Enhancement of Big Data Security in Cloud Computing Using RSA Algorithm

[1]Abel Yeboah-Ofori
School of Computing and Engineering
University of West London
United Kingdom
abel.yeboah-ofori@uwl.ac.uk

[1]Iman Darvishi
School of Computing and Engineering
University of West London
United Kingdom
iman.darvishi@uwl.ac.uk

[2]Azeez Sakirudeen Opeyemi
School of Computing and Engineering
University of West London
United Kingdom
21518893@uwl.ac.uk

Abstract--The enhancement of big data security in cloud computing has become inevitable dues to factors such as the volume, velocity, veracity, Value, and velocity of the big data. These enhancements of big data and cloud technologies have computing enabled a wide rangeof vulnerabilities in applications in organizational business environments leading to various attacks such as denial-of-service attacks, injection attacks, and Phishing among others. Deploying big data in cloud computing environments is a rapidly growing technology that significantly impacts organizations and provides benefits such as demand-driven access to computational services, a distorted version of infinite computing capacity, and assistance with demand-driven scaling up, scaling down, and scaling out. To secure cloud computing for big data processing, a variety of encryption techniques such as RSA, and AES can be applied. However, there are several vulnerabilities during processing.

The paper aims to explore the enhancement of big data security in cloud computing using the RSA algorithm to improve the deployment and processing of the variety, volume, veracity, velocity and value of the data utilizing RSA encryptions.

The novelty contribution of the paper is threefold: First, explore the current challenges and vulnerabilities in securing big data in cloud computing and how the RSA algorithm can be used to address them. Secondly, we implement the RSA algorithm in a cloud computing environment using the AWS cloud platform to secure big data to improve the performance and scalability of the RSA algorithm for big data security in cloud computing. We compare the RSA algorithm to other cryptographic algorithms in terms of its ability to enhance big data security in cloud computing. Finally, we recommend control mechanisms to improve security in the cloud platform. The results show that the RSA algorithm can be used to improve Cloud Security in a network environment.

Keywords-*Cloud Security, Big Data Security, Encryption,RSA Algorithm, Cyber Security*

I. INTRODUCTION

The recent introduction of big data and cloud computing technologies has enabled a wide range of applications in organizational business environments, leading to various attacks such as denial-of-service attacks, injection attacks, and Phishing among others. Cloud computing can be referred to as an online service or resource that utilizes, offers and deliver physical or virtualized resources through the Internet [1]. It enables organizations to focus on their core competencies by totally abstracting processing, storage and network resources to workloads as needed and utilizing a plethora of pre-built services [2]. Big Data is typically characterized as a collection of extremely large data sets of various types, making it challenging to process them using conventional data processing platforms and methods. It refers to enormous data volumes that are orders of magnitude larger in number(Volume), more varied in kind, containing organized, unstructured & semi-structured data (Variety), arriving ata faster rate than any enterprise has ever had to cope with (Velocity), the uncertainty and potential inaccuracies of thedata which can make it difficult to trust the insights gained from the data (Veracity) and the likelihood of data leaks and cyberattacks that could jeopardize the security of big data (Vulnerability) [3]. Many businesses in a variety of sectors use big data to uncover new information and enhance existing processes [4]. Some examples include Technology(Google, Amazon), Retail (Walmart, Sainsbury) among others. However, there are still some significant issues in the 5 Vs of big that need to be addressed:

- Velocity: The speed at which data is generated and processed can make it difficult to keep up, which can lead to missed or inaccurate insights [5] Volume: Storing and processing large amounts of datacan be a significant challenge, as it can require significant amounts of storage and computational resources. This can make the data vulnerable to attacks that attempt to overload systems, such as denial-of-service attacks [6].
- Variety: The wide range of data types and formats can make it difficult to properly process and analyze data, which can lead to errors and inaccuracies in the insights gained from the data.
- Veracity: The data quality and trustworthiness can bea problem for big data, as it can be challenging to confirm the veracity and correctness of the data [7].
- Value: The value of data is determined based on the organizational business goals. The sensitivity and privacy issues arise as any form of attack could compromise confidentiality, integrity and the availability of the big data leading to ID theft, Intellectual property theft, breach of trust and litigation issues.

Vulnerabilities that come with such 5 V's data stored can be vulnerable to breaches and cyberattacks, which can leadto the loss or theft of sensitive information [8]. Deploying big data in cloud computing environments is a rapidly growing technology that has a significant impact on organizations and provides benefits such as demand-driven access to computational services, a distorted version of infinite computing capacity, and assistance with demand-driven scaling up, scaling down, and scaling out. Additionally, deploying servers in the cloud is so convenient and easy that it speeds up the provisioning of big data [9]. To protect big data in cloud computing, researchers have devised many strategies suchas using RSA and MD5, RSA and AES as discussed in section 2. These approaches cover several big data security principles, and all of them aim to create a secure environment.

The paper aims to explore the enhancement of big data security in cloud computing using the RSA algorithm to improve the deployment and processing of the variety, volume, veracity, velocity and the v of the data utilizing RSA encryptions. The novelty contribution of the paper is threefold: First, explore the current challenges and vulnerabilities in securing big data in cloud computing and how can use the RSA algorithm be used to address them. Secondly, we implement the RSA algorithm in a cloud computing environment using the AWS cloud platform to secure big data to improve the performance and scalability of the RSA algorithm for big data security in cloud computing. In addition to RSA itself being used in cloud security, this work investigates integrating RSA encryption technology with cloud services such as AWS. It describes how to set up metrics on CloudWatch to monitor and issue

alerts as needed, launch an EC instance, build an S3 bucket, and create an IAM user. We compare the RSA

algorithm to other cryptographic algorithms in terms of its ability to enhancebig data security in cloud computing. Finally, we recommend control mechanisms to improve security in thecloud. The results show that we RSA algorithm can be used to improve Cloud Security in a network environment.

