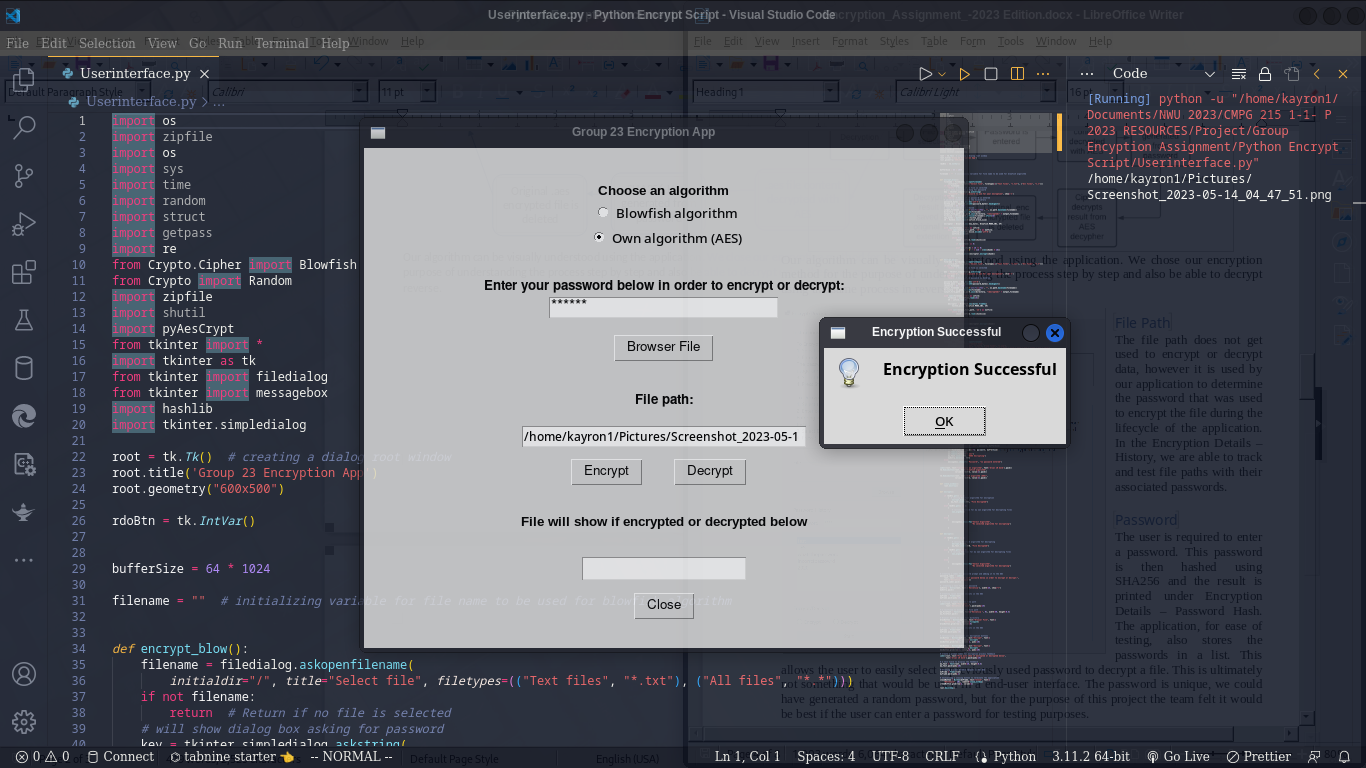
Original .aes

encrypted file is

deleted

Our algorithm can be visually understood using the application. We chose our encryption method for the purpose of understanding the process, step by step, and also be able to decrypt using the same process in reverse.

# File Path

The file path is not really used up in the process of encryption and decryption of data, however, is used by our application to determine the file path to access and save or delete the encrypted or decrypted file during the life-cycle of the application. Figure 1.1 Shows the User interface of the encryption Application designed.

*Figure1.1*

# Password

The user is required to set a password. The password is hashed using SHA256 hash. For the purpose of our project, we felt that the use of password has to be unique in a way that the user sets password before selecting a file path as well as before initiating the process of encryption.

# SHA256 HASH

The password is hashed with the SHA256 hash and is stored in the memory, and passed to the next phase of the process. The hashed password allows for deeper, more extensive encryption as it results in a string that is not easily understood (poly-alphabets). It is considered as an extra layer to the password strength and is then also used to extensively cipher the content of the file.

# Cipher

For every letter in the alphabet set, a new alphabet is generated using the hash. The hash determines the number of bytes every letter or might be a symbol requires moving both on a horizontal and vertical alphabet table.

