CLOUD COMPUTING SECURITY ENHANCEMENT

A PROJECT REPORT

Submitted in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

by

DEBANJAN DEB [Reg. No.:RA1511003010534]
AVIRAL VERMA [Reg. No.:RA1511003010688]

Under the guidance of

Ms. J.V VIDHYA

Assistant Professor, CSE

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM Nagar, Kattankulathur- 603 203

Kancheepuram District

MAY 2019

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this project report titled **"CLOUD COMPUTING SECURITY ENHANCEMENT"** is the bonafide work of **DEBANJAN DEB [Reg. No. RA1511003010534]** and **AVIRAL VERMA [Reg. No. RA1511003010688]** who have carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Signature of the Guide
Ms. J.V VIDHYA
GUIDE
Assistant Professor
Department of CSE
SRM IST

Signature of the HOD
Dr. B. AMUTHA
HEAD OF THE DEPARTMENT
Department of CSE
SRM IST

Internal Examiner

Date:

External Examiner

Date:

Department of Computer Science and Engineering

SRM Institute of Science & Technology

Declaration Form

Degree/ Course	**:**	B.Tech Computer Science & Engineering
Student Name	**:**	Debanjan Deb
Registration Number	**:**	RA1511003010534
Title of Work	**:**	CLOUD COMPUTING SECURITY ENHANCEMENT

I / We hereby certify that this assessment compiles with the University's Rules and Regulations relating to Academic misconduct and plagiarism, as listed in the University Website, Regulations, and the Education Committee guidelines.

I / We confirm that all the work contained in this assessment is my / our own except where indicated, and that I / We have met the following conditions:

- Clearly references / listed all sources as appropriate
- Referenced and put in inverted commas all quoted text (from books, web, etc)
- Given the sources of all pictures, data etc. that are not my own
- Not made any use of the report(s) or essay(s) of any other student(s) either past or present
- Acknowledged in appropriate places any help that I have received from others (e.g. fellow students, technicians, statisticians, external sources)
- Compiled with any other plagiarism criteria specified in the Course handbook / University website

DECLARATION:
I am aware of and understand the University's policy on Academic misconduct and plagiarism and I certify that this assessment is my / our own work, except where indicated by referring, and that I have followed the good academic practices noted above.
If you are working in a group, please write your registration numbers and sign with the date for every student in your group.

ABSTRACT

Cloud Computing is a technology that has seen a spurious growth recently and is being deployed for personal as well as business purposes. Cloud computing is defined as the delivery of computing services – namely servers, storage, databases, networking, software, analytics and intelligence over the Internet to offer faster innovation, flexible resources and economies of scale.

With the feature becoming increasingly integral to various services provided across the inter-connected digital world, it is imperative that its susceptibility be assessed and security made impenetrable to protect the sensitive information Cloud servers store. Cloud security has been vulnerable to threats and in several cases has led to Data Loss, Information Hacking and Denial of Services. These incidents have given rise to widespread concern regarding the data security that these Cloud Services employ. However, security models and security tools are being continually enhanced. This project aims to define the term "Cloud Computing", its functionality and implementation, define the utility and essentiality of Cloud Security and refer to its existence.

Index Terms: Cloud Security, Diffie-Hellman Key Exchange, AES Encryption Standard, Multiple Blocks Cloud Storage.

ACKNOWLEDGEMENT

We express our humble gratitude to **Dr. Sandeep Sancheti**, Vice Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support.

We extend our sincere thanks to **Dr. C. Muthamizhchelvan**, Director, Faculty of Engineering and Technology, SRM Institute of Science and Technology, for his invaluable support.

We wish to thank **Dr. B. Amutha**, Professor & Head, Department of Computer Science and Engineering, SRM Institute of Science and Technology, for her valuable suggestions and encouragement throughout the period of the project work.

We are extremely grateful to our Academic Advisor and Panel Head **Dr. K. Annapurani,** Associate Professor, Department of Computer Science and Engineering, SRM Institute of Science and Technology, for her great support at all the stages of project work.

We register our immeasurable thanks to our Faculty Advisors, **Ms.M.Gayathri** Assistant Professor, Department of Computer Science and Engineering and **Ms. R Anita** Assistant Professor, Department of Computer Science and Engineering, SRM Institute of Science and Technology, for leading and helping us to complete our course.

Our inexpressible respect and thanks to our guide **Ms. J.V. Vidhya** Assistant Professor, Department of Computer Science and Engineering, SRM Institute of Science and Technology, for providing me an opportunity to pursue my project under his mentorship. She provided me the freedom and support to explore the research topics of my interest. Her

passion for solving the real problems and making a difference in the world has always been inspiring.

We sincerely thank staff and students of the Computer Science and Engineering Department, SRM Institute of Science and Technology, for their help during my research. Finally, we would like to thank my parents, our family members and our friends for their unconditional love, constant support and encouragement.

Debanjan Deb

Aviral Verma

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

AES-256	Advanced Encryption Standard 256 bits key
DH	Diffie – Hellman
EU	End User
TPA	Third Party Auditor
CSP	Cloud Service Provider

CHAPTER 1

INTRODUCTION

CLOUD COMPUTING:

Cloud Computing is defined as the delivery of computing services—servers, storage, databases, networking, software, analytics, intelligence and more—over the Internet ("the cloud") to offer faster innovation, flexible resources and economies of scale.

The cloud services are offered majorly in the form of the following three models:

1. Software as a Service (SaaS)
2. Platform as a Service (PaaS)
3. Infrastructure as a Service (IaaS)

Cloud Computing has been in a steep ascent in the recent years due to the flexibility the service provides to both the Service Provider and the User. It provides a shared pool of resources that can be accessed over the internet over multiple devices. Data is stored on a server on the internet instead of storing locally. This enables easier access of data as well as frees up local storage space.

Cloud computing also enables a number of resources to be interconnected and work on a single resource. There are various benefits associated with Cloud Computing. These can be summarized as below,

- Cost : Cloud Computing eliminates the expenses of software and hardware in bulk at datacentres as the service requires limited components. There is no need of

setting up on-site datacentres, instead data is accessible over a shared platform on the internet.

- Speed : Cloud Computing increases the speed of service as the service is provided over the internet on a pool of resources. Thus, the service is not monopolised and dependent on a single system but is collaboration of multiple systems. This helps provide a faster access to information.

- Productivity : Cloud computing minimises on-site operation such as “racking and stacking”—hardware set up, software patching and other time-consuming IT management chores.

- Performance : The greatest benefit of Cloud computing comes in the enhanced performance of the service. As these services are provided over a large network of datacentres, there is regular Updation of software and gives reliable performance.

CLOUD SECURITY:

Cloud Security refers to the mechanisms, technologies and policies that are deployed to protect data, communication and infrastructure of the Cloud computing service. Cloud Security is an integral part of the Cloud computing service. It is a sub-category of Information Security and Network Security.

1.1 Background

As per the current system, cloud security is still an area of concern. With cloud computing, organizations can use services and data is stored at any physical location outside their own control. This facility raised the various security questions like privacy, confidentiality, integrity and demanded a trusted computing environment wherein data confidentiality can be maintained. To be able to sufficiently trust in the service of computing, there is an

imminent need of a system that performs authentication, verification and encrypted data transfer, thereby preserving data confidentiality. Here in our project we are trying to point out some measures, which will make it more safe and secure.

1.2 Innovation Idea

The idea that motivates this project is the collaboration of cryptography with cloud computing to ensure an efficient security mechanism that successfully protects data storage as well as authenticates data access.

At its core, Cloud Services are nothing but widely inter-connected network of configurable system resources.

Cryptography is equipped with various protocols, in our focus namely the Diffie-Hellman Key Exchange and Advanced Encryption Standard (AES-256) that are utilised for secure data transfer over a network.

The idea is to implement this feature of cryptography on cloud servers so as to resolve the concerns over Cloud Security.

Moreover, the purpose is to build an implementation that is capable of partitioning the various data for storage over distributed server systems. The partitioning of data along with encryption and access protocol will provide an efficient encryption scheme for cloud computing enhancing its security.

1.1 EVOLUTION

The work done until date can be distributed into three periods:

i. Idea period

ii. Pre cloud period

iii. Current Cloud period

During1960s, John McCarthy, Douglas Parkhill, and others reconnoitred the idea of computing in the form of a public service. The idea was to use commodity hardware and software, computing resources to deliver an infinite elastic online public utility.

In the late 1960s, J.C.R. Licklider inspired the ARPANET, and in the early 1970s global networking became a reality. In the early 1980s, the TCP/IP suite emerged as the protocol for ARPANET, and the Domain Name System (DNS) established naming designations for websites.

The now widely recognised cloud phase began the year 2007 when the cataloguing of IaaS, PaaS, and SaaS were officially finalized. The cloud-computing chapter of history has seen some peculiar and intriguing breakthroughs initiated by the leading web officialdoms of the digital world.

CHAPTER 2

LITERATURE SURVEY

2.1 SECURITY PROVENENCE: THE ESSENTIAL BREAD AND BUTTER OF DATAFORENSICS IN CLOUD COMPUTING

Rongxing and his team in their publication presented in the ASIACCS suggested a new security system wherein they applied the algorithm called Bilinear Pairing Method. They recommended of a novel mechanism for data and security attribution in cloud security systems. In their proposition, secret standard files shall be included on the tree of the users file system, and the tool will be capable of supporting data security. They provided an authentication mechanism to verify user access so that unauthorized user access can be filtered away. The attribution vision, a work of the 'bilinear pairing method' that has data forensics blocks inbuilt milieu. Data is then paired with these blocks for security and authentication. The idea is to verify access by resolving disputes of data forensics. They tested their model using multiple security techniques and were able to prove functionality. However, they could not implement the said system as there were occurrences of complexities on the mathematical models used.

2.2 CLOUD COMPUTING: SECURITY AND RISK CLASSIFICATION

In their research paper published in the ACMSE, R. La'Quata Sumter stated, the evolution of cloud service execution amounts to suspected net security with constant increase of threats. Cloud services and vendors' clients are earnestly dispirited by the feebleness of cloud safety to secure sensitive information. Users distrust the security mechanisms that are implemented by the service providers. In order to reassure clients, they provided a prototype that trails every move and process that is being performed on the data stored on the servers.

In order to achieve this, these people necessitated a security capture device which shall provide the said functionality and make their model work. The advantages of their system are that they have been dealing with customer reinforcement regarding the security worries. However this prototype is limited lone to minor Cloud Setups as it loses its practicality on large multi-server cloud systems.

