Int. J. Advanced Networking and Applications
Volume: 14 Issue: 05 Pages: 5609 - 5618 (2023) ISSN: 0975-0290

An Efficient Data Protection for Cloud Storage Through Encryption

Adepoju E. Sunday
Department of Mathematical Sciences, Achievers University Owo, Nigeria
Email: sunnybenpoju@gmail.com
Oyekanmi E. Olufunminiyi
Department of Mathematical Sciences, Achievers University Owo, Nigeria
Email: e.oyekanmi@achievers.edu.ng

ABSTRACT

Encryption of data pose data confidentiality, availability and integrity by eliminating all data leakage possibilities which is done by converting data into a cipher text that cannot be understood by unauthorized users through the use of a mathematical function called key. This research developed a secure system used for encryption and decryption process on the data supplied before such data is stored on the cloud storage system. The model used in developing the secure system is 256-bit key encryption and adopts the Base64 character encoding scheme rather than the widely used ASCII standard. Variations of data supplied to the proposed secure system were from several multimedia data (such as video, audio, text, and image) and other types. The model performance was tested and evaluated by brute-force attack and program execution time. The implementation was done using Python programming language and Google Drive as the cloud storage and the encrypted data is supported by the cloud storage system. The adopted model creates a better and efficient mechanism for data supplied by encoding the key to be used for encryption in base64. It gives a better security technique with lesser computational time.

Keywords – **AES-256, Base64, Cloud storage, Cryptography, Google cloud.**

Date of Submission: June 24, 2022 Date of Acceptance: Jan 14, 2023

I. INTRODUCTION

A popular practice involving the storing and management of data on the internet is called cloud computing. Cloud computing is the on-demand delivery of Information Technology (IT) resources such as compute, databases and storage via the internet rather than managing files and services on a local storage device. Instead of buying, owning and maintaining physical datacentres and servers, one can access a wide range of technology services such as computing power and storage on an as-needed basis from a cloud provider like Microsoft Azure. Organizations of every type, size and industry are using the cloud for a wide variety of use cases, such as data backup, disaster recovery, email, virtual desktops, software development and testing, big data analytics and customer facing web applications. For example, health care companies are using the cloud to develop more personalized treatment for patients, financial services companies are using the cloud to power real time fraud detection and prevention, and video game makers are using the cloud to deliver online games to millions of players around the world.

Amidst all the functionality of cloud computing, this study focused on the area of storage as one of the forms of cloud computing. Cloud storage is a service model in which data is transmitted and stored on remote storage systems where it is maintained, managed, backed up and made available to the user over internet. It is based on virtualized infrastructure with accessible interfaces and user can access and retrieve data from storage with the help of Application Packages Interface (API). Some cloud storage systems are for small operations while others are so large that the mechanical equipment can consume an entire warehouse. A cloud storage system requires just one data server joined to the internet, a client (cloud service consumer) subscribes to the cloud storage service, transfer copies of files across the internet to the data server which then writes the information in the cloud storage. When the client wants to recover the information, the client can access the data server through a web-based interface, the server then either transfer the files back to the client or allows the client to locate and manage the files on the server itself.

With the availability of numerous cloud storage provider (such as Dropbox, Google Drive, Proton Drive, Sync, OneDrive and many other options), trusting these companies with users' data seems to be a great risk even though it is safer than keeping such data on the local computer due to ease of accessibility, storage volume and data loss. Most cloud providers do not encrypt data on servers and if there is encryption, the encryption is done with a key owned by the cloud providers which means the cloud provider could offer users' data to law enforcements with a subpoena or a rogue, employee could snoop around and read users' data.

Prior to moving on to the new computing technology, there is an imperative need to have a knowledge of the number of levels of security that the technology provides to the data, considering hacking capabilities are also well versed [1].

According to Miller in [2], from The Washington Post titled the intelligence coup of the century, it reads thus; "For decades, the Central Intelligence Agency (CIA) read the encrypted communications of allies and adversaries". This shows that some of the companies (like Crypto AG) are owned by the law enforcement agencies in a highly classified partnership without the customers knowing.

Cloud storage users speculate that there is a backdoor to the cloud services provided by the companies and trusting third party company to store the data does not make the data more secure. Based on the third party problem, some crypto software or tools offers to encrypt data for users in the best possible way but the problem is that most crypto tools are open source which means anyone can look at the source code, check how the encryption is done which is a backdoor for hackers.

If one ever read a privacy policy, one may have noticed a section that says something about how user's data will be shared with law enforcement, which means if the police demand it and have the necessary paperwork, police will likely get it. But maybe, like most American adults don't read privacy policies very carefully if at all. In that case, user might be surprised to learn how much of one's data is in the hands of third parties, how much access law enforcement has to it, how it might be used against the user, or what users' rights are, if any, to prevent it. The Department of Justice obtained Democratic Reps. Adam Schiff's and Eric Swalwell's subscriber records (and that of their family members) from Apple Company through a grand jury subpoena. This occurred in 2017 and 2018, but the Congress members only found out about it in June 2021, when the gag order expired [3].

This also shows that a data broker could have access to Cloud User's data. A data broker is an individual or company that specializes in collecting personal or company data from either public records or private source, and selling or licensing such information to third parties for a variety of use. Therefore, the need for privacy to user's data arise due to the possibility of data leakage through legal or illegal means. The aim of this paper is to propose an efficient mechanism for data protection for cloud storage.

II. RELATED WORK

Many solutions have been proposed to solve data breaches, privacy risk and many more problems for data security in cloud-based storage.

Prajapati and Shah in [4] discussed about the number of data breaches (through cyber-attacks and other various means) that occurred per month in 2019 and 2020. It revealed that the effect of these data breach varies with some of the organizations that fell victim of it and are able to respond immediately. The researchers gave an overview of cloud storage security and reviewed various approaches for providing secure deduplication techniques in cloud storage. Prajapati and Shah concluded that combining various security approaches with data deduplication, reduces clients' security issues and eliminates redundant data from cloud storage. However, the storage and management overhead caused by cryptographic keys in the breach of client's key is too enormous.

