Data Encryption and File Sharing

[1]Rafi Uz Zaman, [2]Parth Utkarsh, [3]Luv Tanwani, [4]Atul Kumar

*[123]Computer Science & Engineering, IMS Engineering College,
Ghaziabad, 201009, India*

Abstract- This paper is proposing a web based application that would provide a framework to the user to encrypt his data using a symmetric key cryptographic algorithm on the client's local machine itself. This would enable the user to be assured that the confidentiality of the user's data is maintained even from the encryption service providers themselves.

Keywords- Symmetric key encryption/decryption, application architecture, files upload and sharing.

Introduction- With the ever increasing digitalization the risk of data leaks over the internet have increased exponentially. This calls an immediate need for preventative measures against confidentiality breaches. Encryption of data has been looked upon as a promising technique over the previous few decades. The motivation for the development of this application came when a very well known cloud service provider was hacked in the late 2016 which resulted in the leak of sensitive information of the users. Moreover many big reputed giants were accused of sharing the sensitive user data with the several government agencies voluntarily. Keeping all this in mind we have created a web application based on a client-server architecture that provides encryption decryption services to the users without implementing any backdoors. The whole application architecture ensures complete confidentiality of the data provided by the client even from the server itself.

Apart from encryption and decryption, if a user wishes to upload an encrypted file on the server so that they might download it later from any remote machine, he/she can easily do so by using the upload functionality of the application. Only encrypted files are allowed to be uploaded on the server and links to the uploaded files are automatically generated and returned to the user. All the users are allotted some space on the drive where the data is securely stored in separate partitions for each user.

Literature Review

Many papers were studied during the development of this application out of which the main ones were considered and used as a base for this comparison report. Tiny Encryption Algorithm that was developed by David Wheeler and Roger Needham has been used in this application. Few of the papers which were considered are as mentioned-

[1] Wheeler, David J. And Needham, Roger M. TEA, a Tiny Encryption Algorithm Computer Laboratory, Cambridge University, England, November 1994.

[2] Wheeler, David J And Needham, Roger M TEA Extensions. Computer Laboratory, Cambridge University, England. October 1997.

[3] Steil, Michael. 17 Mistakes Microsoft Made in the Xbox Security System. October, 2005.

These papers were referred to analyse the running time complexity, implementation complexity and the strength of the above proposed algorithm.

Methodology-

This project involves providing cryptographic security to user data. Based on functionality the project can be divided into four major modules:

[1]**Data Encryption**:

The data that the user wishes to encrypt can be selected through the GUI of the application. This data is then encrypted on client's machine itself and the encrypted file is then automatically downloaded their systems local drive. As symmetric key encryption is involved an encryption/decryption key is to be decided. This key is automatically generated using the random mouse cursor coordinates of the user in the browser window making it unique for every user. An alternate option is provided for the user to replace the generated key by any other desired key.

Tiny Encryption Algorithm (TEA) has been used as symmetric key encryption algorithm. This is a light weight algorithm that produces results faster and requires low computational power allowing it run on all computer machines easily.

[2] **Data Decryption:**

The encrypted data file that was generated from the previous module which resides on the user local drive needs to be selected and passed along with the respective decryption key in order to decrypt the file. Then the file is decrypted and the automatically downloaded on the users local drive.

[3] **File Upload**:

The user is also given an option to upload the encrypted file on the server. Only the encrypted files are allowed to be uploaded on the server and the links to the files are automatically generated and returned to the user. All the users are allocated some space on the server's memory drive where the data is securely stored in separate partition for each user. Once the encrypted file is uploaded, it is compressed and converted into zip file. This file can later be downloaded from any remote computer using the respective decryption key.