2.3 CLOUD COMPUTING ISSUES, RESEARCH AND IMPLEMENTATION

Mladen A. Vouch, in their publication said that cloud computing evolved in form of a scheme whence several practical ages over computer networking equipment. The paper primarily focuses on trepidations in respect to ‘cloud computing with virtualization, cyber infrastructure, service oriented architecture and end users’. They undertook and stated key concerns and execution study that upheld their study essential. But discontent on the user’s side led them to compose hypothetical papers dependent on security ideas and issues.

2.4 TOWARDS A DATA CENTRIC VIEW OF CLOUD SECURITY

Wenchao et al. in their exploration offered an alternate point of view of arrangements through information driven data the executives. They completely examined the security necessities of verifying information and sharing of information through online applications. They included dialogs on criminology, framework investigation and information the board. They proposed of another security stage known as Declarative Secure Distributive Services condensed as DS2, a stage that bolsters the elements of the offered information verifying strategies. 'Secure Network Data log (SeNDlog)' a programming language that forms systems administration and access control rationale based errands deals with the system conventions and security strategies. Helped by Rapid Net definitive systems administration motor, they had the capacity to create DS2 model and included provenance support, as per their conviction that it will make the security level progressively steady. The strong point

of their work lies in the information driven security that outcomes in secure question handling, framework examination and legal sciences, proficient start to finish check of information. The work done by them anticipates assessment from expert cloud sellers.

2.5 SECURITY AUDITS OF MULTI-TIER INFRASTRUCTURES IN PUBLIC INFRASTRUCTURE CLOUD

The benefits associated with cloud computing services such as protection, safety and secrecy, compelled them to explain the popularity of the effect of cloud services. To provide invulnerability to Cloud systems against threats, complex and good management of web interfaces is essential. Their implementation was over the platform, “Amazon’s Elastic Compute Cloud (EC2)” and connected a safety examination device & reenacted it at genuine variables. They proposed complex abnormal state inquiry language and utilized it to portray the necessities of the arrangement. Python and EC2 were the unmistakable programming used. The system recognizes any ruptures on the safe segments of the foundation and afterward continues to educate the chairmen to break down the issue. In this manner it very well may be said that the product works like an antivirus program. While they researched each conceivable security assault with the proposed apparatus, the burden of the device is that the product is connected to work just with the Amazon EC2 and no other general frameworks.

2.6 TRANSPARENT SECURITY FOR CLOUD

Flavi and Roberto suggested an Architecture and Transparent Cloud Protection System (TCPS) to ensure greater security. They claimed to have accomplished integrity in relation to cloud privacy issues. To recognize these, they built feasibly secure architecture called TCPS. The system had potential to be used in order track every host transfer but as well as maintain the virtualization and transparency of the server. Their resultant work is that they managed to create an intrusion detection mechanism built in the architecture but failed to tackle realistic scenarios thereby lacking validation.

2.7 CLOUD HOOKS: SECURITY AND PRIVACY ISSUES IN CLOUD COMPUTING

Wayne stated the essentiality of configuring security on critical systems. Facing security issues from end user perspective is mandatory. Security policies with strong commands should keep data checked for dangerous actions and prevent unauthorized access to both clouds and data servers. Their paper focuses on public clouds. Key factors are end user trust, insider access, visibility, risk management, client-side protection, server-side protection, and access control and identity management. The weakness in their work is that they did not outcome of a tool, or a solution on real infrastructure.

2.8 MANAGING SECURITY OF VIRTUAL MACHINE IMAGES IN A CLOUD ENVIRONMENT

Jing Peng et al proposed a paper on cloud's image database. Their design addresses the risks and can be easily implemented and prove success. Filters in the system infrastructure capture malware and secondly all sensitive data to crack passwords are removed and replaced by stronger ones. Clients can choose the required images. Repository maintenance decreases the possibility of running illegal software. The testing's of this papers show that filters work efficiently in the image management system. They proposed a system "different" from other cloud architectures and showed with aid of filters and scanners that they could detect malicious traffic. The weakness is that captures of filters are not 100% accurate and could lead to legitimate issues as also the scanner cannot capture every type of virus and it has to be updated constantly.

2.9 A CLIENT-BASED PRIVACY MANAGER FOR CLOUD COMPUTING

Miranda and Siani are confronting issues of information drainage client whine about. This issue puts a genuine deterrent on the acknowledgment of the execution of cloud and its development available. A few scenarios have been contemplated about. A customer based security director device for handling delicate data embedded in the cloud is proposed. The apparatus lessens security issues as at the same time builds protection wellbeing. The device has been tried effectively and utilized in numerous conditions.

2.10 DATA PROTECTION MODELS FOR SERVICE PROVISIONING IN THE CLOUD

Dan and Anna proposed an information security structure for delicate data. Their proposed system contains three essential keys: arrangement positioning, coordination and implementation. Different models have been portrayed for each part. They displayed security information models yet in addition cost capacities. Their work is tried and re-enacted yet not approved on genuine conditions.

LITERATURE SURVEY TABLE

Author/Year	Title	Main Focus	Algorithm	Drawbacks
Rongxing et al **Publisher :** ASIACCS **Year :** 2010	Security Provenance: The Essential Bread & Butter of Data Forensics in Cloud Computing	New Security and Provenance Data Forensics Tool	Bilinear Pairing Method	Complex Mathematical Models
R. La'Quata Sumter **Publisher :** ACMSE **Year :** 2009	Cloud Computing: Security & Risk Classification	Store Information of every Process on Cloud Servers	Security Capture Device	Practical only for Small Cloud Environments
Mladen A. Vouch **Publisher :** IEEE **Year :** 2008	Cloud Computing Issues, Research & Implementation	Cloud Computing with Virtualization, Cyber Infrastructure, SOA and End Users	Issue Authentication	User Dissatisfaction
Wenchao **Publisher :** IEEE **Year :** 2010	Towards a Data Centric View of Cloud Security	Data Centric Perspective, Forensic, System Analysis and Data Management	Declarative Secure distributed Systems (DS2), Secure Network Data Log (SeNDlog)	Complex Implementation. Awaits Evaluation from Cloud Vendors
Soren Bleikertz **Publisher :** CCSW **Year :**	Security Audits of Multi-Tier Infrastructures in Public	Implementation of Cloud Security Analysis and	Complex Query Language over Amazon's	Software connected to work only with EC2 and not generic.

2010	Infrastructure Clouds	simulation to real factors	Elastic Compute Cloud (EC2) and Python	
Flavio Lombard, Roberto Di Pietro **Publisher :** IEEE **Year :**2010	Transparent Security for Cloud	Transparent Cloud Protection System (TCPS) for better Security Management	Secure Architecture system of TCPS	Failed to deploy realistic scenarios and validate their work
Wayne A. Jansen **Publisher :** IEEE **Year :** 2011	Cloud Hooks: Security and Privacy Issues in Cloud Computing	The essentiality of configuring security on critical systems	Revamped Security Policies with Strong Commands	Unknown outcome of tool or a solution on real infrastructure
Jinpeng et al **Publisher :** CCSW **Year :** 2009	Managing Security of Virtual Machine Images in a Cloud Environment	Cloud's Image Repository.	Use a Filter to capture malware and replace all sensitive passwords with stronger ones.	Captures weren't 100 % accurate and could lead to legitimate issues. Required regular Updation
Miranda & Siani **Publisher :** IEEE **Year :** 2009	A Client-Based Privacy Manager for Cloud Computing	Manager tool for information processing in the cloud	Client-side Key Authentication Mechanism	Lack of implementation in all scenarios
Dan Lin, Anna Squicciarini **Publisher :** SACMAT **Year :** 2010	Data Protection Models for Service Provisioning in the Cloud	Data Protection Framework for sensitive information	Policy Ranking, Integration and Enforcement	Not validated on real environments.

Table 2.1 Literature Survey

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Wang J and et al suggested anonymity based method for confidentiality in cloud data. Their algorithm anonymizes the data before storage. Each time a user requests data it uses domain data to analyze anonymous data to obtain that information. Its benefits are that it is simplistic and supple and differs to the traditional methods based on cryptography. It is partial to few number of services.

Gentry displayed encryption strategy for protecting the security of information. This technique for encryption empowers the calculation on the encoded information which is put away in cloud. Cloud supplier doesn't know about information preparing capacity just as aftereffect of calculation. It is an incredible asset for security support yet it neglects to utilize for all intents and purposes.

The Singh and et al. utilized the RC5 encryption calculation to verify capacity information. A framework utilizes the encryption and decoding keys of client's information and stores it on remote server. Every capacity server has an encoded document stockpiling framework which scrambles the information and store. It is anything but difficult to utilize and verify. Guarantees the information is put away on confided in server yet there are difficulties of breaking the calculation.

It is a type of open key encryption wherein the mystery key of a client and the figure content are reliant upon properties (for example the nation of living arrangement, or membership type). In such a framework, the decoding of a figure content is conceivable just if the arrangement of characteristics of the client key matches the qualities of the figure content.

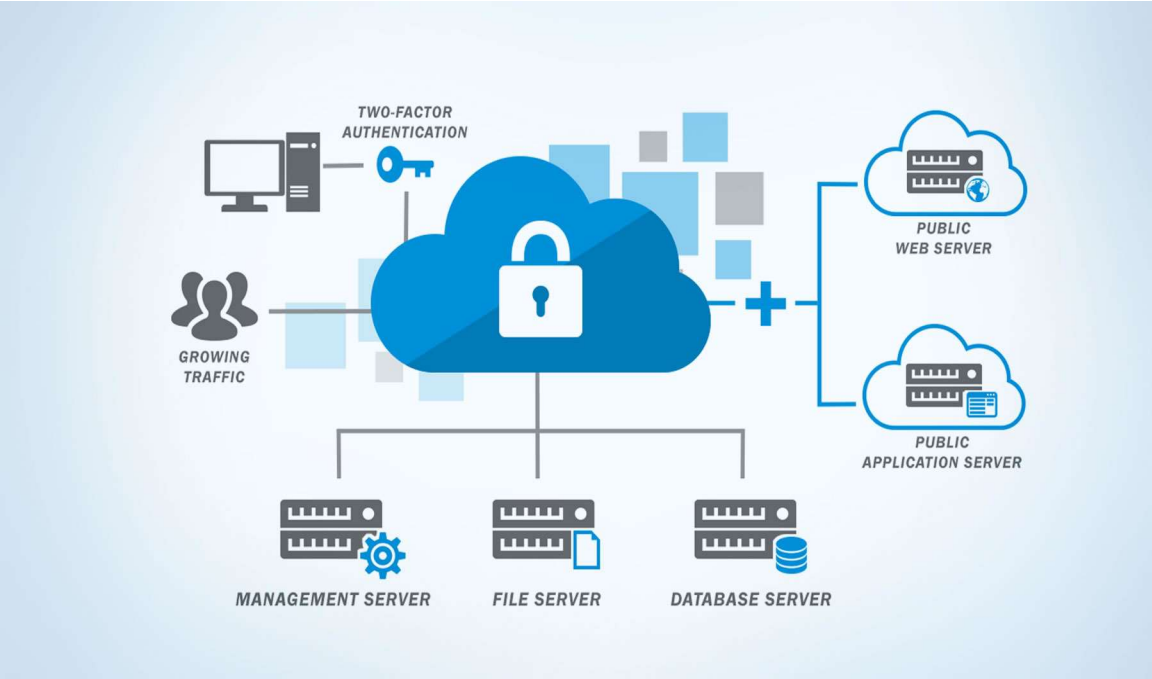

Figure 3.1 Existing Cloud Security System

3.2 PROPOSED SYSTEM

We intend to convey the "Three way system" since it guarantees all the three insurance plan of validation, information security and confirmation, in the meantime. In this paper, we have proposed to utilize computerized mark and Diffie Hellman key trade mixed with (AES-256) Advanced Encryption Standard encryption calculation to secure secrecy of information put away in cloud. Regardless of whether the key in transmission is hacked, the office of Diffie Hellman key trade renders it futile, since key in travel is of no utilization without client's private key, which is restricted just to the genuine client. This proposed engineering of three way instrument makes it intense for programmers to split the security framework, accordingly ensuring information put away in cloud.

