Using role-based access to store data on Cloud securely

Research Project
MSc in Cloud Computing

Suyash Tripathi
Student ID: x21121443

School of Computing
National College of Ireland

Supervisor: Mr. Vikas Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	SUYASH TRIPATHI		
Student ID:	X21121443		
Programme:	MSc in Cloud Computing	**Year:**	2022-23
Module:	Research Project		
Supervisor:	Mr. Vikas Sahni		
Submission Due Date:	15/12/2022		
Project Title:	Using role-based access to store data on cloud securely …………………………………………………………………………………………………….…….		
Word Count:	6347	**Page Count**:	22

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Suyash Tripathi

Date: 14/12/2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Using role-based access to store data on Cloud securely

Suyash Tripathi
X21121443

Abstract

The Internet of Things (IoT) is facing previously unheard-of scalability and security challenges as a result of the volume of data created and shared by IoT platforms and devices. Many critical pieces of information are kept in the cloud and are accessible to users. Thus, access control techniques are used to protect this sensitive material from people who could be harmful. Some of those problems are currently being resolved by combining blockchain technology with IoT applications.

However, the current generation of blockchain-based IoT solutions simply employs the blockchain to record access rules, vastly underutilizing this revolutionary technology's full potential. The issue of creating and maintaining a collection of histograms is addressed by the self-adaptive set of histograms (SASH framework), which is presented in this paper. It combines IoT platforms with blockchain technology to offer several benefits over the current state of the art like by combining the benefits of RSA and AES, it increases security and enables message decryption without requiring the sender and receiver to exchange the AES key.

1 Introduction

Access control choices may be chosen, and access control rules can be stored, using Sash. As a result, access request auditability is made available together with exact regulatory enforcement. They also create a "data marketplace" by utilizing blockchains' capacity to conduct financial transactions and compensating data producers by design. Nevertheless, given the enormous number of IoT stakeholders and possible sharing transactions among them, previous solutions are not adequately addressing for access control challenges. Although they still suggest that owners manage policy revisions, they advise shifting access control policies to the blockchain, a decentralized network of trust (Bhalla, and Prasad, 2022). The magnitude of IoT systems cannot be handled by this strategy. A vast, dispersed network of objects with actuators and sensors is what is meant by the term "Internet of Things" (IoT). IoT devices are anticipated to generate and communicate enormous volumes of data, leading to hitherto unseen challenges concerning scalability and security. Due to the volume of data generated, the number of devices, the absence of confidence between partners, and the lack of openness about data processing, current ways to secure data sharing fall short in these particular circumstances. Blockchain technology has shown that it may be a good solution in many distributed systems where transparency and trust are essential components.

According to the cloud computing paradigm which is expected to become more significant in computing, individuals will be able to easily encounter computability anywhere and at any time by using the Internet (Bhalla, and Prasad, 2022). An increasing number of apps employ cloud storage, which provides users with access control and information storage options that may successfully handle the large data problems mentioned earlier. For instance, in a future smart city transport network with IoT connectivity and cloud storage, transportation data is automatically acquired from automobiles and roadside traffic surveillance terminals and saved in the cloud.

Due to the Internet of Things, traffic police officers and businesses that offer services to cars may simply and efficiently monitor and even remotely regulate traffic flow. However, a motorist takes privacy issues extremely seriously. Therefore, it's essential to keep some delicate information about specific people a secret. The security of the landlord's data may be jeopardised if a member of the inquisitive internal staff or a persistent outside attacker succeeds in obtaining a driver's personal information without authorization (Bhalla, and Prasad, 2022). However, the owner can agree to provide a select group of well-screened individuals access to their data. Due to the adaptability of the access control policy, "ciphertext-policy attribute-based encryption (CP-ABE)" is seen as a potential choice for access control in a cloud storage system for "Internet of Things applications". If a data user (DU) satisfies specific requirements and complies with the DO's access policy, the DU may be granted shared access to the data that the DO has made available.

In many attribute-based encryption systems, attribute maintenance and key distribution are often controlled by a central authority. The CP-ABE strategy has been extensively researched for systems with a single source (Pandian, and Columbus, 2022). These techniques can't be used right away in a cloud storage system because they were developed for a typical setting. Consequently, the multi-authority CP-ABE approach must be implemented in order to provide data network access in a cloud storage system. They provide distributed access control for user attribute management and enable data sharing between such a data owner and a select group of data users by implementing an attribute access control policy. These methods were just not developed for cloud storage systems; thus, they can't be applied right away. Consequently, in order to offer data network access in a cloud storage system, the multi-authority CP-ABE technique must be implemented (Pandian, and Columbus, 2022). By establishing an attribute access control policy, they provide distributed access control entered by the user management and enable data sharing between such a data owner and a defined group of data users.

1.2 Research question

- How can role-based access be used to securely store the data on the cloud?

2 Related Work

2.1 Different types of the access control scheme

To satisfy the needs of various users, a number of access control solutions have been put forth, but the "attribute-based encryption (ABE) paradigm" has generated the most attention for protecting cloud storage. According to Xiong et al. (2020), "Key-policy attribute-based encryption (KP-ABE) and content-based attribute-based encryption" are the two primary subcategories of ABE techniques (CP-ABE). When choosing an access control strategy for a cloud storage system, there are a number of things to consider, such as compute overhead, communication costs, storing attribute keys, and security needs. The use of CP-ABE in this circumstance is a common tactic. Most ABE application research initially concentrates on a single authority model. a method that uses selective group key exchange protocol and CP-ABE for each attribute group. According to Xiong et al. (2020), efficient data control for "multi-authority cloud storage systems (DAC-MACS)" based on CP-ABE was presented, offloading the majority of the decryption computation to a cloud server by creating a decryption token. An inexpensive attribute revocation method that involves less computation and communication was also developed by them. The processing expenses and attribute revocation expenses of the DAC-MACS system are still a hassle for an IoT user with limited resources. () assessed two approaches that entailed monitoring user-submitted secret key changes and intercepting ciphertext updating secrets in a multi-authority cloud storage system employing DACMACS. They suggested a brand-new, expanded DAC-MACS approach (NEDAC-MACS) for multiauthority cloud storage systems to counteract the two threats. The technology performs similarly to DAC-MACS but has a higher level of security.

Given the shortcomings of multi-authority access-control systems, Xiong et al. (2020) developed a reliable, auditable, and multi-attribute authority access control system for public cloud storage. In order to create secret keys for a designated user, it adds a central authority. Using an auditing method, the approach may also identify the attribute authority who knowingly carried out the validity authentication. When the security requirements serve as the foundation for key generation, there are significant performance benefits. Access control for safe cloud storage is the main objective. To protect cloud data storage from financial denial of sustainability (DDoS) attacks, Zhou et al. (2018) developed a CPABE-based technique.

2.2 A Blockchain-Based IoT System with Secure Storage

Blockchain and Internet of Things (IoT)-related technologies are now evolving swiftly in both industry and academics. Central servers are essential to the security and efficiency of IoT systems since they are often centralized. Users must thus have faith in the central servers, and it is challenging to coordinate the usage of outside computer resources to enhance IoT performance. According to Zhou et al. (2018), fortunately, the blockchain may provide high security, a strong reputation, and decentralization. As a result, it could be required to develop a decentralized Internet of Things (IoT) system based on blockchain technology. The number of innovative applications is rising as a result of the Internet of Things (IoT) quick expansion. In today's IoT networks, sensors are typically found on certain low-power, portable gadgets. The devices are in charge of gathering data from the environment, and they may communicate with other devices, servers, or platforms to share information. Traditional IoT systems store the gathered data in centralized cloud systems so that it may be used during future operations.

According to Zhou et al. (2018), users must thus have faith that the centralized servers, which usually lack encryption, will secure their sensitive and private data. Among the popular topics right now is blockchain-powered IoT. Four areas have received the most attention in prior research: data storage, cloud computing, smart apps, and authentication (or access control).

They were not aware that servers may conduct homomorphic operations on encrypted data without first decrypting it. Additionally, earlier research either found that the data was frequently transmitted to servers in an unencrypted state or that servers could decode the data that was being provided to them. Confidential material may be made public since the servers may have access to the details of the data they receive. IoT systems gathered data that might be in unencrypted, cryptosystem, or hash values and stored it using the blockchain in Zhou et al. (2018). To ensure that only specified miners or users who possess certain traits may decode and authenticate encrypted data, Zhou et al. (2018) employed attribute-based encryption in particular. However, due to the fact that attribute authorities are in charge of determining system settings as well as attributes for customers and miners, this protocol is only partially decentralized.

2.3 Access Control Model for Google Cloud IoT

The Internet of Things (IoT) is becoming more and more commonplace and has a bigger impact on our everyday lives than before. Recent research on the privacy and security consequences of IoT and cyber-physical networks has garnered a lot of interest from both academia and business. Academic scholars have created novel access control techniques and concepts for the Internet of Things. According to Gupta et al. (2020), to assure widespread adoption, businesses, particularly cloud services companies like Microsoft, Amazon, and Google, have created cloud-enabled IoT platforms. IoT access control and authorization frameworks are powered by the cloud. For AWS, Azure, and their IoT platforms, access control methods have been researched and implemented in the past. AWS-IoT and Azure-IoT employ a "Policy-Based Access Control (PBAC)" mechanism, according to research on Cloud Enabled IoT access control systems. To manage their access control needs, the Azure cloud platform uses a special "Role-Based Access Control (RBAC)" mechanism with pre-defined roles and organizations.