Han *et al.* in [5] identified that building security mechanism for cloud storage is not an easy task. Because shared data on the cloud is outside the control domain of legitimate participants, making the shared data usable upon the demand of the legitimate users should be solved. To address the security problem of sharing data on the cloud storage, a secret sharing group key (SSGK) management protocol was proposed and the following means are taken by our protocol to help detect or prevent frauds.

Zhang *et al.* in [6], shows that cloud storage is a widely used cloud computing model which allow users to outsource their digital data to some cloud service provider (CSP) for storage, sharing/dissemination, analysis and the likes. While individuals and organizations can benefit from economies of scale by using cloud services, such as cloud storage services, there are underlying security and privacy challenges. Cryptography is a promising and widely applied solution to overcome security challenges in cloud storage services.

Wu *et al.* in [7], worked on cloud storage security Assessment through Equilibrium Analysis: Game Theory model (Nash equilibrium). The model assesses the internal security hazards in CSP/TSP and estimate whether they would behave honestly to user data but the adopted model is based on probability.

Kumar *et al.* in [8], explored data security issues and solutions in cloud computing and explained the need of CIA (Confidentiality, Integrity and Availability) triad in data security and how it can be improved. Application of data encryption when the data is at rest or in transit was also considered. Application of strong encryption algorithms like Advanced Encryption Standard (AES) and Rivest Shamir Adleman (RSA) algorithms.

Vengadapurvaja *et al.* in [9] adopted an efficient homomorphic encryption algorithm to encrypt the medical images and to perform useful operations on them without breaking the confidentiality. Homomorphic encryption is a form of encryption that permits users to perform computations on its encrypted data without first decrypting it. This approach provides an efficient security to original medical data (image) and however requires high computational cost for implementation.

Bokefode *et al.* in [10] describes some of the security issues which hampers the cloud and their resolutions which ensure that data stored on a cloud is secured. The study adopted AES and RSA cryptographic techniques simultaneously.

III. PRELIMENARIES

A. Data Encryption

The important aspect of this research focused on how safe and secure is the data on the cloud storage? Encryption of data is the surest way to store or share confidential information because it eliminates all data leakage possibilities. Its process involves conversion of data into a cipher text (unreadable format) that cannot be understood by unauthorized users. To read an encrypted file, one must have access to a secret key or password that enables decryption of the file. Encryption is used to protect sensitive information during data transmission and data storage. The components of data encryption are plain text, cipher text and encryption keys. The Plain text is the original message in a readable format visible to either the authorized or unauthorized user. The Cipher text is the encrypted message in an unreadable format to every other user except the authorized user(s). Encryption keys are typically a random string of bits generated specifically to scramble/encrypt and unscramble/decrypt data. The Encryption keys are created with algorithms designed to ensure that each key is unique and unpredictable by unauthorized user. The complex the key constructed, the harder it is to break the encryption code. The Encryption objectives are Confidentiality, Integrity, Authentication, Non-repudiation and Availability.

The various type of Encryption are Symmetric encryption, Asymmetric encryption and Hash function. In Symmetric encryption, the same secret key is used to both encrypt and decrypt data while in Asymmetric encryption, public key is used for encryption and private key is used for decryption. Hash function is used for one-way encryption, no keys are required for encryption and decryption. The main advantage of symmetric encryption over asymmetric encryption is that it is relatively fast and efficient for large amounts of data which is considered as the best method to adopt for cloud storage. According to Smirnoff & Turner in [11], Symmetric encryption is a type of encryption where only one key (a secret key) is used to both encrypt and decrypt electronic information. This paper adopts the Symmetric type of encryption. Fig. 1 shows the topology of Symmetric key encryption. Some examples of Symmetric encryption methods include: AES (Advanced Encryption Standard); DES (Data Encryption Standard); IDEA (International Data Encryption Algorithm); Blowfish (Drop-in replacement for DES or IDEA); RC4 (Rivest Cipher 4); RC5 (Rivest Cipher 5); RC6 (Rivest Cipher 6). This research adopted the AES method of all the types of symmetric encryption because it is mathematically efficient and an elegant cryptographic method.

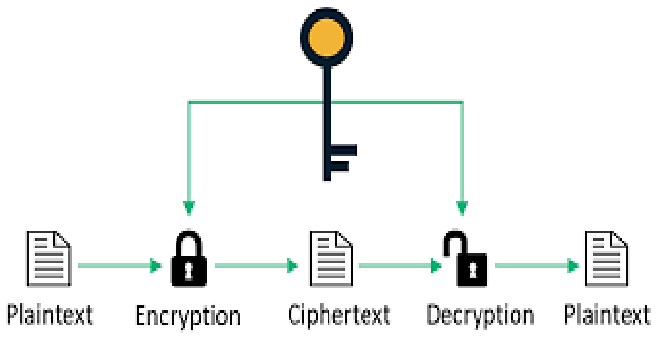


Fig. 1: Symmetric key encryption topology

B. Advanced Encryption Standard (AES)

AES is comparatively much faster than other types of encryption method (like DES) and can encrypt large files in a fraction of seconds compared to DES. Because of the small bit size of the shared key used in DES, it is considered to be less secure than AES. The AES method successively applies a series of mathematical transformations to each 128-bit block of data. Because the computational requirements of this approach are low, AES can be used with consumer computing devices such as laptops and smartphones, as well as for quickly encrypting large amounts of data. The key length is typically specified as a logarithm in form of bits. For instance, 3-bit key length (2 x 2 x 2 = 8) will accept eight different keys. The longer the key, the more secure data can be viewed with it. Since the only way to crack a key is usually a so-called brute-force attack, attempting every possible option, the key length specifies the processing power and computing time.

AES main strength rests in the option for various key lengths. AES is a symmetric method which uses the same 128, 192, or 256-bit key for both encryption and decryption (the security of an AES system increases exponentially with key length) making it exponentially stronger than the 56-bit key of DES. With a 128-bit key, the task of cracking AES by checking each of the 2^{128} possible key values (a “brute force” attack) is so computationally intensive. The higher the number of the key length, the more secure the key becomes. Table 1 shows the possible key combinations with respect to the key size. AES-256, which has a key length of 256 bits, supports the largest bit size and is practically unbreakable by brute force based on current computing power, making it the strongest encryption standard which was adopted in this research.