The proposed framework is an improvement that will give numerous levels of assurance.

- In using the Diffie Hellman key-pair, the Attribute-Key System is not essentially required to be eliminated.

- The Attribute-Key framework can be utilized for Authentication at Server-Side while the proposed framework can be utilized to give get to control at the Client-Side.
- The thought of putting away information at different cloud servers does somewhat entangle systems at Server-Side.
- However, this slight hindrance is in no way, shape or form practically identical to the advantages the proposed framework gives which ensures information regardless of whether a solitary server security fizzles.

3.2.1 ALGORITHMS USED

The AES Algorithm is utilised in the Data Encryption Module for encrypting all the part files that are stored on the provided directory.

AES Algorithm

1. KeyExpansion—round keys are derived from the cipher key using Rijndael's key schedule. AES requires a separate 128-bit round key block for each round plus one more.
2. Initial round key addition:
    1. AddRoundKey—each byte of the state is combined with a block of the round key using bitwise xor.
3. 9, 11 or 13 rounds:
    1. SubBytes—a non-linear substitution step where each byte is replaced with another according to a lookup table.
    2. ShiftRows—a transposition step where the last three rows of the state are shifted cyclically a certain number of steps.
    3. MixColumns—a linear mixing operation which operates on the columns of the state, combining the four bytes in each column.
    4. AddRoundKey
4. Final round (making 10, 12 or 14 rounds in total):
    1. SubBytes
    2. ShiftRows
    3. AddRoundKey

Figure 3.2.1 AES Algorithm

The Diffie-Hellman Key Exchange Algorithm is used to authenticate the data transfer process between two parties. It is used to transfer the secret message i.e. the encryption key used in the Encryption Module.

Diffie-Hellman Algorithm

Step 1: Alice and Bob get public numbers P = 23, G = 9

Step 2: Alice selected a private key a = 4 and

Bob selected a private key b = 3

Step 3: Alice and Bob compute public values

Alice: x =(9^4 mod 23) = (6561 mod 23) = 6

Bob: y = (9^3 mod 23) = (729 mod 23) = 16

Step 4: Alice and Bob exchange public numbers

Step 5: Alice receives public key y =16 and

Bob receives public key x = 6

Step 6: Alice and Bob compute symmetric keys

Alice: ka = y^a mod p = 65536 mod 23 = 9

Bob: kb = x^b mod p = 216 mod 23 = 9

Step 7: 9 is the shared secret.

Figure 3.2.2 Diffie-Hellman Key Exchange

3.3 SYSTEM ARCHITECTURE

The process involved in proposed system is comprised of the following modules,

i. Partitioning

ii. Encryption

iii. Key-Exchange

iv. Decryption

v. Merging

Each section varies greatly in the methodologies.

The system is comprised of three modules namely,

- The Data Encryption Module
- Partitioning Module
- Diffie-Hellman Key exchange Module

The components involved in the system architecture are as follows,

- Customer
- Partitioning and Encryption System
- Cloud Server
- User Data
- Decryption and Merging System
- End User

The data stored on the local machine of the Customer is transferred into the Partitioning and Encryption System that splits the data into multiple parts and then encrypted. The partitioned and encrypted data is then uploaded on the Cloud Server. The User Data stored on the server is hence encrypted by the customer with access restricted to it. Since the data is partitioned and encrypted by the customer he/she possesses control over the data.

The End User downloads the data onto their local machine. This data is required to be decrypted and then merged. This process is carried out by the Decryption and Merging system. The data is passed onto the module which accesses it using the key passed on by the customer over the server. The decryption process is initiated and is followed by merging the decrypted files into the original files. Thus, the complete data is reinstated at the end user machine.

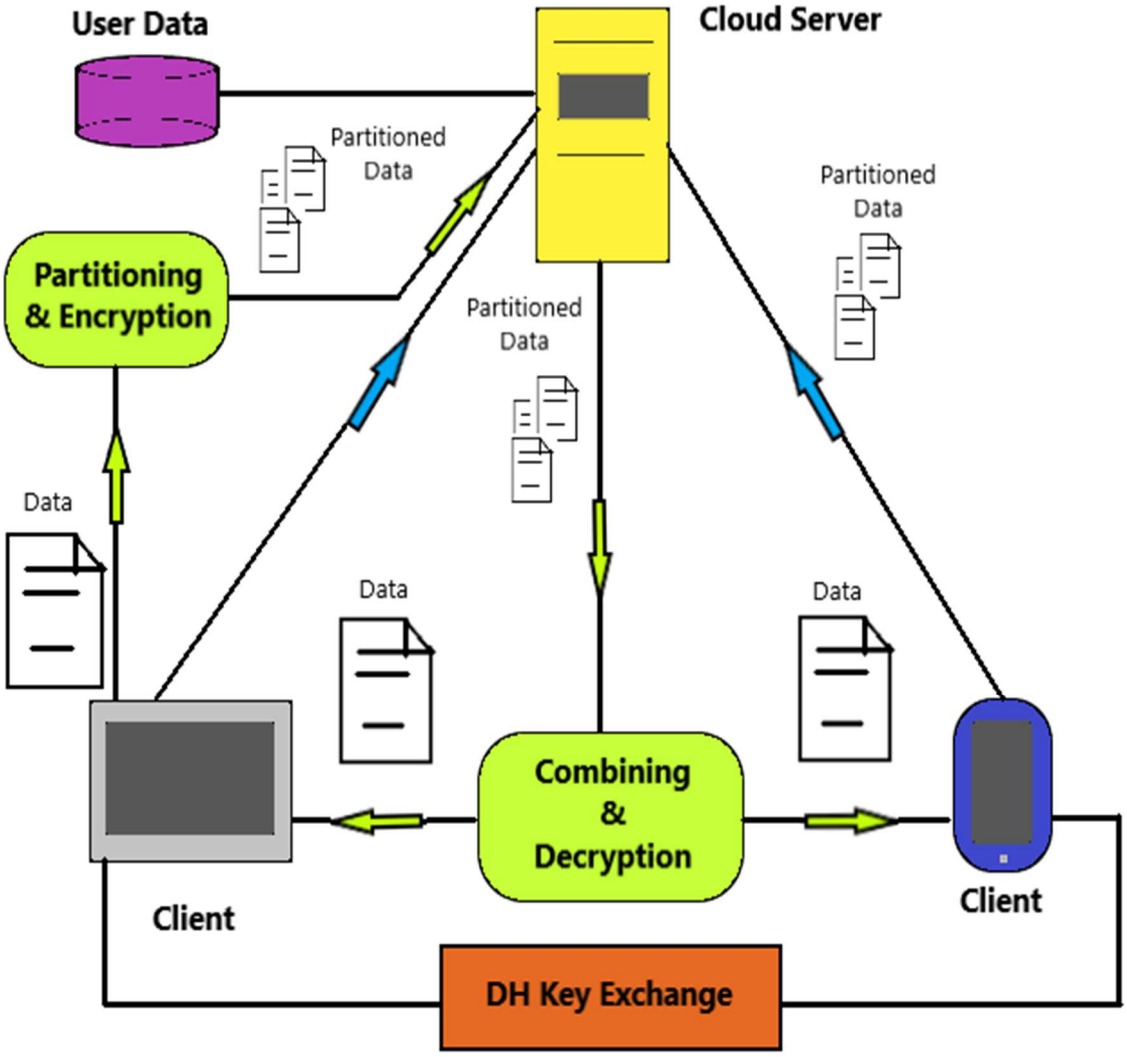

Figure 3.3.1 Architecture Diagram

The process flow is depicted as follows. The user of the system is prompted to select the directory and decide the chunk size into which the parts of the file will be split.

The files are then split. Next up, the encryption module demands the password using which a 32 bit AES key is generated with an IV. The process encrypts files, renames files and creates an index with complete information.

The data is then uploaded onto the server.