[4] **File Sharing**:

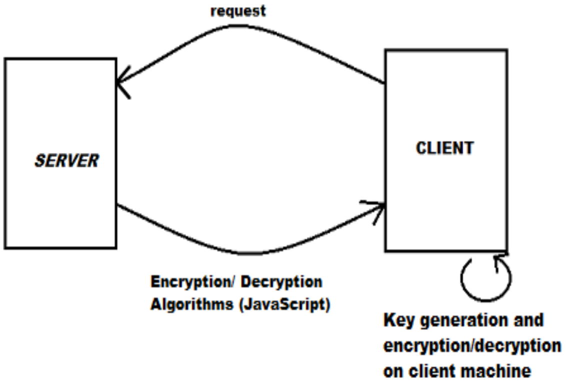
The encrypted files uploaded on the server can be easily shared with any other registered users of the application using the share option provided for each uploaded file. The link of the encrypted file on the server is sent as a message to recipient user. Once the recipient user is online he/she will be notified about the link received through an incoming message in the inbox. The file received can be easily downloaded by a single click and decrypted if the recipient has the respective key.

Implementation:

This project has been implemented using many popular technologies providing the application with simplicity and high scalability.

Platform- This web application has been designed on simple client-server architecture. The front end of the application involves HTML, CSS, and JavaScript. While the back end of this application includes the PHP as the server site scripting language, MySql as the database service provider and the server has been implemented by using Apache open source services.

Architecture- The application follows a typical client-server architecture where the algorithms,user record, file records, etc. are stored on the remote server and fetched on the client machine on user's request.


request
SERVER
CLIENT
Encryption/ Decryption Algorithms (JavaScript)
Key generation and encryption/decryption on client machine

The unique key by default is generated automatically every time a user logs into our server. This unique key is generated by tracking the random cursor co-ordinates produced by the user.

Whenever the user requests, as a response application provides them with the front end markup along with encryption and decryption algorithms encoded in JavaScript. These scripts are loaded in the browser to be executed on client's machine. This application uses browser itself as an interface which makes it platform independent.

Database Model- The application database is stored on a remote server. The database is implemented by using MySql service. All the user registration data such as the user name, passwords and email addresses are stored in the user record table. The file sharing and messaging records are stored in the messages table while the links to the files uploaded by the users on the servers are stored in the file record.

Only the encrypted files of the client are allowed to be uploaded on the server. For each client there exists a unique directory in the server with a specific amount of space which can be accessed by the client using the graphic user interface which is provided by the web application.

Security- There are many security measures implemented in the server to counter any malicious user activity.

Basic authentication has been implemented by checking the user against the username and password given at the time of registration. Passwords stored in the database are hashed with standard algorithm.

A large number of input filters have been implemented in the servers to counter the SQL injection attacks.

To avoid any kind of cross side scripting attack server input modules have been equipped with measures like CSRF tokens.

Conclusion:

This web application is created to overcome the problem of security in this digitalised world. With the advancement in technology everything is nowadays becoming digitalised and thus security of our confidential digital data is becoming our biggest challenge.

The web applications nowadays have access to our confidential data which in turn makes us vulnerable. Now it is the duty of the web application to keep our data secure. Sometimes these applications give access to its database to certain organisations that monitor our data and keep track of our activities, thus depriving us of our confidentiality.

Such was the scenario in case of a popular web communication application that took place in January 2013. It was claiming to be end to end encrypted and was promising security of data to its users. It was later found having a backdoor entry to its database. The access to it database was granted to a certain monitoring organisation. This resulted in users being denied right of confidentiality.

Even when the web applications are not corrupted there may be a case of intrusion of unauthorised users to its database making users data unsecure. This happened in case of a popular cloud storage that was hacked in August 2016 loosing 68 million email addresses and passwords of users in that web attack.

Thus overcoming this problem is the main goal our project that would allow users to encrypt data on their system itself thus denying web servers any access to original data. After the encryption takes place it would be up to the user to upload the encrypted data or share it.

Our web application enhances security of data because the data is always encrypted on the client side and always encrypted data is uploaded. This denies servers to have any access to original data thus always keeping it secure.

References

[1] Wheeler, David J. And Needham, Roger M. TEA, a Tiny Encryption Algorithm Computer Laboratory, Cambridge University, England, November 1994.

[2] Wheeler, David J And Needham, Roger M TEA Extensions. Computer Laboratory, Cambridge University, England. October 1997.

[3] Steil, Michael. 17 Mistakes Microsoft Made in the Xbox Security System. October, 2005.

[4] Mozilla "http://www.mozilla.org/enUS/docs/Web/API/FileReader"

[5] Eli Grey "http://www.eligrey.com/blog/saving-generated-files-on-the-client-server"