Design and Implementation of A Secure File Transfer System Using Encryption and Web-Based Authentication

D. Nandhini[1], T.Surendhar[2]
[1]*Assistant professor, MCA, Department of Master of Computer Applications*
[2]*MCA, Christ College of Engineering and Technology Moolakulam, Oulgaret Municipality, Puducherry – 605010*

***Abstract*—The internet is used more and more for lots of things. Because of this people are sharing files online a lot, whether it is for school, work or personal stuff. The old ways of sending files are not very safe. This means that important information can be easily seen by people who should not have access to it This paper is about a system, for sharing files that is safe. The Secure File Transfer System makes sure that only the right people can see and use the files that are being shared. It also makes sure that the files are not changed or lost during the transfer. The Secure File Transfer System is a way to keep files safe when they are being shared over the internet.**

The system they are talking about uses encryption to keep files safe before they are stored or sent. This system also uses a web-based way to authenticate users and manage their sessions. The people who made this system used Python. Something called the Flask framework. This means the system is easy to use when you want to upload or download files in a way. The system is also really secure. They tried out the system. Found out that it works well to keep data safe without slowing things down too much. This makes the system a good choice for people who want to share files in a way. The system is, about secure file sharing.

***Index Terms*—Secure File Transfer, Encryption, Flask Framework, Authentication, Web Security, Confidential Data**

I. INTRODUCTION

We live in a time where the internet's a big part of our daily lives. People use the internet to share files with each other all the time [11]. Schools and companies use file sharing to work and manage their files. They also use cloud-based services for the reason [11]. File sharing is really convenient. When we send files over the internet without any security it can be very dangerous [11]. Someone might intercept our files change them without our knowledge or even steal our identity [11].

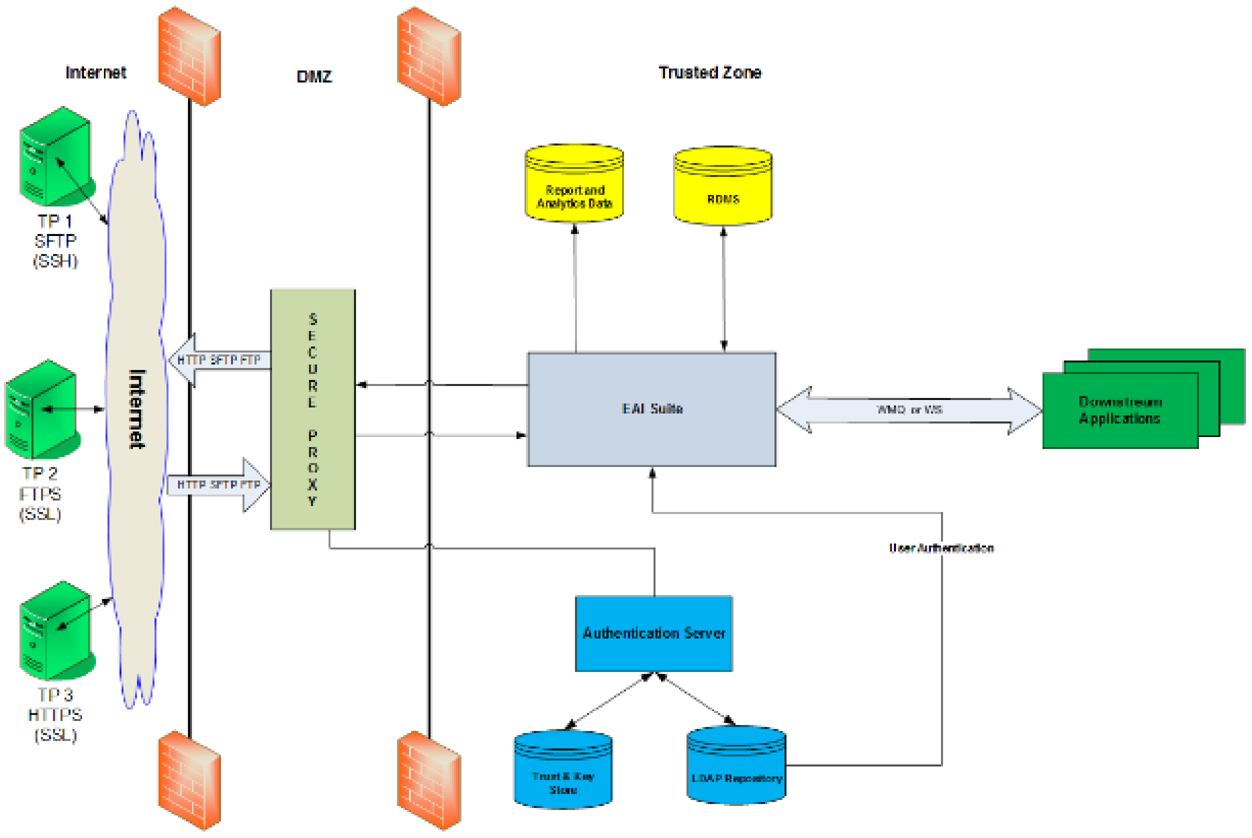
These problems get even worse when we send confidential files without protecting them properly [11]. File sharing is something we do all the time. We need to be careful when we do it. Secure file transfer systems try to stop people from getting our information by using special codes and checks to make sure everything is safe [11]. These codes, called encryption make sure that even if someone gets their hands on our data they will not be able to read it [11]. The system also checks to see if the person trying to get in is really who they say they are and then it decides what that person is allowed to do [11].

