VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Cloud based Secure Text Transfer Using Diffie-Helman Exchange

FINAL REVIEW REPORT

Submitted by

Adi Thakkar *[19BCE0074]*

Dharmik Govani *[19BCE0091]*

For

Information Security Analysis and Audit (CSE3501)

PROJECT COMPONENT

Submitted to

Prof. Siva Shanmugam

School of Computer Science and Engineering

ABSTRACT:

The emergence of cloud computing has been a watershed moment in computer science. It gave the ability to solve numerous problems that were previously thought to be impossible to address by a machine. It removed the pressure from those responsible for manufacturing better machines to keep up with the increasing complexity of the problems that the machines are intended to solve. Cloud computing offered a framework for more efficient use of globally distributed resources. Because it is a new sector, it is teeming with issues that engineers and scientists are frantically attempting to solve. One of the major disadvantagesof cloud computing is security. As a result, this project presents a secure file storage cloud technique based on encryption and Diffie Hellman. The approach encrypts the cloud file and uses Diffie Hellman to authenticate the user in order to decrypt the needed file.

INTRODUCTION:

One of the key concerns in the cloud computing arena is cloud security. Storing personal and sensitive data on a third-party storage media exposes you to the danger of data theft and exploitation by malevolent individuals. The danger is so great that it has deterred governments and many other large organizations from moving their operations to the cloud. In the cloud, traditional techniques of safeguarding files and information are no longer necessary. In order to make cloud more secure and dependable, extensive research and study is being conducted in this subject. Some of the methods that stand out among these, some of the studies include AES encryption and Diffie Hellman Key Exchange. The latter method is so powerful that even today's most powerful computers may take millions of years to crack the code and read the file. Our strategy presents a mechanism that entails encrypting the file using any standard encryption technique and using Diffie Hellman for user authentication. As a result, the files can be safely saved in the public domain without the risk of being accessed by unauthorized individuals.

SURVEY REPORT:

Reference	Author	Idea	Advantages	Disadvantages
19BCE0074 ADI THAKKAR				
Data Security in Cloud Architecture Based on Diffie Hellman and Elliptical Curve Cryptography	Tirthani, N. and Ganesan, R., 2014. Data Security in Cloud Architecture Based on Diffie Hellman and Elliptical Curve Cryptography. *IACR Cryptol. ePrint Arch.*, *2014*, p.49.	They suggested a new architecture for deploying these two approaches, which can be utilized to create a cloud system for. At the same time, cloud servers have improved security and reliability. Time maintaining data integrity from the user's perspective. Then they move on to compute the key for cryptography, one for ECC and one for Diffie Hellman	They have used Diffie Hellman protocol as it significantly better for establishment of connections. They also place a strong emphasis on the implementation of the planned architecture, as well as many comparisons to demonstrate its efficacy.	They can work on the implementation of the proposed architecture along with different comparisons to show the effectiveness of their proposed architecture
A hybrid network security algorithm based on Diffie Hellman and Text-to-Image Encryption algorithm	Abusukhon, A., Anwar, M. N., Mohammad, Z., & Alghannam, B. (2019). A hybrid network security algorithm based on Diffie Hellman and Text-to-Image Encryption algorithm. Journal of Discrete Mathematical Sciences and Cryptography, *22*(1), 65-81.	Java NetBeans is utilized as a vehicle in this article to carry out their investigations. The key exchange algorithm is used to accomplish the key exchange between the client and the server in the first phase of their suggested method. The Diffie Hellman key exchange technique is employed again in the second step of their proposed process to generate three random numbers (R, G, and B) on both sides of the network. The TTIE is upgraded by adding another level of encryption (the COP) to the CIP level in the third phase of their proposed method (the Diffie Hellman Text to Image Encryption algorithm (DHTTIE)).	The TTIE encryption method has been improved by introducing a new level of protection (COP). The Diffie Hellman method was used in this paper to generate the TTIE's key entries (coloured pixels) on both sides of the network without the need to transfer the key over the network. This is done to protect the keys from prying eyes.	In future, they can investigate the DHTTIE when the key size is greater than 624bits and the tested data is a large-scale collection (multiple GBytes).

Symmetric Key Generation and Distribution Using Diffie-Hellman Algorithm	Purohit, K., Kumar, A., Upadhyay, M., & Kumar, K. (2020). Symmetric Key Generation and Distribution Using Diffie-Hellman Algorithm. In Soft Computing: Theories and Applications (pp. 135-141). Springer, Singapore.	In the proposed improved Diffie-Hellman method, they first choose a huge prime number 'p' and a prime number generator 'g', where 'g' is less than 'p' and the power of 'g' must generate all the numbers varied in size. In the proposed algorithm, each and every user have to select a random private key ’k’ on his own such that its public key	The time taken by the existing and new algorithms were noted and compared, resulting in the proposed model having a better complexity and efficiency when tested, resulting in improved safety measures.	The proposed solution is yet to be implemented on a larger scale
A Novel Text Encryption Algorithm using enhanced Diffie Hellman and AES	Kishore, K. N., & Chhetri, S. (2020). A novel text encryption algorithm using enhanced Diffie Hellman and AES. *Int. J*, *8*(6).	They take a primitive root to the secret gained and two private keys, then use the technique to produce a second secret key using the secret key obtained, making it more secure because the attacker won't know we're using a primitive root to the secret key obtained. This secret key can now be passed around and used in the AES algorithm.	The upgraded version of Diffe-Hellman is substantially more secure and cannot be broken by man-in-the-middle, plain-text, or other attacks. Because the attacker has no idea that we're taking a primitive root of the secret key and utilising both to produce new public keys and hence newer secret keys, it's the more secure method.	The only downside of this algorithm, for now, is that it’s execution time is more though the run time is scalable enough.

Use of Digital Signature with Diffie Hellman Key Exchange and AES Encryption Algorithm to Enhance Data Security in Cloud Computing	Rewagad, P., & Pawar, Y. (2013, April). Use of digital signature with diffie hellman key exchange and AES encryption algorithm to enhance data security in cloud computing. In 2013 International Conference on Communication Systems and Network Technologies (pp. 437-439). IEEE.	They use a three-way protection strategy in our proposed architecture. To begin the key exchange process, the Diffie Hellman algorithm is utilized to generate keys. The AES encryption algorithm is then used to encrypt or decode the user's data file after which a digital signature is utilized for authentication. All of this is done to create a secure computing environment and prevent data alteration at the server. For the same reason, two servers are maintained: one for the encryption process, known as (trusted) computing platform, and another for storing user data files, known as storage server.	This proposed architecture of three-way mechanism makes it tough for hackers to crack the security system, thereby protecting data stored in cloud.	Diffie-Hellman key exchange can be enhanced to prevent the its pitfalls. They haven't put up a concrete scalable application on the topic they have done research on.
19BCE0091 DHARMIK GOVANI				
Multi resolution like Data Transmission Security using Diffie-Hellman Protocol	Hanaa Mohsin Ahmed** and Razeen Waheed Jassim*	Parallel multi recipient signcryption that improves the performance in the case when sender transmits distinct messages to multiple recipients in an imbalanced wireless network Phase 1: Hashed Diffie-Hellman Key Exchange Protocol Phase 2: Multi-Resolution Like in Matrix Schema for Data Transmission	Productive and dependable in view of the necessary preparing simultaneously by messages encryption has been done any place the computational time for the proposed cryptosystem is adequate when accepts a couple of milliseconds as real execution time for scramble and unscramble messages came to (5 milliseconds).	Takes more time than expected in running different tests
Prevention of Man-In-The-Middle Attack in Diffie-	Thwe, P.P. and Htet, M., 2019. Prevention of Man-In-The-Middle Attack in Diffie-	This exploration zeroed in on security issues of DHKE calculation and fostered a validated key trade approach. The proposed	Good Security Level Total Execution time of the proposed VRH	hash function is created by using six bitwise operators and operated in a

Hellman Key Exchange Algorithm using Proposed Hash Function	Hellman Key Exchange Algorithm using Proposed Hash Function. Prevention.	key trade approach can create a safer meeting key for the encryption cycle with the assistance of proposed VRH work. In the proposed VRH work, the first message can't be recuperated from the hash code for the explanation that number of round tasks of the inward capacities relies upon the length of the message.	function is faster than some of the existing functions Prevent MITM Attack	variable length of the rounds depending on message length
The Simplest Protocol for Oblivious Transfer	Chou, T. and Orlandi, C., 2015, August. The simplest protocol for oblivious transfer. In International Conference on Cryptology and Information Security in Latin America (pp. 40-58). Springer, Cham.	Basic, effective and secure OT convention. The convention is a basic change of the Diffie-Hellman (DH) key trade convention	Software is a constant-time one: timing attacks are avoided using the high-level strategy as [BDL+11]	Implementation uses the NIST K-283 curve and SHA-1 for hashing, and it is not a constant-time implementation
Oblivious Transfer based on Key Exchange	Parakh, A., 2008. Oblivious transfer based on key exchange. Cryptologia, *32*(1), pp.37-44.	convention for common trade of privileged insights, 1-out-of-2 OT and coin-flipping like Diffie-Hellman convention utilizing the possibility of negligently trading encryption keys.	most OT plans can be stretched out to coin flipping with minor alterations, in which case, just uneven exchange might happen and achievement or disappointment relies upon the restricting party being adequately fortunate to derive the key.	Mutual exchange of secrets, both parties should be trustable
Hardness of Computing the Most Significant Bits of Secret Keys in Diffie-Hellman and Related Schemes	Boneh, D. and Venkatesan, R., 1996, August. Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes. In Annual International Cryptology Conference (pp. 129-142). Springer, Berlin, Heidelberg.	Shows that registering the main pieces of the mysterious key in a Diffie-Hellman key-trade convention from people in general keys of the members is just about as hard as processing the mysterious key itself.	HNP can be utilized to demonstrate that figuring a piece of the Diffie Hellman secret is pretty much as hard as registering the whole confidential.	No disadvantages found as comparing the hardness is achieved.

EXISTING SYSTEMS:

There have been several message passing systems developed that try to achieve message transfer securely and safely. This systems have been improved over the years and various modifications have been made to improve the security and performance.

- Various Key Transfer Algorithms such as Symmetric Key Exchange are used
- Shared Secret Key Mechanism is used in text transfer.
- Many of the existing Systems have been developed that run locally and achieve secure message transfer locally.
- The local Implementations uses a Naive Encryption and Decryption technique and provide less security.
- The Hash algorithms used in encryption and decryption are improved over time to increase security and avoid brute-force approach.
- Many existing systems tried to overcome this disadvantage by either using large values of p and q for AES or by increasing the level of security achieved in this process.
- Many Systems used a naive approach of creating an external key exchange mechanism that is safer than previous one, but leads to longer process for message passing.

GAP ANALYSIS:

The following gaps were identified throughout the survey conducted:

- The system was not implemented, if implemented wasn't viable on a large scale.
- The system implemented exists locally, hence greatly reducing the accessibility of the application.
- Even if the system is hosted online, for better access, such secure transactions are not as secure in the online space, because of a whole new dimension of plausible attacks, attributed simply to the fact that it is now on the internet.
- The connotation of the above, is the Encryption mechanism used is susceptible to various attacks, like NFS (more on this in implementation) , brute force attacks, man in the middle attacks, etc.
- The speed of the such encryption methods, although secure is not very fast, which harms the speed at which applications depending on this can function.
- Diffie Helman can further be improved to mitigate its pitfalls.

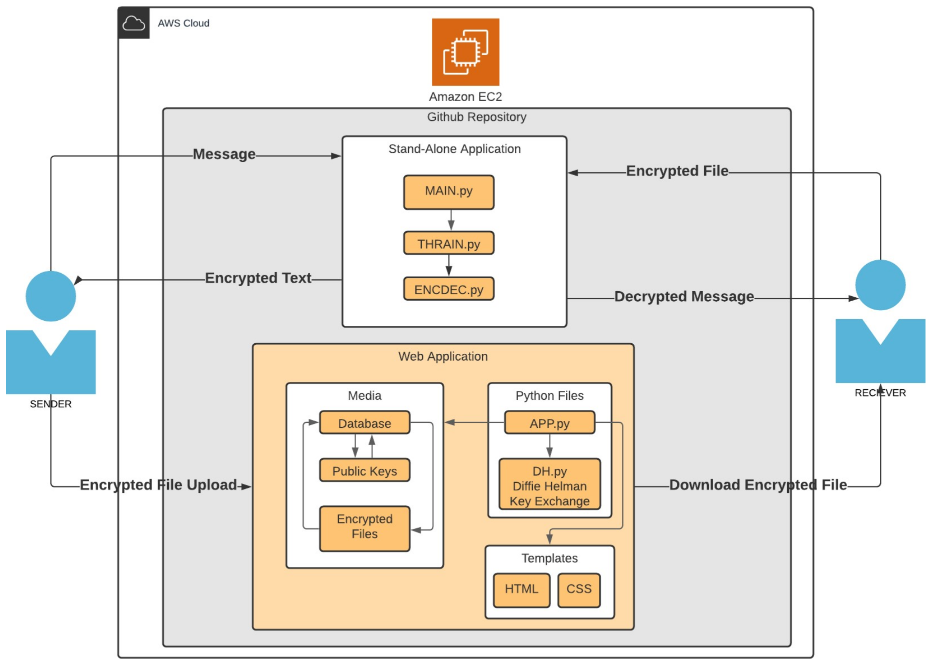
PROPOSED SYSTEM:

The proposed system consists of the following features:-

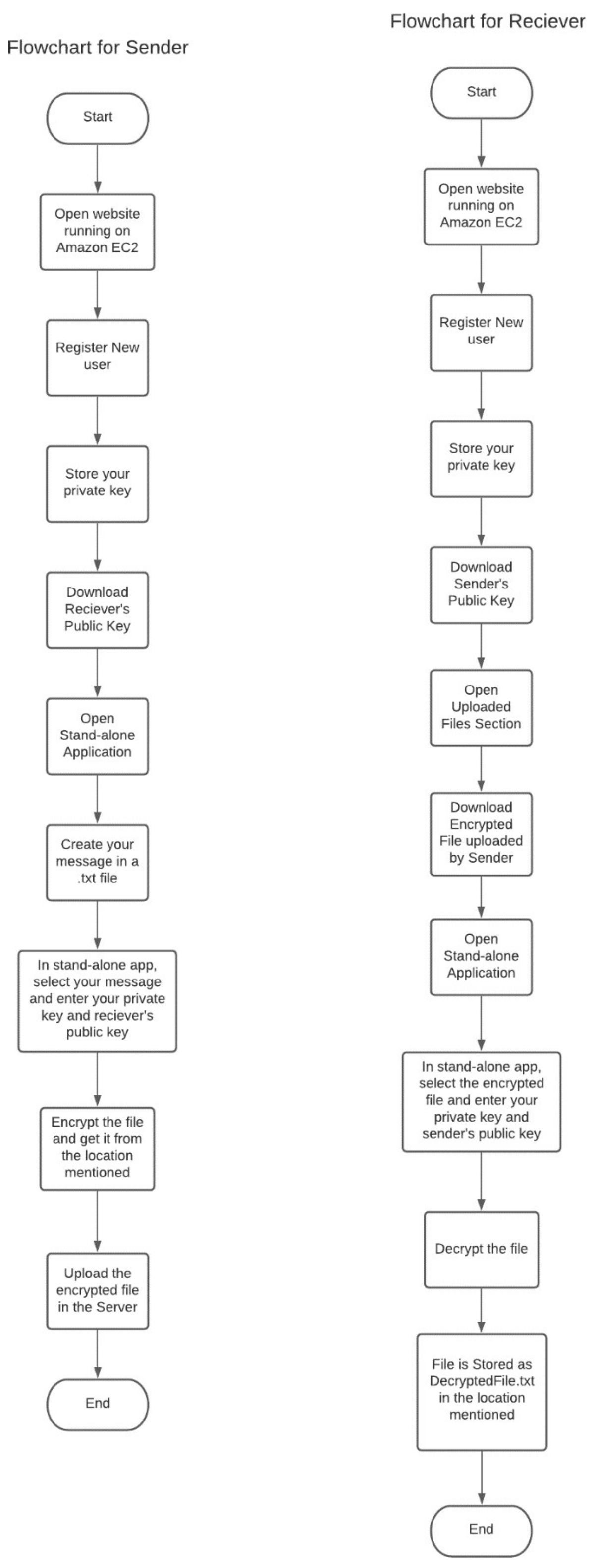
- **Encryption/Decryption**- This is done using AES, and the improved Diffie-Helman Exchange, which mitigates the pitfalls of the Diffie-Helman key Exchange, which was already a secure mechanism to begin with, as seen from the papers reviewed in the survey report. Hence **INCREASED SECURITY** is ensured.
- **Stand-Alone-Application**- This is a python tkinter GUI, where the encryption/decryption will take place. This can be done while offline, to increase the security of the app. Hence the app is **LESS PRONE TO CYBER ATTACKS**.
- **Web-Application**- After the encryption task is done, (which is the most important part, which should have the max. Security), the user interacts with this user-friendly online system to upload text files to the cloud. Even if the system is compromised, **THE ATTACKER WON'T BE ABLE TO DERIVE ANY INFORMATION** from the encrypted text.
- **Cloud-Based**: The above application is hosted in the EC2-instance in the AWS cloud, Hence **SAFE FROM ANY PHYSICAL HARM OR FATAL SYSTEM ERROR.**

ARCHITECTURE DIAGRAM:

Architecture Diagram



FLOW DIAGRAM


Flowchart for Sender
Start
Open website running on Amazon EC2
Register New user
Store your private key
Download Reciever's Public Key
Open Stand-alone Application
Create your message in a .txt file
In stand-alone app, select your message and enter your private key and reciever's public key
Encrypt the file and get it from the location mentioned
Upload the encrypted file in the Server
End
Flowchart for Reciever
Start
Open website running on Amazon EC2
Register New user
Store your private key
Download Sender's Public Key
Open Uploaded Files Section
Download Encrypted File uploaded by Sender
Open Stand-alone Application
In stand-alone app, select the encrypted file and enter your private key and sender's public key
Decrypt the file
File is Stored as DecryptedFile.txt in the location mentioned
End

DETAILED PARAGRAPH EXPLANATION:

Diffie–Hellman key trade (DH) is a technique for safely trading cryptographic keys over a public channel and was one of the principal public-key conventions named after Whitfield Diffie and Martin Hellman. [1] DH is one of the soonest viable instances of public key trade executed inside the area of cryptography.