Table 1: Possible key combinations with key size

Key Size	Possible Combinations
1 bit	2
2 bits	4
4 bits	16
8 bits	256
16 bits	65536
32 bits	4.2×10^{9}
56 bits (DES)	7.2×10^{16}
64 bits	1.8×10^{19}
128 bits (AES)	3.4×10^{38}
192 bits (AES)	6.2×10^{57}
256 bits (AES)	1.1×10^{77}

The basic idea of encryption is to convert data supplied into a form in which the original meaning is masked, and only those who are properly authorized can decipher it. This is done by scrambling the information using mathematical functions based on a number called a key. Since this research work adopts the AES symmetric encryption, AES is based on S-P network in which the entire 128 bits input block is organized as 4x4 bytes array called State array and is processed in several rounds. Number of rounds to be used depend on the length of key. For example, 10 rounds for 128-bit key, 12 rounds for 192-bit key and 14 rounds for 256-bit key. 14 rounds of encryption would be performed due to the use of 256-bit key. For the encryption process, the state array is modified at each round by a round function that defines four different byte-oriented transformations:

i. SubBytes transformation: a non-linear substitution step where each byte is replaced with another byte according to a substitution table (S-box).
ii. ShiftRows transformation: a transposition step where each row of the state is shifted cyclically a certain number of steps.
iii. MixColumns: a mixing operation which operates on the columns of the state, combining the four bytes in each column.
iv. AddRoundKey: a simple bit wise XOR operation is performed between each byte of the state and the round key which is generated from the cipher key using Rijndael key schedule algorithm.

Fig. 2 shows the Advanced Encryption standard (AES) model.

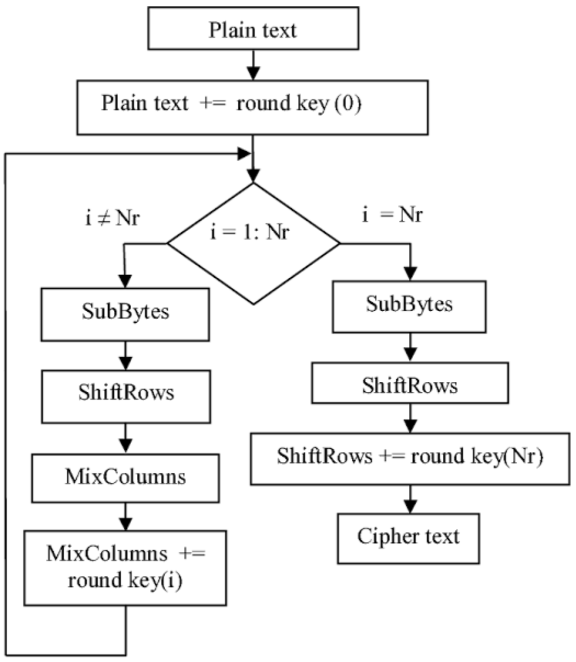


Fig. 2: Advanced Encryption Standard (AES) model

C. *Base64 and ASCII*

Base64 is a way to encode binary data into an ASCII character set known to pretty much every computer system, in order to transmit the data without loss or modification of the contents itself. For example, mail systems cannot deal with binary data because they expect textual (ASCII/Base64) data. Base64 encoding is NOT a way of encrypting, nor a way of compacting data. In fact a base64 encoded piece of data is 1.333… times bigger than the original data piece. It is only a way to be sure that no data is lost or modified during the transfer.
Base64 encoding is not encryption but just a way to transform any given data into a stream of printable characters which can be transmitted over network. Base64 is a more or less compact way of transmitting (encoding, in fact, but with goal of transmitting) any kind of binary data.

ASCII has no genuine advantage over Base64 and vice versa. ASCII and Base64 are both encoding mechanism in which any can be chosen depending on the user or researcher. To the best of our knowledge, no researcher has made use of Base64 for the set of characters as a key to an encryption process. Base64 creates more complexity than ASCII. A 24-bit string is represented by 3 characters in ASCII while the same 24-bit string is represented by 4 characters. 4 ÷ 3 = 1.333.

Why base64?

Years ago, when mailing functionality was introduced, it was mainly a text-based, later on, the need for attachments like image and media (audio, video etc.) arose to make the idea robust. When files are attached over the internet (which is basically in the form of binary data), the probability of binary data getting corrupt is high in its raw form and this cannot be sufficiently handled by ASCII method. Hence, Base64 technique was introduced. viii.

Base64 encoding schemes are commonly used when there is a need to encode binary data that needs to be stored and transferred over media that are designed to deal with ASCII. This is to ensure that the data remain intact without modification during transmission. Base64 images are primarily used to embed image data within other formats like HTML, CSS, or JSON. By including image data within an HTML document, the browser doesn't need to make an additional web request to fetch the file, since the image is already embedded in the HTML document.

When a text is encoded in ASCII, the first step is to convert the text string into a sequence of bytes. On the other hand, an encoded data in Base64, start with a sequence of bytes and convert it to a text string.

Base64 is one of the binary-to-text encoding schemes having 75% efficiency. It is used so that typical binary data (such as images) may be safely sent over legacy "not 8-bit clean" channels. In earlier email networks (till early 1990s), most email messages were plain text in the 7-bit US-ASCII character set. So, early communication protocol standards were designed to work over "7-bit" communication links "not 8-bit clean". Scheme efficiency is the ratio between number of bits in the input and the number of bits in the encoded output. Hexadecimal (Base16) is also one of the binary-to-text encoding schemes with 50% efficiency.

Since an encryption key does not have to be text but raw bytes it is sometimes necessary to store it in a file or database, which Base64 comes in handy for. Same with the resulting encrypted bytes. Although Base64 is often used in cryptography, yet it is not a security mechanism. Anyone can convert the Base64 string back to its original bytes, so it should not be used as a means for protecting data, only as a format to display or store raw bytes more easily.