According to established security standards and laws, access control (AC) entails the authentication and authorization of communication privileges and resource access. According to Gupta et al. (2020), giving authorized organizations access to use resources in line with established regulations is the process of authorizing control (AC). It is difficult to apply the ideal access control strategy to billions of IoT devices. Although authentication and authorization concerns have been thoroughly researched in the literature, the IoT environment is where they are still in their development. According to Gupta et al. (2020), some of the most popular access control techniques in IT infrastructure include the "Access Control List (ACL)", "Role-Based Access Control (RBAC)", and "Attribute-Based Access Control (ABAC)", but they are not entirely suitable for providing the scalable, effective, and manageable potential application in an IoT environment.

Based on their goals, models, design, and processes, Ghani et al. (2020) provided a complete analysis of the present IoT access control solutions (OM-AM). The authors carefully examined a taxonomy, which was also included in the study. In relation to the IoT environment, the study examined the benefits and drawbacks of access control models and protocols. They also introduced Fair Access for IoT, a blockchain-based decentralized access control system. To protect users' privacy, the framework employed a pseudonymous method. Users may now give, receive, delegate, and revoke access based on access tokens thanks to a brand-new type of transaction that was established by the team. Attribute-Based Access Control serves as the foundation for this conceptually. Ghani et al. (2020) suggested a simple architecture for employing private blockchain technology to secure IoT. The suggested method is built on three main models: system infrastructure, overlay networks, and cloud storage, and employs an access control list to ensure authorization. Because all IoT devices in the smart home level are under the control of the miner, this method preserves the rules that govern access in the policy prefix of a local blockchain and does not require a Proof of Work (PoW) consensus to confirm blocks. They argued that the technology's overhead expenses pale in comparison to the security advantages it provides. The suggested design is based on a strategy that is focused on applications for smart homes rather than taking a general approach that may not be applicable in all circumstances.

Alattab et al. (2022) proposed a framework built on a multi-agent system for identity management in the cloud and bringing one’s own devices (BYODs) settings by addressing several security and privacy problems related to BYODs, such as the leakage of sensitive information and unwanted policy modifications. To address security and privacy issues, they proposed an architecture that would subtly and securely incorporate Mandatory Access Control rules in BYOD and cloud environments. Ghani et al. (2020) recommended building an access control and authentication mechanism for an IoT infrastructure utilising an "ECC (Elliptic Curve Cryptosystem)" based security key and employing an RBAC authorization approach based on the user's role. The recommended approach is not scalable in an IoT environment due to a large user base and RBAC's restrictions.

Alattab et al. (2022) proposed a capability-based method to overcome the difficulties in IoT access control brought on by limited storage and processing capacity. This approach calls for simple secure control access rules. A straightforward access control system with trust awareness was also developed by Alattab et al. (2022) to offer an encrypted channel between devices connected in an IoT environment. Based on the trust values discovered through "Quality of Service (QoS)", security precautions, and reputation, they created a unique trust model. When device identification is unknown, trust management in a based on distributed IoT system becomes a significant problem.

Table 1. Comparison of some related papers and their findings

Articles	Pros	Cons	Method used	Findings
Alattab et al. (2022)	This article helped to understand cloud computing and has assisted in completing the research effectively.	The article only focused on cloud computing and not IoT.	In order to provide a control measure, coarse-grained, and role privacy security for a cloud storage system, the study employed the privacy-preserving RBAC mechanism to forecast the roles of the smart grid and EV users.	The recommended method implements the standard model's security and adds a number of significant features, such as coarse-grained access control, privacy, enforceability, unidentifiable authentication, and public compliance.

Ghani et al. (2020)	The study has helped to include different information related to cloud storage.	The study did not provide information role-based access.	No such method used.	It turns out that despite its simplicity of use and financial advantages, cloud storage technology still has a lot of drawbacks. Security and data management concerns are the main factors obscuring the cloud storage architecture. The literature suggests a variety of countermeasures to these dangers.
Gupta et al. (2020)	The study focused on role-based access IoT. Hence helped to include all the relatable information.	The study did not include information related to cloud computing.	The GCPAC model, which is a formal access control paradigm for the Google Cloud Platform, is studied and developed in this work. The Google Cloud Platform IoT Access Control (GCP-IoTAC) model is then expanded from the GCPAC model to include IoT-specific components.	With this contribution, formal IoT access control models and practical Cloud-Enabled IoT Platforms will be able to reach consensus.

3 Research Methodology

The research methodology used in this part to create the study will be outlined. The next part additionally discusses the suggested method's study design. Below Figure illustrates the typical research technique used, from evaluating current systems through demonstrating the suggested one.

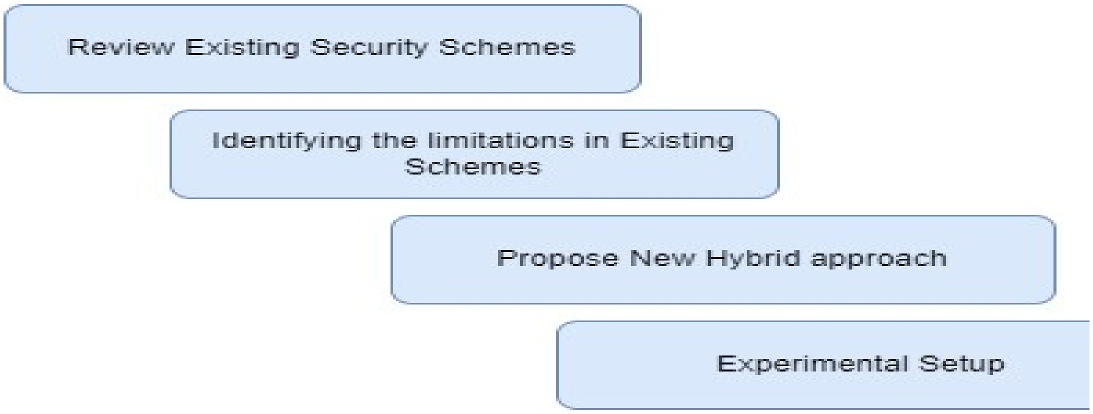

Review Existing Security Schemes
Identifying the limitations in Existing Schemes
Propose New Hybrid approach
Experimental Setup

Figure 1: Research approach

It is clear from the above Figure that article follows the flow of the traditional research approach. First, research has been done on current systems in the literature. Several shortcomings in these current methods will be discovered after research. It is discovered that the time and computation overhead costs for present techniques are both higher. Other limitations were discovered, but we only address a handful of the major ones with our new hybrid technique (AES+RSA), which secures data stored in the cloud.

In addition to combining the strengths of both algorithms, using a hybrid AES+RSA algorithm can also provide additional security benefits. Because RSA is an asymmetric algorithm, it is much more difficult to break than a symmetric algorithm like AES. This added security can provide an additional layer of protection for the encrypted data. Additionally, because the keys used for AES and RSA are generated and managed separately, it can be more difficult for an attacker to gain access to both keys and therefore to the encrypted data. Overall, using a hybrid AES+RSA algorithm can provide enhanced security and efficient encryption and decryption of data.

By using both algorithms together, we can take advantage of the speed and efficiency of AES for the actual encryption of the data, while using RSA to securely exchange the AES key between the sender and the recipient. This means that the recipient can decrypt the data without ever having to share the AES key with the sender.

Another advantage of using a hybrid AES+RSA algorithm is that it provides additional security. Because the AES key is encrypted using RSA, it is much more difficult for an attacker to obtain the key and decrypt the data. This is because RSA is considered to be a much more secure algorithm than AES, and it would require a much larger amount of computing power to break RSA encryption.

In order to use this methodology, the sender and the recipient must both have access to the appropriate RSA keys. The sender will need the public RSA key in order to encrypt the AES key, while the recipient will need the private RSA key in order to decrypt the AES key and the data.

Once the data is encrypted using this methodology, it can be uploaded to a cloud storage service like AWS. This provides a secure and convenient way to store the data and share it with others. Because the data is encrypted, it will be secure even if the cloud storage service is hacked or otherwise compromised.

Overall, using a hybrid AES+RSA algorithm is an effective way to securely encrypt and share data. By combining the strengths of both algorithms, we can achieve a high level of security and convenience.

4 Design Specification

Hybrid encryption is a method of security that combines two or more encryption systems in order to take advantage of the benefits of both symmetric and asymmetric encryption. This approach allows for both security and speed, as the symmetric encryption provides security while the asymmetric encryption provides convenience. As long as the public and private keys are kept secure, hybrid encryption is considered a highly effective form of encryption. By combining the strengths of both types of encryptions, a hybrid encryption technique can provide the best of both worlds, allowing for the secure transmission of sensitive data without sacrificing speed or convenience.

The use of numerous different cryptography approaches may provide a mechanism for the solution of extra flaws in security cryptography, including such as:

- All security goals are met due to key cryptography administration.
- to establish a safe communications application that enables Web users to transfer information and data.
- There should be a reliable cryptographic technique that was developed as a lot of data has been transmitted by users.

The proposed algorithm combined the advantages of both asymmetric key encryption and symmetric key encryption to solve the problems related to administration in AES and the high computational costs associated with RSA due to its long key length, accordingly (Rehman, et. al., 2021). It employs a wide range of applications to generate accurate results. The data decryption and encryption process use the AES key technique. The RSA key method will be employed to encrypt the encrypted message generated by the AES-Key, guaranteeing data integrity, authenticity, and non-repudiation.