In public key cryptosystem, enciphering and deciphering are governed by distinct keys, E and D, such that computing D from E is computationally infeasible (e.g., requiring more than 10100 instructions). The enciphering key E can thus be publicly disclosed without compromising the deciphering key D. This was the main ideology behind Diffie-Hellman Key Exchange Protocol. Each user of the network can, therefore, place his enciphering key in a public directory. This enables any user of the system to send a message to any other user enciphered in such a way that only the intended receiver can decipher it. As such, a public key cryptosystem is a multiple access cipher. A private conversation can therefore be held between any two individuals regardless of whether they have ever communicated before. Each one sends messages to the other enciphered in the receiver's public enciphering key and deciphers the messages he receives using his own secret deciphering key.

Diffie Hellman Key Exchange

Diffie–Hellman key exchange (DH) is a method of securely exchanging cryptographic keys over a public channel and was one of the first public-key protocols named after Whitfield Diffie and Martin Hellman. [1] DH is one of the earliest practical examples of public key exchange implemented within the field of cryptography.

In public key cryptosystem, enciphering and deciphering are governed by distinct keys, E and D, such that computing D from E is computationally infeasible (e.g., requiring more than 10100 instructions). The enciphering key E can thus be publicly disclosed without compromising the deciphering key D. This was the main ideology behind Diffie-Hellman Key Exchange Protocol. Each user of the network can, therefore, place his enciphering key in a public directory. This enables any user of the system to send a message to any other user enciphered in such a way that only the intended receiver can decipher it. As such, a public key cryptosystem is a multiple access cipher. A private conversation can therefore be held between any two individuals regardless of whether they have ever communicated before.

Each one sends messages to the other enciphered in the receiver's public enciphering keyand deciphers the messages he receives using his own secret deciphering key.

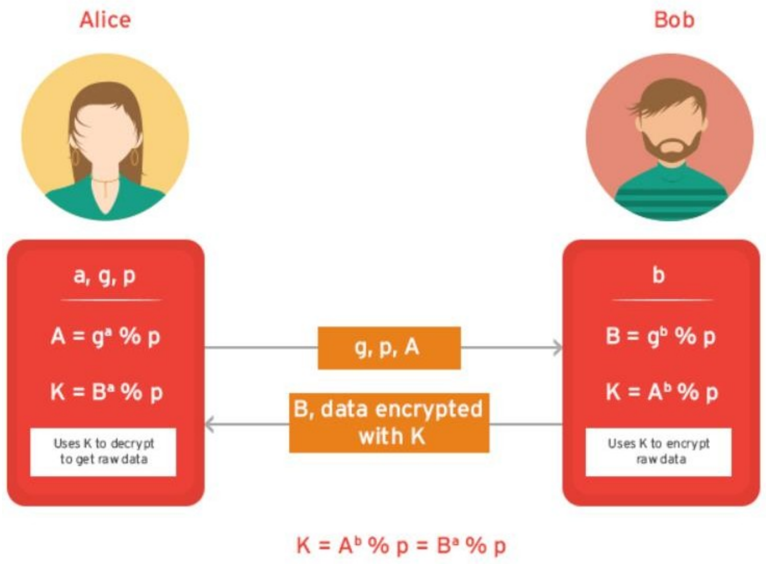


Figure 1: Illustration of idea behind Diffie-Hellman

Diffie–Hellman key exchange establishes a shared secret between two parties that can be used for secret communication for exchanging data over a public network. The above conceptual diagram illustrates the general idea of the key exchange by using colors instead of large numbers.

The process begins by having the two parties, Alice, and Bob, agree on an arbitrary starting color that does not need to be kept secret in this example the color is yellow. Each of them selects a secret color that they keep to themselves – in this case, orange and blue-green. The crucial part of the process is that Alice and Bob each mix their own secret color together with their mutually shared color, resulting in orange-tan and light-blue mixtures respectively, and then publicly exchange the two mixed colors. Finally, each of the two mixes the color he or she received from the partner with his or her own private color. The result is a final color mixture (yellow-brown in this case) that is identical to the partner's final color mixture. If a third party listened to the exchange, it would be computationally difficult for this party to determine the secret colors. In fact, when using large numbers rather than colors, this action is computationally expensive for modern supercomputers to do in a reasonable amount of time. [2]

Prime Number

A prime number (or a prime) is a natural number greater than 1 that cannot be formed by multiplying two smaller natural numbers.

The only user-defined pre-existing parameter in theDiffie-Hellman protocol is the selection of prime number. The prime number p should be large enough to defend against the known attacks against it. The most efficient attack is NFS (attack on the network file system); that has been used against numbers on the order of 2^768 (a 232-digit number). It would appear wise to pick a p that is bigger than that; around 1024 bits at a minimum, and more realistically at least 1536 bits. Another property about p is that p−1 should have a large prime factor q, and one should know what the factorization of p−1 is. If we pick a random prime p, and a random generator g, well, we are secure, but we will not be certain (and we might leak a few bits of the private exponent if the order of your random g happens to have some small factors).

Method

Follow the mathematical implementation of Diffie Hellman key exchange protocol.

p is a prime number.

g is a primitive root modulo of p

1. Alice and Bob agree to use a modulus p = 23 and base g = 5
2. Alice gets her private key (key which she should not share with anyone) generated as 4.
3. Thus, public key generated for Alice shall be 54%23 = 625%23 = 4
4. Bob gets his private key (key which he should not share with anyone) generated as 3.
5. Thus, public key generated for Bob shall be 53%23 = 125%23 = 10
6. Now, Alice gets the public key of Bob and generates a secret key. i.e. (public key of Bob Private Key of Alice) mod p

=> (104) % 23 => 10000 % 23 => 18

7. On the other side, Bob also uses a similar method to generate a secret key i.e. (public key of Alice Private Key of Bob) mod p

=> (43) % 23 => 64 % 23 => 18

Thus, it is proven that mathematically, Alice and Bob generate the same key without each one of them knowing other one’s private key. This is the implementation of Diffie-Hellman Key Exchange Protocol.

Encryption

Encryption is widely used on the internet to protect user information being sent between a browser and a server, including passwords, payment information and other personal information that should be considered private. Organizations and individuals also commonly use encryption to protect sensitive data stored on computers, servers and mobile devices like phones or tablets. There are various encryption techniques that are present some of which are:

- Triple DES
- Blowfish
- RSA
- Twofish
- AES

The technique that we have used in our project is AES and it is described below.

Advanced Encryption Standard

The more popular and widely adopted symmetric encryption algorithm nowadays is the Advanced Encryption Standard (AES). It is found to be at least six time faster than triple DES.

A replacement for DES was needed as its key size was too small. With increasing computing power, it was considered vulnerable against exhaustive key search attack. Triple DES was designed to overcome this drawback, but it was found to be slow.

The AES has three fixed 128-bit block ciphers with cryptographic key sizes of 128, 192 and 256 bits. Key size is unlimited, whereas the block size maximum is 256 bits. The AES design is based on a substitution-permutation network (SPN) and does not use the Data Encryption Standard (DES) Feistel network. The diagram below shows the implementation of AES encryption technique.

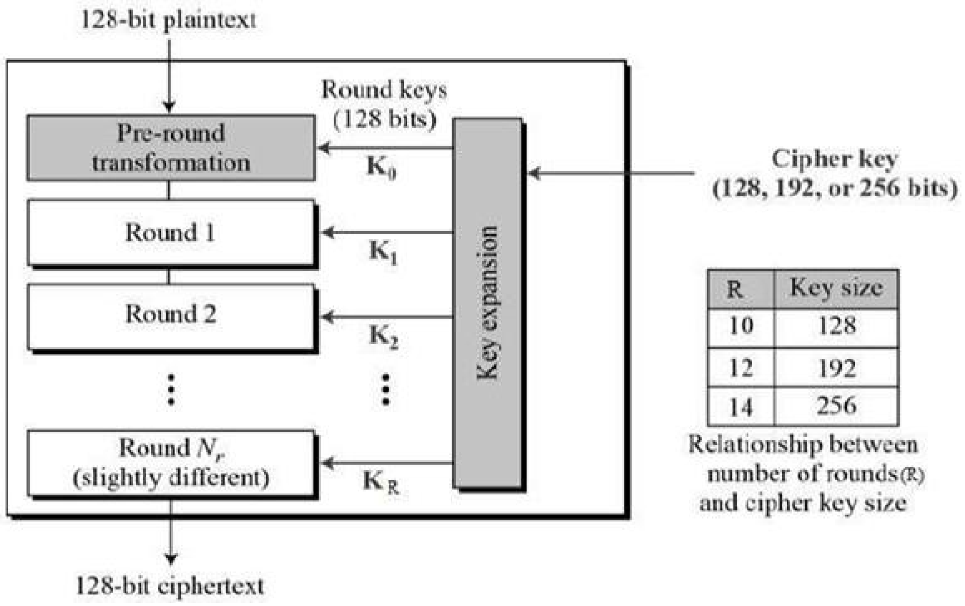


Figure 2: The schematic of AES structure

Python

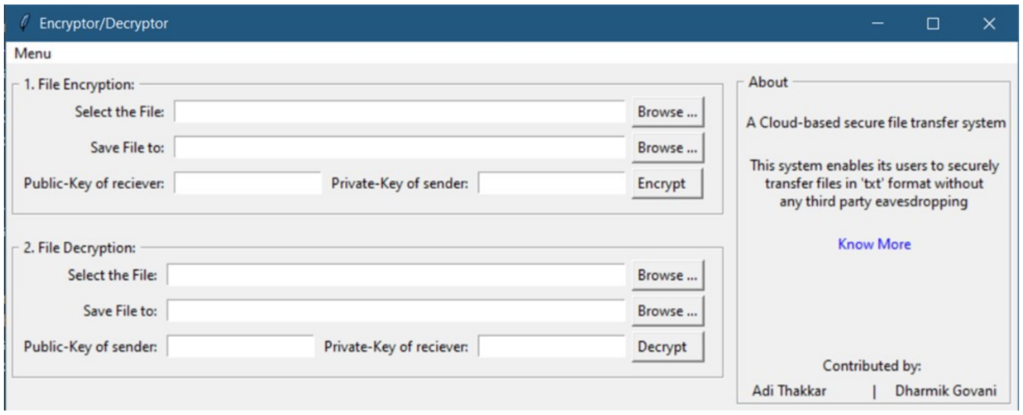
Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespaces. It provides constructs that enable clear programming on both small and large scales. [3] Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, and procedural, and has a large and comprehensive standard library.

Such large and comprehensive standard libraries along with the documented support helped us to choose python as the language which we used to build both, the stand alone as well as web-based application.

Below is a brief description of the frameworks and libraries extensively used to build the desired framework.

Python tkinter

Tkinter is Python's de-facto standard GUI (Graphical User Interface) package. It is a thin object- oriented layer on top of Tcl/Tk. [5] Tkinter is not the only GUI Programming toolkit for Python. It is however the most used one. Cameron Laird calls the yearly decision to keep Tkinter "one of the minor traditions of the Python world." Tkinter is a GUI (graphical user interface) widget set for Python. This document was written for Python 2.7 and Tkinter 8.5 running in the X Window system under Linux. Tkinter helps users to build a cross-platform application and is easy to use, thus, we used it to build the GUI of our stand-alone application. Figure 3 shows the GUI of Cloud-based Secure Text Transfer, the framework of the hour.



A Cloud-based secure file transfer system

This system enables its users to securely transfer files in 'txt' format without any third party eavesdropping

Figure 3: GUI built using python-tkinter

Python flask

Flask is a BSD licensed microframework for Python based on Werkzeug and Jinja 2. “Micro” does not mean that your whole web application must fit into a single Python file (although it certainly can), nor does it mean that Flask is lacking in functionality. The “micro” in microframework means Flask aims to keep the core simple but extensible. Flask will not make many decisions for you, such as what database to use. Those decisions that it does make, such as what templating engine to use, are easy to change. Everything else is up to you, so that Flask can be everything you need and nothing you don’t.

By default, Flask does not include a database abstraction layer, form validation or anything else where different libraries already exist that can handle that. Instead, Flask supports extensions to add such functionality to your application as if it were implemented in Flask itself. Numerous extensions provide database integration, form validation, upload handling, various open authentication technologies, and more. Flask may be “micro,” but it is ready for production use on a variety of needs. Flask has many configuration values, with sensible defaults, and a few conventions when getting started. By convention, templates and static files are stored in subdirectories within the application’s Python source tree, with the name's templates and static, respectively. While this can be changed, you usually do not have to, especially when getting started.

Python crypto

Another python’s extensively maintained library is PyCrypto. It is an actively developed library that provides cryptographic recipes and primitives.

One application of the modules is writing secure administration tools. Another application is in writing daemons and servers. Clients and servers can encrypt the data being exchanged and mutually authenticate themselves; daemons can encrypt confidential data for added security. Python also provides a pleasant framework for prototyping and experimentation with cryptographic algorithms; thanks to its arbitrary-length integers, public key algorithms are easily implemented. It provides secure hash functions and various encryption algorithms. Hash functions played a vital role in the formation of the framework.

This library was also useful as it already had implementation of various symmetric- and asymmetric-key encryption algorithms like AES, blowfish, ARC2 and many more. Encryption algorithms transform plaintext in some way that is dependent on a key or key pair, producing ciphertext. We employed AES and blowfish (symmetric-key encryption) for encrypting the text. We performed a double layered encryption.

Amazon Web Services

Amazon Web Services (AWS) is a subsidiary of Amazon.com that provides on-demand cloud computing platforms to individuals, companies, and governments, on a paid subscription basis. The technology allows subscribers to have at their disposal a virtual cluster of computers, available all the time, through the Internet. AWS's version of virtual computers emulate most of the attributes of a real computer including hardware (CPU(s) & GPU(s) for processing, local/RAM memory, hard- disk/SSD storage); a choice of operating systems; networking; and pre-loaded application software such as web servers, databases, CRM, etc. Each AWS system also virtualizes its console I/O (keyboard, display, and mouse), allowing AWS subscribers to connect to their AWS system using a modern browser.[2]

Elastic Compute Cloud [EC2]

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers.

Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment. Amazon EC2 reduces the time required to obtain and boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change. Amazon EC2 changes the economics of computing by allowing you to pay only for capacity that you use. Amazon EC2 provides developers the tools to build failure resilient applications and isolate them from common failure scenarios. For compute, AWS' main offering is its EC2 instances, which can be tailored with many options. It also provides related services such as Elastic Beanstalk for app deployment, the EC2 Container service, AWS Lambda and Autoscaling.

In general terms, prices are comparable, especially since AWS shifter from by-the-hour to by-the-second pricing for its EC2 and EBS services in 2017. This economic pricing, reliable and secure infrastructure and vast online support enabled us to use Amazon to host and deploy the service on it is IaaS platform.

Implementation

This section describes the methodologies and implementation details of the targeted framework.