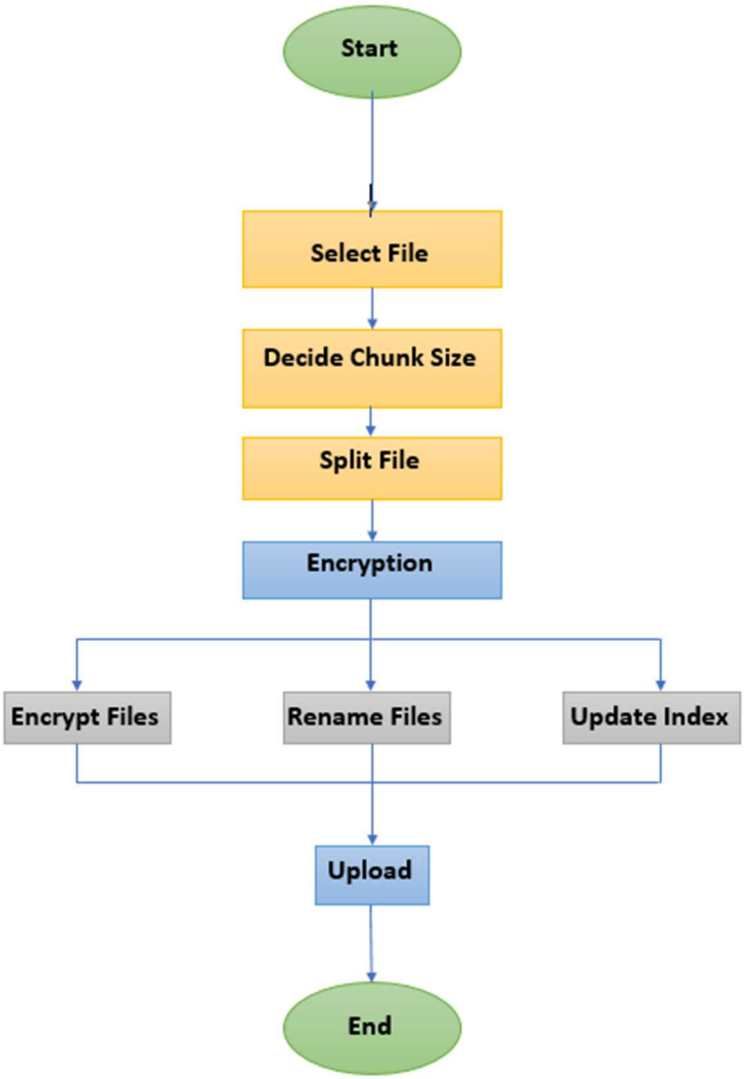

Figure 3.3.2 Process Flow Diagram

Figure 3.3.3 ER Diagram

3.4 BENEFITS OF THE PROPOSED SYSTEM

- The user energises control over data access, by implementation of the Diffie-Hellman Key Pair.
- Only the user is possession of Private Key from the data owner can regain data on the cloud server.
- The keys created by Diffie-Hellman Protocol are not dependent over attributes of users as in current system.
- Thus, any hacker with significant information shall still be a failure in figuring the key out.
- The key-pair generated by the DH Algorithm are mathematical sets that can be cracked only through enormous and virtually impossible unrealistic approach.
- At the Cloud Server, the Data is encrypted using AES-256 Encryption. Thus, the data is protected at the cloud server as well.
- Even if a chunk of data is decrypted, it gives incomplete information and renders any threat futile.
- Both the algorithms utilised are complex calculations extremely difficult to crack using brute force algorithms and hence very secure.
- The AES-256 is the most advance encryption standard that is still unbroken.
- In theory, it would entail around 3×10^{51} years to deplete the key space utilized in AES-256.
- Defining what two random primes go together to create the DH number is a computationally infeasible problem. This computational complexity secures the issue from cryptanalysis.

3.5 SYSTEM REQUIREMENTS

The requirements of the system are broken down into the following.

3.5.1 SOFTWARE REQUIREMENTS

The Software Requirements of the system are as follows

1. Python 2.7
2. PyCrypto Module for Python
3. Tonido Application for Cloud Server Setup
4. Optional Cloud Services (Google Drive, OneDrive etc.)
5. Web Browser

3.5.2 HARDWARE REQUIREMENTS

The Hardware Requirements are listed below,

1. Windows/Linux PC or Laptop
2. RAM 2 GB or More
3. Memory 10 MB

CHAPTER 4

IMPLEMENTATION AND DESIGN

4.1 DATA ENCRYPTION MODULE

The Data Encryption Module is responsible for encryption and decryption of data at the client side. It is a simple module that serves the purpose of encrypting each file individually. It is designed with cloud storage in mind.

The Data Encryption Module uses the AES-256 (32 Bit Key) for encrypting the data. Using the EncryptFS module, the client can specify a chosen directory that contains the various files required to be encrypted before upload on the cloud storage. The said directory can also be the primary cloud sync folder.

4.1.1 SERVER SIDE ENCRYPTION

The module has been built using Python 2.7 and the PyCrypto module. At execution it asks for the complete address of the directory containing sensitive files and demands a 16/24/32-bit key. The key specified by the user is then used to create and encrypted index file and various files with randomly generated names. The data in the files along with its metadata is safely encrypted. Later, the existing EncryptFS can be used to decrypt the files in the index.

This module also provides with renaming the various files in a directory with a random 1-bit hex number thereby hiding the original file names and metadata such as file type, date etc.

Indexing is used to communicate the original filenames to the receiver's end for proper decryption. The module creates a dictionary with the original file names and new file names as a key-value pair. This index is stored as a Java Synchronised Object file and further encrypted to be uploaded along with the other files.

a) The Data Encryption Module holds the responsibility of encryption and decryption of a data at the client side. It is a simple module that serves the purpose of encrypting each file individually. It is designed with cloud storage in mind.

b) The Data Encryption Module uses the AES-256 (32 Bit Key) for encrypting the data. Using the module, the client can specify a chosen directory that contains the various files required to be encrypted before upload on the cloud storage. The said directory can also be the primary cloud sync folder.

c) The module has been built using Python 2.7 and the PyCrypto module. At execution, it asks for the complete address of the directory containing sensitive files and demands a 32-bit key.

d) The key specified by the user is then used to create and encrypted index file and various files with randomly generated names.

e) The data in the files along with its metadata is safely encrypted.

4.1.2 CLIENT SIDE DECRYPTION

At the site where the encrypted files are downloaded from the cloud server, the Data Encryption module is responsible for decrypting the files. It asks the client for the 32-bit key using which the data was encrypted. Once the key is provided the module proceeds to decrypt all the files in the directory.

a) At the receiver's end, the encrypted files when downloaded are decrypted using the AES key and the index.

b) The index provides the information pertaining to various source file names and their corresponding randomly generated names [7]. It also gives information regarding original file formats.

c) The files are then decrypted in similar manner as the AES Decryption process.

d) The result is a series of decrypted files that require merging to access the original files.

4.2 DIFFIE-HELLMAN KEY EXCHANGE MODULE

The Diffie-Hellman Key Exchange module is responsible for generating a public-private key pair using the Diffie-Hellman algorithm for authentication of access. The key pair thus generated is also utilised as the key in the data partitioning module.

4.2.1 KEY-EXCHANGE

a) The Diffie-Hellman Key Exchange module is responsible for generating a public-private key pair using the Diffie-Hellman algorithm for authentication of access.

b) The key pair thus generated is used to exchange the 256 bit AES key for decryption.

4.3 DATA PARTITIONING MODULE

The Data Partitioning Module serves the purpose of partitioning the various files to be stored on the cloud into unrecognisable chunks that can be then regrouped at the receiver's end only through access by the key generated by the Diffie-Hellman process. In the existing system, the data is stored on the cloud utilising dynamic operations on data along with computations that requires the user to create a copy for further updation and verification of the loss in data. An efficient distributed storage auditing mechanism is proposed which has the ability to overcome the limitations of data loss handling.

The files once decrypted are matched using regular expressions to facilitate the organising of the various part files matched as objects into a single file that replicates the original.

4.3.1 SERVER SIDE PARTITIONING

a) The first phase of the system proposes partitioning of the data that is to be stored over a cloud storage server into several chunks.

b) The said functionality is achieved by splitting files of all formats such as *.txt, *.jpg, *.mp4 etc. into smaller fixed size chunks using the partitioning module.

c) The system reads the files selected by the user and requests the chunk size i.e. the maximum size of each of the chunks the file will be divided into.

d) The optimal chunk size is dependent on the cloud service the customer is utilising. For example, Amazon S3 requires chunk size of 5 MB minimum whereas Azure demands they must be much less than 4 MB.

e) The program then reads the source file up to the specified chunk size and writes on to an output part file iteratively until the end of file is reached.

f) Thus, the client is provided with numerous divisions of a single file that can be further encrypted and uploaded on to a cloud storage.

4.3.2 CLIENT SIDE MERGING

a) The system merges the various part files by accessing the manifest to gain information regarding the original file names.

b) When the user provides the original source file name along with the command to merge it's part files, the system obtains all the part files downloaded.

c) The system uses regular expression to match all the part files in the directory with the source file name.

d) For each chunk matched, the system reads the data stored in them and writes them onto a new file that replicates the source fil, thereby merging all the chunks. [12]

e) The part files are then deleted from the directory.

CHAPTER 5

RESULTS

5.1 SERVER SIDE

In several test runs across multiple devices, we proceeded to select a directory containing various files of different sizes and types such as images, videos, documents, PDFs, audios etc.

The directory as stated by the user in the program is then used as the home directory of the File Splitting module. For each file present in the directory, the program demands of the user the optimal chunk size that will become the maximum size of each chunk file the original file is split into.

In our test run depicted in the Appendix 2 of this report, we selected three files namely, *696.pdf* (PDF Document), *Batman.mkv* (Video) and *Sunrise.mp3* (Audio). With an optimal chunk size provided by the user of 500 KB, the said files were split into 4, 3 and 2 parts respectively.

Each chunk file limited at a maximum of 500 KB while few with the residual size observed as the last part files. These part files are named in the format *file_name-(part number).file_type.*

Thus once all the files in the directory are partitioned, the program moves onto the second module i.e. the Directory Encryption Module.

The user is then prompted by the program to provide with a key of amenable size.

In this next stage, the directory as provided by the user is encrypted using the AES-256 algorithm with a 32 bit key generated by the password provided by the user and an Initialisation Vector produced using SHA algorithm.

The key value is then used to encrypt all the files in the directory also renaming each file by a random 16 bit Hexagonal number. This function adds to the security feature while maintaining the metadata associated with each file. Eg, *26fci97hjres45asutahijgiod887.enc*

The program also creates an index of all the files containing a dictionary with a key-value pair that imitates the *original filename – new filename* storage. This index is stored in the directory into an *index.json* file wich is also further encrypted.

The index file is deemed necessary for decoding original file names at the receiving end.

The user is finally prompted to communicate the password through DH Key Exchange Protocol as stated previously.

The whole directory is then ready to be uploaded onto any cloud server.

5.2 CLIENT SIDE

The client then downloads the said files onto their local machine from the cloud server. The process of decryption starts with the Directory Decryption Module.

In this module, the user now is asked of their keys regarding the DH Key Exchange Process to retrieve the password required to decrypt all the files. With the correct password in place the user enters this password prompted by the program to initiate the decryption process.

Firstly, the index file is read to rename all the files in the directory from random number to their original post encryption file names. The file is read onto the program and the dictionary is retrieved to splat out the required information.

The Decryption process then begins using the same 32 bit key to decrypt all the files at each thread into original data. As the process continues each file is restored to its original form i.e. their original names and types.

Thus once the decryption is complete we are left with the various part files such as *696.pdf-1, Batman.mkv-3* etc. The index file is the only exception of the process. It is left as is.

The part files are then merged in the File Merging Module wherein the user is prompted to provide the filename they wish to be merged. Using regular expressions, all the part files of the file demanded are matched and are rewritten as a single file.

As seen, *Batman.mkv-1, Batman.mkv-2* and *Batman.mkv-3* are merged into singular file *Batman.mkv.*

Similarly all the files are merged in the directory and the original data is restored.

5.3 PERFORMANCE ANALYSIS

Implementing Threading into the Encryption Process enhances the performance of the program as depicted by the graph below.