The steps of the suggested hybrid encryption process are as follows:

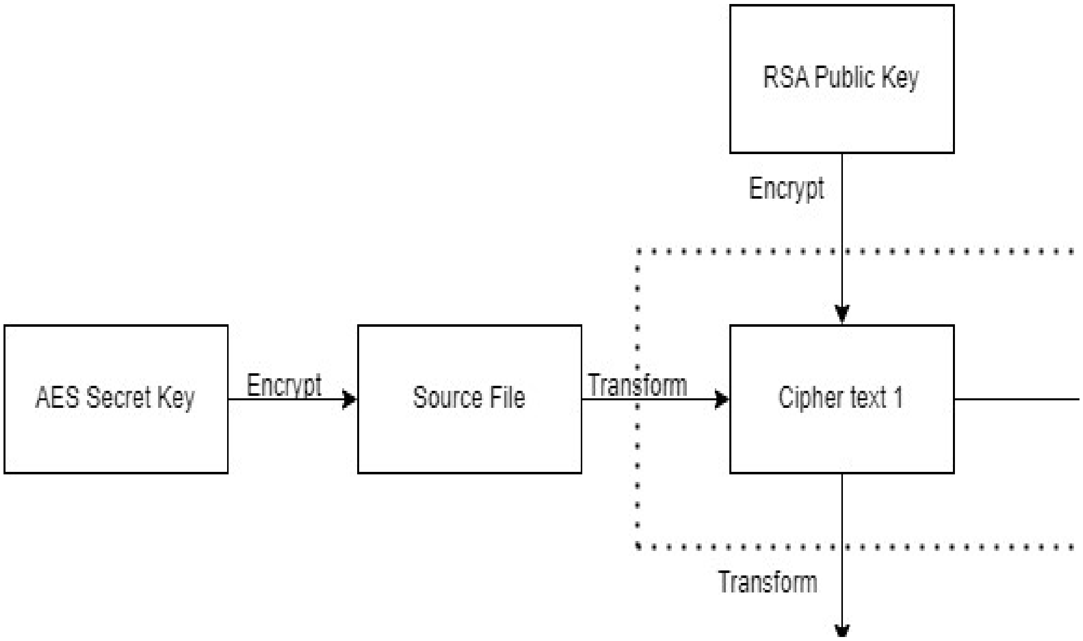


Figure 2: Flow chart of hybrid Encryption

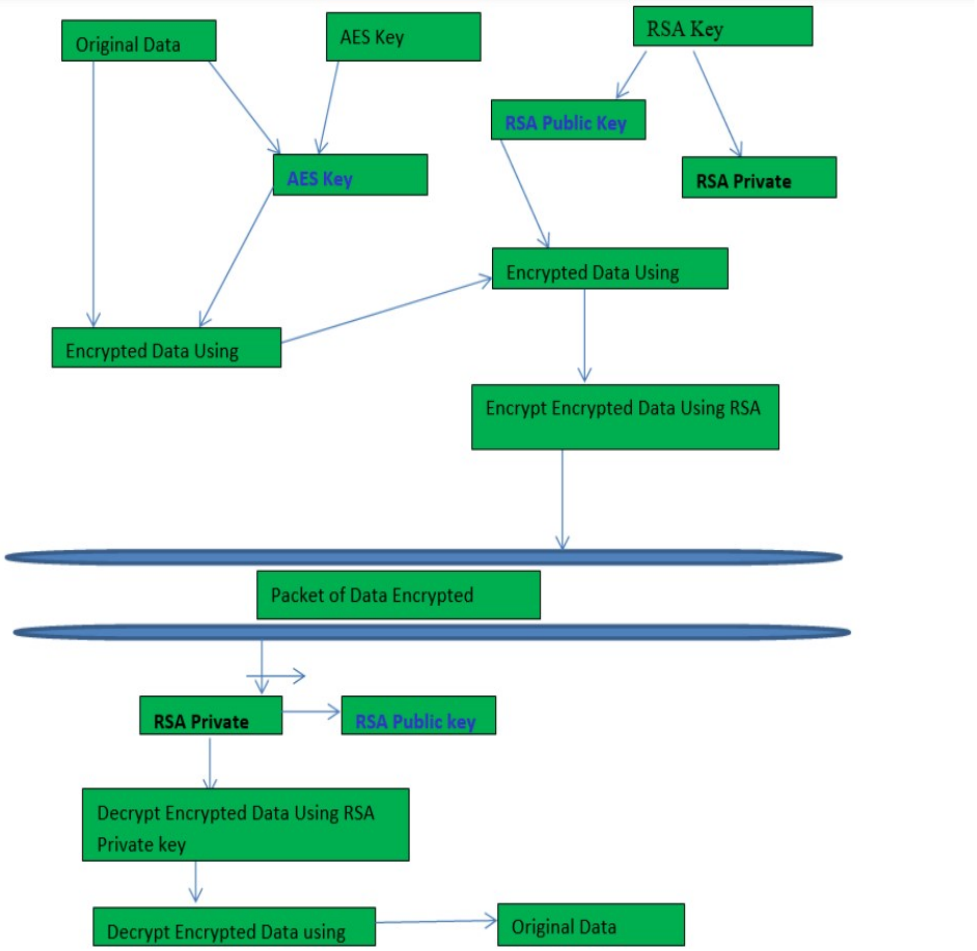


Figure 3: Detailed Model of Ideal tool

The above picture outlined how the ideal tool decrypts and encrypts files using various steps to ensure good file security. The suggested algorithm's effectiveness would've been assessed over time. Only if it is assumed that the particular file size encoded can be properly decrypted without losing any of it can reliability be established. When the above condition was met, the algorithms provided the high accuracy required because of the level of security and short process time.

The framework can be broken down into the following improvements. The sender's side, third-party confirmation, and the recipient's side are isolated from each other. A framework is software with many strings (Rashid, et. al., 2022). Three sub-strings are handled by the project's primary string. To accept from the user for encryption and sending to the third party is one of the sub-strings. The third-party sub-string is employed to validate the sender's approach signal. The recipient sub-string next receives the document and examines for a critical success that is set to come.

5 Implementation

- First, generate a secure AES key that will be used to encrypt the file. The key should be long enough to provide sufficient security (e.g., 256 bits).
- Use the AES key to encrypt the file using the AES algorithm. This will produce an encrypted file that can only be decrypted using the original AES key.
- Next, generate a secure RSA key pair (consisting of a private key and a public key). The private key should be kept secret and should not be shared with anyone, while the public key can be shared freely. Use the public RSA key to encrypt the AES key. This will ensure that the AES key can only be decrypted using the corresponding private RSA key.
- Store the encrypted AES key and the encrypted file in a secure location, such as on a cloud storage service like AWS. To decrypt the file, the recipient will need to have access to the private RSA key. They can use this key to decrypt the AES key, and then use the decrypted AES key to decrypt the file.
- In order to ensure the security of the encrypted data, it is important to regularly rotate the RSA key pair. This means generating a new key pair and updating the encryption and decryption processes to use the new keys. This will help to prevent an attacker from gaining access to the keys and decrypting the data. It is also important to use strong passwords and other security measures to protect the private RSA key. This key should be kept in a secure location, such as a password-protected file or a hardware security module.
- In order to share the encrypted data with others, the sender can simply provide them with the encrypted file and the encrypted AES key. The recipient can then use their private RSA key to decrypt the AES key and the data. To further enhance the security of the system, the sender can use a key management service to securely store and manage the RSA keys. This can help to ensure that the keys are always available when needed, while also providing additional security controls and auditing capabilities.
- Finally, it is important to regularly monitor the security of the system and the encrypted data. This can include conducting regular security audits and penetration testing to identify potential vulnerabilities, and implementing measures to address any issues that are discovered.

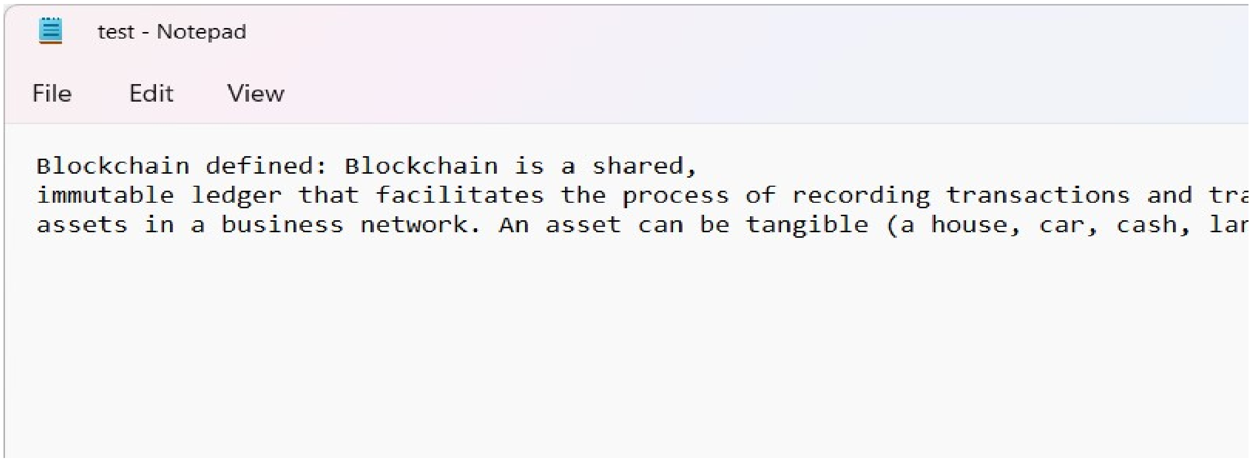
6 Evaluation

The proposed algorithm can encrypt and decrypt files of any size using AES and RSA keys. The system requires the user to import the file from any place on the system that must be encrypted and decrypted. The client will specify the encrypted file title with the. enc extension after exporting the file.