Base64 can be used in a variety of contexts, for instance:

i. Evolution and Thunderbird use Base64 to obfuscate e-mail passwords.
ii. Base64 can be used to transmit and store text that might otherwise cause delimiter collision.
iii. Base64 is often used as a quick but insecure shortcut to obscure secrets without incurring the overhead of cryptographic key management.
iv. Spammers use Base64 to evade basic anti-spamming tools, which often do not decode Base64 and therefore cannot detect keywords in encoded messages.
v. Base64 is used to encode character strings in LDIF files.
vi. Base64 is sometimes used to embed binary data in an XML file, using a syntax similar to Firefox's bookmarks.html for example.
vii. Base64 is also used when communicating with government Fiscal Signature printing devices (usually, over serial or parallel ports) to minimize the delay when transferring receipt characters for signing.

Base64 is used to encode binary files such as images within scripts, to avoid depending on external files.

It can be used to embed raw image data into a CSS property such as background-image.

Base64 encoding scheme is commonly used when there is a need to encode binary data that needs be stored and transferred over media that are designed to deal with textual data. The only downside is that base64 encoding will require around 33% more space than regular strings (Base64 encoding uses 33% more storage).

Base64 Procedure

Base64 encoding method requires that every three 8-bit bytes be converted into four 6-bit bytes. Among them, every six valid bits in the four bytes after conversion are valid data, and the remaining two bits use 0 Make up a byte. The general strategy is to choose 64 characters that are common to most encodings and that are also printable. In an example given, the encoded value of “Man” is “TWFu”. Encoded in ASCII, the characters M, a, and n are stored as the byte values 77, 97, and 110, which are the 8-bit binary values 01001101, 01100001, and 01101110. These three values are joined together into a 24-bit string, producing 010011010110000101101110. Groups of 6 bits (6 bits have a maximum of $2^6 = 64$ different binary values) are converted into individual numbers from start to end (in this case, there are four numbers in a 24-bit string), which are then converted into their corresponding Base64 character values. As this example illustrates, Base64 encoding converts three ASCII characters into four encoded characters. Fig. 3 is an example showing the three ASCII characters.

Source (ASCII)	**M**								**A**								**N**							
	77 (0x4d)								97 (0x61)								110 (0x6e)							
Bits	0	1	0	0	1	1	0	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0
Base64 Encoded	19						22						5						46					
	T						**W**						**F**						**u**					

Fig 3: Three ASCII characters

But if there are only two significant input octets (e.g., 'Ma'), or when the last input group contains only two octets, all 16 bits will be captured in the first three Base64 digits (18 bits); the two least significant bits of the last content-bearing 6-bit block will turn out to be zero, and discarded on decoding (along with the succeeding = padding character). Fig. 4 is an example showing the two ASCII characters.

Source (ASCII)	**M**								**A**															
	77 (0x4d)								97 (0x61)															
Bits	0	1	0	0	1	1	0	1	0	1	1	0	0	0	0	1	0	0						
Base64 Encoded	19						22						4						Padding					
	T						**W**						**E**						=					

Fig 4: Two ASCII characters

Since Base64 is not a model or algorithm, it does not have complexity, hence the complexity of AES algorithm is considered. With respect to Table 1, 256-bit key size with require 1.1×10^{77} possible combinations by brute-force attack, therefore the time complexity of the algorithm is $O(2^n)$. This complexity is calculated by;
n-bit key → 2^n

4-bit key → 2^4
8-bit key → 2^8
16-bit key → 2^{16}
64-bit key → 2^{64}

Therefore, the time complexity of the model with respect to brute-force is $O(2^n)$.

IV. DESIGN OF THE PROPOSED SYSTEM

The system architecture is designed such that the data stored on the public cloud provided by the Cloud Service Provider (CSP) is safe and protected in the best possible way. For a data to be safe, it means privacy. Data privacy generally means the ability of a person to determine for themselves when, how and to what extent personal information about them is shared with or communicated to others. For data to be protected, it means security. Data security refers to the process of protecting data from unauthorized access and data corruption throughout its lifecycle. This includes data encryption, hashing, tokenization and key management practices that protect data across all applications and platforms.
Study shows that most cloud service providers encrypt cloud service consumer's data with a key they own which means the cloud service providers provides security mechanisms to protect customer's data in which the security mechanism can also be manipulated or breached by them without the knowledge of the cloud service consumer. Having this alone means the cloud service consumer trusts the cloud service provider totally with its data no matter the encryption process stated in their terms and agreement. Many CSPs use different types of encryption method known to them in which some methods are susceptible to hacking and exhibit some limitations. With this understanding, this pops a question; "*instead of giving cloud service provider total control of your data, why not have a higher percent control of your data than the cloud service provider?"*
The focus of the system architecture gives more control of the data security encryption to the cloud service consumer than the cloud service provider in such a way that the data goes through hybrid encryption method (more than one encryption process).

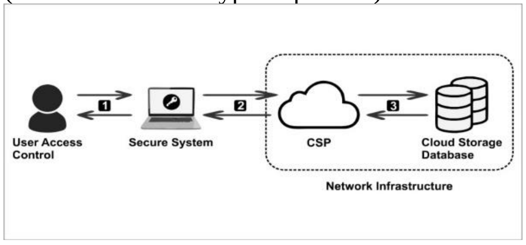


Fig 5: System Architecture Design

The system architecture in Fig. 5 shows different events ranging from stage 1 to stage 3 and the parties to the whole process. Only an authorized user can have access to the whole system and must have passed the user access control verification. The different events to secure data storage is explained below;

Stage 1: As per the forward arrow, this is the first process in which a user selects a readable data to be stored safely and protected, the data is sent to the secure system for cryptography process so as to make the data unreadable after the encryption process. For the backward arrow, the data coming from the secure system is also readable by the user. The readable data is called plaintext in cryptography. This first event can be called plaintext phase because the original message is readable.

Stage 2: For the forward arrow, the original message is now masked which is unreadable, the unreadable data is sent to the cloud service provider (CSP) for storage. For the backward arrow, unreadable data coming from the CSP is sent to the secure system. Unreadable data is called ciphertext in cryptography. This is the ciphertext phase because the file is encrypted and cannot be understood by any party.