Planning and Analysis

The main task of this project was to provide as secure a file storage on the cloud as possible. So, several issues had to be sorted out like:

- Where should the file be encrypted?
- How should user must be verified?
- What AES encryption key look like?
- How will this key be related to the DH key Exchange?

We considered encrypting the text across the internet, but the attack man in the middle convinced us otherwise. We also learned about a new type of assault known as NFS. In an attack where a key of order 2768 could be computed, NFS was used. There were 232 digits in this number. As a result, we came to the conclusion that a higher prime number is required in the process. As a result, we chose a 600-digit prime number.

We came up with this plan of action after analyzing all of these questions. To provide the owner of the file more control, we used an application to encrypt the file on the owner's computer. Only people who have the final same key from this process will be able to decrypt the file, which was generated using the Diffie Hellman algorithm. The final key created by Diffie Hellman was used because it was the same for both intended participants, was used as the basis for the key for AES encryption.

The following sub-sections describe the various parts of the project in greater detail.

Stand-alone-application (Cloud-based Secure Text Transfer/src/stand-alone-application)

Stand-alone application was built to encrypt and decrypt the text using various symmetric encryption algorithms. It is a GUI based application which uses sender's key and receiver's key to encrypt and decrypt the text. This text is later uploaded on the online directory.

Diffie-Hellman (Cloud-based Secure Text Transfer/src/stand-alone-application/DH.py)

This section discussed the implementation of the Diffie-Hellman algorithm. The working and background details about the algorithm are already described in Section 3.0 of the report. This module had four methods to execute. The tasks were

Generate a private key of given length for a new user Generate a public key for a user using his private key Generate secret key based on a given public and private key

First, we needed a prime number. We already discussed the NFS attack in section 7.1 and need of a strong prime number in section 3.1 of the document. Thus, we hard coded the prime number

This prime number is 600 characters long, which is substantially larger and more resistant to NFS attacks. It can accommodate a large number of users; for example, a large number of users could be allocated a unique key that is smaller than this prime number. We also discovered that one of the prime number's primitive roots is 2, so we hard coded it and utilized it in all of the above functions.

Encryption (Cloud-based Secure Text Transfer/src/stand-alone-application/ENCDEC.py)

The implementation of the encryption algorithm used to secure the text coded into this file. We used the PyCrypto library, which already had the AES algorithm implemented.

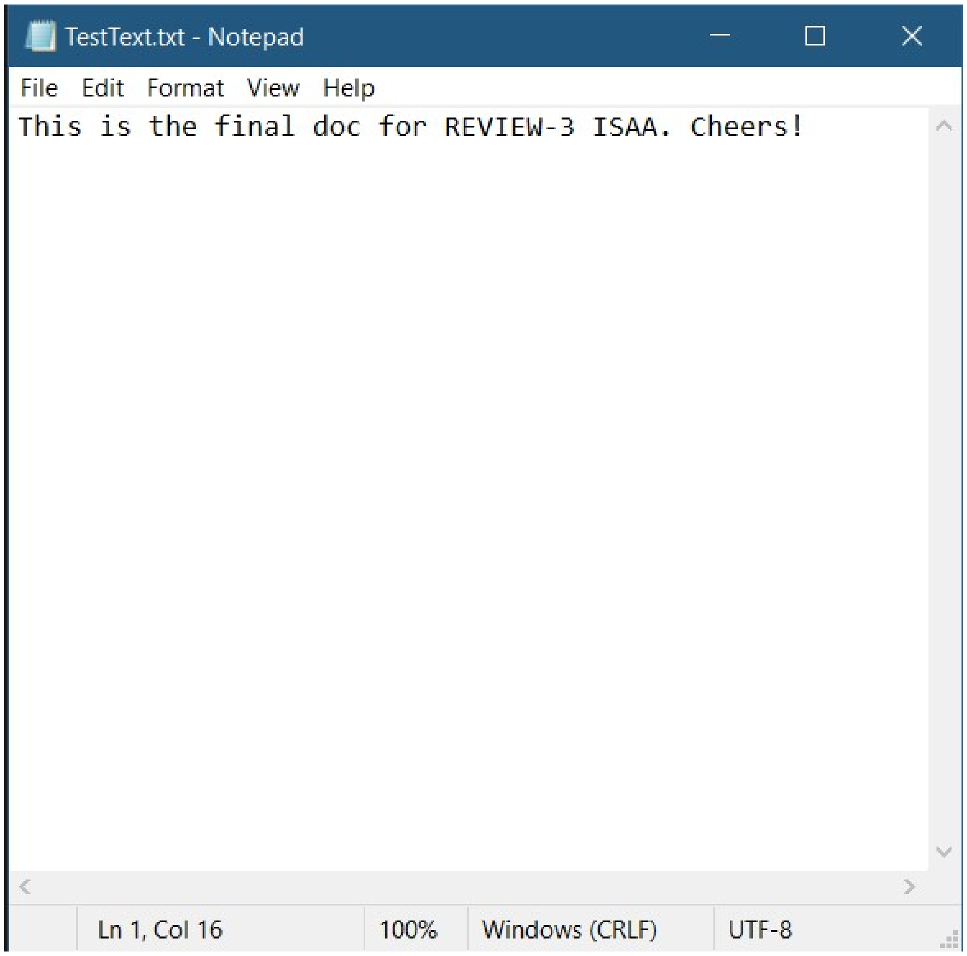
GUI (Cloud-based Secure Text Transfer/src/stand-alone-application/main.py)

The user's ability to run the application is improved by the GUI. Tkinter was used to create the user interface for our Cloud-based Secure Text Transfer application. This program provides a framework for encrypting, decrypting, uploading, and downloading encrypted data from an internet directory. It also includes a link that instructs the user on how to use the software. Note: main.py must only be run on the terminal.

Here, there are 2 flows

1. Sender: they select the plain-text file and enter the public key of receiver available on the cloud (see web-application section), enter their private key, available when the user registers (see web-application section).
    a. After clicking on Encrypt, and encoded file is automatically downloaded in the location you set.
    b. You can now upload this encrypted file on the cloud (see web-application section).
2. Receiver: They must download the encrypted text file from the cloud, select the file in the stand-alone application, enter their private key, enter the sender's public key (available on the server).
    a. After clicking on Decrypt, the decrypted file will be available in the set location by the name DecodeFile.txt.
    b. If the details entered were correct and the encryption was done using their public key the receiver must be able to view the plaintext as was intended by the sender.

1. Plaintext file



TestText.txt - Notepad

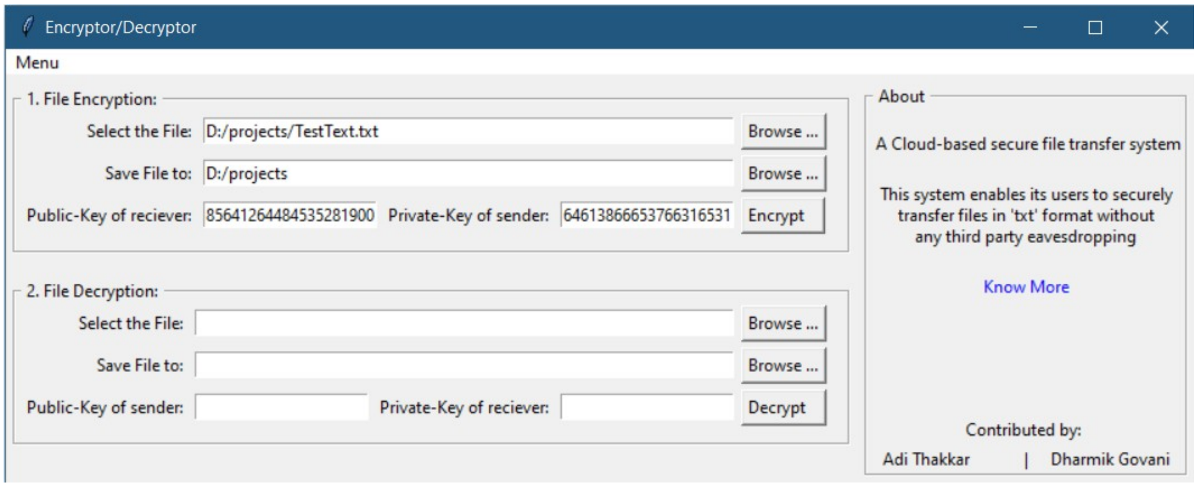
File Edit Format View Help

```
This is the final doc for REVIEW-3 ISAA. Cheers!
```

Ln 1, Col 16 | 100% | Windows (CRLF) | UTF-8

The plaintext file as intended by the sender.

2. Encryption process



Encryptor/Decryptor

Menu

1. File Encryption:

Select the File: D:/projects/TestText.txt

Save File to: D:/projects

Public-Key of reciever: 85641264484535281900 Private-Key of sender: 64613866653766316531

2. File Decryption:

Select the File:

Save File to:

Public-Key of sender: Private-Key of reciever:

About

A Cloud-based secure file transfer system

This system enables its users to securely transfer files in 'txt' format without any third party eavesdropping

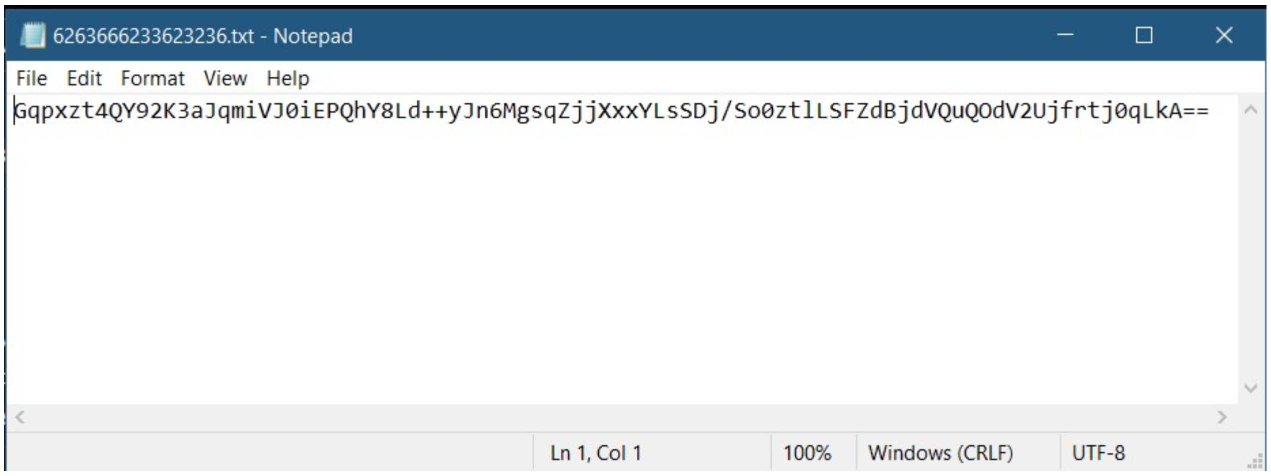
Know More

Contributed by:

Adi Thakkar | Dharmik Govani

Encryption of the plaintext file by the sender.

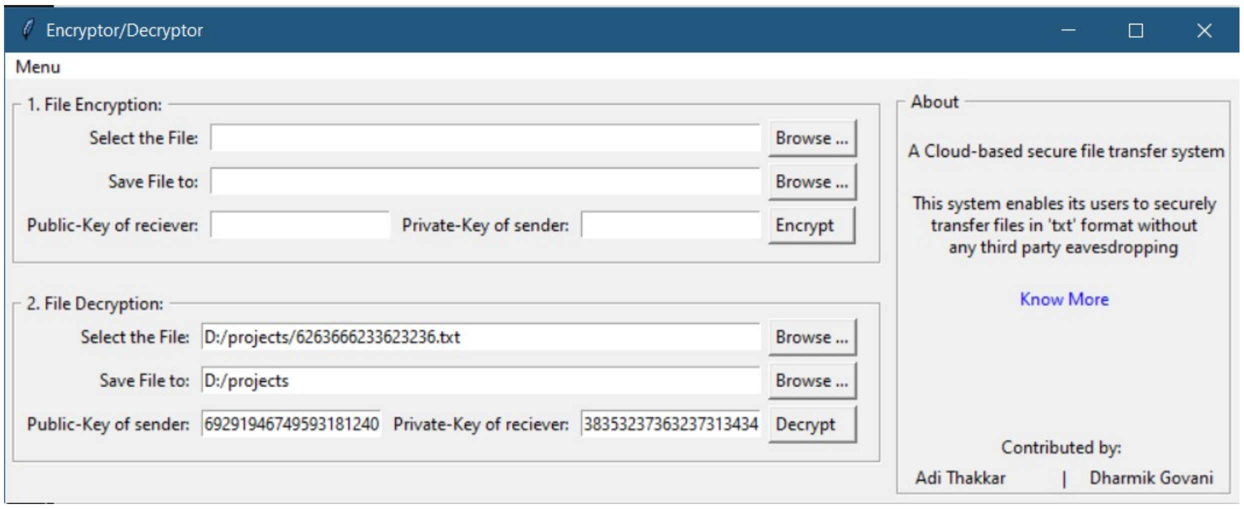
3. Encrypted file



```
Gqpxzt4QY92K3aJqmiVJ0iEPQhY8Ld++yJn6MgsqZjjXxxYLsSDj/So0ztlLSFZdBjdVQuQOdV2Ujfrtj0qLkA==
```

The encrypted file after encryption in the stand-alone-application, which is to be uploaded on the cloud (see web-application section).

4. Decryption process



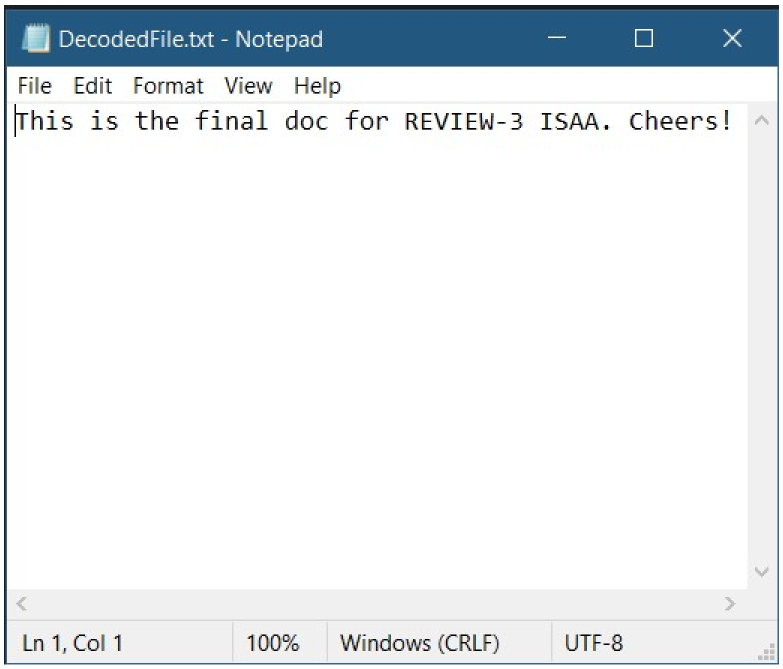
1. File Encryption:
Select the File:
Save File to:
Public-Key of reciever: Private-Key of sender:

2. File Decryption:
Select the File: D:/projects/6263666233623236.txt
Save File to: D:/projects
Public-Key of sender: 69291946749593181240 Private-Key of reciever: 38353237363237313434

About
A Cloud-based secure file transfer system
This system enables its users to securely transfer files in 'txt' format without any third party eavesdropping
Contributed by:
Adi Thakkar | Dharmik Govani

The Receiver downloads the encrypted file (see web-application section) and performs decryption on the file.

5. Decrypted File



```
This is the final doc for REVIEW-3 ISAA. Cheers!
```

This is the decrypted file. The communication is complete.