II. RELATED WORKS

This section discusses the stet of the art and related works regarding big data security and cloud computing. The related work considers encryptions including RSA, and AES among others used in enhancing big data. Regarding the use of RSA in enhancing cloud security, [10] the researchers proposed a hybrid cryptographic model combining AES and RSA as a means to recognize some of the dangers to information confidentiality, identify, and conceal sensitive data and securely transmit big data over the cloud. The RSA public key is used by the sender to encrypt the concealed data to create cypher data, and the receiver's key is then used by the AES technique to encryptthe RSA public key. Once the key has been encrypted andthe cypher data has been transmitted, it is then sent only tothe receiver. The receiver decrypts the RSA keys using their private keys, and they translate the encrypted data back to their original form using the decrypted AES symmetric keys. Further, [11] the authors propose a hybridcryptosystem based on RSA and Paillier for encrypting HDFS files. In this hybrid system, public and private keysare generated by the HDFS client, and the file cache in HDFS encrypts the file using the unstructured data. Data isthen saved in the cloud and subsequently clustered in the HADOOP File System (HDFS) after applying the proposed encryption strategy. Using the hybrid system, the user will utilize their private key to obtain the decrypted data after the server has presented the encrypted data to the decryption process. In [12] presented an improved CP-ABE with an RSA algorithm tosecure outsourced big data stored on cloud servers and to restrict access to, disseminate, and safeguard information. The five stages of the proposed model's activities setup, user key generation, data encryption & decryption, and revocation perform actions that successfully identify the users who decipher the ciphertexts during decryption. Before allowing users to access the secret key and the ciphertext to compute decryption, the cloud environment uses the users' IDs to determine whether or not they are malevolent. The ciphertext will receive updates from boththe public key and the master key during the revocation procedure. New users will be assigned a secret key based on the MK to gain access to the publicly available data andreversibly acquire secrecy. Furthermore, [13] proposed the use of an enhanced RSA algorithm (ERSA) to provide data privacy in cloud environments. To prevent attacks or intrusion, the speed of the encryption and decryption procedures was improved, and the key used for the processes was tightened. The proposed ERSA method uses two alternatives "N" values for encryption and decryption in contrast to the traditional RSA. Similar to theHigh Speed and Secure RSA technique, the generated N1 and N2 values use prime numbers rather than two Random numbers. In [14] the authors proposed a method for handling authentication and data security in Cloud computing that combined digital signatures and RSA. TheRSA algorithm was used to encrypt data uploaded to the Cloud, while the MD5 algorithm was used to verify authenticity. User requests are encrypted by using the system's RSA public key before being sent to cloud providers to ensure secure communication. In [15] the authors implemented a multi-prime RSA algorithm as a middle secure layer of storage to enhance user data securityin the cloud. Using prime numbers selected at random, the multi-prime RSA technique creates a public and private key. The recipient receives the encrypted data once it has been removed from the storage service. In [16] the authors proposed an RSA algorithm to provide data security that could be accessed by the user who requests it to prevent unauthorized access it encrypts user data before it is uploaded to the cloud. The user submits a request to the cloud provider for the requested data, and the cloud provider confirms the user's identity before sending the requested data. The RSA Public Key is shared by everyonein their cloud environment. However, the Private Key is only known by the user who originally held the data.

All the works of literature are relevant and contribute to encryptions in enhancing encryptions in the cloud. However, none of them used RSA in the AWS platform to enhance big data security in the cloud.

III. APPROACH

This section discusses the approach used for the implementation of the proposed RSA algorithm. We used Amazon Web Services (AWS) for the cloud computing platform. The rationale for using AWS for our implementation is that it offers tools such as EC2, S3 bucket and Cloudwatch. The platform provides scalability, security, and cost-effectiveness, making it a suitable choice for this study. The RSA encryption algorithm will be implemented through Elastic Compute Cloud (EC2) instances and Simple Storage Service (S3) providing storage for both encrypted & decrypted datasets. These strategies allow EC2 instances to manage virtual environments designed specifically for executing the RSA encryption algorithm while S3 acts as a storage hub. With various monitoring tools available within AWS resources; we remain confident in optimizing decryption speed throughout implementations irrespective of variations in file sizes. NOAA/PMEL has been selected for this research owing to its reputation for providing credible environmental and oceanographic data. By leveraging their datasets' accuracy, our big data security analysis was fortified with contextual information related to climate variations, oceanic states atmospheric trends among others. This additional information helped us unearth uncommon traits or irregular occurrences which aided immensely towards informed decisions on pertinent security matters. The dataset consists of 100 records with20 variables per record. The choice to use this dataset wasmade based on its applicability to the study's goals and research topic. Also, it was selected because of its size anddiversity, which enable a thorough investigation and evaluation of the suggested RSA algorithm for protecting big data stored in the cloud. To obtain the dataset, the following steps were taken: we access the AWS Open Data Registry website (https://registry.opendata.aws/). We search for the NOAA/PMEL dataset using the search bar on the website. Python was used to carry out these preprocessing processes, and the generated dataset was then utilized to carry out the analysis and assessment of thesuggested RSA algorithm.