Stage 3: This event is done by the CSP in which data given to them is stored in their self-owned datacenter or third party database. For both the forward and backward arrow, the user encrypted data is further encrypted by the CSP and decrypted respectively. This event exists and functions on the cloud service (network infrastructure).

One thing that has become abundantly clear in the internet age is that preventing unauthorized people from gaining access to the data stored in web-enabled computer systems is extremely difficult. All it takes is for a worker to click on the wrong link in an email, or respond unwarily to a seemingly legitimate request for information, and an intruder could gain complete access to all your data. In today's regulatory and public relation environments, that kind of breach can be catastrophic. But what if you could be assured that even if an attacker get access to your information, they can't use it? That's the role of data encryption.

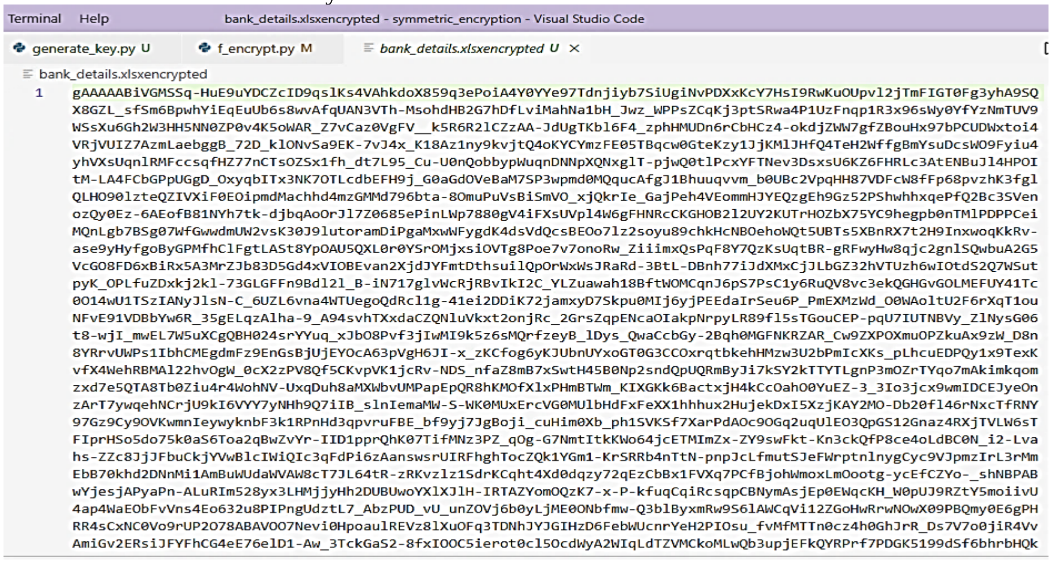
The secure system performs encryption process on the data supplied but the work of this secure system is not limited to encryption only because it also performs decryption process too. The secure system is designed such that it generates a powerful private key to encrypt the data and as well as using the same key to decrypt the data, only the key that is used to encrypt data can decrypt the data else the data will not be decrypted to the plain text. This research adopts the AES-256 bits key method of all the types of symmetric encryption because it is mathematically efficient and an elegant cryptographic method. This study tends to encode the AES private key in Base 64 scheme rather than the well-known ASCII (American Standard Code for Information Interchange).

V. IMPLEMENTATION OF THE PROPOSED SYSTEM

The proposed system is implemented of multimedia files using the python programming language.

A. Encryption

The encryption program makes a procedural call to both the file holding the key and the data to be encrypted. The data to be encrypted (bank_details.xlsx) is allocated to a variable name (filename), then the variable name is open as read byte to another variable name (e_file). Thereafter the e_file is encrypted using the key (a procedural call is made to the file holding the key) and the result is allocated to a new variable name (encrypted_file). The encrypted data (encrypted_file) is saved into a new file by opening the file as write byte and write the encrypted data to that file and give it an extension of 'encrypted' in order to be able to differentiate the encrypted data (bank_details.xlsxencrypted) from the original data (bank_details.xlsx). Fig. 6 is an image showing the encrypted data and Fig. 7 is an encrypted data on the CSP.