This project is about making a way to send files over the web that uses these security ideas in a way that is easy to use and works well [11]. The secure file transfer system is what we are trying to build. We want the secure file transfer system to be simple and good at what it does. The system is designed to balance strong security with usability, ensuring ease of adoption without compromising protection [11].

II. SYSTEM OVERVIEW

The Secure File Transfer System is a website that lets users upload, store, and download files in a secure way [11]. Users have to sign up and log in before they can use the system, which means only certain people can get in [16]. When users are logged in, they can upload files. These files are automatically locked with a code before they are saved on the server [11]. This way, even if someone gets into the server, the files will still be safe [19].The Secure File Transfer System is really good at keeping files protected.

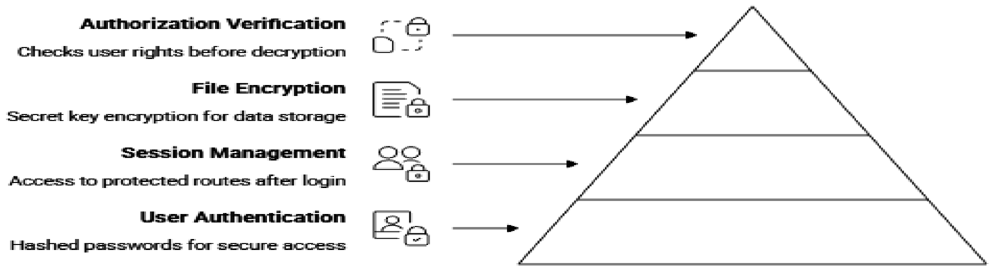
The system also lets certain people get access to files in a controlled way, so only those who are allowed can unlock the files [24]. There are tools for people in charge to manage the users and keep an eye on what's going on in the system, which helps make sure everyone is doing what they should be doing and that the system is secure [16][19].The way the system is designed is like building blocks, so parts such as checking who people are, locking and unlocking files, and managing the database can all work on their own while still being part of the whole system [11].



III. SYSTEM ARCHITECTURE

The Secure File Transfer System is set up in a client–server way [11]. It has a client and a server. The client part is what users see on the web. It uses HTML, CSS, and JavaScript to make an interface [8]. People can use any web browser to work with the Secure File Transfer System [11].The server part is where the magic happens. It is made using the Flask framework [8]. This framework does a lot of things, like managing what the application can do and checking if users are really who they say they are [16]. It also helps with moving files around on the Secure File Transfer System [11].