A. Selection of AWS Technologies

Several technologies were chosen based on their suitability with the AWS cloud platform and the project's goals to

implement the suggested RSA method for securing big data stored in the cloud. These technologies include:

- Amazon S3: The encrypted data are stored S3 which is an extremely robust and scalable object storage service that enables customers to store and access any volumeof data. It enables access control policies.
- Amazon EC2: The encryption and decryption procedures are carried out via a virtual machine that was built and managed using this technology. The cloud's scalable computing capability offered by EC2 makes it simple to deploy apps and services.
- Amazon CloudWatch: This technology was utilized to monitor the S3 bucket and EC2 instance for any possible difficulties or issues already present. To simplify the operation of the infrastructure, it gathers and displays real-time logs, metrics, and event data in automated dashboards.
- Python: The Python programming language was usedto write the encryption and decryption code. Python is a popular language for data analysis and scientific computing and provides a wide range of libraries and tools for encryption and decryption.
- OpenSSL: The public and private key pair necessary for the RSA algorithm was created using OpenSSL. OpenSSL is a widely-used open-source software library for secure communication.
- Sublime Text: The Python code was written using Sublime Text because of its ease of use, customizable interface, and compatibility with numerous programming languages. Figure 1 illustrates the code used to generate the encryption process, and the validated key pair.

```python
import OpenSSL
from cryptography.hazmat.primitives.asymmetric import rsa, padding
from cryptography.hazmat.primitives import serialization

def verify_key_pair(pub_key_path, priv_key_path):
    # Load the public key
    with open(pub_key_path, 'rb') as pub_key_file:
        pub_key_data = pub_key_file.read()
    pub_key = OpenSSL.crypto.load_publickey(OpenSSL.crypto.FILETYPE_PEM, pub_key_data)

    # Load the private key
    with open(priv_key_path, 'rb') as priv_key_file:
        priv_key_data = priv_key_file.read()
    priv_key = OpenSSL.crypto.load_privatekey(OpenSSL.crypto.FILETYPE_PEM, priv_key_data)

    # Convert OpenSSL keys to cryptography keys
    pub_key_cryptography = serialization.load_pem_public_key(
        OpenSSL.crypto.dump_publickey(
            OpenSSL.crypto.FILETYPE_PEM,
            pub_key
        )
    )

    priv_key_cryptography = serialization.load_pem_private_key(
        OpenSSL.crypto.dump_privatekey(
            OpenSSL.crypto.FILETYPE_PEM,
            priv_key
        ),
        password=None
    )

    # Verify if the key pair match
    if pub_key_cryptography.public_numbers() == priv_key_cryptography.public_key().public_numbers():
        return True
    else:
        return False

# Example usage
pub_key_path = 'public_key.pem'
priv_key_path = 'private_key.pem'
key_pair_match = verify_key_pair(pub_key_path, priv_key_path)
if key_pair_match:
    print("Key pair matches")
else:
    print("Key pair does not match")
```

Fig 1. Python Code used to Verify the Key Pair in Sublime Text

- Kali Linux VM Terminal: Kali Linux VM terminal was employed to connect to the EC2 instance using SSH as it provides a secure shell connection and allows for remote access to the EC2 instance. Figure 2 shows the command used to list the files in the EC2 instance

Fig 2 Linux VM for Connecting to the EC Instance

B. RSA Implementation Process

We used RSA encryption techniques to encrypt big data in a cloud-based environment to ensure security so that access is restricted to authorized users. AWS is one ofthe main providers of large-scale cloud-based data processing, which can effectively store and distribute extremely huge datasets on servers that can run concurrently. It offers a unique computing strategy based on distributed file systems and saves data in clusters. Before storing data in the cloud, users' data will be encrypted. Authentication of the user is performed by the cloud provider when the user requests data from it. As part of thisstudy, we will implement the RSA algorithm, and analyze its performance based on time & space complexity, and throughput parameters. The suggested work will be executed using Python and Kali Linux VM terminal to obtain the results for various evaluation settings. Implementing the RSA algorithm requires the following steps: Key generation, Encryption and Decryption. Each message is represented by a unique integer in the RSA block cypher that uses both public-key and private-key components. While the private key is solely known to the original owner of the data, the public key is universally known in the cloud environment. As a result, the encryption of data is carried out by the cloud service provider while the decryption of the data is performed by the cloud user. After being encrypted with the public key, the data can only be decrypted with the accompanying private key.

C. Key Generation:

The key pair is generated by connecting the user EC instance using its public DNS to an SSH client on the CLI before the encryption of the data. The pseudocode breakdown for creating a public-private key pair with OpenSSL is as follows:

Step 1: Generate a private key using OpenSSL

```
openssl gen-key -algorithm RSA -out
private_key.pem
-aes256
```

"This command generates a private key file named "private_key.pem" using the RSA algorithm with AES 256-bit encryption".

Step 2: Extract the public key from the private key using OpenSSL openssl rsa -in private_key.pem -pubout -outpublic_key.pem

"This command extracts the public key from the previously generated private key file and saves it to a new file named "public_key.pem".

D. Encryption Phase

This is the process of transforming original plain datainto cypher text.