Terminal
Help
bank_details.xlsxencrypted - symmetric_encryption - Visual Studio Code
generate_key.py U
f_encrypt.py M
bank_details.xlsxencrypted U
bank_details.xlsxencrypted

```
gAAAAABiVGMSSq-HuE9uYDCZcID9qslKs4VAhkdoX859q3ePoiA4Y0YYe97Tdnjiyb7SiUgiNvPDXxKcY7HsI9RwKuOUpvl2jTmFIGT0Fg3yhA9SQ
X8GZL_sfSm6BpwhYiEqEuUb6s8wvAfqUAN3VTh-MsohdHB2G7hDfLviMahNa1bH_Jwz_WPPsZCqKj3ptSRwa4P1UzFnqp1R3x96sWy0YfYzNmTUV9
WSsXu6Gh2W3HH5NN0ZP0v4K5oWAR_Z7vCaz0VgFV__k5R6R2lCZzAA-JdUgTKbl6F4_zphHMUDn6rCbHCz4-okdjZWW7gfZBouHx97bPCUDWxtoi4
VRjVUIZ7AzmLaebggB_72D_klONvSa9EK-7vJ4x_K18Az1ny9kvjtQ4oKYCYmzFE05TBqcw0GteKzy1JjKMlJHfQ4TeH2WffgBmYsuDcsWO9Fyiu4
yhVXsUqnlRMFccsqfHZ77nCTsOZSx1fh_dt7L95_Cu-U0nQobbypWuqnDNNpXQNxglT-pjwQ0tlPcxYFTNev3DsxsU6KZ6FHRLc3AtENBuJl4HPOI
tM-LA4FCbGPpUGgD_OxyqbITx3NK7OTLcdbEFH9j_G0aGdOVeBaM7SP3wpmd0MQqucAfgJ1Bhuuqvvm_b0UBc2VpqHH87VDFcW8fFp68pvzhK3fgl
QLHO90lzteQZIVXiF0EOipmdMachhd4mzGMMd796bta-8OmuPuVsBiSmVO_xjQkrIe_GajPeh4VEommHJYEQzgEh9Gz52PShwhhxqePfQ2Bc3SVen
ozQy0Ez-6AEofB81NYh7tk-djbqAoOrJl7Z0685ePinLWp7880gV4iFXsUVpl4W6gFHNRcCKGHOB2l2UY2KUTrHOZbX75YC9hegpb0nTMlPDPPCei
MQnLgb7BSg07WfGwwdmUW2vsK30J9lutoramDiPgaMxwWFygdK4dsVdQcsBEOo7lz2soyu89chkHcNBOehoWQt5UBTs5XBnRX7t2H9InxwoqKkRv-
ase9yHyfgoByGPMfhClFgtLASt8YpOAU5QXL0r0YSrOMjxsiOVTg8Poe7v7onoRw_ZiiimxQsPqF8Y7QzKsUqtBR-gRFwyHw8qjc2gnlSQwbuA2G5
VcGO8FD6xBiRx5A3MrZJb83D5Gd4xVIOBEvan2XjdJYFmtDthsuilQpOrWxWsJRaRd-3BtL-DBnh77iJdXMxCjJLbGZ32hVTUzh6wIOtdS2Q7WSut
pyK_OPLfuZDxkj2kl-73GLGFFn9Bdl2l_B-iN717glvWcRjRBvIkI2C_YLZuawah18BftWOMCqnJ6pS7PsC1y6RuQV8vc3ekQGHGvGOLMEFUY41Tc
0O14wU1TSzIANyJlsN-C_6UZL6vna4WTUegoQdRcl1g-41ei2DDiK72jamxyD7Skpu0MIj6yjPEEdaIrSeu6P_PmEXMzWd_O0WAoltU2F6rXqT1ou
NFvE91VDBbYw6R_35gELqzAlha-9_A94svhTXxdaCZQNluVkxt2onjRc_2GrsZqpENcaOIakpNrpyLR89fl5sTGouCEP-pqU7IUTNBVy_ZlNysG06
t8-wjI_mwEL7W5uXCgQBH024srYYuq_xJbO8Pvf3jIwMI9k5z6sMQrfzeyB_lDys_QwaCcbGy-2Bqh0MGFNKRZAR_Cw9ZXPOXmuOPZkuAx9zW_D8n
8YRrvUWPs1IbhCMEgdmFz9EnGsBjUjEYOcA63pVgH6JI-x_zKCfog6yKJUbnUYxoGT0G3CCOxrqtbkehHMzw3U2bPmIcXKs_pLhcuEDPQy1x9TexK
vfX4WehRBMAl22hvOgW_0cX2zPV8Qf5CKvpVK1jcRv-NDS_nfaZ8mB7xSwtH45B0Np2sndQpUQRmByJi7kSY2kTTYTLgnP3mOZrTYqo7mAkimkqom
zxd7e5QTA8Tb0Ziu4r4WohNV-UxqDuh8aMXWbvUMPapEpQR8hKMOfXlxPHmBTWm_KIXGKk6BactxjH4kCcOahO0YuEZ-3_3Io3jcx9wmIDCEJyeOn
zArT7ywqehNCrjU9kI6VYY7yNHh9Q7iIB_slnIemaMW-S-WK0MUxErcVG0MUlbHdFxFeXX1hhhux2HujekDxI5XzjKAY2MO-Db20fl46rNxcTfRNY
97Gz9Cy9OVKwmnIeywyknbF3k1RPnHd3qpvruFBE_bf9yj7JgBoji_cuHim0Xb_ph1SVKSf7XarPdAOc9OGq2uqUlEO3QpGS12Gnaz4RXjTVLW6sT
FIprHSo5do75k0aS6Toa2qBwZvYr-IID1pprQhK07TifMNz3PZ_qOg-G7NmtItkKWo64jcETMImZx-ZY9swFkt-Kn3ckQfP8ce4oLdBC0N_i2-Lva
hs-ZZc8JjJFbuCkjYVwBlcIWiQIc3qFdPi6zAanswsrUIRFhghTocZQk1YGm1-KrSRRb4nTtN-pnpJcLfmutSJeFWrptnlnygCyc9VJpmzIrL3rMm
EbB70khd2DNnMi1AmBuWUdaWVAW8cT7JL64tR-zRKvzlz1SdrKCqht4Xd0dqzy72qEzCbBx1FVXq7PCfBjohWmoxLmOootg-ycEfCZYo-_shNBPAB
wYjesjAPyaPn-ALuRIm528yx3LHMjjyHh2DUBUwoYXlXJlH-IRTAZYomOQzK7-x-P-kfuqCqiRcsqpCBNymAsjEp0EWqcKH_W0pUJ9RZtY5moiivU
4ap4WaEObFvVns4Eo632u8PIPngUdztL7_AbzPUD_vU_unZOVj6b0yLjME0ONbfmw-Q3blByxmRw9S6lAWCqVi12ZGoHwRrwNOwX09PBQmy0E6gPH
RR4sCxNC0Vo9rUP2O78ABAVOO7Nevi0HpoaulREVz8lXuOFq3TDNhJYJGIHzD6FebWUcnrYeH2PIOsu_fvMfMTTn0cz4h0GhJrR_Ds7V7o0jiR4Vv
AmiGv2ERsiJFYFhCG4eE76elD1-Aw_3TckGaS2-8fxIOOC5ierot0cl5OcdWyA2WIqLdTZVMCkoMLwQb3upjEFkQYRPrf7PDGK5199dSf6bhrbHQk
```