The process utilizes the combined strategy for the two encryption computations and is a protected information transmission system. The Advanced Encryption Standard (AES) and RSA are the two techniques used in the framework. Any information file can be encoded and decoded using the framework. It is expected that it will be utilized in a realistic programming role. The system produces a random AES key and performs encryption for the identified information document. The RSA encryption scheme is then employed to encrypt the applied AES key. The encrypted information document and the jumbled AES key are merged and sent over to the communication media. The joined document is recognized by the recipient.

The file will be successfully decrypted using the following code if the need for decryption occurs once it has been properly encrypted using the provided keys and stored in the project folder (Rashid, et. al., 2022). Because it was instantly saved there to be encrypted, the file should be chosen from the project folder. The keys should also be used. The results of files of various sizes and formats that were encrypted using AES, RSA, and the suggested technique. To assess the effectiveness of the suggested method, the results analysis was based on the time parameter. The hybrid algorithm provides increased security by combining the strengths of both AES and RSA. AES is a widely-used and secure symmetric encryption algorithm, while RSA is a widely-used and secure public-key encryption algorithm. By using RSA to encrypt the AES key, the hybrid algorithm adds an additional layer of encryption to the data, making it more difficult for an attacker to access the unencrypted message.

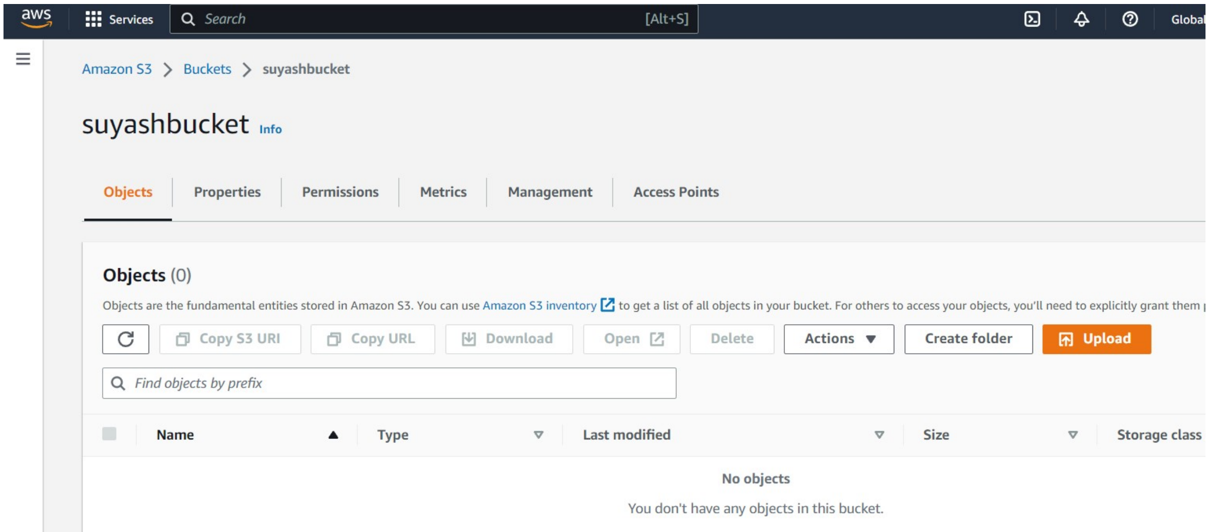
6.1 Experiment 1



```
Blockchain defined: Blockchain is a shared,
immutable ledger that facilitates the process of recording transactions and tra
assets in a business network. An asset can be tangible (a house, car, cash, lar
```

Figure 1: Test data file

This the data file for testing. Here some content has been written the file which will be encrypted and uploaded to the AWS.



Objects (0)

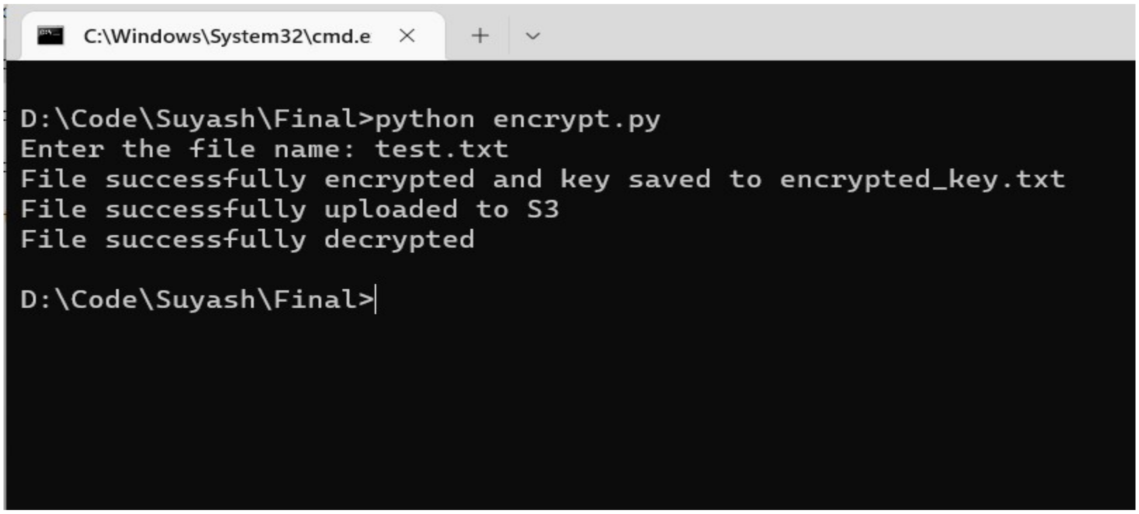
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them

No objects

You don't have any objects in this bucket.

Figure 2: AWS Bucket

All the encrypted data will get uploaded on the bucket.

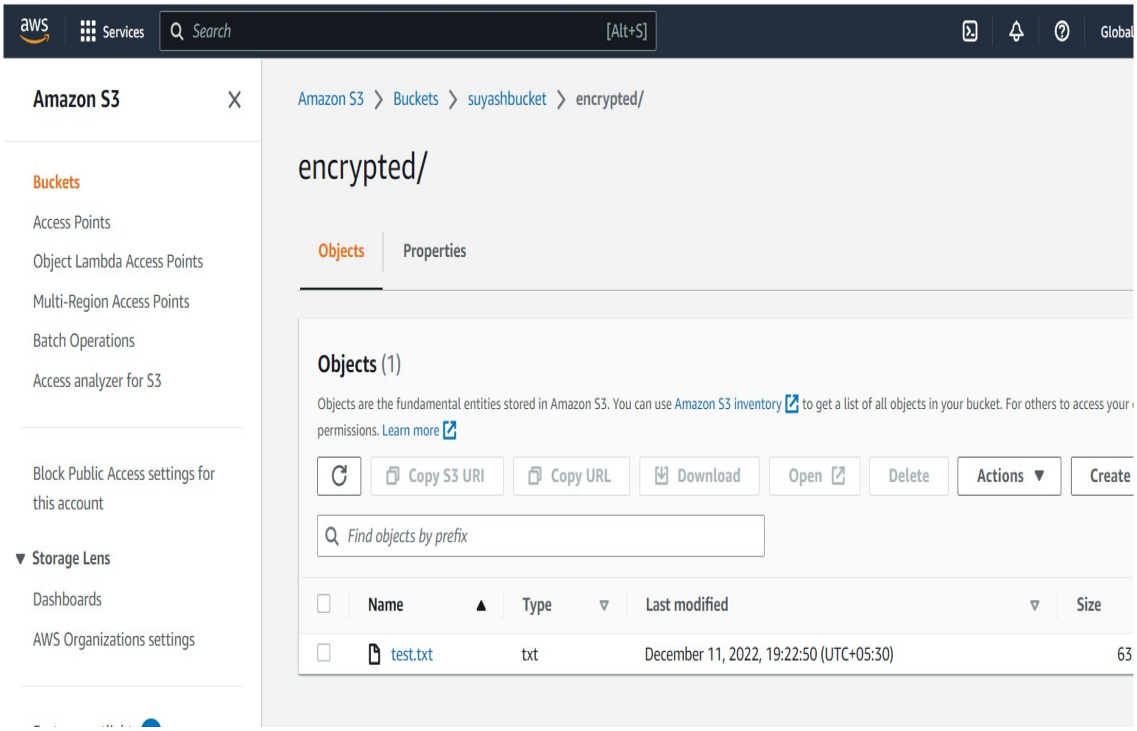


```
D:\Code\Suyash\Final>python encrypt.py
Enter the file name: test.txt
File successfully encrypted and key saved to encrypted_key.txt
File successfully uploaded to S3
File successfully decrypted

D:\Code\Suyash\Final>
```