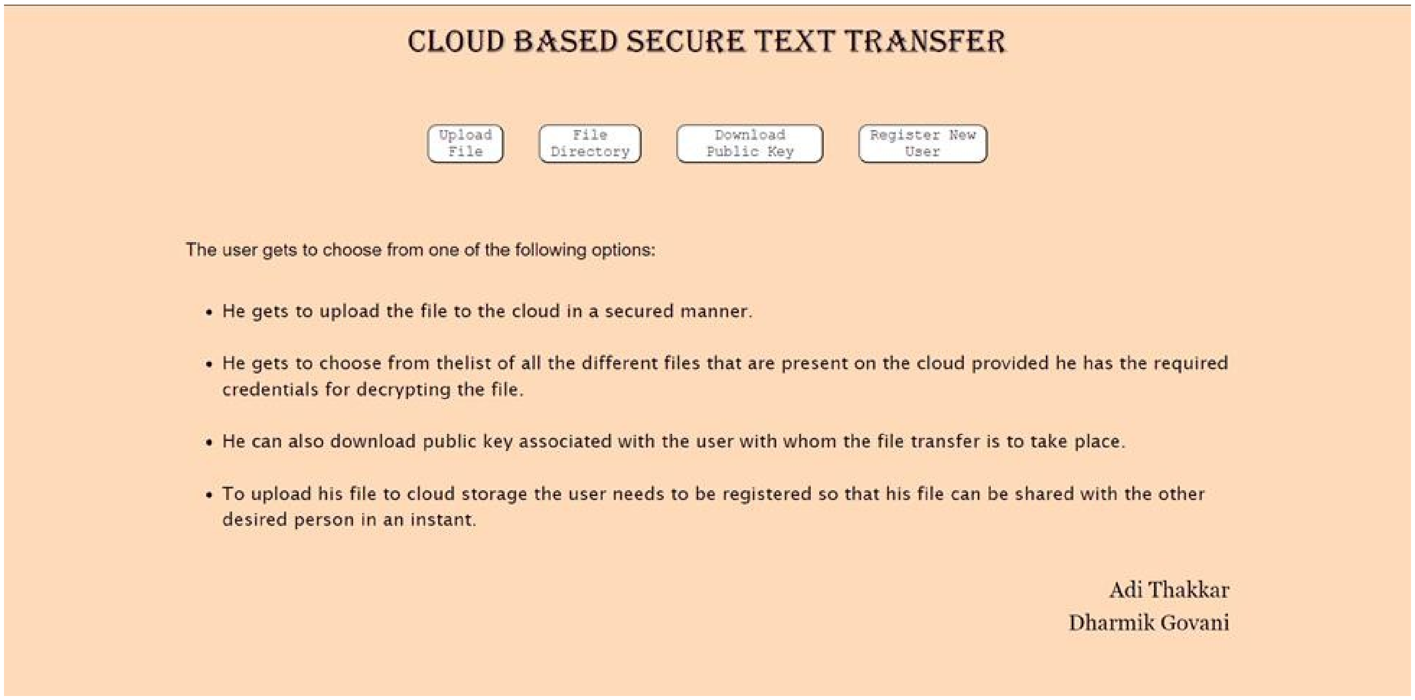
Web-application

The web application of the project was used for secure storage on the cloud. The major steps included in the web application are as follows:

- The first step for the user was to register on our platform. On registering the user would be given an automatically generated private key that would be used by the user for transactions. For security purposes the private key of the user is not stored in the database.
- On selecting the upload file option, the user is taken to another page that allows the user to submit the encrypted file.
- Clicking the file directory option on the page takes the user to the above page which displays all the different files stored on the cloud.
- Selecting different options like download-public key lists the public keys of the registered users which the user can download and use for authenticating the user when someone tries to open his uploaded file.
- Clicking on register user tab take the user to a page where the user registers himself with the Platform.

Images:-

1. Home

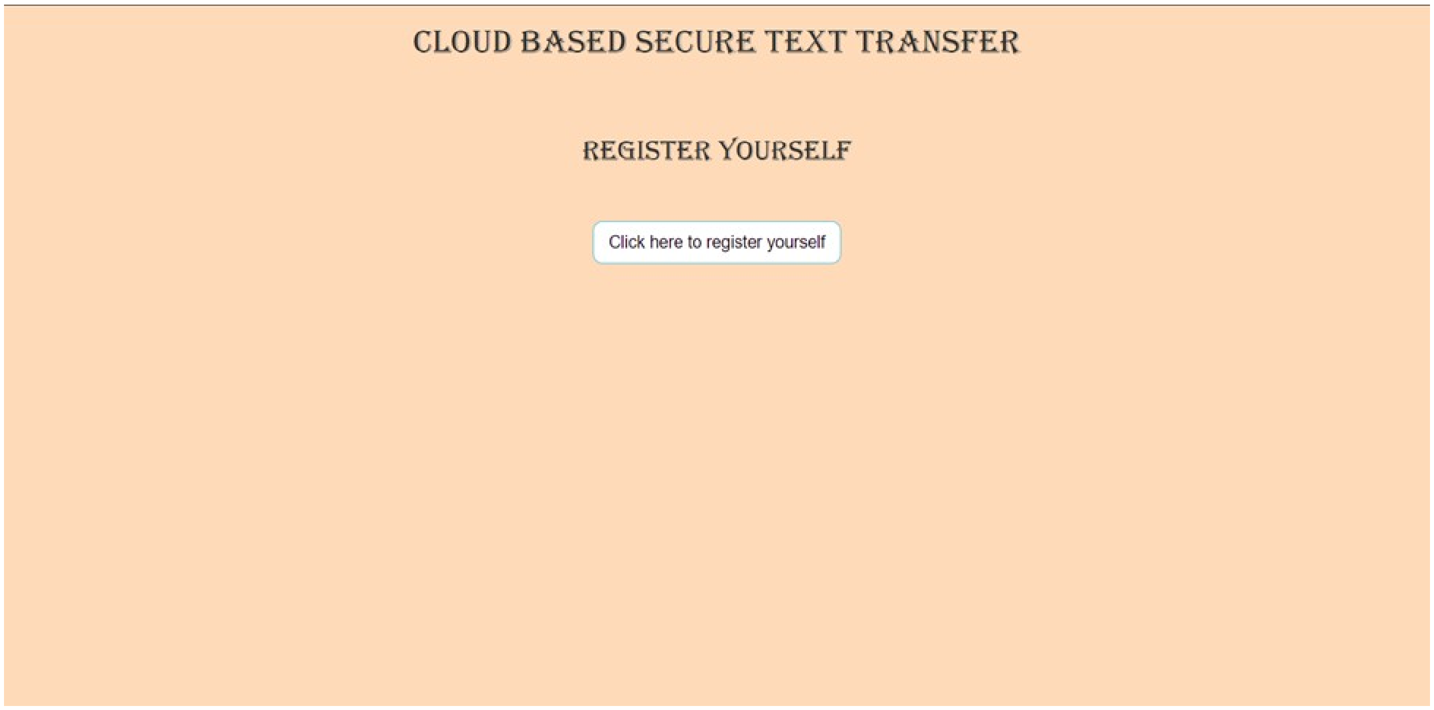

CLOUD BASED SECURE TEXT TRANSFER

The user gets to choose from one of the following options:

- He gets to upload the file to the cloud in a secured manner.
- He gets to choose from thelist of all the different files that are present on the cloud provided he has the required credentials for decrypting the file.
- He can also download public key associated with the user with whom the file transfer is to take place.
- To upload his file to cloud storage the user needs to be registered so that his file can be shared with the other desired person in an instant.

Adi Thakkar
Dharmik Govani

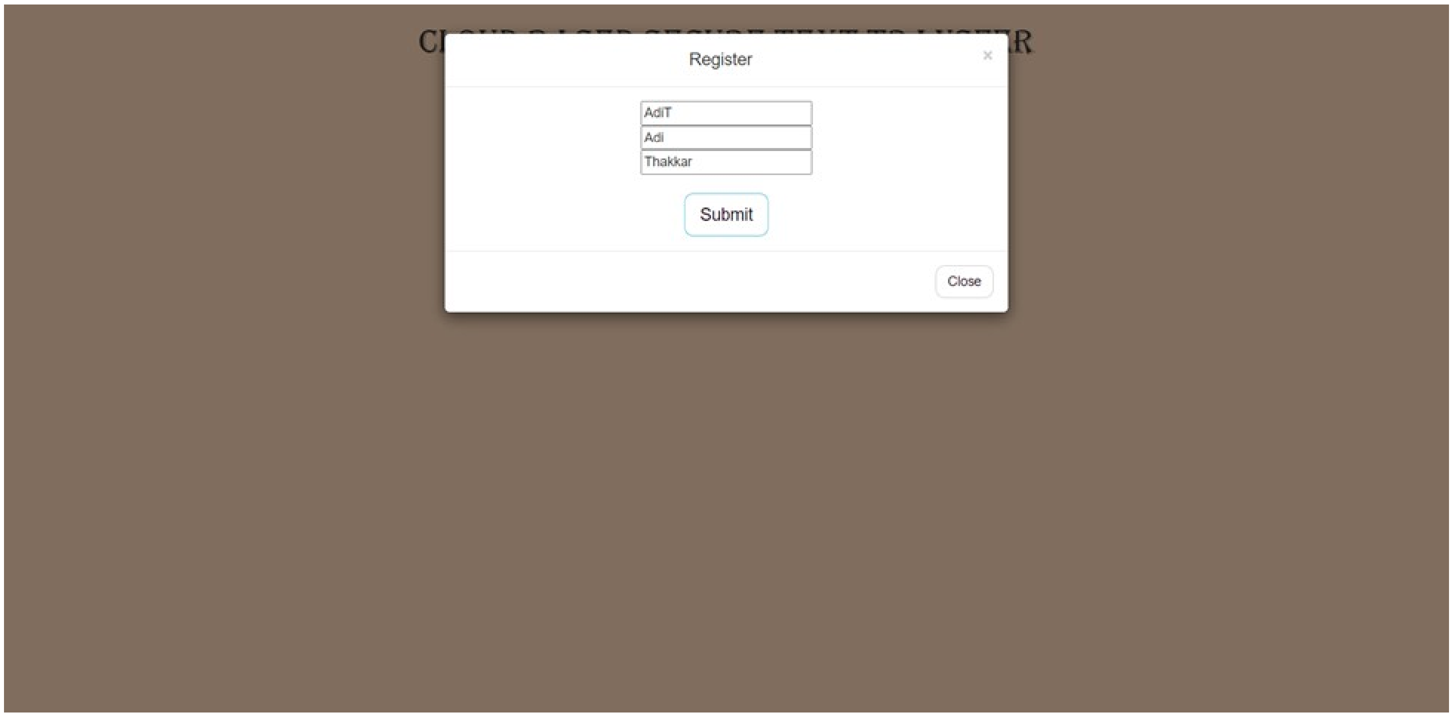
2. Register page


CLOUD BASED SECURE TEXT TRANSFER

REGISTER YOURSELF

The user first must register themselves, for the application to generate a public key-private key pair. Which enables all the other use-cases in the project

Register modal


Register

×

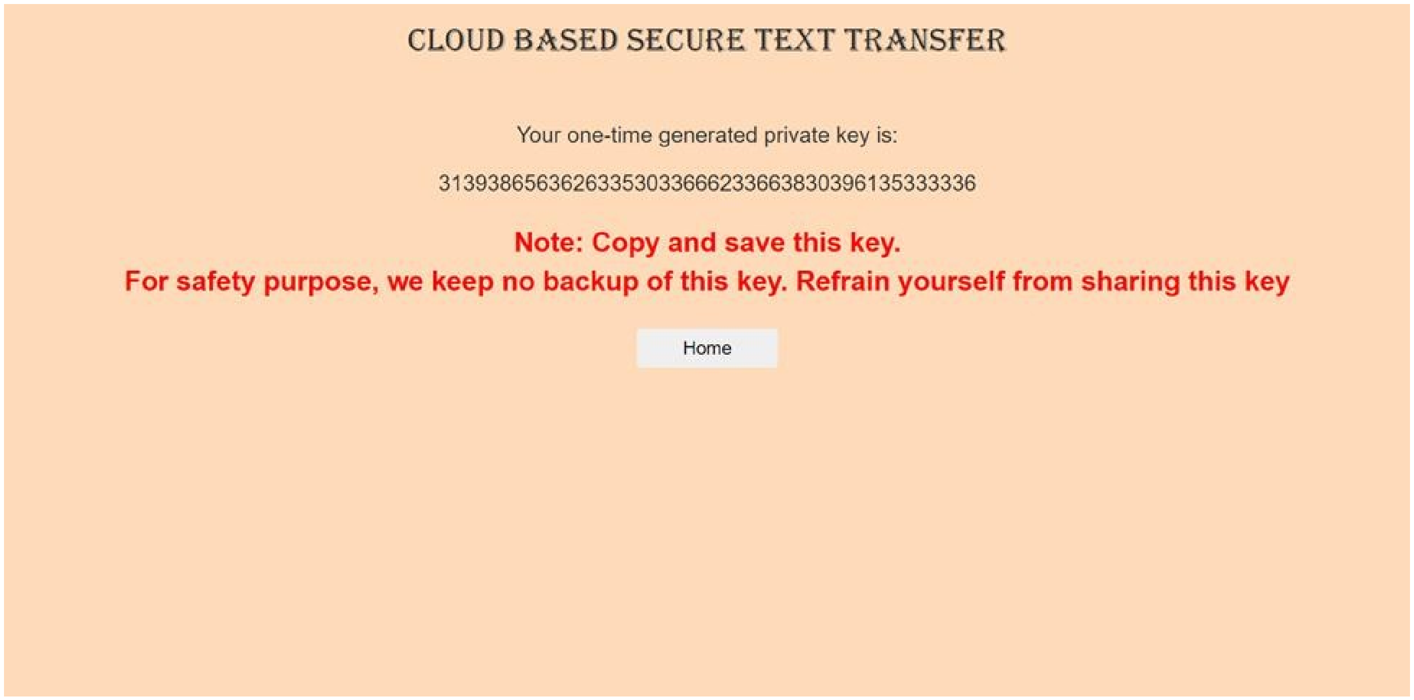
AdiT

Adi

Thakkar

The username is unique. Hence can only be used by one user.

3. Post- Register page


CLOUD BASED SECURE TEXT TRANSFER

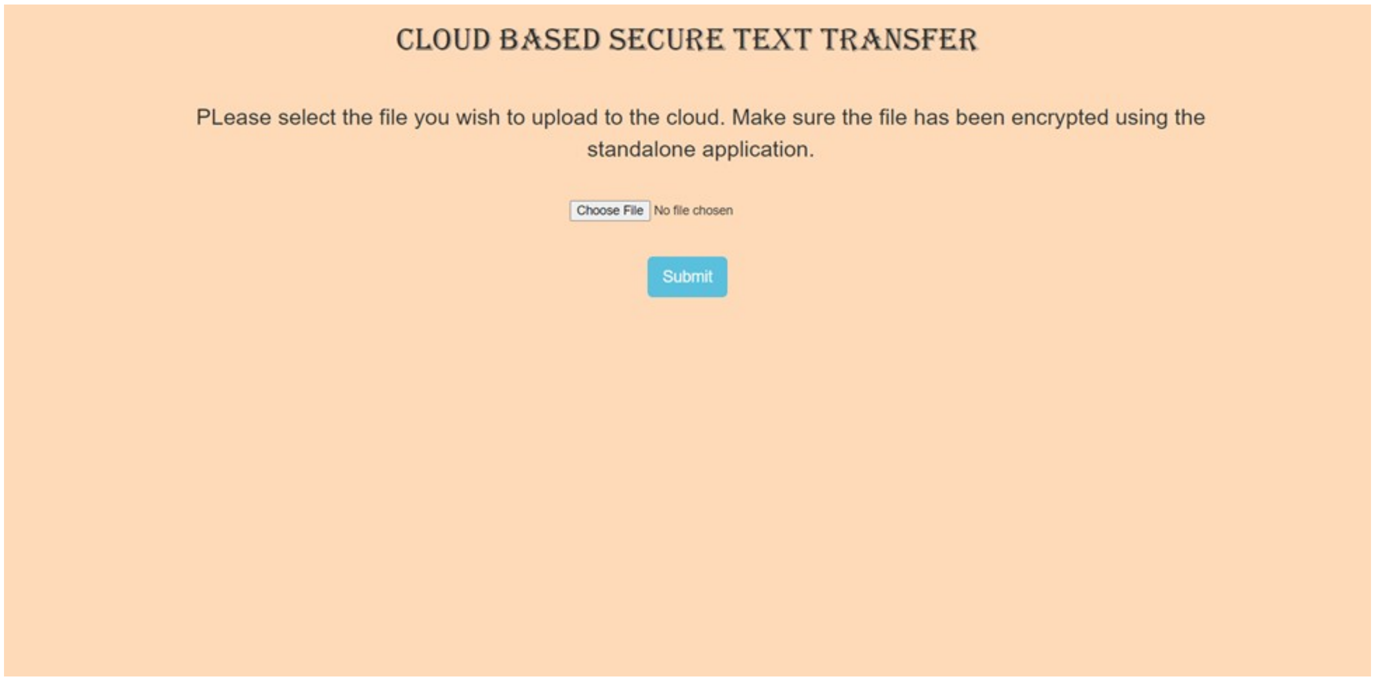
Your one-time generated private key is:

3139386563626335303366623366383039613533336

Note: Copy and save this key.
For safety purpose, we keep no backup of this key. Refrain yourself from sharing this key

The system returns your private key, the user must store it and keep it private because this private key will not be available later.