The encryption module is in charge of encrypting and decrypting files. It uses cryptographic techniques to do this [2][5]. The database layer is where user credentials and file information are stored. It also stores information about who can access which files [16]. This way of doing things helps keep everything organized and secure. It makes the system easier to maintain and allows it to handle more users [11].The encryption module and database layer work together to make sure the file encryption and decryption processes are secure [2][5]. The database layer is very important because it stores user credentials and file metadata [16], while the encryption module uses cryptographic techniques to handle file encryption and decryption processes [2]


Authorization Verification
Checks user rights before decryption
File Encryption
Secret key encryption for data storage
Session Management
Access to protected routes after login
User Authentication
Hashed passwords for secure access

IV. METHODOLOGY AND IMPLEMENTATION

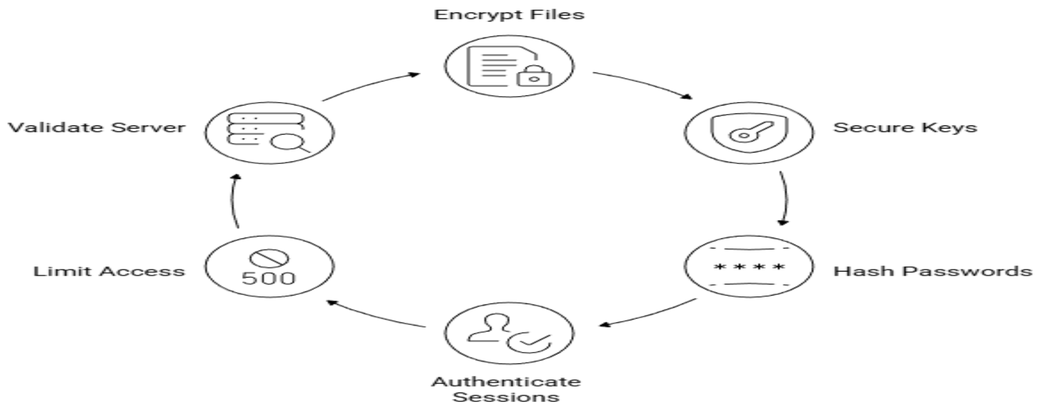
The system starts with people signing up and logging in [16]. When users create a password, the system scrambles it so it is not stored in plain text [18]. This helps to prevent someone from getting the password and using it [7]. When the user login is successful, the system lets users see the protected parts of the application [11]. The system uses a way to manage what the user can see and do after they log in; this is called session management for the web application system [7]. The web application system uses this to keep the user safe [16].

When a file is uploaded, the system uses a key to make the file secure before it puts the file on the server [2][3]. The system makes sure that the file is never stored without being encrypted [14]. The raw file contents are always protected [18]. When a user wants to download a file, the system checks to see if the user is allowed to do that before it decrypts the file and sends it in a secure way [24]. This ensures that the file is always protected both when it is being stored and when it is being accessed [18]. The system uses file encryption as the main method to keep files safe, and this is how file encryption protects user data [2].

V. ENCRYPTION AND SECURITY MECHANISMS

Encryption is really important for keeping files safe when we send them [2]. The system uses encryption because it works well and is good for protecting files on the server [3][14]. We make sure to handle the encryption keys in a secure way so that nobody can get to them without permission [5]. We also use password hashing and session-based authentication to keep user credentials and active sessions safe [7][16]. This way, the encryption and the whole system are protected [18]. Encryption is an important part of keeping everything secure [2].

We follow security rules to keep our system safe [7]. For example, we limit who can access files and control how people move around the system [16]. We also check everything on the server to make sure it is safe [19]. This helps stop people from doing things like looking at files they should not see, downloading things they should not have, and taking over someone's session [7][18]. Security best practices such as restricted file access paths, controlled routing, and server-side validation are used to prevent common attacks including directory traversal, unauthorized downloads, and session hijacking [6][7].



VI. RESULT AND DISCUSSION

The Secure File Transfer System was checked in different situations [14]. This included cases where many users signed up, files were uploaded with encryption, and authorized people downloaded files [2][14]. The results of these tests show that files on the server stay encrypted and are unreadable to anyone outside of the application [18]. People who were not allowed to access the files could not get to them or download them [24]. This shows that the Secure File Transfer System has strong control over who can access the files [16][24].