Step 1: Connect to EC instance using SSH client
Step 2: Load the public key from a file
Step 3: Obtain text file to encrypt from the S3 bucket
Step 4: Convert the text data to binary format
Step 5: Break the binary data into chunks of 190 bytes (lessthan the RSA key length)
Step 6: For each chunk: a. Pad the chunk to 192 bytes withrandom bytes
b. Convert the padded chunk to an integer M
c. Compute ciphertext C = M ^ E mod (N1)
d. Store ciphertext in a list of encrypted chunks
Step 7: Upload encrypted data back to the S3 bucket
Step 8: Close the SSH connection

E. Decryption Phase

This is the process of returning the encrypted data to itsoriginal plain form.

A. Setting Up EC2 Instance and S3 Bucket for Big Data Encryption in the Cloud:

This section outlines the exact steps we used to establish an EC2 instance and S3 bucket, set up the security groups, and upload a dataset for encryption to the S3bucket. Implementing the RSA method for big data security in cloud computing requires completing this crucial step.

Launching an EC2 Instance: The first step is to launch an EC2 instance. This involves logging in to the IAM userAWS Management Console that we have created using the root account, we selected EC2 from the list of services, andthen clicking on the Launch Instance button. Next, we select the Amazon Machine Image (AMI) for the instance: Amazon Linux 2023 AMI 2023.0.20230315.0 x86_64 HVM kernel-6.1, and choose the appropriate instance typeas: m5.2xlarge Then, configure the instance details, add storage, and configure security groups. To perform any actions on an EC2 instance, the IAM user was allocated the following permissions in JSON format. The EC instance console shows the instance's running statusduring the tests as follows.

```
{
"Version": "2012-10-17",
"Statement": [
{
"Effect": "Allow","Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:RunInstances",
                "ec2:TerminateInstances"
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect":
        "Allow",
        "Action": [
          "s3:CreateBuck
          et",
          "s3:DeleteBuck
          et",
          "s3:GetObject",
          "s3:PutObject",
          "s3:ListBucket"
        ],
        "Resource": [
          "arn:aws:s3:::bigdatasecuritys3bucket",
          "arn:aws:s3:::bigdatasecuritys3bucket/*"
        ]
      }
    ]
}
          ],
          "Resource": "*"
        }
      ]
    }
```

Configuring Security Groups: A crucial component of protecting an EC2 instance is using security groups that specify inbound and outbound traffic rules. We setup the security groups that specify theinbound and outbound rules (SSH & TCP traffic) and associating the security group with the EC2 instance.

Creating and Uploading Dataset to the S3 bucket: The next step is to create an S3 bucket. This involves logging in to the AWS Management Console, we selected S3 fromthe list of services and then clicked on the Create Bucket button. We choose a unique name for the bucket (bigdatasecuritys3bucket), selected the region for the bucket as eu-west-2, and configure the bucket properties. Once the S3 bucket is created, the next thing we did was upload the NOAA/PMEL dataset to the bucket for encryption. Then configured the permissions for the uploaded files, including the owner and access permissions. To perform any actions on an S3 bucket, the

Step 1: Connect to EC instance using SSH client
Step 2: Load the private key from a file
Step 3: Obtain the text file to be decrypted from the S3 bucket
Step 4: Decrypt the file using the private key
Step 5: Upload the decrypted data back to the S3 bucket
Step 6: Close the SSH connection

Once the file has been encrypted, it will be automatically uploaded to the storage service in the cloud environment. The encrypted file can only be decrypted by authorized users using the corresponding private key. We implement the encryption/decryption algorithm in the next chapter.

IV. IMPLEMENTATION

This section discusses the implementation process used for the paper. Data transmission across a communication channel can be made secure by using the RSA encryption technique. The computationally demanding RSA algorithm necessitates the use of a programming languagewith strong mathematics libraries and functions. Python offers a large range of libraries and packages for data encryption and security, such as PyCrypto and cryptography [17], which provide the capabilitiesneeded to implement the RSA method. The libraries enabled to create of RSA key pairs, encrypting and decrypt data with the RSA technique, and communication with AWS.

IAM user was allocated the following permissions in JSON format. Figure 3 is an illustration of the S3 bucket console showing the NOAA/PERM dataset.

B. Installation of Dependencies for RSA Algorithm

This section outlines the installation process for the necessary dependencies required to implement the RSA installed before we begin to implement the RSA algorithm

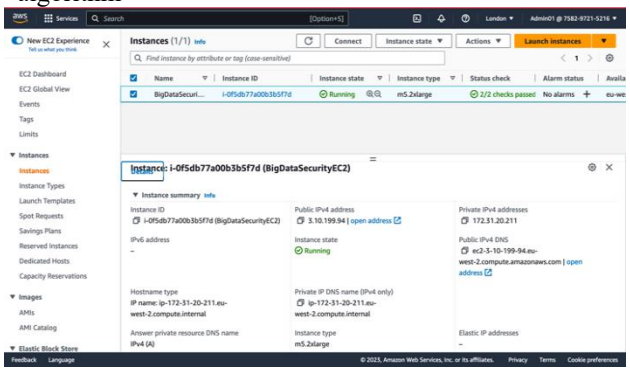

Fig 3. Creating an S3 Bucket for Object Storage on AWS

Steps:

1. Opened a terminal window on the EC2 instance.
2. Updated the package index by running the command: sudo apt update.
3. Installed Python 3 and pip3 by running the command: sudo apt-get install python3 python3-pip. Install Python dependencies on the Kali Linux VM
4. Installed the Python cryptography library by running the command: sudo apt-get install cryptography
5. Installed the required Python packages by running the following command: sudo apt-get install pycrypto boto3 (pycrypto is a Python package that providescryptographic services, including the RSA algorithm. boto3 is a Python package that provides an interface to interact with Amazon Web Services (AWS) services, including S3.
6. Installed the Amazon Web Services Command Line Interface (AWS CLI) by running the command: sudo apt-get install awscli.
7. We configured the AWS CLI by running the command:aws configure. Entered our Access Key ID, Secret Access Key, default region name, and default output format. Figure 4 shows the command for configuring AWS on Kali Linux VM