4. Upload file to cloud

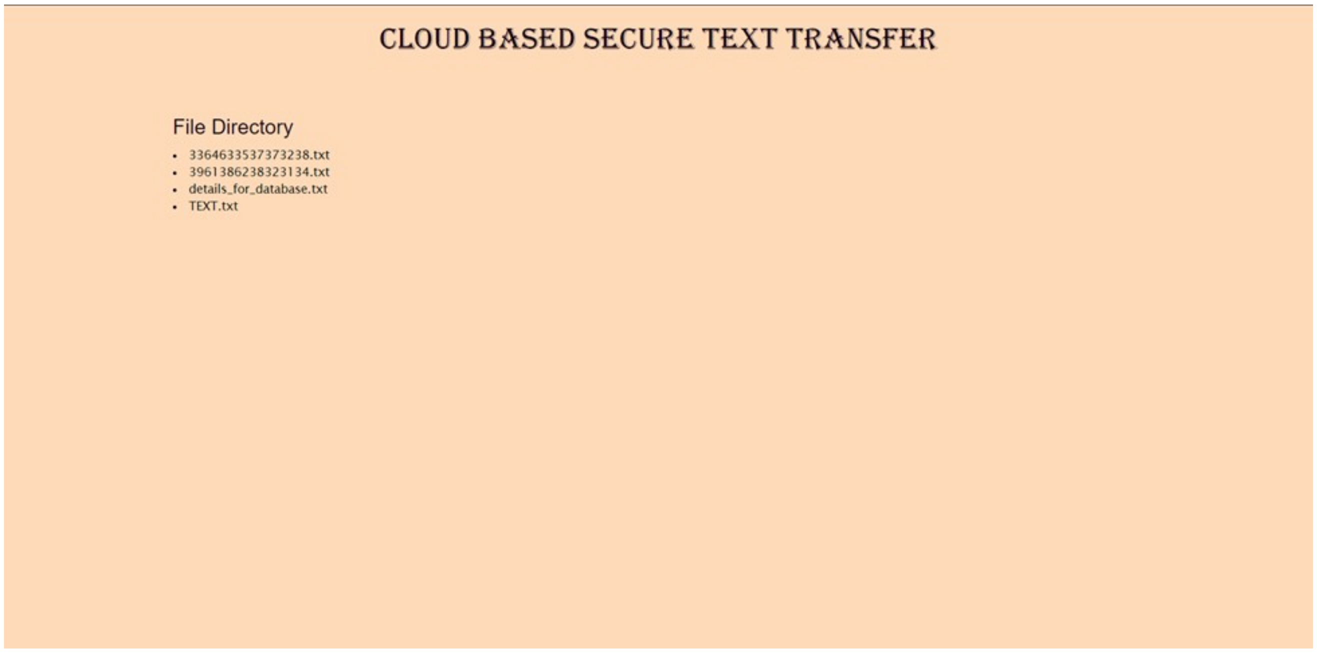


CLOUD BASED SECURE TEXT TRANSFER

PLease select the file you wish to upload to the cloud. Make sure the file has been encrypted using the standalone application.

The user than must select the .text file that they want to securely share over the cloud. (Note: The encrypted file must be uploaded here, NOT the plaintext file). Encryption can be done as discussed in the stand-alone-application.

5. Uploaded files directory



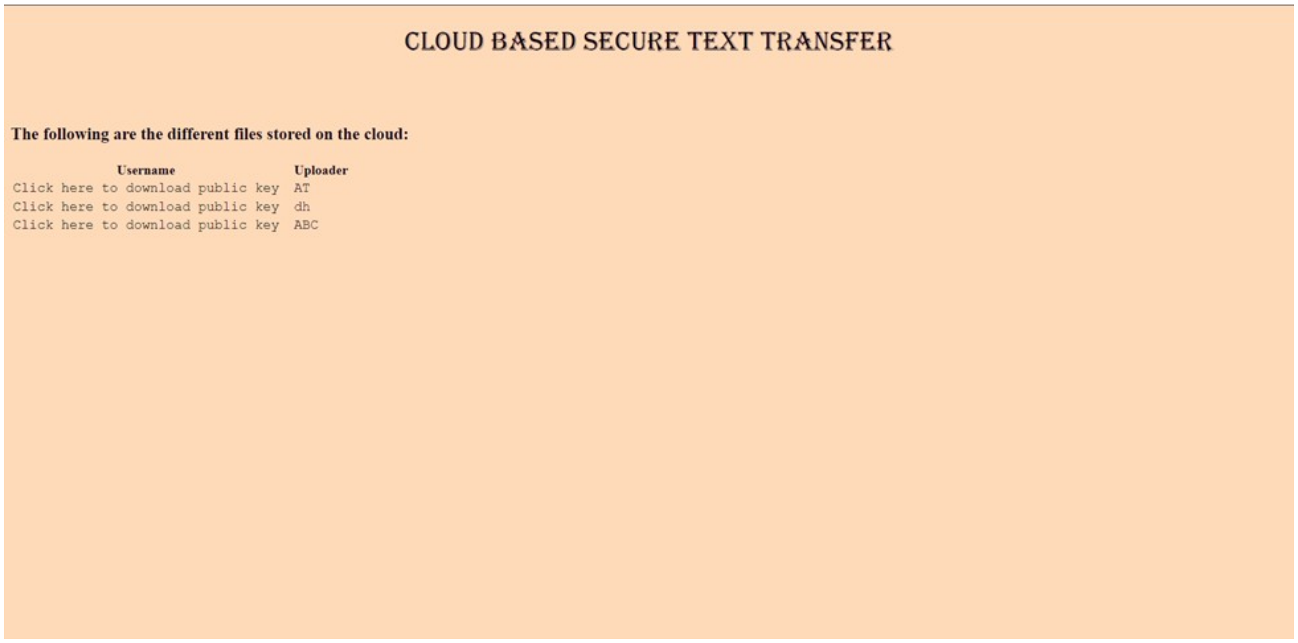
CLOUD BASED SECURE TEXT TRANSFER

File Directory

- 3364633537373238.txt
- 3961386238323134.txt
- details_for_database.txt
- TEXT.txt

After Encrypted file is uploaded, all the users can view it in the file directory. But even if they try to access it, it will not make sense because of the encryption. Now the receiver (assuming they have already registered) can download the encrypted file that was meant to be communicated.

6. List of Public keys of various users



CLOUD BASED SECURE TEXT TRANSFER

The following are the different files stored on the cloud:

Username	Uploader
Click here to download public key	AT
Click here to download public key	dh
Click here to download public key	ABC

This is the list of public keys of various users, this has its use when running the stand-alone-application, when performing encryption or decryption.

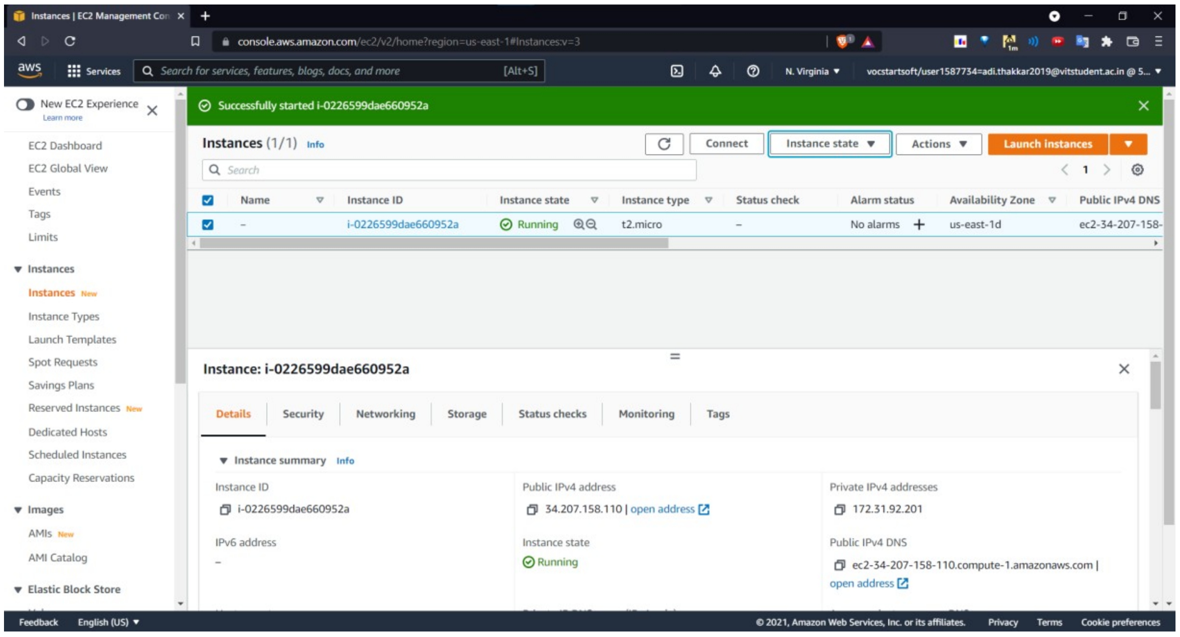
Hosting on Amazon AWS-EC2

The following steps are to be performed to host the machine on AWS cloud.

1. First log in to the AWS account and go to EC2 console and select a machine to deploy.
2. Then configure the machines ports as displayed above.
3. When the machine is up and running copy the public IP address of the machine that will be used further.
4. Now open the Putty Key Generator and used the downloaded file to produce private key to be used for connection. Save the private key file.
5. Copy and paste the public IP address in Putty Configuration software and then go to SSH and give the path of the private key file and then click open. This will open the terminal of the remote machine. Once the terminal is open log into the machine.
6. Once you are logged in then install the dependencies like pip, flask, and then host the application on the remote machine.
7. Once the application is running use the public Ip address of the machine to access the application from anywhere.

Images:-

AWS console


Successfully started i-0226599dae660952a

Instances (1/1)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
–	i-0226599dae660952a	Running	t2.micro	–	No alarms	us-east-1d	ec2-34-207-158-

Instance: i-0226599dae660952a

Instance summary

Instance ID: i-0226599dae660952a

Public IPv4 address: 34.207.158.110 | open address

Private IPv4 addresses: 172.31.92.201

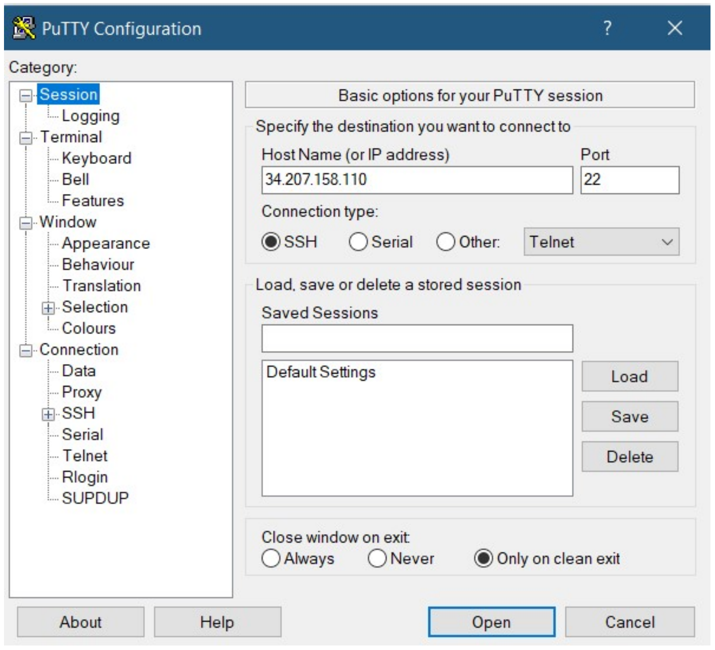
IPv6 address: –

Instance state: Running

Public IPv4 DNS: ec2-34-207-158-110.compute-1.amazonaws.com | open address

Here we launch the ec2 instance, and we get the public IP address for the hosted terminal.

1. Putty Config


PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): 34.207.158.110

Port: 22

Connection type: [✓ SSH] Serial Other: Telnet

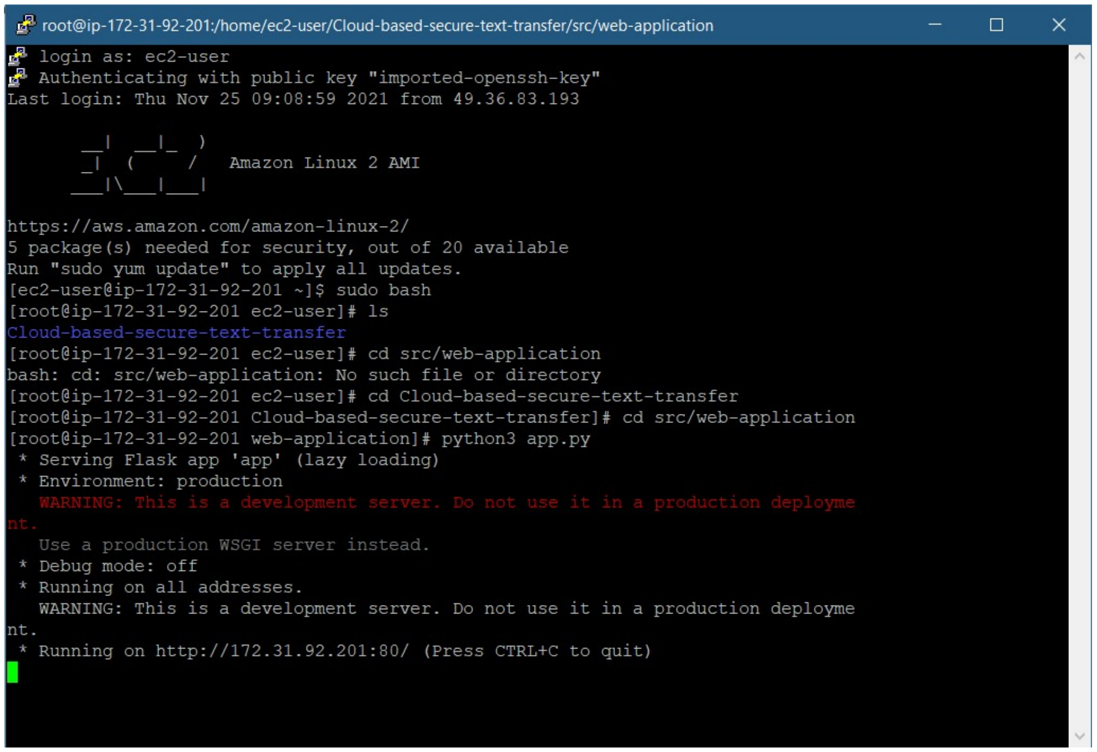
Load, save or delete a stored session

Saved Sessions: Default Settings

Close window on exit: Always Never [✓ Only on clean exit]

Putty is used to login to the AWS terminal via the private-key pair generated and the IP address of the hosted instance.

2. EC2 instance terminal



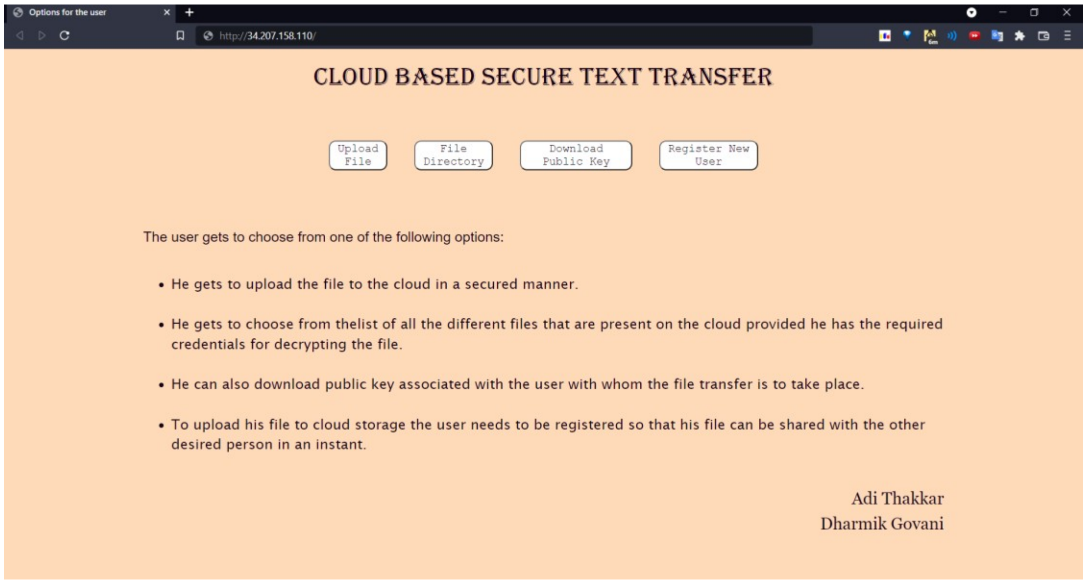
```
root@ip-172-31-92-201:/home/ec2-user/Cloud-based-secure-text-transfer/src/web-application
login as: ec2-user
Authenticating with public key "imported-openssh-key"
Last login: Thu Nov 25 09:08:59 2021 from 49.36.83.193

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
5 package(s) needed for security, out of 20 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-92-201 ~]$ sudo bash
[root@ip-172-31-92-201 ec2-user]# ls
Cloud-based-secure-text-transfer
[root@ip-172-31-92-201 ec2-user]# cd src/web-application
bash: cd: src/web-application: No such file or directory
[root@ip-172-31-92-201 ec2-user]# cd Cloud-based-secure-text-transfer
[root@ip-172-31-92-201 Cloud-based-secure-text-transfer]# cd src/web-application
[root@ip-172-31-92-201 web-application]# python3 app.py
 * Serving Flask app 'app' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a production deployme
nt.
 * Running on http://172.31.92.201:80/ (Press CTRL+C to quit)
```

Here we login to the terminal using the username as ‘ec2-user’, Here we clone the github repository, install dependencies and run the app.py file in thw web-application folder.