Figure 3: Encrypting the file

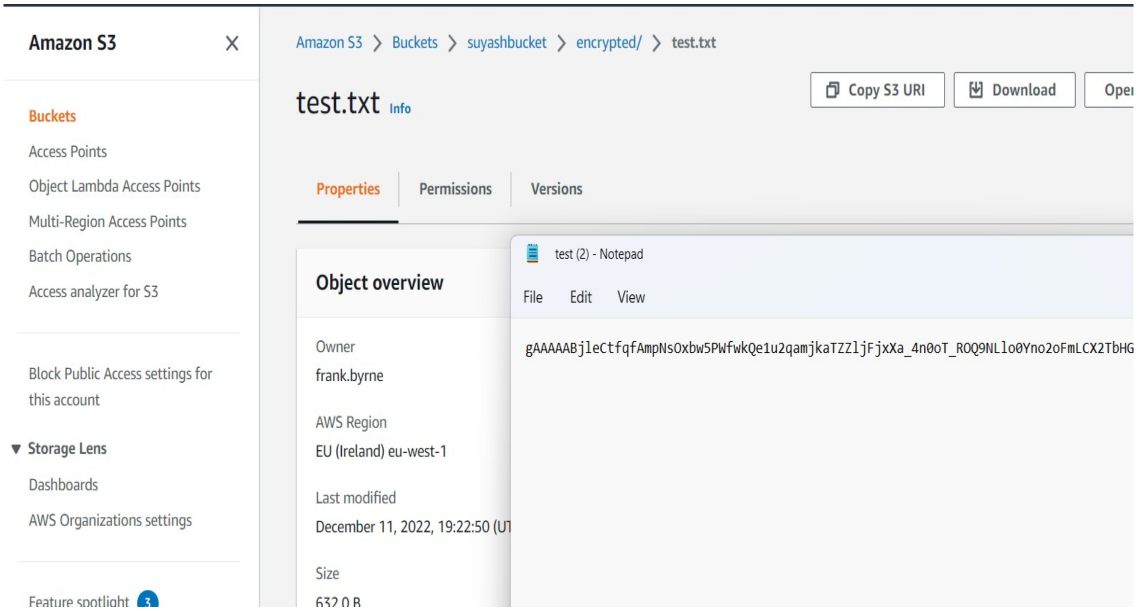
When the script run, it will ask for the user input to enter the file name and after providing the file name it will start the encryption process, the key for decryption will get saved on the user system and the encrypted file will get uploaded on the AWS bucket.



Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your
permissions. Learn more

□	Name ▲	Type ▽	Last modified ▽	Size
□	test.txt	txt	December 11, 2022, 19:22:50 (UTC+05:30)	63

Figure 4: Encrypted file uploaded



Owner
frank.byrne

AWS Region
EU (Ireland) eu-west-1

Last modified
December 11, 2022, 19:22:50 (UT

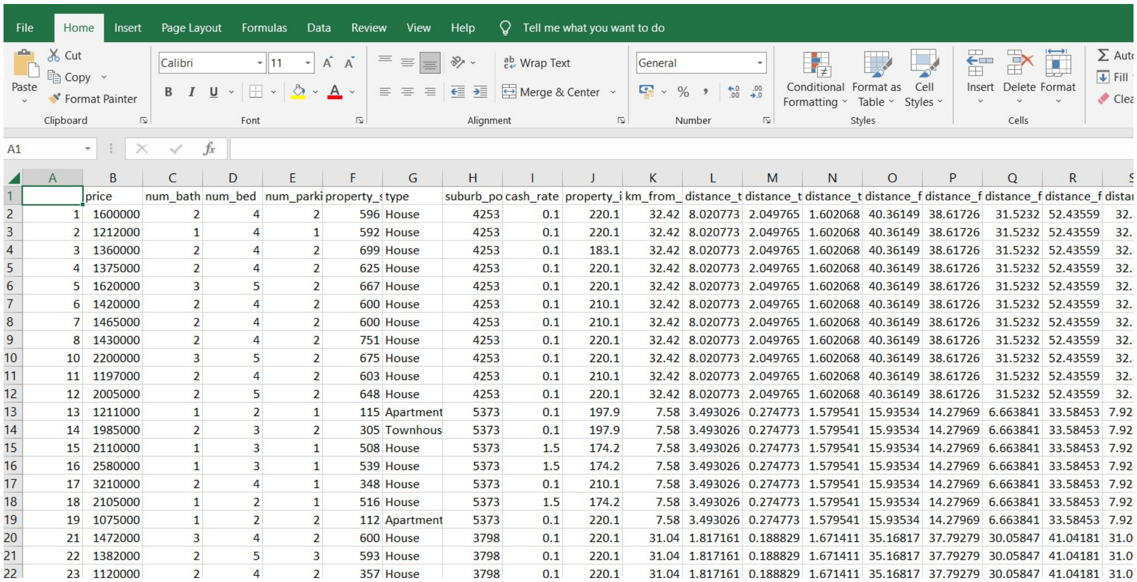
Size
632.0 B

```
gAAAAABjleCtfqfAmpNsOxbw5PWfwkQe1u2qamjkaTZZljFjxXa_4n0oT_ROQ9NLlo0Yno2oFmLCX2TbHG
```

Figure 5: Encrypted file

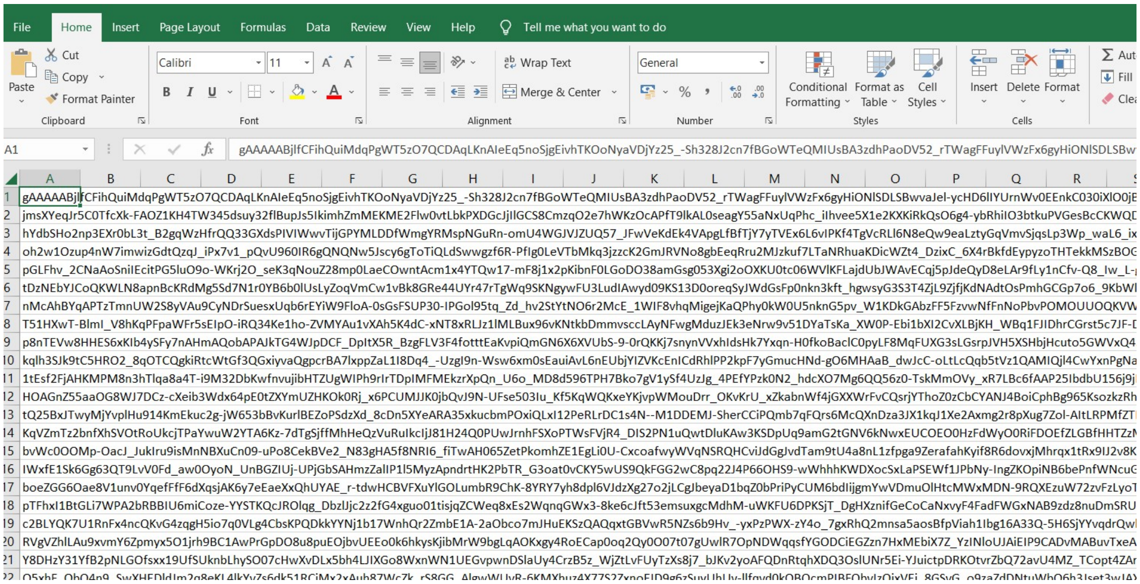
When some one downloads the datafile from the bucket it will be in encrypted form So no can able to see the real data until it gets decrypted.

6.2 Experiment 2



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1		price	num_bath	num_bed	num_parki	property_s	type	suburb_po	cash_rate	property_i	km_from_	distance_t	distance_t	distance_t	distance_f	distance_f	distance_f	distance_f	distar
2	1	1600000	2	4	2	596	House	4253	0.1	220.1	32.42	8.020773	2.049765	1.602068	40.36149	38.61726	31.5232	52.43559	32.
3	2	1212000	1	4	1	592	House	4253	0.1	220.1	32.42	8.020773	2.049765	1.602068	40.36149	38.61726	31.5232	52.43559	32.
4	3	1360000	2	4	2	699	House	4253	0.1	183.1	32.42	8.020773	2.049765	1.602068	40.36149	38.61726	31.5232	52.43559	32.
5	4	1375000	2	4	2	625	House	4253	0.1	220.1	32.42	8.020773	2.049765	1.602068	40.36149	38.61726	31.5232	52.43559	32.
6	5	1620000	3	5	2	667	House	4253	0.1	220.1	32.42	8.020773	2.049765	1.602068	40.36149	38.61726	31.5232	52.43559	32.
7	6	1420000	2	4	2	600	House	4253	0.1	210.1	32.42	8.020773	2.049765	1.602068	40.36149	38.61726	31.5232	52.43559	32.
8	7	1465000	2	4	2	600	House	4253	0.1	210.1	32.42	8.020773	2.049765	1.602068	40.36149	38.61726	31.5232	52.43559	32.
9	8	1430000	2	4	2	751	House	4253	0.1	220.1	32.42	8.020773	2.049765	1.602068	40.36149	38.61726	31.5232	52.43559	32.
10	10	2200000	3	5	2	675	House	4253	0.1	220.1	32.42	8.020773	2.049765	1.602068	40.36149	38.61726	31.5232	52.43559	32.
11	11	1197000	2	4	2	603	House	4253	0.1	210.1	32.42	8.020773	2.049765	1.602068	40.36149	38.61726	31.5232	52.43559	32.
12	12	2005000	2	5	2	648	House	4253	0.1	220.1	32.42	8.020773	2.049765	1.602068	40.36149	38.61726	31.5232	52.43559	32.
13	13	1211000	1	2	1	115	Apartment	5373	0.1	197.9	7.58	3.493026	0.274773	1.579541	15.93534	14.27969	6.663841	33.58453	7.92
14	14	1985000	2	3	2	305	Townhous	5373	0.1	197.9	7.58	3.493026	0.274773	1.579541	15.93534	14.27969	6.663841	33.58453	7.92
15	15	2110000	1	3	1	508	House	5373	1.5	174.2	7.58	3.493026	0.274773	1.579541	15.93534	14.27969	6.663841	33.58453	7.92
16	16	2580000	1	3	1	539	House	5373	1.5	174.2	7.58	3.493026	0.274773	1.579541	15.93534	14.27969	6.663841	33.58453	7.92
17	17	3210000	2	4	1	348	House	5373	0.1	210.1	7.58	3.493026	0.274773	1.579541	15.93534	14.27969	6.663841	33.58453	7.92
18	18	2105000	1	2	1	516	House	5373	1.5	174.2	7.58	3.493026	0.274773	1.579541	15.93534	14.27969	6.663841	33.58453	7.92
19	19	1075000	1	2	2	112	Apartment	5373	0.1	220.1	7.58	3.493026	0.274773	1.579541	15.93534	14.27969	6.663841	33.58453	7.92
20	21	1472000	3	4	2	600	House	3798	0.1	220.1	31.04	1.817161	0.188829	1.671411	35.16817	37.79279	30.05847	41.04181	31.0
21	22	1382000	2	5	3	593	House	3798	0.1	220.1	31.04	1.817161	0.188829	1.671411	35.16817	37.79279	30.05847	41.04181	31.0
22	23	1120000	2	4	2	357	House	3798	0.1	220.1	31.04	1.817161	0.188829	1.671411	35.16817	37.79279	30.05847	41.04181	31.0