Fig 4. Configuring AWS CLI on Kali Linux VM

C. Steps to generate RSA keys using OpenSSL

RSA keys are an important component of the RSA encryption algorithm. We have to generate a pair of keys before we can encrypt files in the S3 bucket. In this section,we will explain how we generated the RSA key pair usingOpenSSL and the significance of these keys in the encryption process. OpenSSL is a popular encryption toolkit that is used to generate RSA keys

1. Installed OpenSSL on our Kali Linux VM using the command: sudo apt-get install openssl. Figure 5 shows thecommand used to install OpenSSL on Kali Linux

algorithm. These dependencies are essential for the proper functioning of the algorithm. The installation procedure for the requisite dependencies needed to implement the RSA algorithm was described in this section. It is crucial to make sure that all dependencies have been correctly

2. To generate a private key, we ran the following command: openssl genrsa -out private_key.pem 3072. This command will generate a 3072-bit RSA private key and save it in the **private_key.pem** file. The private key will be used to decrypt any data encrypted by the corresponding public key in the S3 bucket. Figure 6 shows the command used to generate the private key needed for the decryption of the encrypted dataset.

Fig 5. Generating a 3072-bit private key

To generate a public key from the private key, we ran the following command: openssl rsa -in private_key.pem -outform PEM -pubout -out public_key.pem. This command will generate a public key and save it in the *public_key.pem* file. The public key will be used to encrypt the data set in the S3 bucket. Figure 7 is anillustration of the command used to generate the public key needed for encrypting the dataset

Fig 6. Generating the Public key from the private key

The private key has to be protected to prevent unauthorized access. To do this, we set the appropriate file permissions on the private_key.pem file by running the following command. Chmod 400 private_key.pem

This command will set the file permissions so that only the owner of the file can read it. In conclusion, creating RSA keys using OpenSSL is a straightforward process that entails creating a private key, creating a public key from the private key, and safeguarding the private key to thwart unauthorized access. Data will be encrypted using thepublic key and then decrypted using the private key.

D. Source Code for Encryption & Decryption of Data sets stored in the S3 Bucket

The source code for encrypting the data presented is a Python script named bigdatasecurityencryption.py. It is used to access data stored in an S3 bucket, break the data into chunks, pad each chunk, and encrypt the data using the RSA algorithm with a public key. The encrypted data is then uploaded back to the S3 bucket.

The S3 bucket and other crucial information for accessing the data are initially set by the script. After that,it sets up an S3 client using the Python Boto3 package. Using the get object() function, it downloads the requestedobject from the S3 bucket, and the 'Body' key is used to read the binary data from the response. The open() functionis then used by the script to load the public key from the supplied file path and save it in a variable. It divides the binary data into 190-byte chunks and adds arbitrary padding to make each chunk 192 bytes long. The ciphertextis then calculated using the RSA method and the pow() function and is then stored in a list after each padded chunkhas been converted to an integer.

Afterwards, the original data is then encrypted using the RSA technique and the public key that was

previously loaded using the Openssl command line program. The rsautl command is executed with the correct arguments for encryption using the check output() function from the subprocess library. The encrypted information is kept in a variable. Finally, the script uses the put object() method to upload the encrypted data to the S3 bucket and outputs a success message if the upload was successful. Figure 8 shows the Python code used to encrypt the dataset in the AWS S3 bucket and Figure 11 shows the result of the encryption for each data set in the bucket

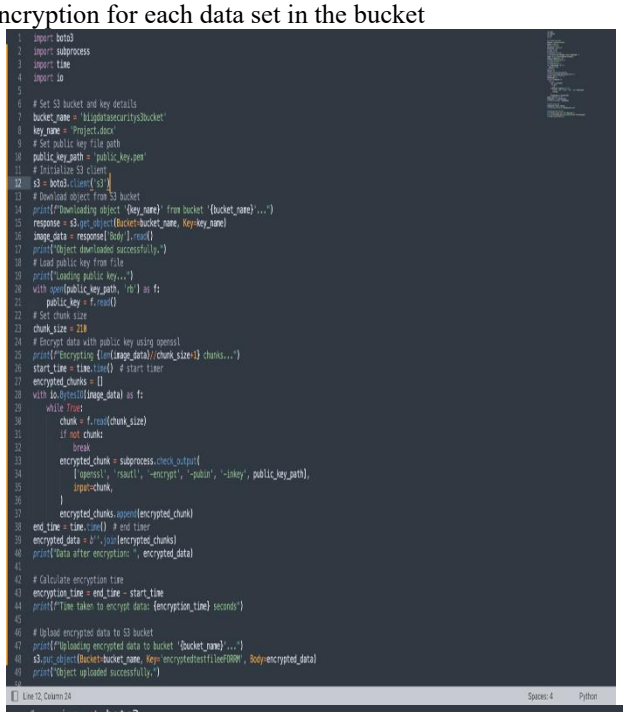

Fig 7. Encryption code in Sublime text

The source code for decrypting the encrypted data set in the S3 bucket is also presented in the Python script. It uses the private key to decrypt the data, which was previously encrypted with the public key. The S3 bucket name, encrypted object key name, and desired decoded object key name are first set, along with the object details. The path to the private key file is also specified.