3. The hosted website



CLOUD BASED SECURE TEXT TRANSFER

The user gets to choose from one of the following options:

- He gets to upload the file to the cloud in a secured manner.
- He gets to choose from thelist of all the different files that are present on the cloud provided he has the required credentials for decrypting the file.
- He can also download public key associated with the user with whom the file transfer is to take place.
- To upload his file to cloud storage the user needs to be registered so that his file can be shared with the other desired person in an instant.

Adi Thakkar
Dharmik Govani

The Web-application is now accessible as soon as we type in the public IP address of the ec2 instance in our browser.
This can be accessed from any browser, (not local). Now the communication can begin.

After these instructions are done, any user can now access this code, by putting the IP address of the machine (AWS) in the browser. And hence Cloud-based-Secure-Text-Transfer can now take place.

CONCLUSION:

The goal of the proposed project is to solve the problem of safe cloud file storage. This approach is a simple implementation of the recommended methodology that can be tweaked and adjusted to meet specific requirements. It proposes to use encryption and Diffie Hellman to provide double layer of security to the files that are stored on the cloud.

Github Repository link- https://github.com/AT-LEGEND/Cloud-based-secure-text-transfer

REFERENCES:

https://www.trendmicro.com/content/dam/trendmicro/global/en/migrated/security-intelligence-migration-spreadsheet/trendlabs-security-intelligence/2015/09/anglerek_dh_01.jpg

https://i.stack.imgur.com/AEx0X.png

https://www.academia.edu/download/36607691/049.pdf

https://www.tandfonline.com/doi/abs/10.1080/09720529.2019.1569821

https://www.academia.edu/download/64601037/Kishore%20Karanam%20-%20V8

https://link.springer.com/chapter/10.1007/978-981-15-4032-5_14I6-0010.pdf

https://ieeexplore.ieee.org/abstract/document/6524434/

https://www.iasj.net/iasj/download/edc53e687e7742d6

https://www.researchgate.net/profile/Phyu-Thwe-2/publication/342672410_Prevention_of_Man-In-The-Middle_Attack_in_Diffie-Hellman_Key_Exchange_Algorithm_using_Proposed_Hash_Function/links/5efffc98299bf1881600486f/Prevention-of-Man-In-The-Middle-Attack-in-Diffie-Hellman-Key-Exchange-Algorithm-using-Proposed-Hash-Function.pdf

https://link.springer.com/chapter/10.1007/978-3-319-22174-8_3

https://www.tandfonline.com/doi/abs/10.1080/01611190701593228

https://link.springer.com/chapter/10.1007/3-540-68697-5_11

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html

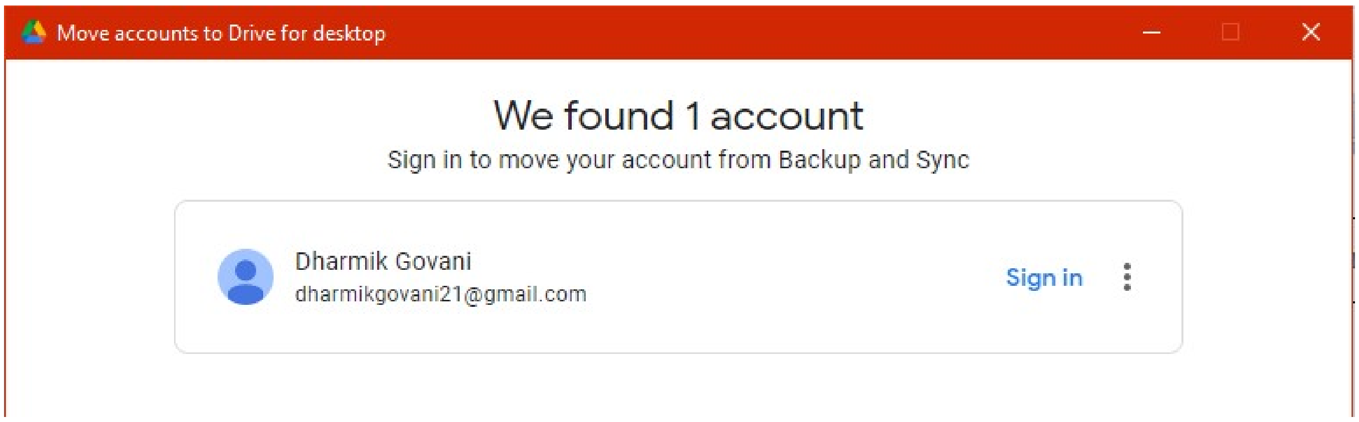
Audit Report:

1. RISK ASSESSMENT:

This is a Low Security System and is prone to any new external attacks

2. BACKUP PROCEDURE:

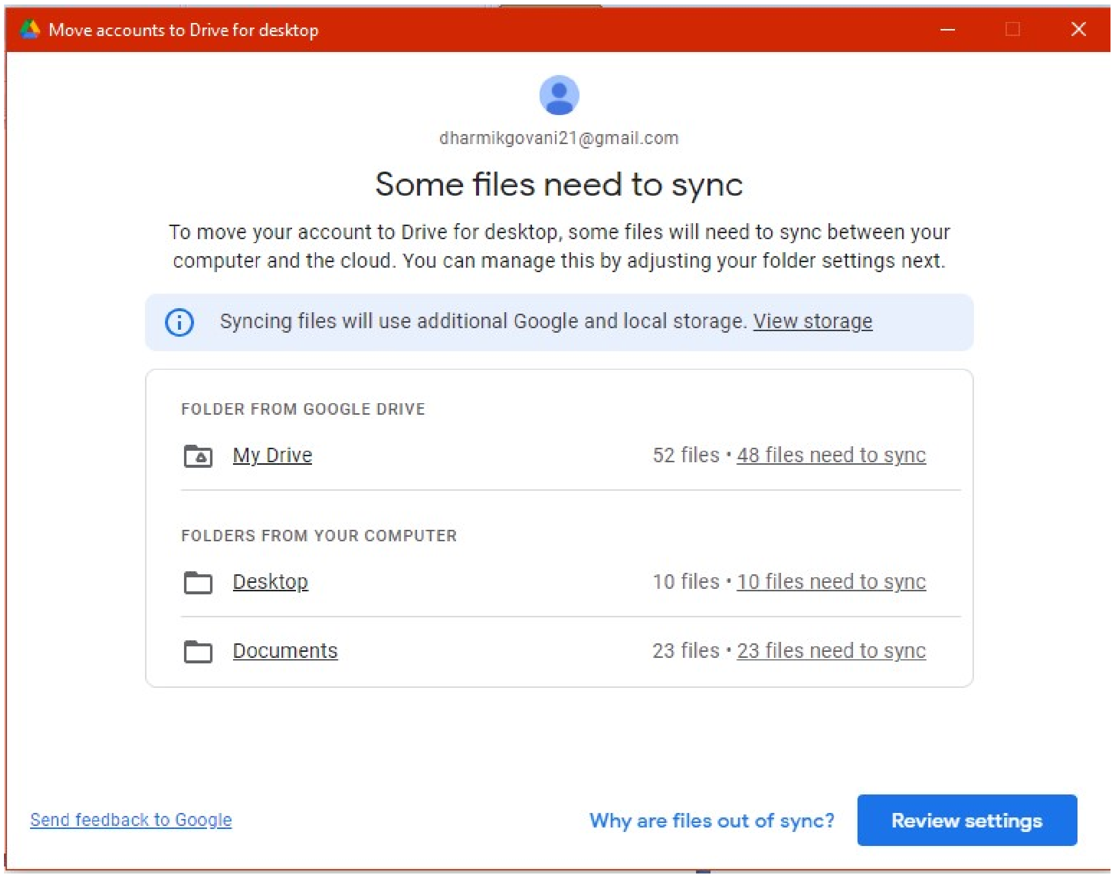
Google Drive takes backup of important files and media stored on admin laptop



We found 1 account

Sign in to move your account from Backup and Sync

Dharmik Govani
dharmikgovani21@gmail.com



Move accounts to Drive for desktop

dharmikgovani21@gmail.com

Some files need to sync

To move your account to Drive for desktop, some files will need to sync between your computer and the cloud. You can manage this by adjusting your folder settings next.

Syncing files will use additional Google and local storage. View storage

FOLDER FROM GOOGLE DRIVE

My Drive 52 files • 48 files need to sync

FOLDERS FROM YOUR COMPUTER

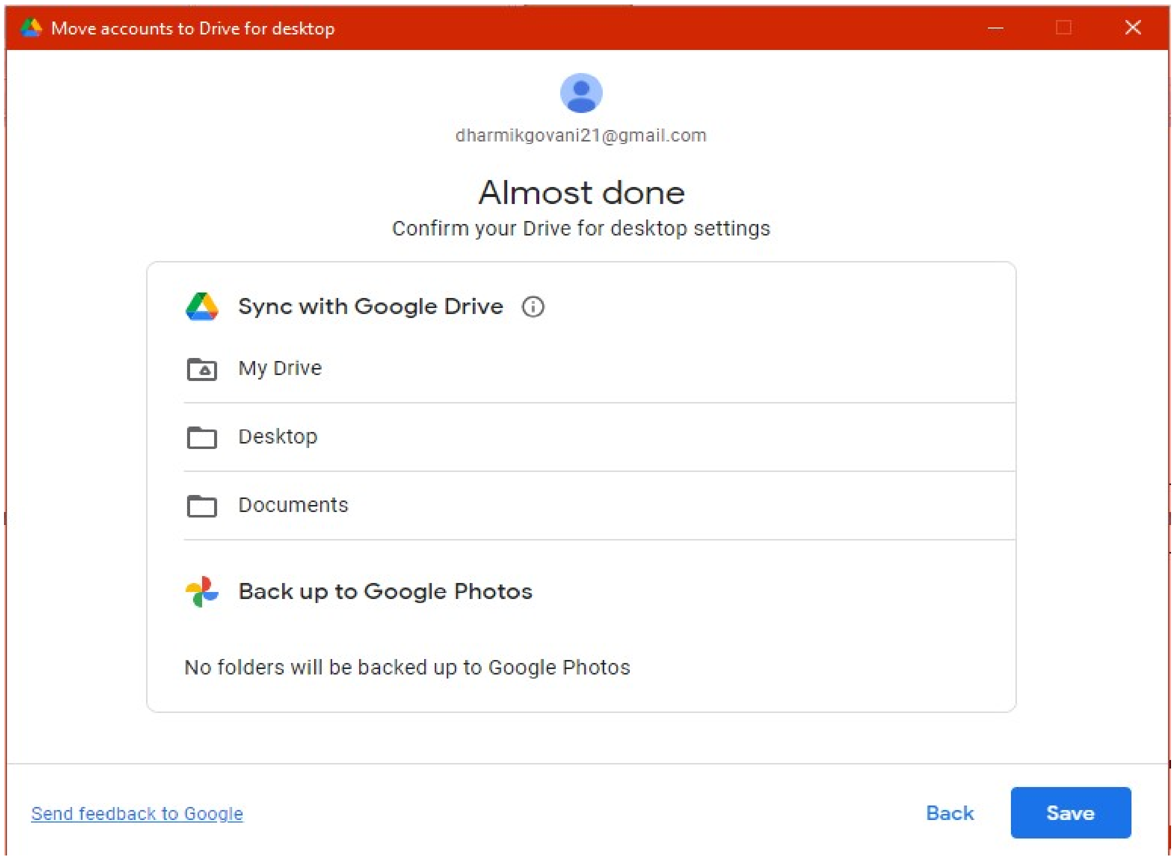
Desktop 10 files • 10 files need to sync

Documents 23 files • 23 files need to sync

Send feedback to Google

Why are files out of sync?

Review settings



Move accounts to Drive for desktop

dharmikgovani21@gmail.com

Almost done

Confirm your Drive for desktop settings

Sync with Google Drive

My Drive

Desktop

Documents

Back up to Google Photos

No folders will be backed up to Google Photos

Send feedback to Google

Back

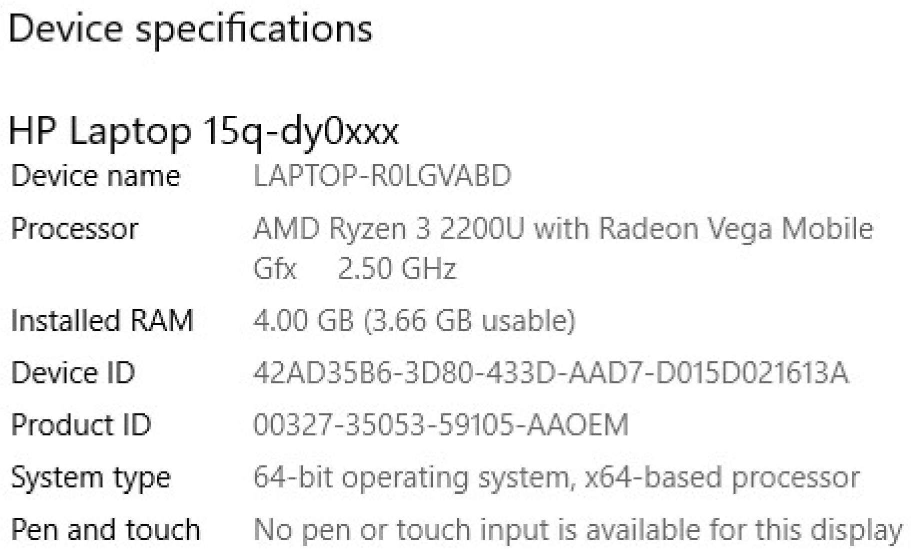
Save

3. PASSWORD POLICY SECTION

Password has been setup on the admin laptop. It follows all the criteria for a strong password:

- 1 Uppercase Letter
- 1 Lowercase Letter
- Minimum length: 8
- One number and one special character