Fig 6: Encrypted data

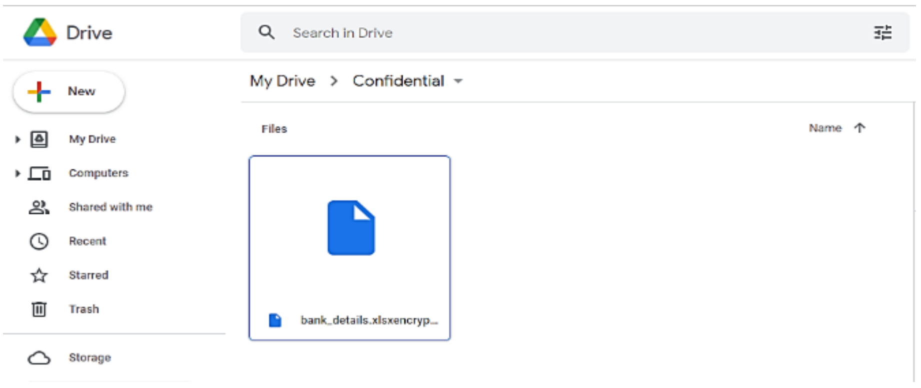
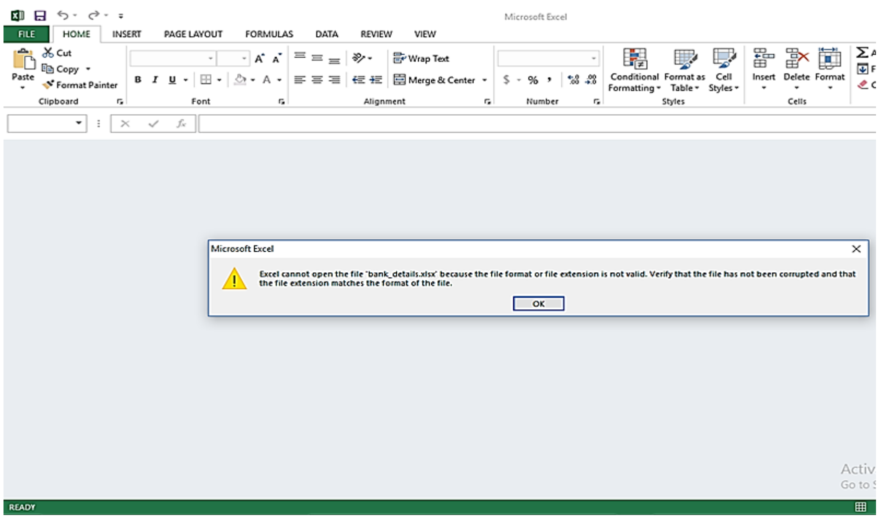


Fig 7: Encrypted data on the CSP

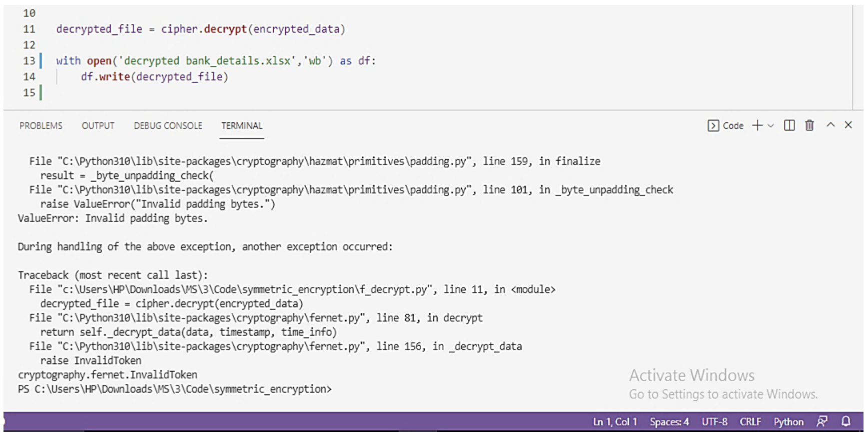
In Fig 8, having '.xlsxencrypted' as the encrypted data extension file type, could be thought of that editing (removing) the 'encrypted' from the extension and leaving '.xlsx' alone would make the file readable thereby decrypting the file without the cipher-key. The outcome as shown in Fig. 8 shows that decryption process needs a key to open correctly. This was tested using the brute-force attack by trying all possible ways to break into the data. Fig. 9 shows the encryption program with a wrong key.


Microsoft Excel
FILE HOME INSERT PAGE LAYOUT FORMULAS DATA REVIEW VIEW
Clipboard Font Alignment Number Styles Cells
OK
READY

Microsoft Excel

Excel cannot open the file 'bank_details.xlsx' because the file format or file extension is not valid. Verify that the file has not been corrupted and that the file extension matches the format of the file.

Fig. 8: Error message after editing the encrypted data file extension


PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python

```
decrypted_file = cipher.decrypt(encrypted_data)

with open('decrypted bank_details.xlsx','wb') as df:
    df.write(decrypted_file)
```

```
  File "C:\Python310\lib\site-packages\cryptography\hazmat\primitives\padding.py", line 159, in finalize
    result = _byte_unpadding_check(
  File "C:\Python310\lib\site-packages\cryptography\hazmat\primitives\padding.py", line 101, in _byte_unpadding_check
    raise ValueError("Invalid padding bytes.")
ValueError: Invalid padding bytes.

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "c:\Users\HP\Downloads\MS\3\Code\symmetric_encryption\f_decrypt.py", line 11, in <module>
    decrypted_file = cipher.decrypt(encrypted_data)
  File "C:\Python310\lib\site-packages\cryptography\fernet.py", line 81, in decrypt
    return self._decrypt_data(data, timestamp, time_info)
  File "C:\Python310\lib\site-packages\cryptography\fernet.py", line 156, in _decrypt_data
    raise InvalidToken
cryptography.fernet.InvalidToken
PS C:\Users\HP\Downloads\MS\3\Code\symmetric_encryption>
```