After initializing the S3 client, the encrypted object is downloaded from the S3 bucket. This is followed by loading the private key from the given file. The encrypted data was decrypted piece by piece before being concatenated into binary data. The RSA decryption algorithm is applied to each encrypted chunk iteratively by the code. The padding bytes are then subtracted before the result is added to the binary data and transformed back to the original binary chunk format. Following converting binary data back to text file content, OpenSSL uses the private keyto decrypt the encrypted data. The decrypted data is storedin the decrypted data variable. The decrypted data is then uploaded to the S3 bucket with the chosen key name. Figure 9 is an illustration of the Python code used to decrypt the encrypted data in the AWS S3 and Figure 11 shows the result of the decryption process for each dataset.

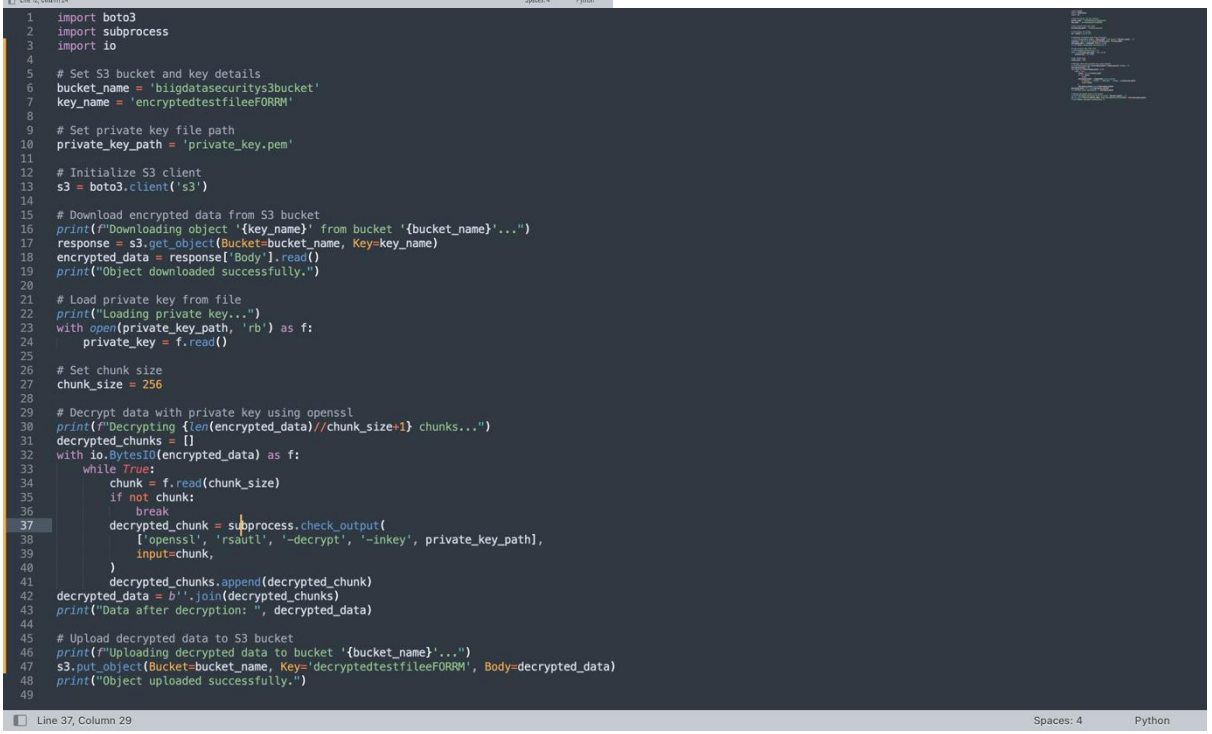

Fig 8. Decryption Code in Sublime Text

E. Configuring CloudWatch

CloudWatch is a monitoring service provided by Amazon Web Services (AWS) that can be used to collect and track metrics, collect and monitor log files, and set alarms. In this project, the EC2 instance and S3 bucket were monitored using CloudWatch to make sure they were operating normally and to get alerts when issues arises. To configure CloudWatch for our EC and S3, we followed the steps below: After the alarms are configured, we will be notified via email when the conditions specified in the alarm are fulfilled. This will make it possible for us to take the necessary steps to guarantee that both our EC2 instance and S3 bucket are functioning properly. Figure 10 depicts statistics and alerts of the metrics configured on AWS CloudWatch.

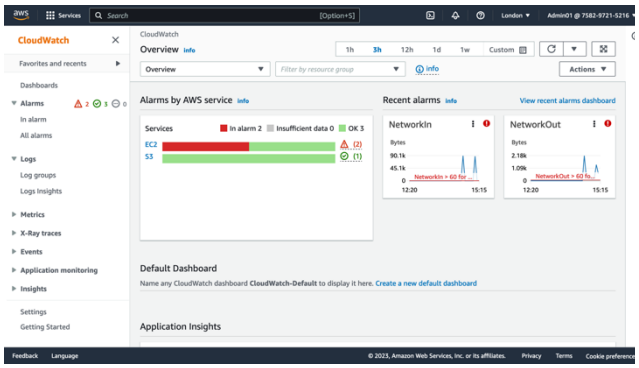

Fig 9. Configuring CloudWatch to Monitor EC2 Instance and S3 Bucket Performance

F. Result Analysis

We conducted several experiments on NOAA/PMEL dataset to assess the efficiency and performance of the developed RSA method for big data security in cloud computing. The dataset consisted of 20 objects with varying sizes ranging from 252KB to 584MB. We contrasted the performance of the RSA algorithm withthat of AES. The test results demonstrated that the RSA method offered a high level of security with little performance cost. For large data files, RSA was shown to perform encryption and decryption more quickly than AES. This is because RSA encryption is a one-way operation, and after encryption is complete, decoding happens quickly.