Figure 6: Data in CSV Format

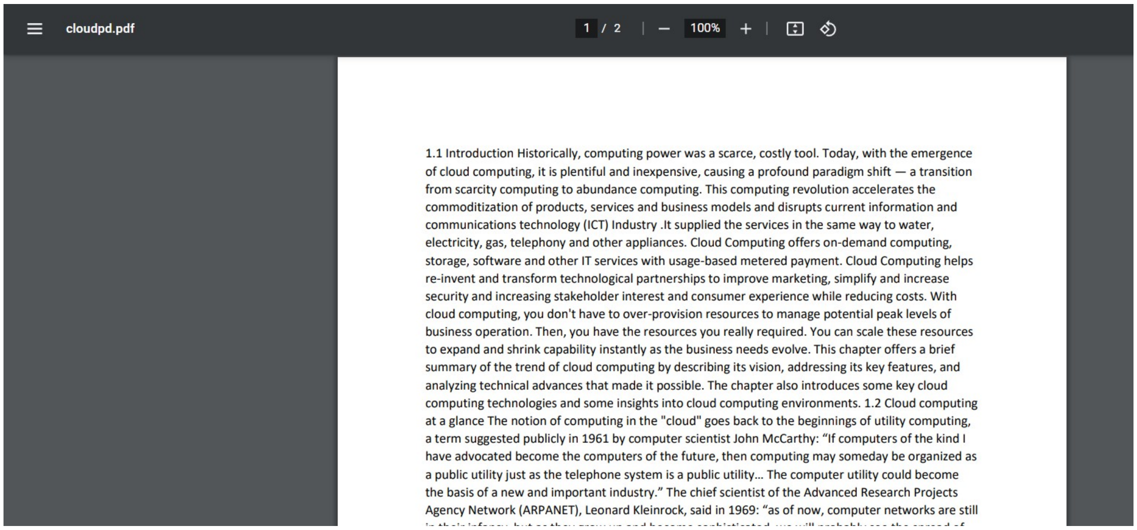


gAAAAABjlfCFihQuiMdqPgWT5zO7QCDAqLKnAIeEq5noSjgEivhTKOoNyaVDjYz25_-Sh328J2cn7fBGoWTeQMIUsBA3zdhPaoDV52_rTWagFFuylVWzFx6gyHiONlSDLSBw

	A
1	gAAAAABjlfCFihQuiMdqPgWT5zO7QCDAqLKnAIeEq5noSjgEivhTKOoNyaVDjYz25_-Sh328J2cn7fBGoWTeQMIUsBA3zdhPaoDV52_rTWagFFuylVWzFx6gyHiONlSDLSBwvaJel-ycHD6lIYUrnWv0EEnkC030iXlO0j
2	jmsXYeqJr5C0TfcXk-FAOZ1KH4TW345dsuy32flBupJs5IkimhZmMEKME2Flw0vtLbkPXDGcJjIlGCS8CmzqO2e7hWKzOcAPfT9lkAL0seagY55aNxUqPhc_iIhvee5X1e2KXKiRkQsO6g4-ybRhiIO3btkuPVGesBcCKWQD
3	hYdbSHo2np3EXr0bL3t_B2gqWzHfrQQ33GXdsPIVIWwvTijGPYMLDDfWmgYRMspNGuRn-omU4WGJVJZUQ57_JFwVeKdEk4VApgLfBfTjY7yTVEx6L6vIPKf4TgVcRLl6N8eQw9eaLztyGqVmvSjqsLp3Wp_waL6_ix
4	oh2w1Ozup4nW7imwizGdtQzqJ_iPx7v1_pQvU960IR6gQNQNw5Jscy6gToTiQLdSwwgzf6R-PfIg0LeVTbMkq3jzzcK2GmJRVNo8gbEeqRru2MJzkuf7LTaNRhuaKDicWZt4_DzixC_6X4rBkfdEypyzoTHTekkMSzBOG
5	pGLFhv_2CNaAoSniIEcitPG5luO9o-WKrj2O_seK3qNouZ28mp0LaeCOwntAcm1x4YTQw17-mF8j1x2pKibnF0LGoDO38amGsg053Xgi2oOXKU0tc06WVlKFLajdUbJWAvECqj5pJdeQyD8eLAr9fLy1nCfv-Q8_Iw_L-
6	tDzNEbYJCoQKWLN8apnBcKRdMg5Sd7N1r0YB6b0lUsLyZoqVmCw1vBk8GRe44UYr47rTgWq9SKNgywFU3LudIAwyd09KS13D0oreqSyJWdGsFp0nkn3kft_hgwsyG3S3T4ZjL9ZjfjKdNAdtOsPmhGCGp7o6_9KbWl
7	nMcAhBYqAPTzTmnUW2S8yVAu9CyNDrSuesxUqb6rEYiW9FloA-0sGsFSUP30-IPGol95tq_Zd_hv2StYtNO6r2McE_1WIF8vhqMigejKaQPhy0kW0U5nknG5pv_W1KDkGAbzFF5FzvwNfFnNoPbvPOMOUUOQKVW
8	T51HXwT-BlmI_V8hKqPFpaWFr5sEIpO-iRQ34Ke1ho-ZVMYAu1vXAh5K4dC-xNT8xRLJz1lMLBux96vKNtkbDmmvsccLAyNFwgMduzJEk3eNrw9v51DYaTsKa_XW0P-Ebi1bXI2CvXLBjKH_WBq1FJIDhrCGrst5c7JF-D
9	p8nTEVw8HHES6xKIb4ySFy7nAHmAQobAPAJkTG4WJpDCF_DpItX5R_BzgFLV3F4fotttEaKvpiQmGN6X6XVUbS-9-0rQKKj7snynVVxhIdsHk7Yxqn-H0fkoBaclC0pyLF8MqFUXG3sLGsrpJVH5XSHbjHcuto5GWVxQ4
10	kqlh3SJk9tC5HRO2_8qOTCQgkiRtcWtGf3QGxiyvaQgpcrBA7lxppZaL1I8Dq4_-UzgI9n-Wsw6xm0sEauiAvL6nEUbjYIZVKcEnICdRhlPP2kpF7yGmucHNd-gO6MHAaB_dwJcC-oLtLcQqb5tVz1QAMIQjl4CwYxnPgNa
11	1tEsf2FjAHKMPM8n3hTlqa8a4T-i9M32DbKwfnvujibHTZUgWIPh9rIrTDpIMFMEkzrXpQn_U6o_MD8d596TPH7Bko7gV1ySf4UzJg_4PEfYPzk0N2_hdcXO7Mg6QQ56z0-TskMmOVy_xR7LBc6fAAP25IbdbU156j9j
12	HOAGnZ55aaOG8WJ7DCz-cXeib3Wdx64pE0tZXYmUZHKOk0Rj_x6PCUMJJK0jbQvJ9N-UFse503Iu_Kf5KqWQKxeYKjvpWMouDrr_OKvKrU_xZkabnWf4jGXXWrFvCQsrjYThoZ0zCbCYANJ4BoiCphBg965KsozkzRh
13	tQ25BxJTwyMjYvplHu914KmEkuc2g-jW653bBvKurlBEZoPSdzXd_8cDn5XYeARA35xkucbmPOxiQLxI12PeRLrDC1s4N--M1DDEMJ-SherCCiPQmb7qFQrs6McQXnDza3JX1kqJ1Xe2Axmg2r8pXug7Zol-AltLRPMfZT
14	KqVZmTz2bnfXhSVOtRoUkcjTPaYwuW2YTA6Kz-7dTgSjffMhHeQzVuRuIkcIjJ81H24Q0PUwJrnhFSXoPTWsFVjR4_DIS2PN1uQwtDluKAw3KSDpUq9amG2tGNV6kNwxEUCOEO0HzFdWyO0RiFDOEfZLGBfHHTZz
15	bvWc0OOMp-OacJ_JukIru9isMnNBXuCn09-uPo8CekBVe2_N83gHA5f8NRI6_fiTwAH065ZetPkomhZE1EgLi0U-CxcoafwyWVqNSRQHCviJdGgJvdTam9tU4a8nL1zfpga9ZerafahKyif8R6dovxjMhrqx1tRx9IJ2v8K
16	IWxfE1Sk6Gg63QT9LvV0Fd_aw0OyoN_UnBGZIUj-UPjGbSAHmzZalIP1l5MyzApndrtHK2PbTR_G3oat0vCKY5wUS9QkFGG2wC8pq22J4P66OHS9-wWhhhKWDXocSxLaPSEWf1JPbNy-IngZKOpiNB6bePnfWNcuG
17	boeZGG6Oae8V1unv0YqefFfF6dXqsjAK6y7eEaeXxQhUYAE_r-tdwHCBVFXuYlGOLumbR9ChK-8YRY7yh8dpl6VJdzXg27o2jLCgJbeyaD1bqZ0bPriPyCUM6bdIijgmYwVDmuOlHtcMWxMDN-9RQXEzuW72zvFzLyoT
18	pTFhxI1BtGLi7WPA2bRBBIU6miCoze-YYSTKQcJROlqg_DbzlJjc2z2fG4xguo01tisjqZCWeq8xEs2WqnqGWx3-8ke6cJft53emsuxgcMdhM-uWKFU6DPKSjT_DgHXznifGeCoCaNxvyF4FadFWGxNAB9zdz8nuDmSRU
19	c2BLYQK7U1RnFx4ncQKvG4zqgH5io7q0VLg4CbsKPQDkkYYNj1b17WnhQr2ZmbE1A-2aObco7mJHuEKSzQAQqxtGBVwR5NZs6b9Hv_-yxPzPWX-zY4o_7gxRhQ2mnsa5aosBfpViah1Ibg16A33Q-5H6SjYYvqdrQw
20	RVgVZhlLAu9xvmY6Zpmyx5O1jrh9BC1AwPrGpDO8u8puEOjbvUEEo0k6hkysKjibMrW9bgLqAOKxgy4RoECap0oq2Qy0O07t07gUwlR7OpNDWqqsfYGODCiEGZzn7HxMEbiX7Z_YzINloUJAiEIP9CADvMABuvTxeA
21	Y8DHzY31YfB2pNLGOfsxx19UfSUknbLhySO07cHwXvDLx5bh4LJIXGo8WxnWN1UEGvpwnDSlaUy4CrzB5z_WjZtLvFUyTzXs8j7_bJKv2yoAFQDnRtqhXDQ3OslUNr5Ei-YJuictpDRKOtvrZbQ72avU4MZ_TCopt4ZAr
22	O5xhF_ObO4n9_SwXHFDldJm2q8eKL4lkYv7s6dk51RCiMx2xAuh87Wc7k_rS8GG_AlgwWLJvR-6KMXhuz4X77S27xnoFID9g6zSuvUhUv-llfgvd0kOBOcmPIBFOhvIzOixVFi_8GSvG_o9zaZdDNtuWhO6h3Jset3wJAY