Fig. 9: Decryption program with the wrong key

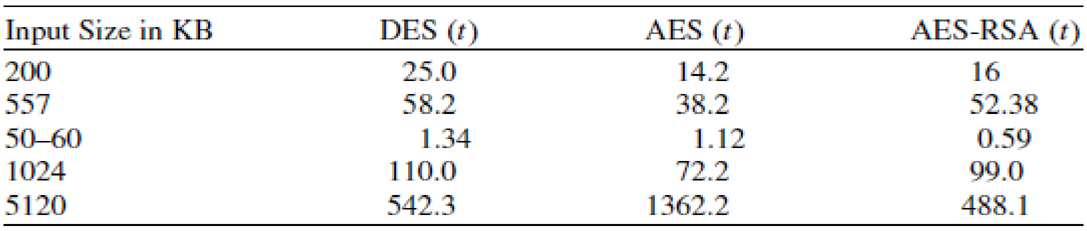
B. Decryption

The decryption process is the same with the encryption process but in the opposite direction. The data to be decrypted is an unreadable file with .xlsxencrypted as its extension file type. To carry out the decryption process on the data supplied, the encrypted data is placed in the same file directory with the program in case it has been moved, then the encryption program is run. If the encrypted data is not in the same folder with the program, the file directory is used to call out the encrypted data. The decryption program makes a procedural call to both the file holding the key and the encrypted data, thereafter decryption process takes place. Note that the file holding the key is not tampered with because the same key that was used for encryption is the same key that would be used for decryption (same key encryption == Symmetric encryption).

VI. COMPARISON OF THE PROPOSED SYSETEM WITH RELATED WORK

According to Bokefode *et al.* in [10], Table 2 is the analysis of the cryptography techniques.

Table 2: Encryption time in ASCII by Bokefode et al. [10]



Input Size in KB	DES (t)	AES (t)	AES-RSA (t)
200	25.0	14.2	16
557	58.2	38.2	52.38
50–60	1.34	1.12	0.59
1024	110.0	72.2	99.0
5120	542.3	1362.2	488.1

Using the same input size in KB as a yardstick for this project model, the following results were gotten. Table 3 is encryption time in base64.

Table 3: Encryption time in Base64

Input Size in KB	AES	XOR(t)	AES(256) in Base64 (t)
203	0.04	0.08	0.37
564	0.06	0.31	0.41
59	0.03	0.02	0.33
1030	0.08	0.45	0.42
5122	0.30	2.29	0.58

AES gives better security than all other forms of cryptography techniques, using the hybrid method (AES-RSA) could give higher security but requires more time. In the AES (256 bit-key) base64, it requires lesser time than in ASCII.

VII. CONCLUSION

Importance of encryption cannot be over emphasized with increase in data integrity and low cost of implementation. The adopted model creates a better and efficient mechanism for data supplied by encoding the key to be used for encryption in base64. It gives a better security technique with lesser computational time. Any Cloud Service Provider can be used for the user's convenience since most CSP perform the same basic activities but the most important point is for the Cloud Service Consumer to take control of its data.

REFERENCES

[1] Subramanian, N. & Jeyaraj, A. (2018). Recent Security challenges in Cloud Computing. Computers & Electrical Engineering. Volume 71, October 2018, pp. 28-42. https://doi.org/10.1016/j.compeleceng.2018.06.006.

[2] Miller, G. (2020). The intelligence coup of the century. National Security. The Washington Post. https://www.washingtonpost.com/graphics/2020/world/national-security/cia-crypto-encryption-machines-espionage/

[3] Morrison, S. (2021). Here's how police can get your data, even if you aren't suspected of a crime. And you may never know they did it. Recode by Vox, VoxMedia. https://www.vox.com/platform/amp/recode/22565926/police-law-enforcement-data-warrant

[4] Prajapati, P. & Shah, P. (2020). A Review on Secure Data Deduplication: Cloud Storage Security Issue. Journal of King Saud University – Computer and Information Sciences. https://doi.org/10.1016/j.jksuci.2020.10.021

[5] Han, S., Han, K., & Zhang, S. (2019). A Data Sharing Protocol to Minimize Security and Privacy Risks of Cloud Storage in Big Data Era. IEEE (Institute of Electrical and Electronics Engineers) Access. Volume 7, May 21, 2019. 10.1109/ACCESS.2019.2914862.

[6] Zhang, L., Xiong, H., Huang, Q., Li, J., Choo, K.R. & Li, J. (2019). Cryptographic Solutions for Cloud Storage: Challenges and Research Opportunities. DOI 10.1109/TSC.2019.2937764, IEEE Transactions on Services Computing.

[7] Wu, Y., Lyu, Y., & Shi, Y. (2019). Cloud Storage Security Assessment through Equilibrium Analysis. Tsinghua Science and Technology. ISSNll1007-0214 09/10 Number 6, Volume 24, December 2019. pp. 738–749 DOI: 10.26599/TST.2018.9010127

[8] Kumar, R.P., Raj, H.P. & Jelciana P. (2018). Exploring Data Security Issues and Solutions in Cloud Computing. 6th International Conference on Smart Computing and Communications, ICSCC 2017, 7-8 December 2017, Kurukshetra, India. Procedia Computer Science 125 (2018), pp. 691–697. http://www.elsevier.com/locate/procedia

[9] Vengadapurvaja, A.M., Nisha, G., Aarthy, R. & Sasikaladevi, N. (2017). An Efficient Homomorphic Medical Image Encryption Algorithm for Cloud Storage Security. 7th International Conference on Advances in Computing & Communications, ICACC-2017, 22- 24 August 2017, Cochin, India. Procedia Computer Science 115 (2017), pp 643–650.

[10] Bokefode, J.D., Bhise, A.S., Satarkar, P.A. & Modanic D.G. (2016). Developing A Secure Cloud Storage System for Storing IoT Data by Applying Role Based Encryption. Twelfth International Multi-Conference on Information Processing-2016 (IMCIP-2016). Procedia Computer Science 89 (2016), pp. 43 – 50.

[11] Bommala, H., Kiran, S; Pujitha, M; Reddy, R.P.K. Performance of Evaluation for AES with ECC in Cloud Environment. International Journal of Advanced Networking and Applications, Vol. 10, Iss. 5, (Mar/Apr 2019): 4019-4025. DOI:10.35444/IJANA.2019.10056

[12] Suresh, S R. An Electronic Digital Library Using Integrated Security Methods and Cloud Storages International Journal of Advanced Networking and Applications Vol. 13, Iss. 1, (Jul/Aug 2021): 4839-4844.