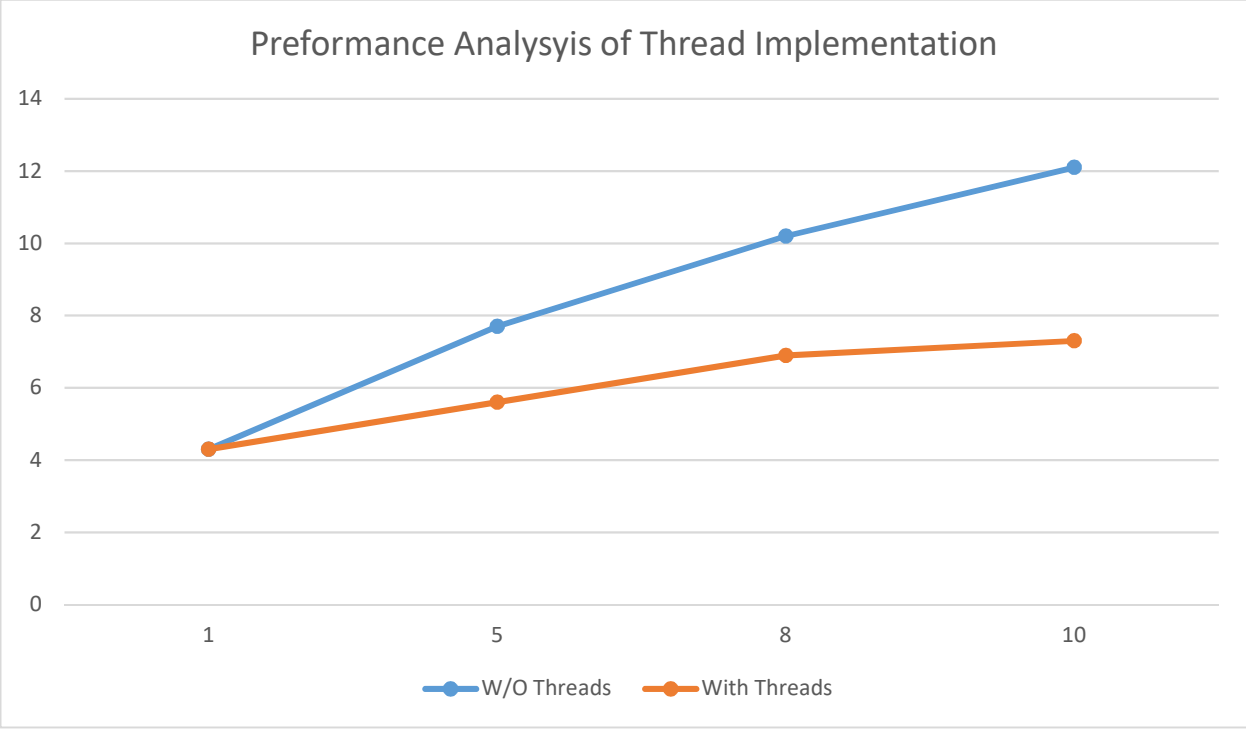

Figure 5.3 Performance Analysis Graph

CHAPTER 6

CONCLUSION

In conclusion, we have represented the security means that possess the ability to secure the data of users stored on cloud servers against malicious entities. Data Partitioning is observed providing an efficient scheme to split the customers' data into changeable chunks.

Data Encryption is covered through utilisation of the AES Encryption Algorithm to encrypt the user data on client machine. Files are further protected through renaming into random file names that are indexed and communicated across the servers. The index file further encrypted is a source of valuable information.

Insecure breach into the server is completely avoided by Diffie-Hellman key exchange algorithm. Implementation of AES cryptographic algorithms over a cloud computing environment is also ventured. The data partitioning scheme provides a dynamic perspective towards protection & security over cloud systems.

An efficient program is presented that promises to give the much required power and control to the most at risk party of Cloud Computing, i.e. the customer.

Providing security to the customer's data on the cloud will empower the standing of Cloud Computing & Storage.

REFERENCES

[1]. V. Fusenig, A. Sharma 2012. Security architecture for cloud networking, in: 2012 International Conference on Computing, Networking and Communications (ICNC). Presented at the 2012 ICNC Conference.

[2]. B.R. Kandukuri, V.R. Paturi, A. Rakshit, 2009. Cloud Security Issues, in: IEEE International Conference on Services Computing, SCC 2009.

[3]. S. Ramgovind, M.M. Eloff, E. Smith, 2010. The management of security in Cloud computing, in: Information Security for South Africa (ISSA), 2010.

[4]. Karun Handa, Uma Singh, 2015. Data Security in Cloud Computing using Encryption and Steganography. International Journal of Computer Science and Mobile Computing. IJCSMC, 2015.

[5]. H. Tianfield, 2011. Cloud computing architectures, in: 2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC). Presented at the 2011 IEEE SMC Conference.

[6]. Op – ed : Encryption, not restriction, is the key to safe cloud computing. https: / / www. nextgov.com/cloud-computing/2012/10/oped-encryp-tion-not-restriction-key-safe-cloudcomputing/58608

[7]. " Cloud Security and Privacy " , Tim Mather, Shahed Latif and Subra Kumaraswamy. – O'Reilly Book.

[8]. David Talbot. " How Secure Is Cloud Computing?" Technology Review [Online].Available: http://www.technologyreview.com/computing/23951/ 2009

[9]. Tania Gaur, Divya Sharma, 2016. A Secure and Efficient Client-Side Encryption Scheme in Cloud Computing. I.J. Wireless and Microwave Technologies. IJWMT 2016.

[10]. Elminaam, DiaaSalama Abdul et al. "Performance Evaluation of Symmetric Encryption Algorithms." IJCSNS International Journal of Computer Science and Network Security 8.12. 2008

[11]. “Heuristics in Physical Design Partitioning “, Bhargab Sinha, Naushad Manzoor Laskar, Rahul Sen.

[12]. “Security Aware Partitioning for efficient file system search”, Aleatha Parker Wood, Christina Strong, Ethan L. Miller.

APPENDIX A

CODE

encrypt_directory.py

```
import os, random, struct, hashlib, json
import threading, time, thread
from Crypto.Cipher import AES


def encrypt_file(key, in_filename, out_filename=None, chunksize=64*1024):

    if not out_filename:
        out_filename = in_filename + '@'

    iv = ''.join(chr(random.randint(0, 0xFF)) for i in range(16))
    encryptor = AES.new(key, AES.MODE_CBC, iv)
    filesize = os.path.getsize(in_filename)

    with open(in_filename, 'rb') as infile:
        with open(out_filename, 'wb') as outfile:
            outfile.write(struct.pack('<Q', filesize))
            outfile.write(iv)
```

```
            while True:
                chunk = infile.read(chunksize)
                if len(chunk) == 0:
                    break
                elif len(chunk) % 16 != 0:
                    chunk += ' ' * (16 - len(chunk) % 16)

                outfile.write(encryptor.encrypt(chunk))
    os.remove(in_filename)
def decrypt_file(key, in_filename, filename, out_filename=None, chunksize=24*1024):

    if not out_filename:
        out_filename = filename.replace('@','')

    with open(in_filename, 'rb') as infile:
        origsize = struct.unpack('<Q', infile.read(struct.calcsize('Q')))[0]
        iv = infile.read(16)
        decryptor = AES.new(key, AES.MODE_CBC, iv)

        with open(out_filename, 'wb') as outfile:
            while True:
                chunk = infile.read(chunksize)
                if len(chunk) == 0:
                    break
```

```
                outfile.write(decryptor.decrypt(chunk))


            outfile.truncate(origsize)
    os.remove(in_filename)
def renamefiles(pth):
        os.chdir(pth)
        ls=os.listdir(pth)
        index={}
        for a in ls:
                temp=a
                b=os.urandom(32).encode('hex')
                index[temp]=b
                os.rename(a,b)
        j=json.dumps(index)
        f=open("index.json","w")
        f.write(j)
        f.close()


def main():
        ps=raw_input("Enter the key : ")
        key = hashlib.sha256(ps).digest()
        pth=raw_input("Enter the path : ")
        os.chdir(pth)
        dirs=os.listdir(pth)
```

```
op=int(raw_input("Encrypt(0) or Decrypt(1) "))
if(op==0) :
        for in_filename in dirs :
                t=threading.Thread(target=encrypt_file,args=(key, in_filename,))
                t.start()
                t.join()
        print("Encrypted")
        renamefiles(pth)

else :
        indexfile=open("index.json","r")
        newindex=json.load(indexfile)
        #print(newindex)
        infiles=list(newindex.values())
        outfiles=list(newindex.keys())
        dirs.remove('index.json')
        for i in dirs :
                in_filename=outfiles[infiles.index(i)]
                #print (in_filename)
                t=threading.Thread(target=decrypt_file,args=(key, i, in_filename,))
                t.start()
                t.join()
                #os.remove(i)
        print("Decrypted")
```