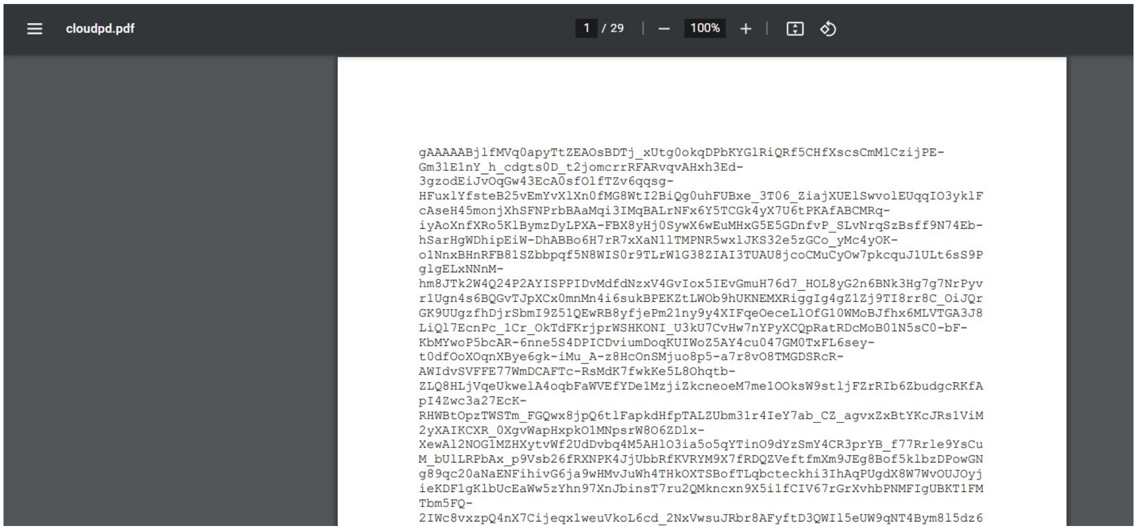
Figure 7: Data after Encryption

6.3 Experiment 3



1.1 Introduction Historically, computing power was a scarce, costly tool. Today, with the emergence of cloud computing, it is plentiful and inexpensive, causing a profound paradigm shift — a transition from scarcity computing to abundance computing. This computing revolution accelerates the commoditization of products, services and business models and disrupts current information and communications technology (ICT) Industry .It supplied the services in the same way to water, electricity, gas, telephony and other appliances. Cloud Computing offers on-demand computing, storage, software and other IT services with usage-based metered payment. Cloud Computing helps re-invent and transform technological partnerships to improve marketing, simplify and increase security and increasing stakeholder interest and consumer experience while reducing costs. With cloud computing, you don't have to over-provision resources to manage potential peak levels of business operation. Then, you have the resources you really required. You can scale these resources to expand and shrink capability instantly as the business needs evolve. This chapter offers a brief summary of the trend of cloud computing by describing its vision, addressing its key features, and analyzing technical advances that made it possible. The chapter also introduces some key cloud computing technologies and some insights into cloud computing environments. 1.2 Cloud computing at a glance The notion of computing in the "cloud" goes back to the beginnings of utility computing, a term suggested publicly in 1961 by computer scientist John McCarthy: "If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is a public utility… The computer utility could become the basis of a new and important industry." The chief scientist of the Advanced Research Projects Agency Network (ARPANET), Leonard Kleinrock, said in 1969: "as of now, computer networks are still

Figure 8: Data in PDF format



```
gAAAAABjlfMVq0apyTtZEAOsBDTj_xUtg0okqDPbKYGlRiQRf5CHfXscsCmMlCzijPE-
Gm3lE1nY_h_cdgts0D_t2jomcrrRFARvqvAHxh3Ed-
3gzodEiJvOqGw43EcA0sfOlfTZv6qqsg-
HFuxlYfsteB25vEmYvXlXn0fMG8WtI2BiQg0uhFUBxe_3T06_ZiajXUElSwvolEUqqIO3yklF
cAseH45monjXhSFNPrbBAaMqi3IMqBALrNFx6Y5TCGk4yX7U6tPKAfABCMRq-
iyAoXnfXRo5KlBymzDyLPXA-FBX8yHj0SywX6wEuMHxG5E5GDnfvP_SLvNrqSzBsff9N74Eb-
hSarHgWDhipEiW-DhABBo6H7rR7xXaN1lTMPNR5wxlJKS32e5zGCo_yMc4yOK-
o1NnxBHnRFB81SZbbpqf5N8WIS0r9TLrW1G38ZIAI3TUAU8jcoCMuCyOw7pkcquJ1ULt6sS9P
glgELxNNnM-
hm8JTk2W4Q24P2AYISPPIDvMdfdNzxV4GvIox5IEvGmuH76d7_HOL8yG2n6BNk3Hg7g7NrPyv
r1Ugn4s6BQGvTJpXCx0mnMn4i6sukBPEKZtLWOb9hUKNEMXRiggIg4gZ1Zj9TI8rr8C_OiJQr
GK9UUgzfhDjrSbmI9Z51QEwRB8yfjePm21ny9y4XIFqeOeceLlOfG10WMoBJfhx6MLVTGA3J8
LiQl7EcnPc_1Cr_OkTdFKrjprWSHKONI_U3kU7CvHw7nYPyXCQpRatRDcMoB01N5sC0-bF-
KbMYwoP5bcAR-6nne5S4DPICDviumDoqKUIWoZ5AY4cu047GM0TxFL6sey-
t0dfOoXOqnXBye6gk-iMu_A-z8HcOnSMjuo8p5-a7r8vO8TMGDSRcR-
AWIdvSVFFE77WmDCAFTc-RsMdK7fwkKe5L8Ohqtb-
ZLQ8HLjVqeUkwelA4oqbFaWVEfYDe1MzjiZkcneoeM7me1OOksW9stljFZrRIb6ZbudgcRKfA
pI4Zwc3a27EcK-
RHWBtOpzTWSTm_FGQwx8jpQ6tlFapkdHfpTALZUbm31r4IeY7ab_CZ_agvxZxBtYKcJRs1ViM
2yXAIKCXR_0XgvWapHxpkO1MNpsrW8O6ZDlx-
XewAl2NOG1MZHXytvWf2UdDvbq4M5AHlO3ia5o5qYTinO9dYzSmY4CR3prYB_f77Rrle9YsCu
M_bUlLRPbAx_p9Vsb26fRXNPK4JjUbbRfKVRYM9X7fRDQZVeftfmXm9JEg8Bof5klbzDPowGN
g89qc20aNaENFihivG6ja9wHMvJuWh4THkOXTSBofTLqbcteckhi3IhAqPUgdX8W7WvOUJOyj
ieKDF1gKlbUcEaWw5zYhn97XnJbinsT7ru2QMkncxn9X5i1fCIV67rGrXvhbPNMFIgUBKT1FM
Tbm5FQ-
2IWc8vxzpQ4nX7Cijeqx1weuVkoL6cd_2NxVwsuJRbr8AFyftD3QWIl5eUW9qNT4Bym8l5dz6
```

Figure 9: PDF after encryption

From the above experiment it can be seen that the script that has been created will work for all types of formats.