When compared to AES, the RSA algorithm also offered a high level of security. The enormous key size of RSA made brute force attacks more difficult to defeat, andeven if an attacker managed to obtain the public key, it would take a long time to decrypt the data. Additionally, the RSA method implementation for big data security in cloud computing added another level of security to the data stored in S3 buckets. The only way to decrypt the data is with the appropriate private key, which was used to encryptit using a public key. The risk of data breaches was decreased because of the additional layer of security this gave the data. Figure 11 gives information about the encrypted and decrypted files in the S3 bucket.

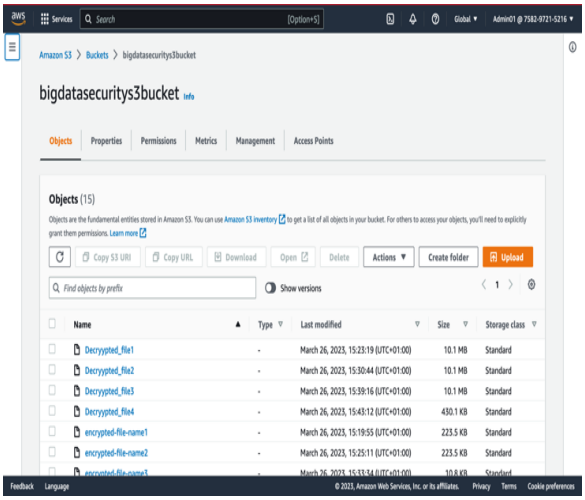

Fig 10. Sample of Encrypted & Decrypted Files in our S3 bucket

The findings in the test demonstrated that using the RSA technique to encrypt sensitive data kept in S3 bucketswas a reliable and secure option. Additionally, the encryption and decryption processes were fast and provided a high level of security against unauthorizedaccess to data.

G. RSA Algorithm and AES Security Comparison

RSA and AES are two popular encryption algorithms that provide security to data by transforming it into unreadable formats that can only be deciphered by authorized users. A comparison of the two algorithms based on their securityand performance characteristics shows that RSA is moresecure than AES, while AES is faster than RSA. However,the security of RSA depends on the length of the key, whilethe security of AES depends on the strength of the key andthe number of rounds used in the encryption process.

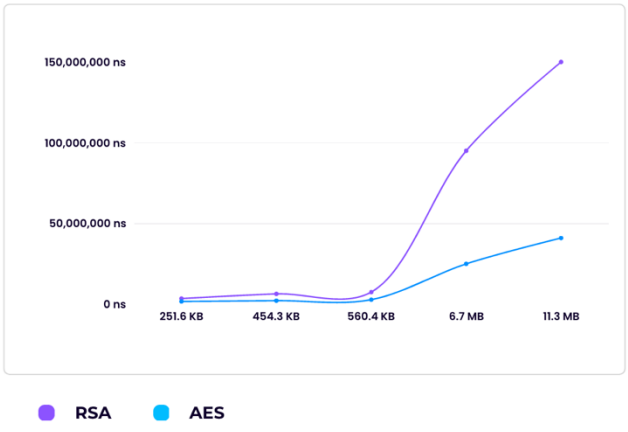

Fig 11. Chart Showing the Encryption Compile Time for RSA & AES (nanoseconds)

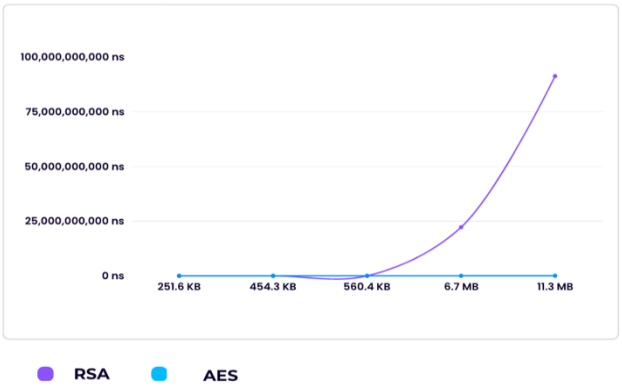

Fig12. Chart showing the decryption compile time for RSA and AES(nanoseconds)

Figure 13 compares the proposed RSA and AES in terms of the time required to compile encryption, whereas Figure 14 compares the time required to compile decryption. Based on this data, AES appears to be a faster and more efficient encryption algorithm than RSA, especially for large file sizes. However, it's important to note that, RSA remains a critical algorithm for securing communication, digital signature, key exchange and data security which is the goal of this work.