APPENDIX B

PROCESS SCREENSHOTS

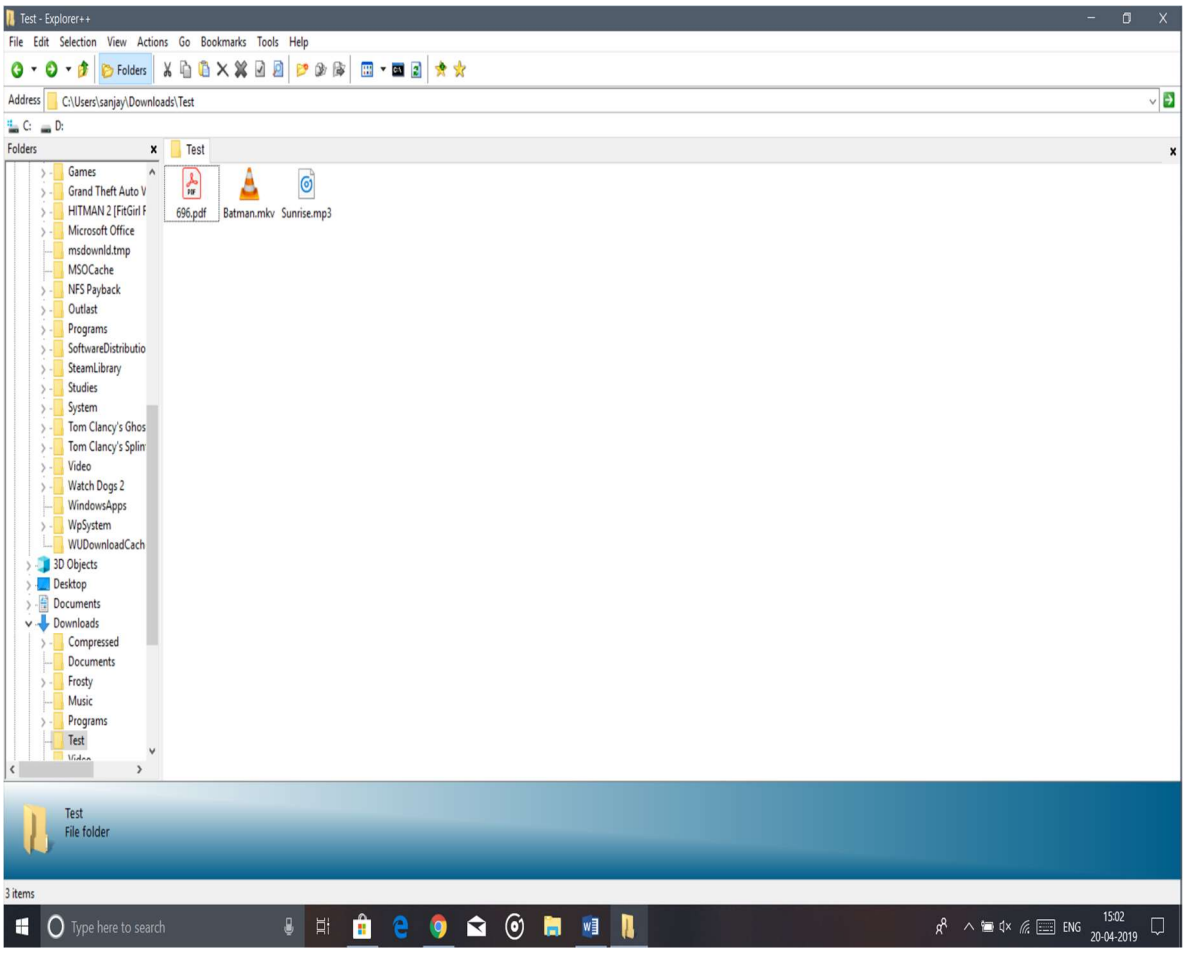

Fig 7.1 Original Directory

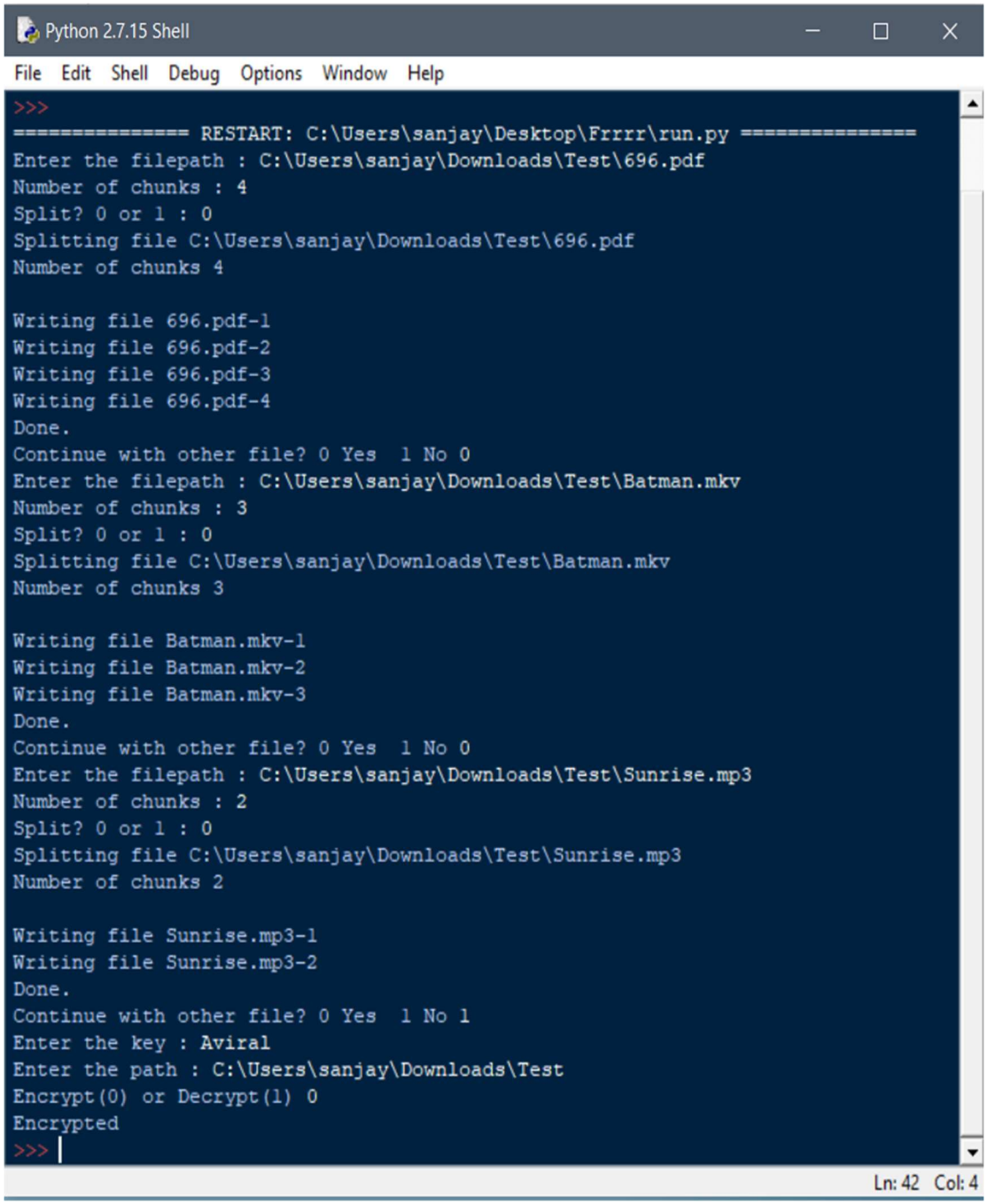

Fig 7.2 Server-Side Program

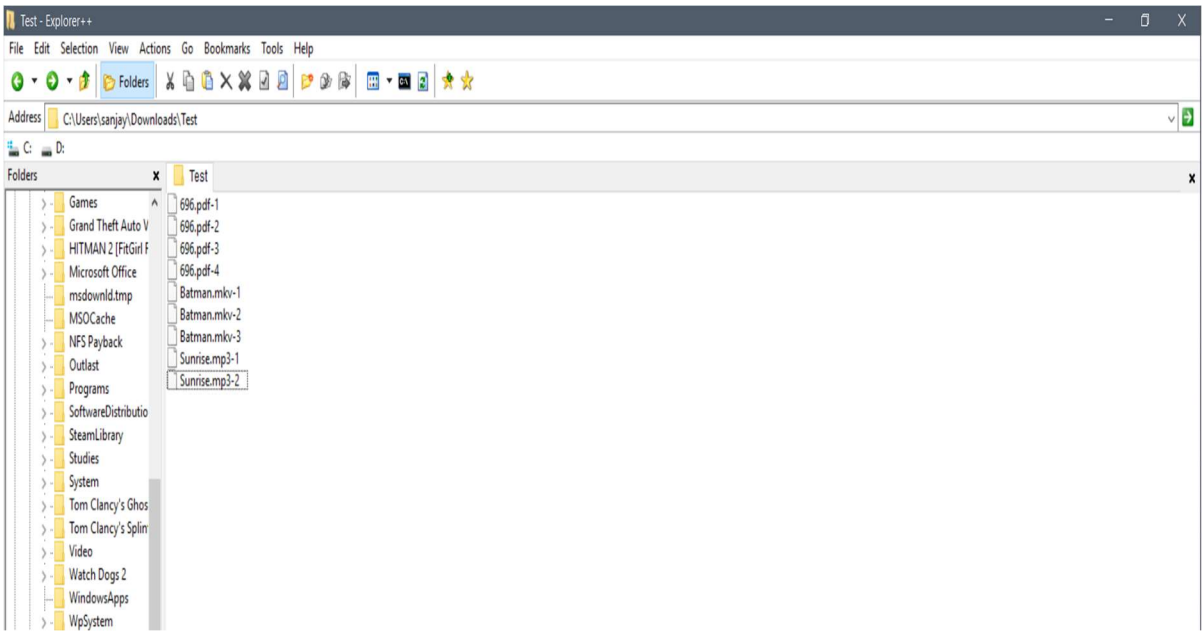

Fig 7.3 Partitioned Files

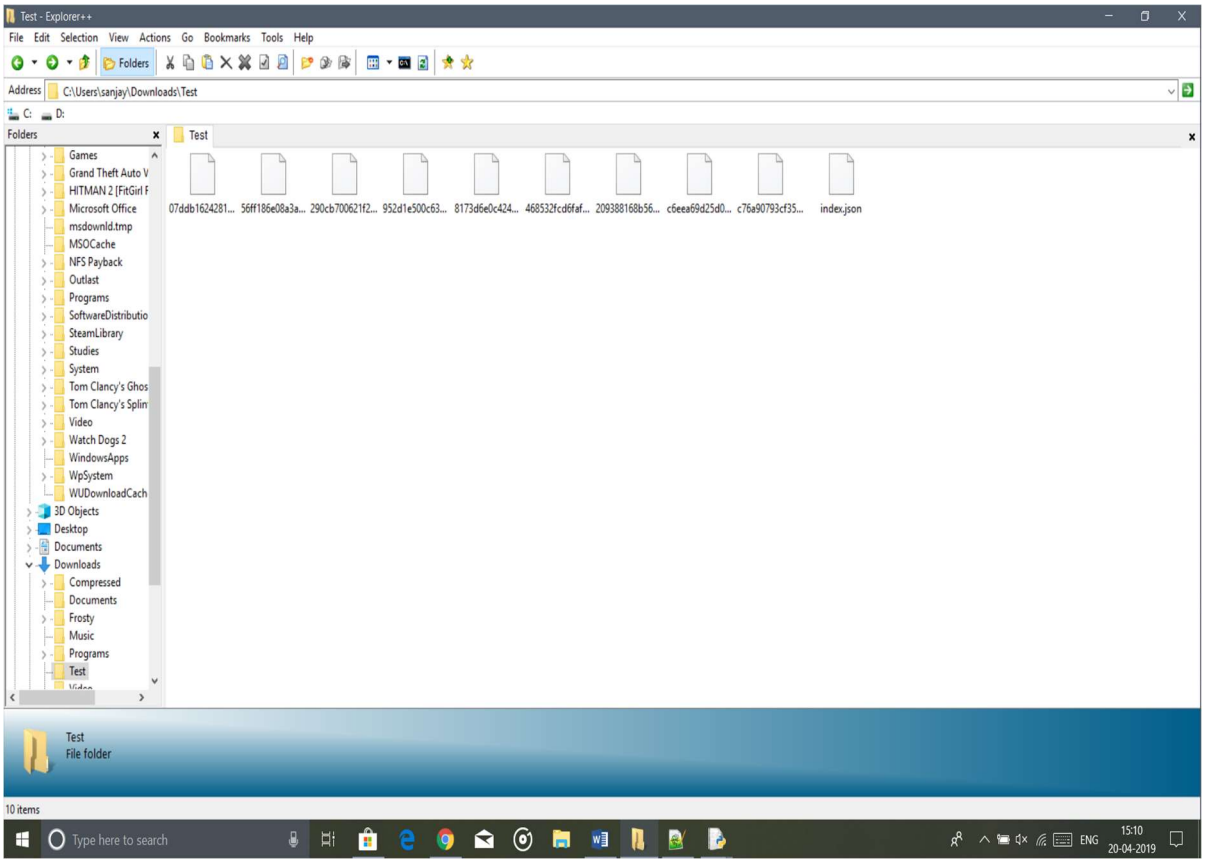