6.4 Discussion

The hybrid encryption algorithm combining AES and RSA is a popular and widely-used method of encrypting data. It provides increased security by combining the strengths of both AES and RSA, and allows the recipient to decrypt the message without having to share the AES key with the sender. This makes it a useful and convenient tool for securely transmitting sensitive information.

However, the hybrid algorithm also has some potential disadvantages. It can be more complex to implement than a single encryption algorithm, and may not be as efficient in some cases. Additionally, it requires implementations of both the AES and RSA algorithms, as well as a way to combine and manage the encrypted data.

Despite these disadvantages, the hybrid algorithm remains a valuable tool for encrypting data. In many cases, the increased security and convenience provided by the hybrid algorithm may justify the additional complexity and potential loss of efficiency. However, the decision to use the hybrid algorithm should be based on a careful evaluation of the specific needs and requirements of the application.

Plaintext size (KB)	Plaintext size (Byte)	RSA+AES Encryption Time (seconds)
32	32710	0.009
64	65420	0.010
128	130840	0.012
256	261680	0.014
512	523360	0.016
1024	1048460	0.018
2048	2096920	0.020
4096	4193840	0.021

The above table shows the time taken by the proposed hybrid algorithm for the given size. Let compare it with the previous algorithm presented.

Plaintext size (KB)	Plaintext size (Byte)	RSA+Blowfish Encryption Time (seconds)
32	32710	0.19
64	65420	0.22
128	130840	0.25
256	261680	0.28
512	523360	0.30
1024	1048460	0.33
2048	2096920	0.35
4096	4193840	0.38

From the above tables it can be seen that the proposed hybrid algorithm has faster time as compared to previously created hybrid algorithms.

7 Conclusion and Future Work

Since RSA encryption has an extremely low data cap because of the significant energy usage while encryption and decryption, it is somewhat difficult to encrypt and decode files just using the RSA method.. This makes it impractical to use for encrypting large amounts of data. In order to address this problem, a hybrid encryption algorithm can be used that combines both symmetric and asymmetric encryption. By using a symmetric algorithm like AES for the actual encryption and decryption of the data, and an asymmetric algorithm like RSA for securely exchanging the keys, it is possible to encrypt larger amounts of data while still maintaining security. This approach also solves the issues with symmetric encryption's key mobility and asymmetric encryption's excessive power use. Hybrid encryption allows for the benefits of both types of encryptions to be utilized, providing both security and efficiency. The research in this study has demonstrated the ability to encrypt and decrypt files of a certain size, and has also measured the time required for each operation. However, future studies will need to address potential problems with the current approach, such as the risk that data manipulation and forgery may be possible if the double key is compromised. Further research will be needed to explore potential solutions to these issues.

In future more algorithms like Blowfish, RC6 can be added to make it more secure and also right now it’s a command line-based program in future a GUI application can be made for easiness of the user where the user needs to enter the login details and bucket name and upload the file. This method may also be used directly to the data coming in from IOT devices. In order to assure total data security while it is saved in the cloud, this technique may be used to IOT data that is directly stored there. In many cases, incoming IOT data is immediately stored in the cloud and accessible by team members. However, since the whole team has access to a given cloud storage, anybody may read that data. With the method suggested in this article, only authentic users will receive the decryption key, and data stored in the cloud will be encrypted, allowing only those members to see or use the data. Rest of the team members will the not be able to view this data.

References

Ahmad, S.A. and Garko, A.B., 2019, December. Hybrid cryptography algorithms in cloud computing: A review. In *2019 15th International Conference on Electronics, Computer and Computation (ICECCO)* (pp. 1-6). IEEE.

Alattab, A.A., Irshad, R.R., Yahya, A.A. and Al-Awady, A.A., 2022. Privacy Protected Preservation of Electric Vehicles' Data in Cloud Computing Using Secure Data Access Control. *Energies*, *15*(21), p.8085.https://www.mdpi.com/1917702

Albahar, M.A., Olawumi, O., Haataja, K. and Toivanen, P., 2018. Novel hybrid encryption algorithm based on aes, RSA, and twofish for bluetooth encryption.

Bhalla, M. and Prasad, S., 2022, October. A security implementation framework for IoT ecosystem. In *AIP Conference Proceedings* (Vol. 2519, No. 1, p. 030041). AIP Publishing LLC.https://aip.scitation.org/doi/abs/10.1063/5.0110595

Chandu, Y., Kumar, K.R., Prabhukhanolkar, N.V., Anish, A.N. and Rawal, S., 2017, August. Design and implementation of hybrid encryption for security of IOT data. In *2017 International conference on smart technologies for smart nation (SmartTechCon)* (pp. 1228-1231). IEEE.

Ekka, D., Kumari, M. and Yadav, N., 2019. Enrichment of security using hybrid algorithm. In *International Conference on Computer Networks and Communication Technologies* (pp. 867-873). Springer, Singapore.

Ghani, A., Badshah, A., Jan, S., Alshdadi, A.A. and Daud, A., 2020. Issues and challenges in cloud storage architecture: a survey. *arXiv preprint arXiv:2004.06809*.https://arxiv.org/abs/2004.06809

Ghosh, S.N., 2015. Performance analysis of AES, DES, RSA and AES-DES-RSA hybrid algorithm for data security. *International Journal of Innovative and Emerging Research in Engineering*, *2*(5), pp.83-88.

Gupta, D., Bhatt, S., Gupta, M., Kayode, O. and Tosun, A.S., 2020, May. Access control model for google cloud iot. In *2020 IEEE 6th Intl conference on big data security on cloud (BigDataSecurity), IEEE Intl conference on high performance and smart computing,(HPSC) and IEEE Intl conference on intelligent data and security (IDS)* (pp. 198-208). IEEE.https://ieeexplore.ieee.org/abstract/document/9123054/

Guru, M.A. and Ambhaikar, A., 2021. AES and RSA-based Hybrid Algorithms for Message Encryption & Decryption. *Information Technology in Industry*, *9*(1), pp.273-279.

Harini, M., Gowri, K.P., Pavithra, C. and Selvarani, M.P., 2017, April. A novel security mechanism using hybrid cryptography algorithms. In *2017 IEEE International Conference on Electrical, Instrumentation and Communication Engineering (ICEICE)* (pp. 1-4). IEEE.

Jintcharadze, E. and Iavich, M., 2020, September. Hybrid implementation of Twofish, AES, ElGamal and RSA cryptosystems. In *2020 IEEE East-West Design & Test Symposium (EWDTS)* (pp. 1-5). IEEE.

Kumar, B., Boaddh, J. and Mahawar, L., 2016. A hybrid security approach based on AES and RSA for cloud data. *International Journal of Advanced Technology and Engineering Exploration*, *3*(17), p.43.

Kuswaha, S., Waghmare, S. and Choudhary, P., 2015. Data Transmission using AES-RSA Based Hybrid Security Algorithms. *International Journal on Recent and Innovation Trends in Computing and Communication*, *3*(4), pp.1964-1969.

Liu, Y., Gong, W. and Fan, W., 2018, June. Application of AES and RSA Hybrid Algorithm in E-mail. In *2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)* (pp. 701-703). IEEE.

Mahalle, V.S. and Shahade, A.K., 2014, October. Enhancing the data security in Cloud by implementing hybrid (Rsa & Aes) encryption algorithm. In *2014 International Conference on Power, Automation and Communication (INPAC)* (pp. 146-149). IEEE.

Murad, S.H. and Rahouma, K.H., 2021. Implementation and Performance Analysis of Hybrid Cryptographic Schemes applied in Cloud Computing Environment. *Procedia Computer Science*, *194*, pp.165-172.

Pandian, R. and Columbus, C., 2022. An Analytical approach for optimal secured data storage on cloud server for online education platform. *Geoscientific Instrumentation, Methods and Data Systems Discussions*, pp.1-36.https://gi.copernicus.org/preprints/gi-2022-15/

Rashid, M.N., Abed, L.H. and Awad, W.K., 2022. Financial information security using hybrid encryption technique on multi-cloud architecture. *Bulletin of Electrical Engineering and Informatics*, *11*(6), pp.3450-3461.

Rehman, S., Talat Bajwa, N., Shah, M.A., Aseeri, A.O. and Anjum, A., 2021. Hybrid AES-ECC Model for the Security of Data over Cloud Storage. *Electronics*, *10*(21), p.2673.

Xiong, S., Ni, Q., Wang, L. and Wang, Q., 2020. SEM-ACSIT: secure and efficient multiauthority access control for IoT cloud storage. *IEEE Internet of Things Journal*, *7*(4), pp.2914-2927.https://ieeexplore.ieee.org/abstract/document/8949526/

Zhang, F., Chen, Y., Meng, W. and Wu, Q., 2019. Hybrid encryption algorithms for medical data storage security in cloud database. *International Journal of Database Management Systems (IJDMS) Vol*, *11*.

Zhou, L., Wang, L., Sun, Y. and Lv, P., 2018. Beekeeper: A blockchain-based iot system with secure storage and homomorphic computation. *IEEE Access*, *6*, pp.43472-43488.https://ieeexplore.ieee.org/abstract/document/8386749/

Zou, L., Ni, M., Huang, Y., Shi, W. and Li, X., 2019, July. Hybrid encryption algorithm based on AES and RSA in file encryption. In *International Conference on Frontier Computing* (pp. 541-551). Springer, Singapore.