# AES

The Advanced Encryption Standard on its own employs an array of encryption methods and stands as an encryption algorithm on its own. Our algorithm uses AES as the final step to encrypt the contents of a file. When the file is encrypted, the content is generated in the aforementioned Ciphered poly-alphabetic substitution, this content is not displayed on the interface. The poly-alphabetic substituted content is then encrypted using AES and this result can then be seen in the center white space below encrypt and decrypt buttons. AES process involves KeyExpansion, an initial round of combining each byte with a byte of the round key, and then up to 13 rounds of SubBytes, ShiftRows, MixColumns and AddRoundKey (Daemen & Rijmen, 1999). The final round consists of creating the final SubBytes, ShiftRows and AddRoundKey. Every step can be illustrated by the Visualization of the AES round function A picture containing text

Description automatically generated(Daemen & Rijmen, 1999):

# Comparison

Compared to standard algorithms, our encryption/decryption is a combination of standard encryption algorithms. Generally, a constant key is set by applications that is to be used to hash and encrypt using the variety of methods such as Triple-DES, DES, AES, Blowfish, RSA, Twofish and Ciphering. Our algorithm generates this key dynamically and uses it in conjunction with AES and Ciphering to encrypt and decrypt files. AES and Blowfish are symmetric ciphers as they use one key for both encryption and decryption (Agrawal e*t al*., 2012). In terms of speed, the blowfish and AES are considered efficient and fast in encrypting and decrypting files (Karthikeyan & Kumar, 2012). However, AES is currently one of the strongest encryption algorithms since it combines and balances speed and security properly.

# Advantages and Disadvantages

AES allows us to encrypt using a block size of 128-bit, as opposed to Triple-DES encrypting only at 64-bit block size. Encryption with AES is faster than that of any DES and ultimately is more secure than DES (Penchalaiah & Seshadri, 2010:1641; Abdulgader *et al*., 2015:2).

Some of the **disadvantages** includes the implementation and the fact that the algebraic structure is simple. Every block is always encrypted the same way, which ultimately makes AES a little more insecure if the hacker can find out what the key is that is used to encrypt. Even as though AES is a renowned block cipher that has several advantages in data encryption, yet, however, it is not suitable for real-time applications (Lakshmi *et al*., 2016:1). Some image encryption methods cannot meet the demands of the image encryption, such as Arnold cat map and Hilbert transformation (Luo *et al*., 2006:349). Again, Triple-DES’s main drawback is its slow software implementation, for both efficiency and security, and a large block size is desirable (Penchalaiah & Seshadri, 2010:1641). Penchalaiah & Seshadri (2010:1642) emphasis how AES has an excellent key setup time and good key agility, but, more importantly, without sacrificing performance. Performance is vital when it comes to encryption algorithm and it serves as one of the criteria used on their selection. Hence, AES is the encryption method we chose to employ for our task, it is for its distinct features. It is the most powerful encryption algorithm in the first level of security, which is very complex to break (Damodaram *et al*., 2009:768).

# Conclusion

Our algorithm is considered secure and dependable when encrypting and decrypting the required file formats. The user interface is intuitive to allow simple and illustrative encryption and decryption. Our encryption algorithm allows for various levels of security, from a custom user-set password, hashing, ciphering and finally AES standards, each required to successfully decrypt a file.

# Reference list

Abdulgader, A., Idbeaa, T., Ismail, M., & Zainal, N. 2015. Enhancement of aes algorithm based on chaotic maps and shift operation for image encryption. *Journal of theoretical and applied information technology*, 71(1):1-12. <http://www.jatit.org/volumes/Vol71No1/1Vol71No1.pdf>

Agrawal, M., & Mishra, P. (2012). A comparative survey on symmetric key encryption techniques. *International Journal on Computer Science and Engineering*, *4*(5), 877.

Atteya, A.M., & Madian, A.H. 2014. A hybrid chaos-aes encryption algorithm and its impelmention based on fpga. *IEEE*. (2014 IEEE 12th International on New Circuits and Systems Conference (NEWCAS)) pp. 217-220. **doi:**[10.1109/NEWCAS.2014.6934022](https://doi.org/10.1109/NEWCAS.2014.6934022)

Damodaram, A., Sridevi, R., & Narasimham, S. 2009. Efficient method of audio steganography by modified lsb algorithm and strong encryption key with enhanced security. *Journal of theoretical and applied information technology*. Pg 768-771.

Deamen, J. & Rijndael, V. 1999. AES Proposal: Rijndeal. Gaithersburg, Maryland, USA. <https://www.cs.miami.edu/home/burt/learning/Csc688.012/rijndael/rijndael_doc_V2.pdf>

Karthikeyan, S., Kumar, M. A. (2012). Investigating the efficiency of Blowfish and Rejindael (AES) algorithms. *International Journal of Computer Network and Information Security*, *4*(2), 22.

Lakshmi, E., Priyanka, M.P., & Reddy, E.R. 2016. FPGA implementation of image encryption and decryption using aes 128-bit core. *IEEE* Xplore. (International Conference on Communication and electronics systems (ICCES))

Luo, X.Y., Ren, H.E., Xu, G.S., Yu, X.Y., & Zhang, J. 2006. Chaotic image scrambling algorithm based on s-des. *IOP Science*. (Journal of Physics Conference series in instrumentation science and technology, 48)

Nelson Darwin Pak Tech. 2022. 6 Jan*. How to create a radio button in python GUI | Radio Button in tkinter GUI* [YouTube video]. <https://youtu.be/YR2yi99yLVc> Date of Access: 01 May. 2023.

Penchalaiah, N., & Seshadri, R. 2010. Effective comparison and evaluation of des and rijndael algorithm (aes). *International journal on computer science and engineering*, 2(5): 1641-1645.