Fig 7.4 Encrypted Directory

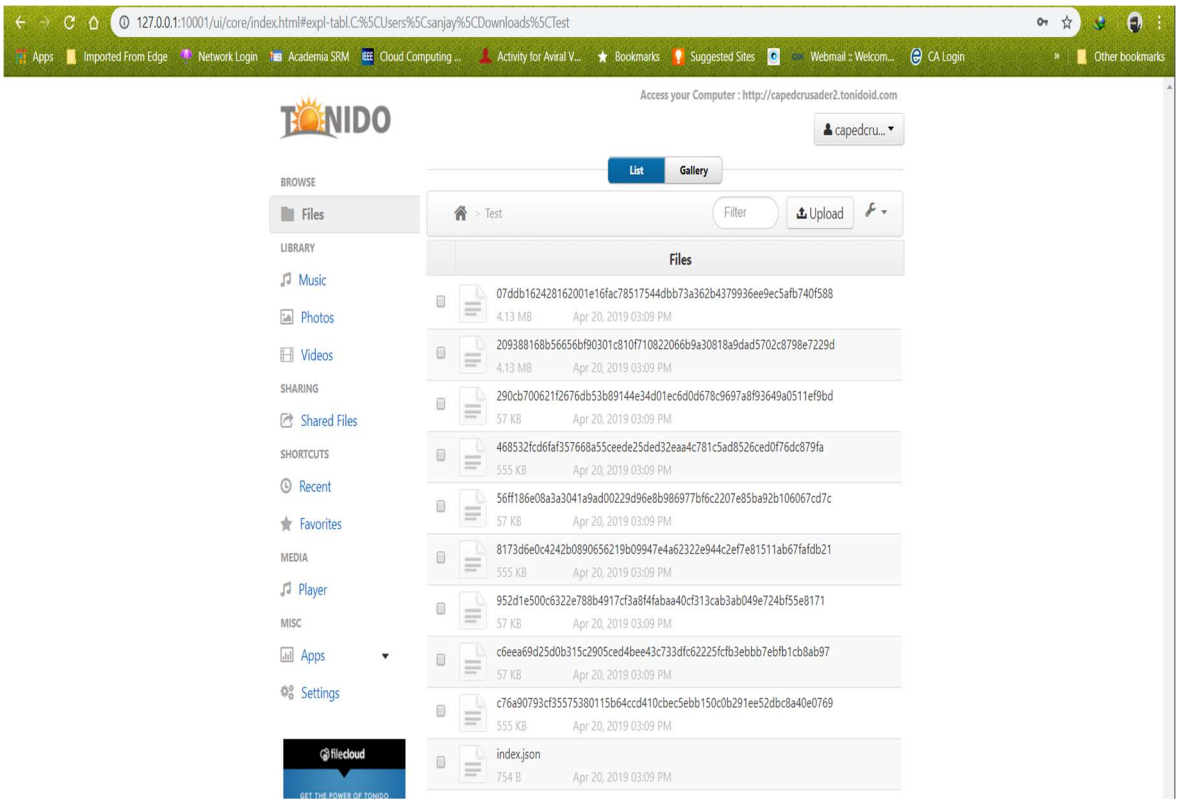

Fig 5.5 Uploaded Directory on Tonido Server

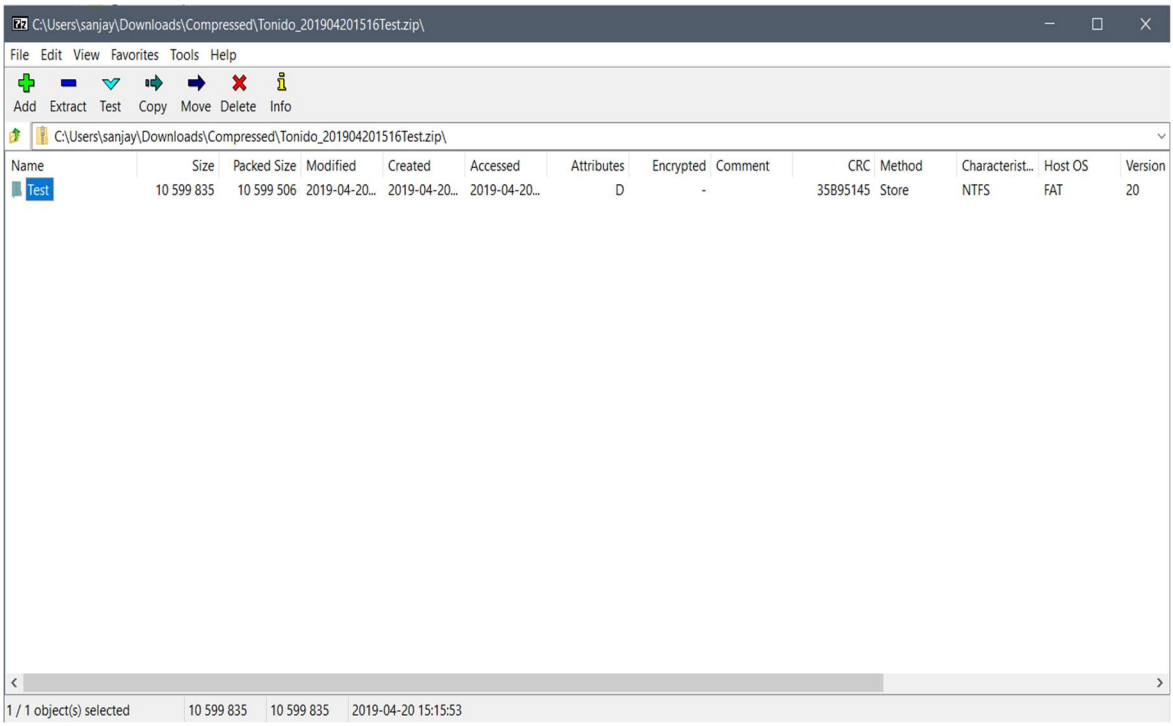

Fig 7.6 Downloaded Files Client Side (Zipped Download)

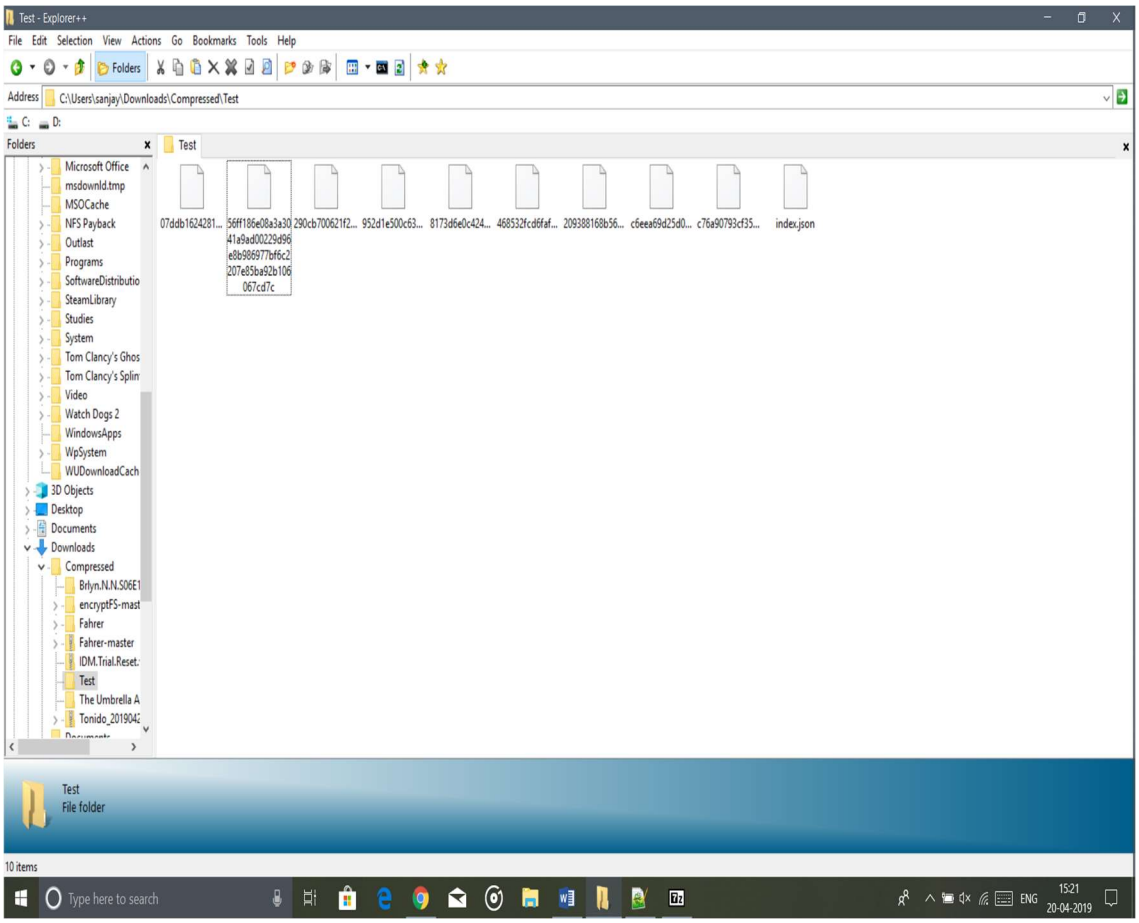

Fig 7.7 Downloaded Files Client Side (Extracted)