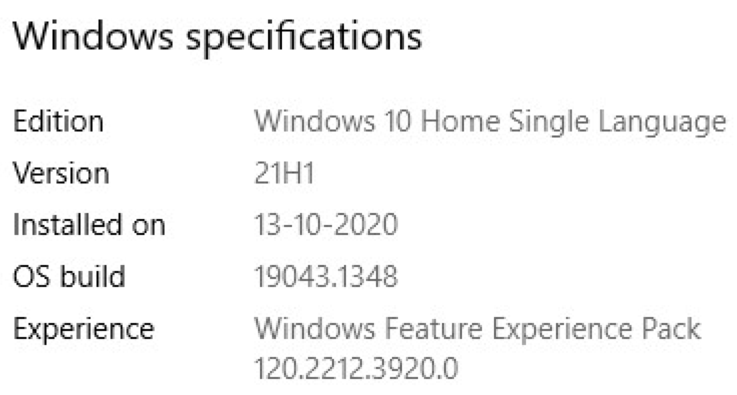
4. CHANGE CONTROL SECTION



Device specifications

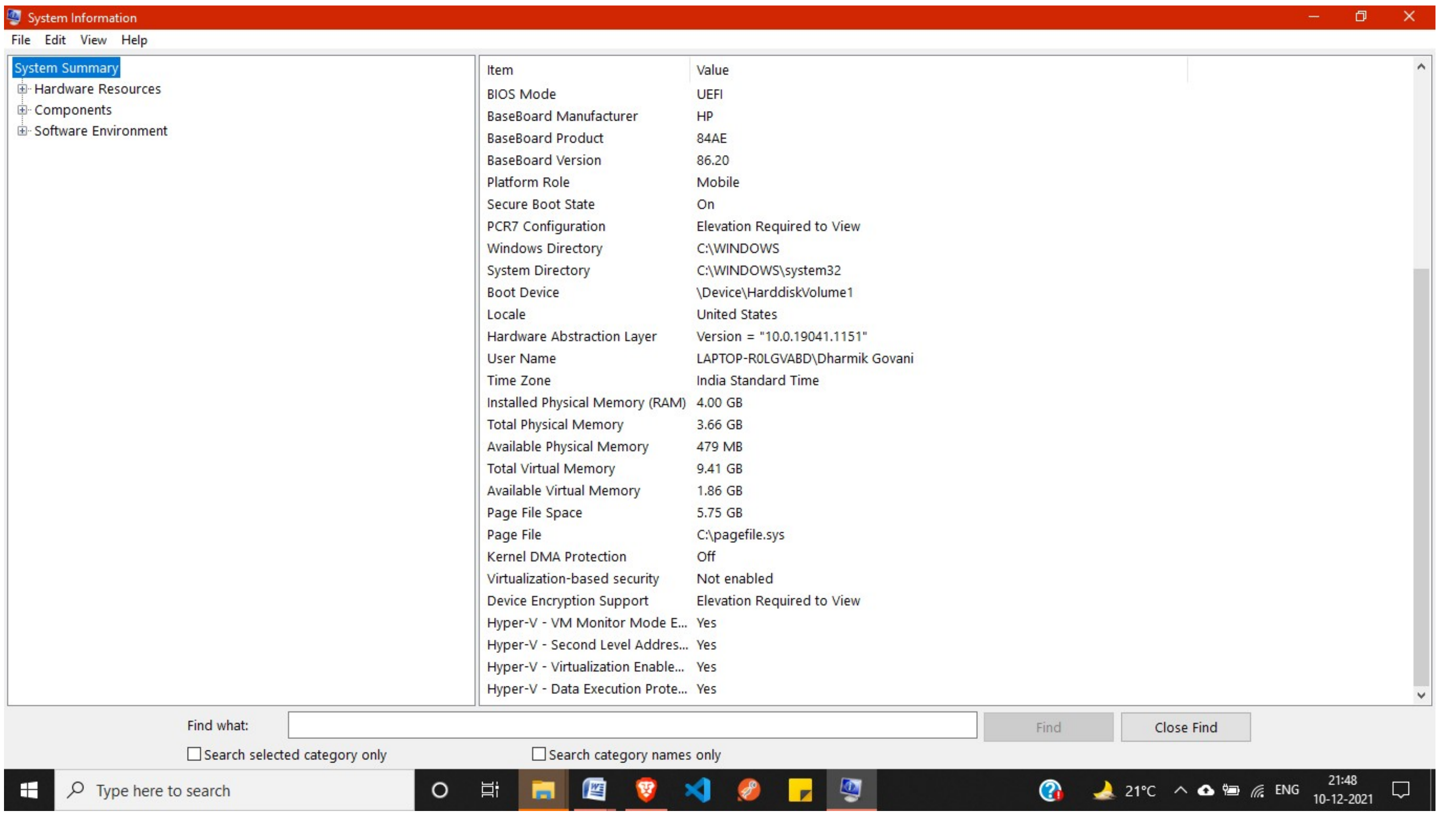
HP Laptop 15q-dy0xxx

Device name	LAPTOP-R0LGVABD
Processor	AMD Ryzen 3 2200U with Radeon Vega Mobile Gfx 2.50 GHz
Installed RAM	4.00 GB (3.66 GB usable)
Device ID	42AD35B6-3D80-433D-AAD7-D015D021613A
Product ID	00327-35053-59105-AAOEM
System type	64-bit operating system, x64-based processor
Pen and touch	No pen or touch input is available for this display



Windows specifications

Edition	Windows 10 Home Single Language
Version	21H1
Installed on	13-10-2020
OS build	19043.1348
Experience	Windows Feature Experience Pack 120.2212.3920.0



Item	Value
BIOS Mode	UEFI
BaseBoard Manufacturer	HP
BaseBoard Product	84AE
BaseBoard Version	86.20
Platform Role	Mobile
Secure Boot State	On
PCR7 Configuration	Elevation Required to View
Windows Directory	C:\WINDOWS
System Directory	C:\WINDOWS\system32
Boot Device	\Device\HarddiskVolume1
Locale	United States
Hardware Abstraction Layer	Version = "10.0.19041.1151"
User Name	LAPTOP-R0LGVABD\Dharmik Govani
Time Zone	India Standard Time
Installed Physical Memory (RAM)	4.00 GB
Total Physical Memory	3.66 GB
Available Physical Memory	479 MB
Total Virtual Memory	9.41 GB
Available Virtual Memory	1.86 GB
Page File Space	5.75 GB
Page File	C:\pagefile.sys
Kernel DMA Protection	Off
Virtualization-based security	Not enabled
Device Encryption Support	Elevation Required to View
Hyper-V - VM Monitor Mode E...	Yes
Hyper-V - Second Level Addres...	Yes
Hyper-V - Virtualization Enable...	Yes
Hyper-V - Data Execution Prote...	Yes

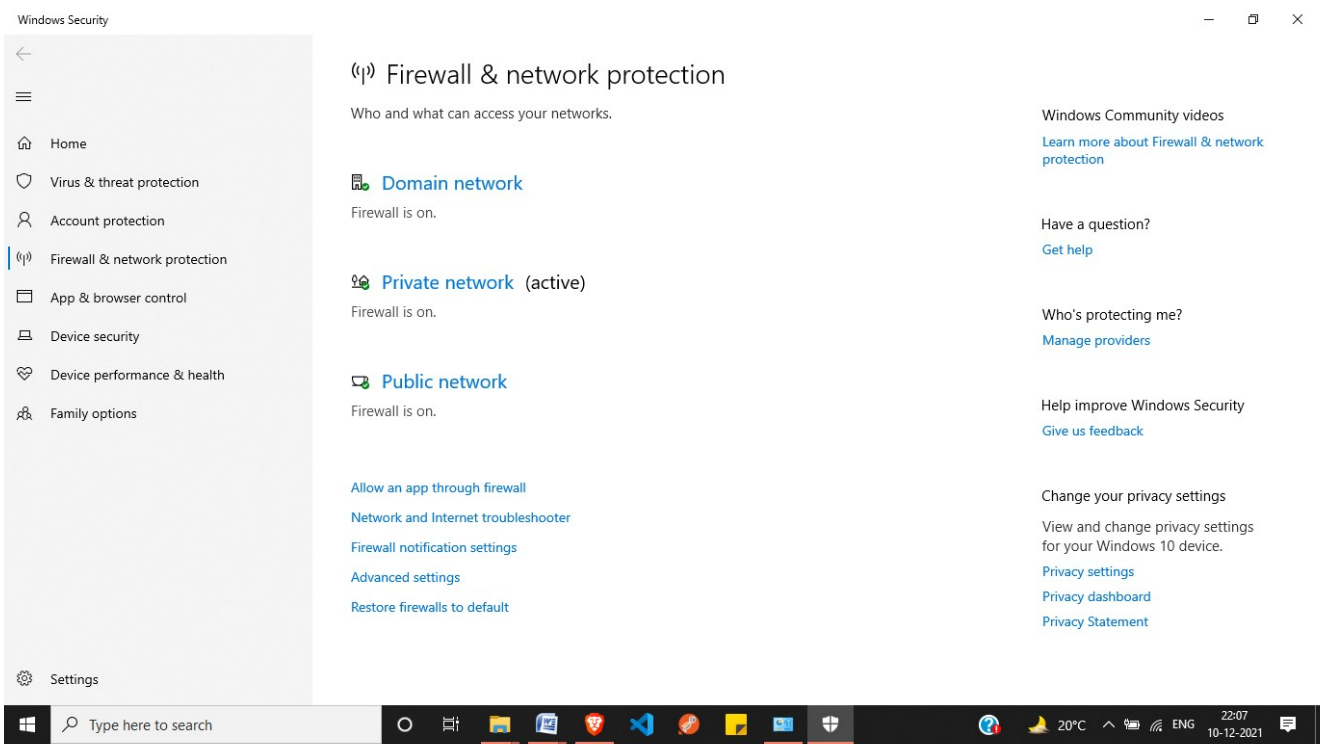
5. INCIDENT RESPONSE SECTION

Security Posture: NA

6. REPORT

I. Network Security Audit:

This system is protected by Windows Defender Firewall that is capable of protecting against various network security threats. The Firewall always remains active and is upgrading regularly.



Windows Security

Firewall & network protection

Who and what can access your networks.

Domain network
Firewall is on.

Private network (active)
Firewall is on.

Public network
Firewall is on.

Allow an app through firewall
Network and Internet troubleshooter
Firewall notification settings
Advanced settings
Restore firewalls to default

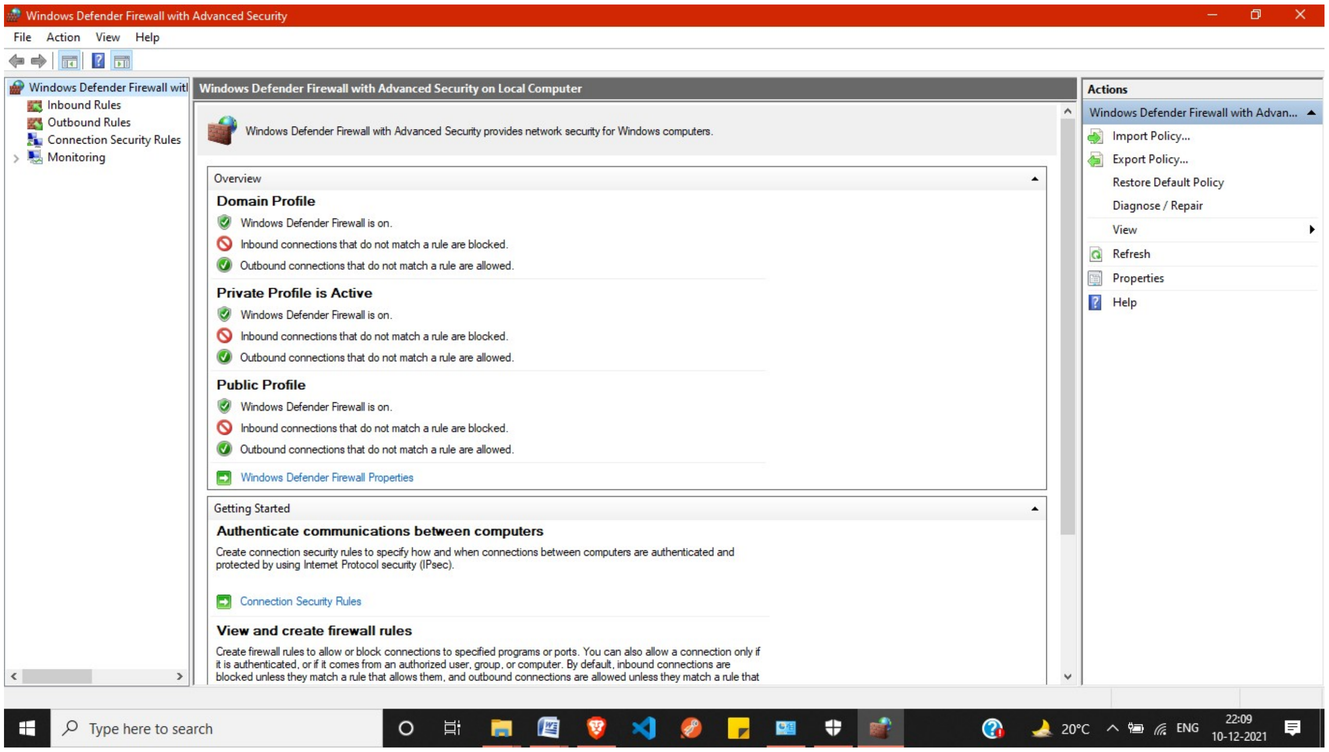
Windows Community videos
Learn more about Firewall & network protection

Have a question?
Get help

Who's protecting me?
Manage providers

Help improve Windows Security
Give us feedback

Change your privacy settings
View and change privacy settings for your Windows 10 device.
Privacy settings
Privacy dashboard
Privacy Statement


Windows Defender Firewall with Advanced Security on Local Computer

Windows Defender Firewall with Advanced Security provides network security for Windows computers.

Overview

Domain Profile

- Windows Defender Firewall is on.
- Inbound connections that do not match a rule are blocked.
- Outbound connections that do not match a rule are allowed.

Private Profile is Active

- Windows Defender Firewall is on.
- Inbound connections that do not match a rule are blocked.
- Outbound connections that do not match a rule are allowed.

Public Profile

- Windows Defender Firewall is on.
- Inbound connections that do not match a rule are blocked.
- Outbound connections that do not match a rule are allowed.

Windows Defender Firewall Properties

Getting Started

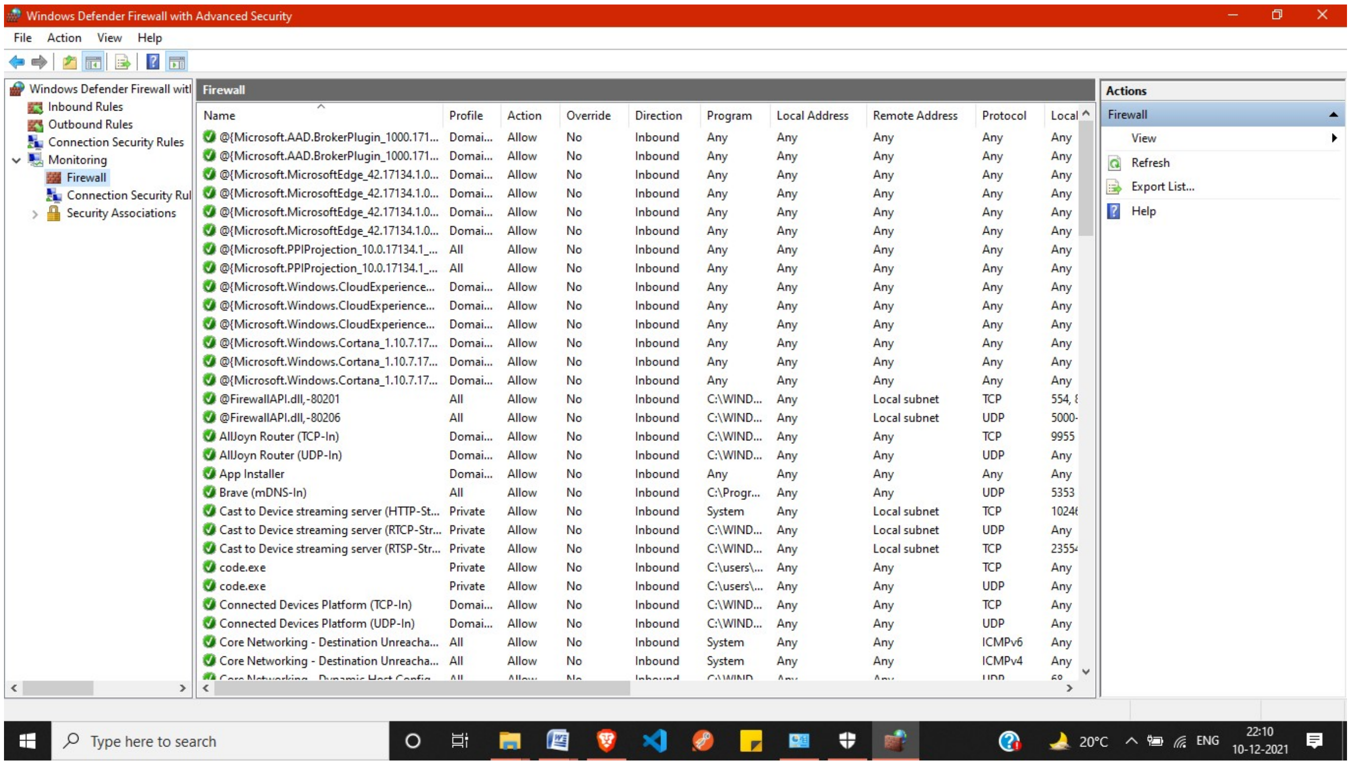
Authenticate communications between computers

Create connection security rules to specify how and when connections between computers are authenticated and protected by using Internet Protocol security (IPsec).