When we look at how the system performs, we see that encrypting and decrypting files does not really slow things down for large files that people commonly use [2]. The system works smoothly, and users do not notice significant delays [11]. At the same time, encryption and decryption provide strong security guarantees, which means users can feel safe when using the system [18]. This shows that the proposed approach is a practical solution for sharing files securely in real-world environments, especially when it comes to encryption and decryption for secure file sharing [14].

VII. COMPARATIVE ANALYSIS

Compared to traditional file transfer methods such as basic FTP or unsecured cloud sharing, the proposed system offers enhanced security through encryption and authentication [12][13]. Unlike conventional approaches that rely solely on network-level security, this system provides application-level protection [2][18]. This ensures data security even if network defenses are compromised [19]. The modular design also allows easier customization and future upgrades [11][14].

VIII. CONCLUSION

This paper presented a Secure File Transfer System that integrates encryption and web-based authentication to protect sensitive data [2][6]. The system successfully addresses major security challenges associated with file sharing by ensuring confidentiality, integrity, and controlled access [18][24]. The implementation demonstrates that strong security mechanisms can be achieved without sacrificing usability [11]. The proposed solution is well-suited for academic institutions, small organizations, and secure personal data exchange [15][17].

IX. FUTURE ENHANCEMENTS

- When we use Public Key Infrastructure, it helps us exchange keys in a secure way and it also supports authentication with digital certificates [4][5][9]. This makes the system more secure and trustworthy. Public Key Infrastructure is very important for the security of the system, as it helps to keep the system strong and safe [16].
- Using Multi-Factor Authentication can really help make things more secure [7]. This is because Multi-Factor Authentication combines passwords with ways to verify who you are, like special codes that are only good for one time or biometric methods [16].
- Adding cloud-based storage to a system can make it work better and be available all the time [17]. This also makes sure that all the data stored is kept confidential and secure [19].
- When dealing with files, it is a good idea to break them up into smaller parts and encrypt each part [2][14]. These parts can also be processed at the same time. This approach can improve upload and download performance for large files and reduce waiting time, which is especially helpful for efficient file handling [11].
- Putting a system in place to track and record everything that happens can really help with accountability and compliance [16]. This is important for following security standards such as ISO/IEC 27001 [16]. Having continuous monitoring ensures better visibility into system activities and helps maintain strong security controls [19].
- Deploying the system over HTTPS using TLS protocols and integrating intrusion detection mechanisms can provide additional protection against advanced network-level and cyber threats [22][19].

REFERENCES

[1] IEEE Computer Society, “Security and Privacy in Networked Systems.”

[2] W. Stallings, Cryptography and Network Security: Principles and Practice, Pearson.

[3] A. Menezes et al., Handbook of Applied Cryptography.

[4] NIST, “Digital Signature Standard (DSS).”

[5] NIST, “Guide to Cryptographic Key Management.”

[6] OWASP, “Secure File Upload and Download Guidelines.”

[7] OWASP, “Top 10 Web Application Security Risks.”

[8] Flask Documentation, “Flask Web Framework.”

[9] D. Boneh, “Twenty Years of Attacks on the RSA Cryptosystem.”

[10] Kahn Academy, “Introduction to Symmetric Encryption.”

[11] S. Garfinkel, “Design Principles for Secure Systems.”

[12] RFC 959, “File Transfer Protocol (FTP).”

[13] RFC 4253, “Secure Shell (SSH) Transport Layer Protocol.”

[14] IEEE, “Secure Data Storage and Transmission Techniques.”

[15] A. Tanenbaum, Computer Networks.

[16] ISO/IEC 27001, “Information Security Management Systems.”

[17] Cloud Security Alliance, “Best Practices for Secure Data Storage.”

[18] M. Bishop, Computer Security: Art and Science.

[19] ENISA, “Data Protection Engineering.”

[20] ACM, “Secure Web Application Development.”