V. CONCLUSION

The paper has considered various encryption algorithms that can be used to enhance big data security in the cloud environment. However, our work leverage on the RSA algorithms. The paper has explored the enhancement of big data security challenges in cloud computing using the RSA algorithm to improve the deployment and processing of the variety, volume, veracity, velocity and value of the data utilizing RSA encryptions. We have discussed challenges that come with the 5 V's storage and vulnerabilities that lead to breaches and cyberattacks including the loss or theft of sensitive information. The paper has discussed the security of cloud computing for big data processing, and the variety of encryption techniques including RSA, and AES to prevent vulnerabilities during processing. We implement the RSA algorithm in a cloud using the AWS cloud platform to improve the performance and scalability of the RSA algorithm for big data security by comparing the RSA algorithm to other cryptographic algorithms in terms of their

ability to enhance big data security in the cloud. The paper has recommended control mechanisms to improve security in the cloud using the RSA algorithm to improve the Cloud Security. Although the RSA algorithm offers encryption, a thorough security strategy also incorporates other security measures such as authentication, access control, and data integrity mechanisms. Future research can examine how these measures can be combined to offer a comprehensive and reliable security framework for big data in the cloud.

REFERENCES

[1] Surbiryala, J. and Rong, C. (2019). *Cloud Computing: History and Overview*. [online] IEEE Xplore. doi:10.1109/CloudSummit47114.2019.00007.

[2] Muniswamaiah, M., Agerwala, T. and Tappert, C. (2019). Big Data in Cloud Computing Review and Opportunities. *International Journal of Computer Science and Information Technology*, 11(4), pp.43–57. doi:10.5121/ijcsit.2019.11404.

[3] Stoycheva, Z. (2019) *The contemporary challenge: 4V's of Big Data blog.datumize.com*. [online]. Available from: https://blog.datumize.com/the-contemporary-challenge-4vs-of-big-data.

[4] Ajah, I. and Nweke, H. (2019). Big Data and Business Analytics: Trends, Platforms, Success Factors and Applications. *Big Data and Cognitive Computing*, [online] 3(2), p.32. doi:10.3390/bdcc3020032.

[5] Gandomi, A. and Haider, M. (2015). Beyond the Hype: Big Data Concepts, Methods, and Analytics. *International Journal of Information Management*, 35(2), pp.137–144.

[6] Koseleva, N. and Ropaite, G. (2017). Big Data in Building Energy Efficiency: Understanding of Big Data and Main Challenges. *Procedia Engineering*, 172, pp.544–549. Doi https://doi.org/10.1016/j.proeng.2017.02.064.

[7] Cai, L. and Zhu, Y. (2015). The Challenges of Data Quality and Data Quality Assessment in the Big Data Era. *Data Science Journal*, [online] 14(0), p.2. doi https://doi.org/10.5334/dsj-2015-002.

[8] Alvarez Mendoza, Y., Londoño Gomez, T.J. and Leguizamón Páez, M.A. (2020). Risks and security solutions exist in the Internet of Things (IoT) in relation to Big Data. *INGENIERÍA Y COMPETITIVIDAD*, 23(1), p.e9484. doi:10.25100/iyc.v23i1.9484.

[9] El-Seoud, S.A., El-Sofany, H.F., Abdelfattah, M.A.F. and Mohamed, R. (2017). Big Data and Cloud Computing: Trends and Challenges. *International Journal of Interactive Mobile Technologies (iJIM)*, 11(2), p.34. doi:10.3991/ijim.v11i2.6561.

[10] Basapur, S.B. (2021). A Hybrid Cryptographic Model Using AES and RSA for Sensitive Data Privacy Preserving. *Webology*. [online] Available at: https://www.semanticscholar.org/paper/A-Hybrid-Cryptographic-Model-Using-AES-and-RSA-for-Basapur/583392b2243a7bc0056785ec17e4afe30757d52a [Accessed 18 Dec. 2022].

[11] Abdalwahid, S.M., Yousif, R. and Kareem, S. (2019). ENHANCING APPROACH USING HYBRID PAILLER AND RSA FOR INFORMATION SECURITY IN BIGDATA. *Applied Computer Science*. [online] Available at: https://www.semanticscholar.org/paper/ENHANCING-APPROACH-USING-HYBRID-PAILLER-AND-RSA-FOR-Abdalwahid Yousif/c9c8e8d86083670b19090ceb9a12f7812db41b6e [Accessed 18 Dec. 2022].

[12] Basavarajegowda, R.M. and Sundaram, S.M. (2022). Enhanced CP-ABE with RSA for Secure and Revocable Data Transmission of Big Data in the Cloud. *International Journal of Intelligent Engineering and Systems*, 15(2), pp.47–56. doi:10.22266/ijies2022.0430.05.

[13] Amalarethinam, I.G. and Leena, H.M. (2017). Enhanced RSA Algorithm with Varying Key Sizes for Data Security in Cloud. *2017 World Congress on Computing and Communication Technologies (WCCCT)*. Doi https://doi.org/10.1109/wccct.2016.50.

[14] Lenka, S.R. and Nayak, B. (2014). Enhancing Data Security in Cloud Computing Using RSA Encryption and MD5 Algorithm. *www.scinapse.io*. [online] Available at: https://www.scinapse.io/papers/2186938371

[15] M., Srivenkatesh, M. and Vanitha, K. (2015). Implementing Multiprime RSA Algorithm to Enhance the Data Security in Federated Cloud Computing. *IJARCCE*, 4(4), pp.647–650. doi:10.17148/ijarcce.2015.44149.

[16] Yellamma, P., Narasimham, C. and Sreenivas, V. (2013). *Data security in the cloud using RSA*. [online] IEEE Xplore. doi:10.1109/ICCCNT.2013.6726471.

[17] Acar, Y., Backes, M., Fahl, S., Garfinkel, S., Kim, D., Mazurek, M.L. and Stransky, C. (2018). Comparing the Usability of Cryptographic APIs. *2017 IEEE Symposium on Security and Privacy (SP)*.