Connection Security Rules

View and create firewall rules

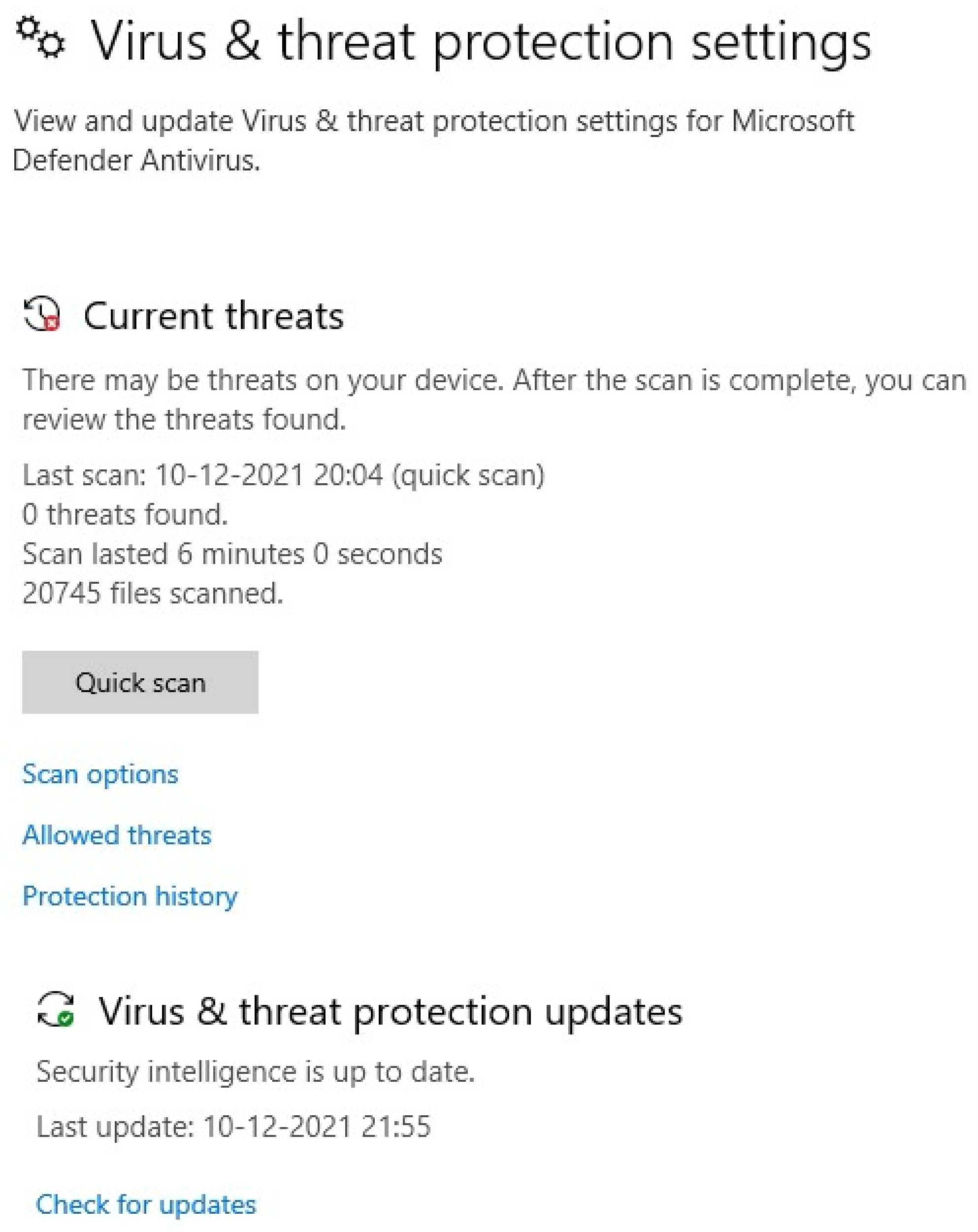
Create firewall rules to allow or block connections to specified programs or ports. You can also allow a connection only if it is authenticated, or if it comes from an authorized user, group, or computer. By default, inbound connections are blocked unless they match a rule that allows them, and outbound connections are allowed unless they match a rule that


Firewall

Name	Profile	Action	Override	Direction	Program	Local Address	Remote Address	Protocol	Local
@{Microsoft.AAD.BrokerPlugin_1000.171...	Domai...	Allow	No	Inbound	Any	Any	Any	Any	Any
@{Microsoft.AAD.BrokerPlugin_1000.171...	Domai...	Allow	No	Inbound	Any	Any	Any	Any	Any
@{Microsoft.MicrosoftEdge_42.17134.1.0...	Domai...	Allow	No	Inbound	Any	Any	Any	Any	Any
@{Microsoft.MicrosoftEdge_42.17134.1.0...	Domai...	Allow	No	Inbound	Any	Any	Any	Any	Any
@{Microsoft.MicrosoftEdge_42.17134.1.0...	Domai...	Allow	No	Inbound	Any	Any	Any	Any	Any
@{Microsoft.MicrosoftEdge_42.17134.1.0...	Domai...	Allow	No	Inbound	Any	Any	Any	Any	Any
@{Microsoft.PPIProjection_10.0.17134.1_...	All	Allow	No	Inbound	Any	Any	Any	Any	Any
@{Microsoft.PPIProjection_10.0.17134.1_...	All	Allow	No	Inbound	Any	Any	Any	Any	Any
@{Microsoft.Windows.CloudExperience...	Domai...	Allow	No	Inbound	Any	Any	Any	Any	Any
@{Microsoft.Windows.CloudExperience...	Domai...	Allow	No	Inbound	Any	Any	Any	Any	Any
@{Microsoft.Windows.CloudExperience...	Domai...	Allow	No	Inbound	Any	Any	Any	Any	Any
@{Microsoft.Windows.Cortana_1.10.7.17...	Domai...	Allow	No	Inbound	Any	Any	Any	Any	Any
@{Microsoft.Windows.Cortana_1.10.7.17...	Domai...	Allow	No	Inbound	Any	Any	Any	Any	Any
@{Microsoft.Windows.Cortana_1.10.7.17...	Domai...	Allow	No	Inbound	Any	Any	Any	Any	Any
@FirewallAPI.dll,-80201	All	Allow	No	Inbound	C:\WIND...	Any	Local subnet	TCP	554, 8
@FirewallAPI.dll,-80206	All	Allow	No	Inbound	C:\WIND...	Any	Local subnet	UDP	5000-
AllJoyn Router (TCP-In)	Domai...	Allow	No	Inbound	C:\WIND...	Any	Any	TCP	9955
AllJoyn Router (UDP-In)	Domai...	Allow	No	Inbound	C:\WIND...	Any	Any	UDP	Any
App Installer	Domai...	Allow	No	Inbound	Any	Any	Any	Any	Any
Brave (mDNS-In)	All	Allow	No	Inbound	C:\Progr...	Any	Any	UDP	5353
Cast to Device streaming server (HTTP-St...	Private	Allow	No	Inbound	System	Any	Local subnet	TCP	1024
Cast to Device streaming server (RTCP-Str...	Private	Allow	No	Inbound	C:\WIND...	Any	Local subnet	UDP	Any
Cast to Device streaming server (RTSP-Str...	Private	Allow	No	Inbound	C:\WIND...	Any	Local subnet	TCP	2355
code.exe	Private	Allow	No	Inbound	C:\users\...	Any	Any	TCP	Any
code.exe	Private	Allow	No	Inbound	C:\users\...	Any	Any	UDP	Any
Connected Devices Platform (TCP-In)	Domai...	Allow	No	Inbound	C:\WIND...	Any	Any	TCP	Any
Connected Devices Platform (UDP-In)	Domai...	Allow	No	Inbound	C:\WIND...	Any	Any	UDP	Any
Core Networking - Destination Unreacha...	All	Allow	No	Inbound	System	Any	Any	ICMPv6	Any
Core Networking - Destination Unreacha...	All	Allow	No	Inbound	System	Any	Any	ICMPv4	Any
Core Networking - Dynamic Host Config...	All	Allow	No	Inbound	C:\WIND...	Any	Any	UDP	68

II. Cyber Security Audit

The system is configured with Defender Antivirus. This checks for all the threats and malwares that the laptop has and responds and warns the admin immediately. This Antivirus is updated regularly and it provides Real-time protection, Cloud-delivered protection, Automatic sample submission, Tamper Protection and Controlled Folder Access.



Virus & threat protection settings

View and update Virus & threat protection settings for Microsoft Defender Antivirus.

Current threats

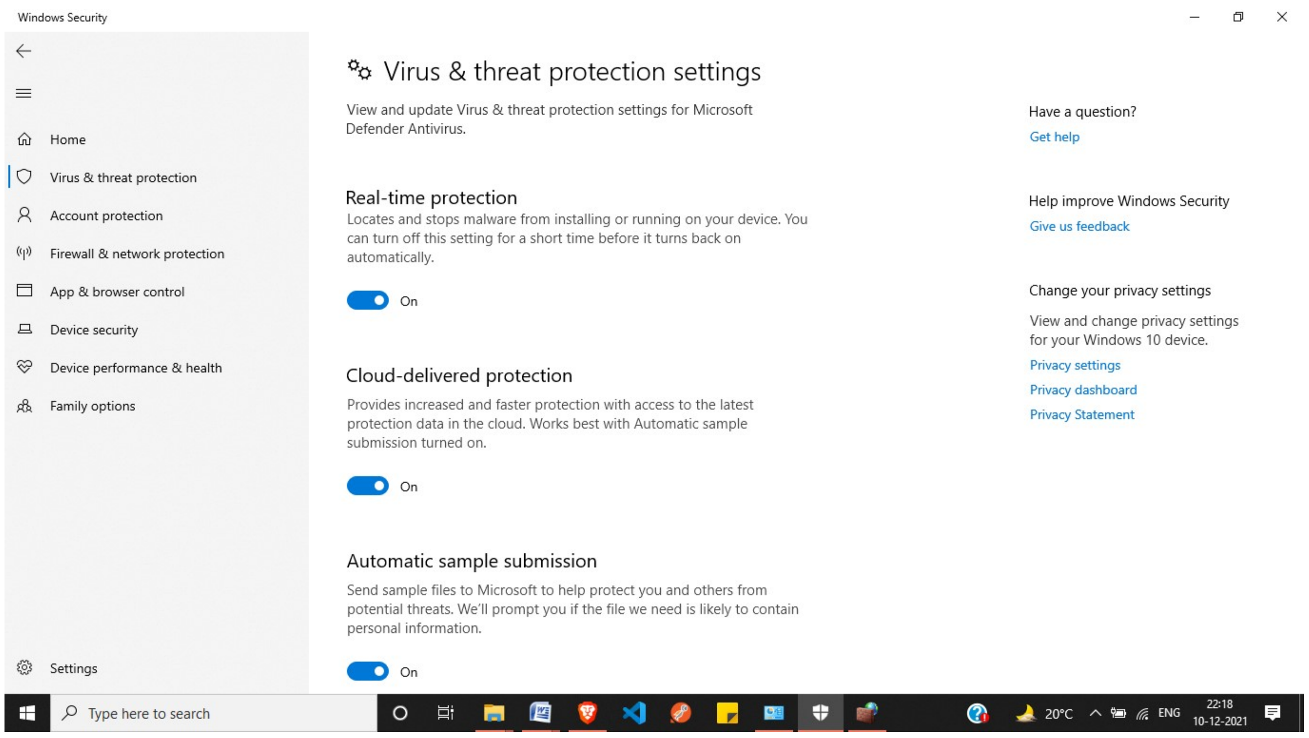
There may be threats on your device. After the scan is complete, you can review the threats found.

Last scan: 10-12-2021 20:04 (quick scan)
0 threats found.
Scan lasted 6 minutes 0 seconds
20745 files scanned.

Virus & threat protection updates

Security intelligence is up to date.

Last update: 10-12-2021 21:55


Windows Security

Virus & threat protection settings

View and update Virus & threat protection settings for Microsoft Defender Antivirus.

Real-time protection

Locates and stops malware from installing or running on your device. You can turn off this setting for a short time before it turns back on automatically.

On

Cloud-delivered protection

Provides increased and faster protection with access to the latest protection data in the cloud. Works best with Automatic sample submission turned on.

On

Automatic sample submission

Send sample files to Microsoft to help protect you and others from potential threats. We'll prompt you if the file we need is likely to contain personal information.

On

Have a question?
Get help

Help improve Windows Security
Give us feedback

Change your privacy settings

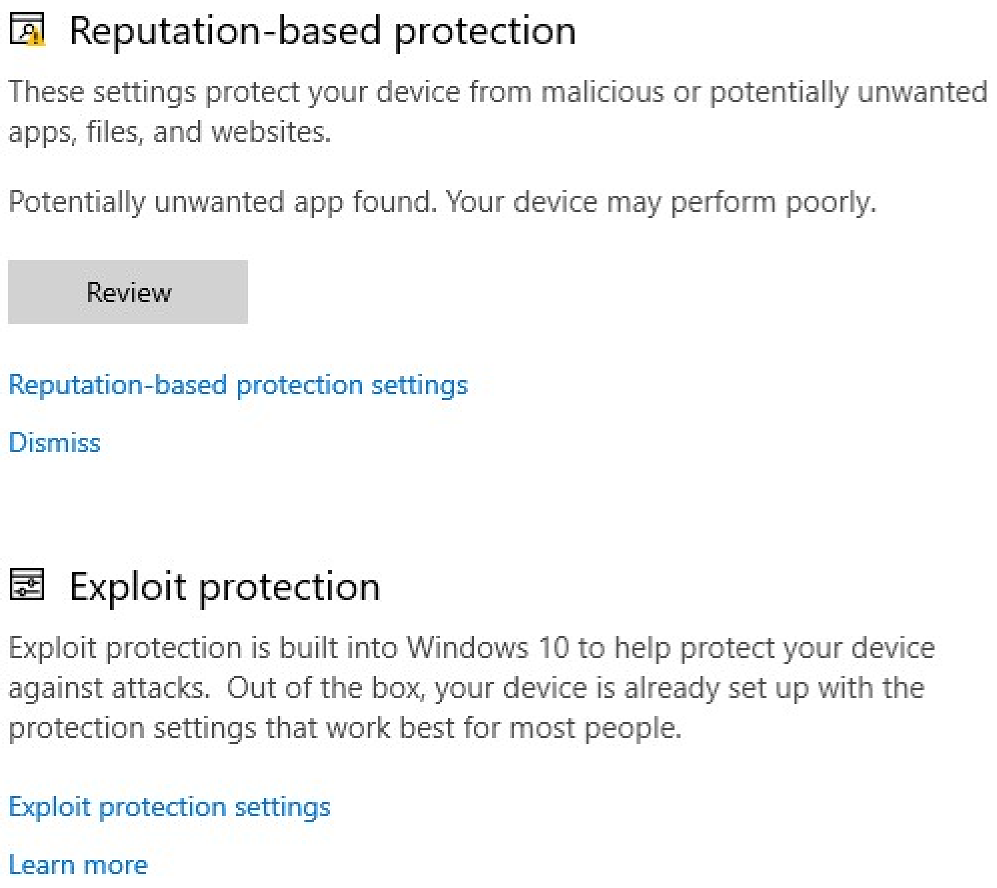
View and change privacy settings for your Windows 10 device.

Privacy settings
Privacy dashboard
Privacy Statement

III. Web Application Security

The hosted web-application runs on a http site using Amazon EC2 instance. So, it becomes easy for a malicious user to break into the site from outside as there is no SSL Encryption layer. Since the file stored on the Database are encrypted using AES, the malicious user cannot steal any data even after accessing it.

App and Browser Control are enabled for windows that detect any suspicious activity.



Reputation-based protection

These settings protect your device from malicious or potentially unwanted apps, files, and websites.

Potentially unwanted app found. Your device may perform poorly.

Exploit protection

Exploit protection is built into Windows 10 to help protect your device against attacks. Out of the box, your device is already set up with the protection settings that work best for most people.

The Secure Cloud Text transfer system is hosted on a deployment service, so although it can be accessed by malicious users, it'll be futile if downloaded from the server. Because it is encrypted using AES algorithm, which is very hard to crack using brute-force.

Moreover, the data is stored on amazon web services, which is a pioneer in cloud security, Hence the data stored on their servers on the cloud is also arguably secure.

Hence the web application is also secure in itself

IV. Compliance Audit:

Answer the following questions in YES/NO

1) Your environment under protection of Disaster Management: YES
2) Peoples Involved are educated: YES
3) All the components involved are following Indian Standards: YES
4) All Components are MADE IN INDIA: NO
5) Regular Internal Audit was covered: NO

VERDICT:

The System that runs this application locally is secure.

The ec2 terminal that is used to run this server is run via SSL and puTTy using a private key, which is secure.

If the application is accessed by some other user over the internet, the security of the files and folders on their system depends on the user's security, the files and folders in the cloud directory are safe because of 1. AWS and 2. AES encryption.

Hence it is safe to say that the hosted web application is also relatively secure.