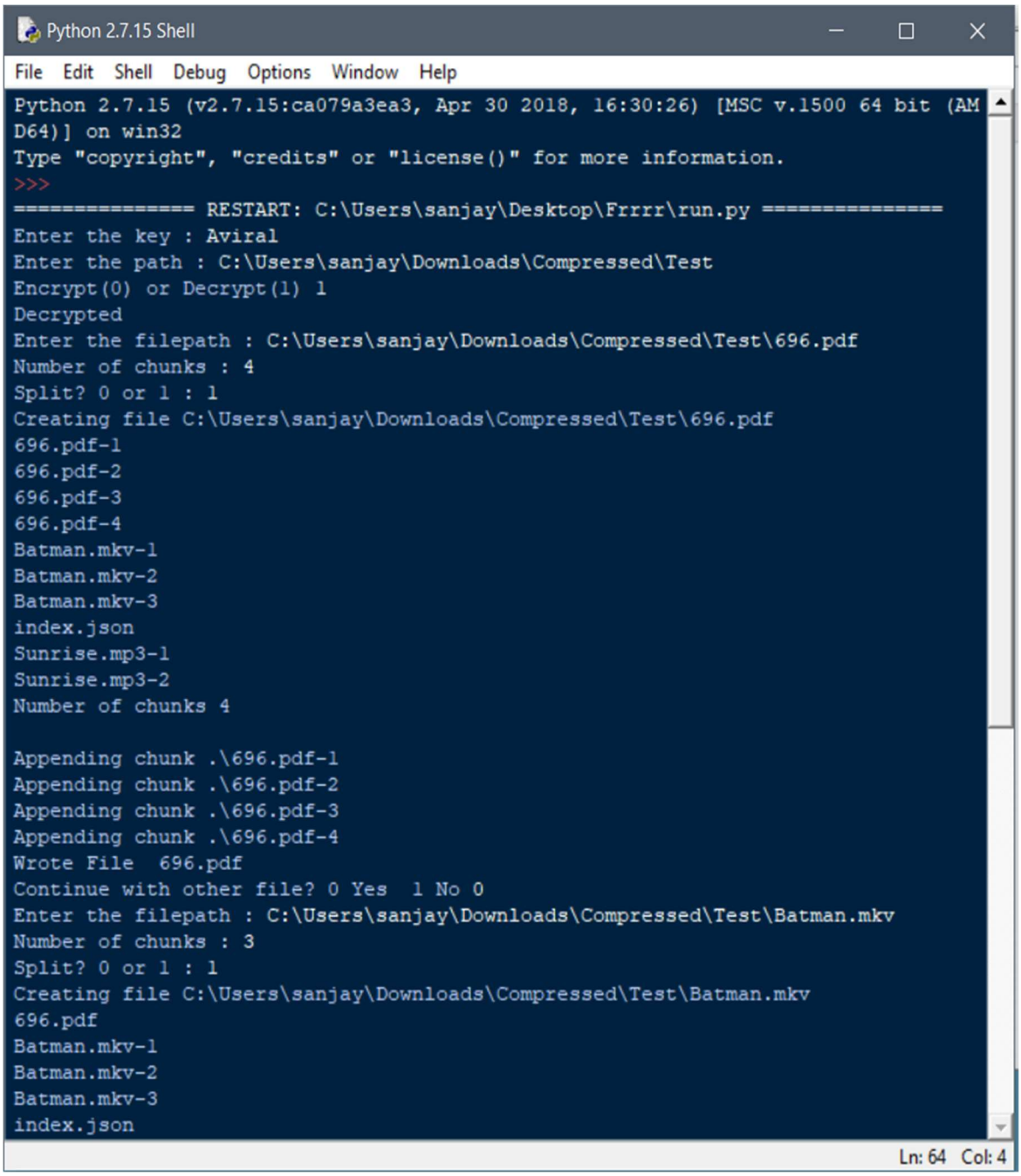

Fig 7.8 Client-Side Program

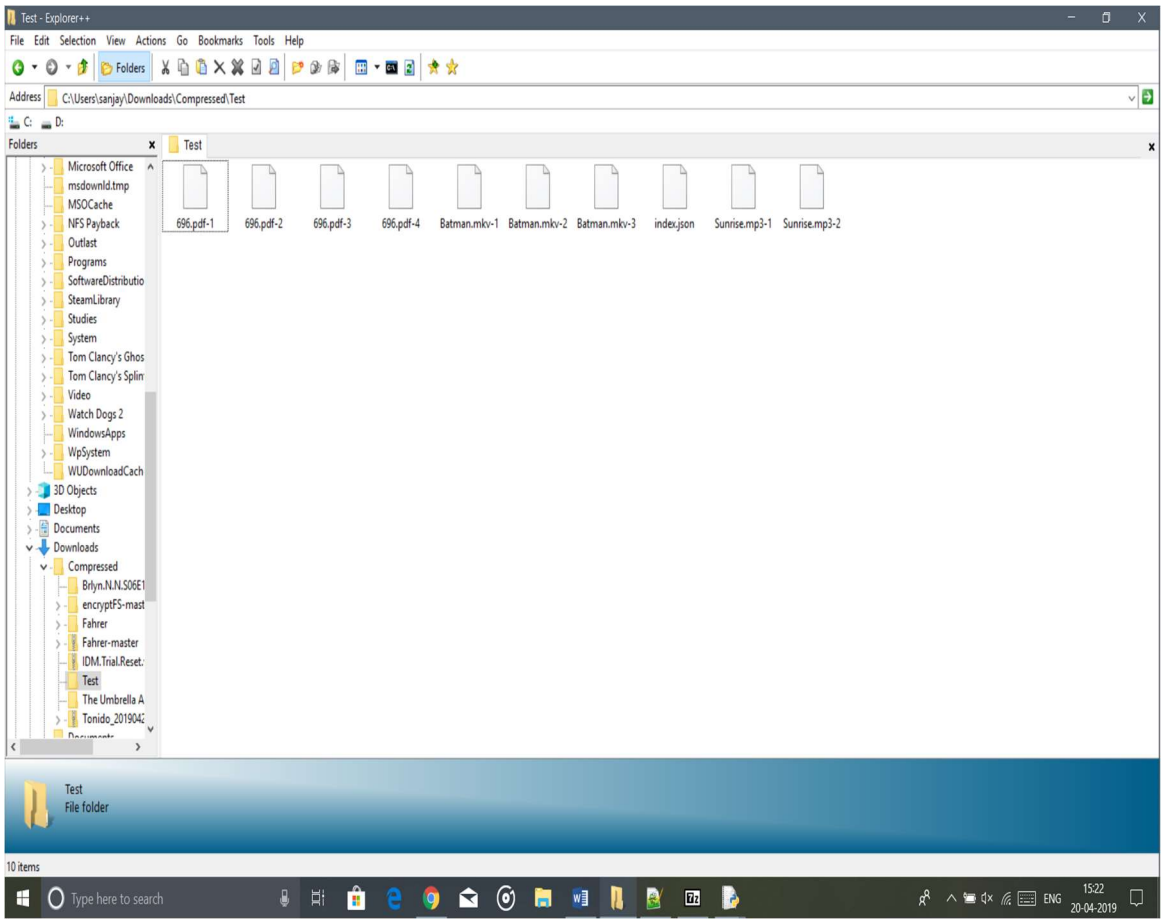

Fig 7.9 Decrypted Directory

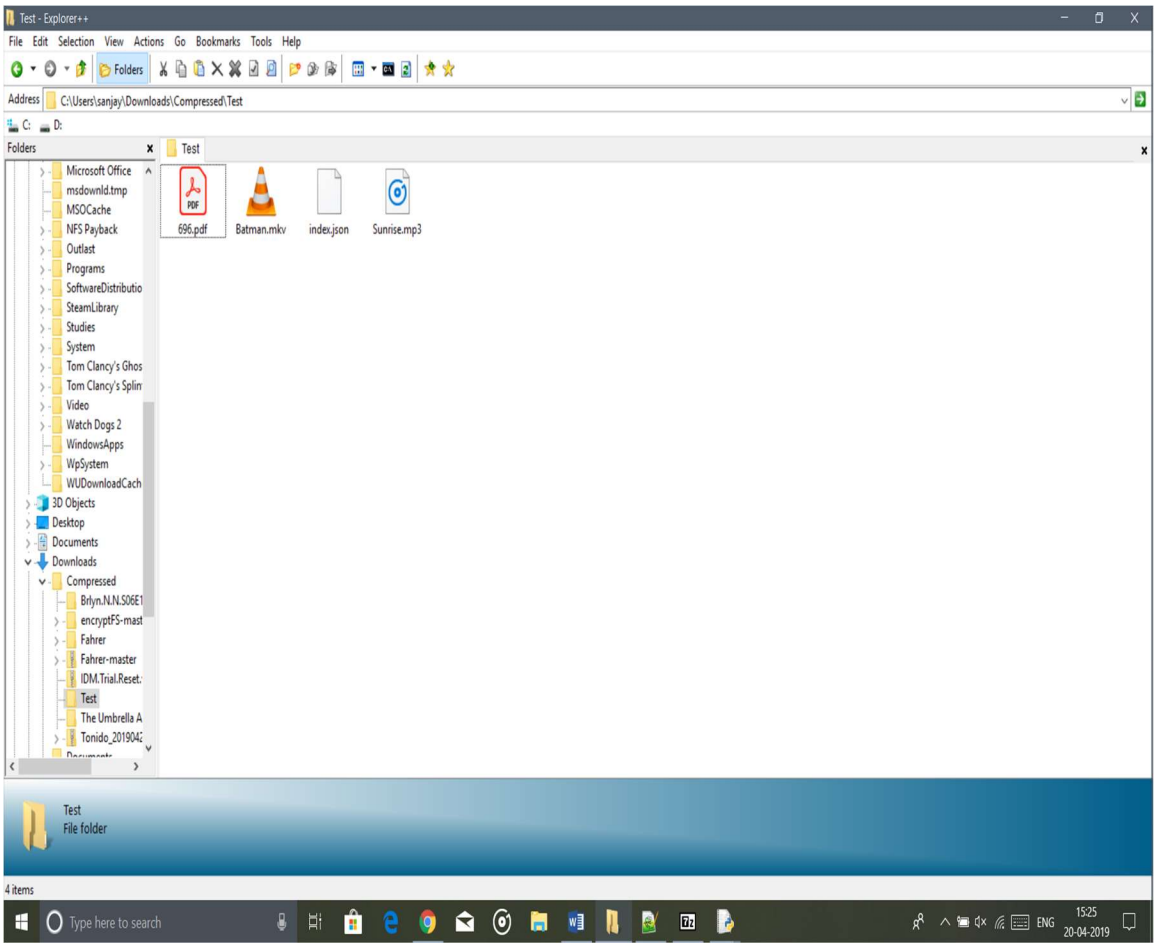

Fig 7.10 Merged Files

CONTRIBUTION SUMMARY

The table below summarises the contribution of each student in the formation of this Project.

Debanjan Deb **RA1511003010534**	1. Diffie Hellman Key Exchange Module. 2. UI Design of each Module.
Aviral Verma **RA1511003010688**	1. Directory Encryption Module. 2. File Partitioning Module.