Secure File Transfer Application

Juyoul Lee

Master of Science in Computer Information Systems
Florida Institute of Technology
juyoul2023@my.fit.edu

GitHub repository: https://github.com/ljy4499/secure-file-transfer-app

Abstract—The Secure File Transfer Application is designed to address the growing need for secure data exchange by utilizing modern cryptographic techniques. The application ensures confidentiality through AES encryption for file contents and secures key transmission using RSA encryption. Data integrity is maintained with SHA-256 hashing, and a user-friendly GUI simplifies the process of secure file transfer between systems. The application is versatile, supporting both local and public network environments.

I. INTRODUCTION

As digital communication continues to evolve, the need for secure file transfer has grown significantly. Sensitive files must be protected against unauthorized access, tampering, and loss during transfer. It is equally important to ensure that only the intended recipient can access and decrypt the file.

This project focuses on developing a Secure File Transfer Application that addresses these challenges by combining symmetric and asymmetric encryption techniques. AES is used to encrypt file contents, ensuring confidentiality, while RSA facilitates the secure exchange of the AES encryption key. Additionally, SHA-256 hashing provides a mechanism to verify file integrity, preventing unauthorized alterations. Designed for ease of use, the application includes a graphical user interface and supports secure file exchanges across both local and public networks, making it suitable for a variety of environments.

II. RATIONALE FOR ALGORITHMS SELECTION

The Secure File Transfer Application utilizes a combination of AES-CFB 256-bit encryption, RSA for key exchange, SHA-256 for file integrity verification, and WebSockets for communication. These technologies were selected based on their security, performance, and suitability for the project's requirements.

A. AES-CFB 256-bit with 32-byte Key

AES (Advanced Encryption Standard) with a 256-bit key is a widely adopted and highly secure symmetric encryption algorithm. The 256-bit key provides strong protection against brute-force attacks, ensuring robust confidentiality for file contents. AES in CFB (Cipher Feedback) mode was chosen over other modes like ECB (Electronic Codebook) or CBC (Cipher Block Chaining) due to its suitability for variable-length files. CFB mode is more flexible as it can encrypt partial blocks, which is crucial for file transfers where the size of the data is not fixed. Additionally, CFB mode prevents identical

plaintext blocks from generating identical ciphertext blocks, thereby offering better protection against cryptanalysis.

B. RSA for Key Exchange

RSA (Rivest-Shamir-Adleman) is an asymmetric encryption algorithm used for securely transmitting the AES key. In this project, RSA allows the sender to encrypt the AES key using the recipient's public key, ensuring that only the recipient, with their private key, can decrypt the AES key. This eliminates the need for a pre-shared secret, enhancing security by mitigating risks associated with key distribution and management. RSA provides a reliable mechanism for secure key exchange, facilitating safe transmission of the AES key over insecure networks.

C. SHA-256 for File Integrity Verification

SHA-256 (Secure Hash Algorithm 256-bit) is a cryptographic hash function chosen for verifying the integrity of files during transfer. SHA-256 produces a unique 256-bit hash value for each file, ensuring that any modification to the file during transmission can be detected by comparing the hash values before and after transfer. This provides an additional layer of security, ensuring that files remain untampered with and are transmitted without corruption.

D. WebSockets for Communication

WebSockets were chosen over FTP for their efficiency and simplicity in real-time, bidirectional communication. Unlike FTP, which requires multiple connections and is more complex to configure, WebSockets provide a persistent, low-latency connection, making file transfer faster and more reliable. They also integrate seamlessly with cryptographic algorithms such as AES, RSA, and SHA-256, allowing for straightforward encryption and secure data transfer.

WebSockets also support secure communication via SSL/TLS, ensuring encrypted data transfer across both local and public networks. In contrast, FTP requires additional configurations for secure transfer (e.g., FTPS), making WebSockets a more flexible and customizable solution for secure file exchange.

III. FLOWCHART

The workflow of the Secure File Transfer Application is illustrated in Figure 1. The process begins with the receiver opening a port to run the server. On the sender's side, the user

selects a file for transfer. The file is then hashed using the SHA-256 algorithm to ensure data integrity. Next, the file is encrypted using AES in CFB mode with a 256-bit key. The 32-byte AES encryption key is then secured by encrypting it with RSA using the receiver's public key.

After the encryption process, the sender transmits the encrypted file, the file's hash, and the RSA-encrypted AES key to the receiving system via TCP sockets. On the receiver's end, the RSA-encrypted AES key is decrypted using the receiver's private key to obtain the AES key. This key is then used to decrypt the file. Finally, the integrity of the received file is verified by comparing its SHA-256 hash with the hash computed at the sender's side.

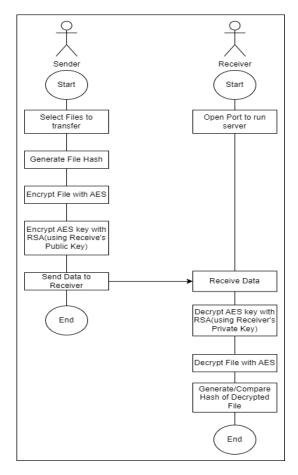


Fig. 1. Flowchart of the Secure File Transfer Application

IV. APPLICATION GUI

The application incorporates a GUI to enhance user experience. The interface is divided into two primary sections: one for sending files and another for receiving files. The sending section allows users to select a file, encrypt it, and transfer it to a specified recipient. The receiving section facilitates the decryption and integrity verification of received files. Screenshots of these sections are provided in Figures 2

The debugging terminal displays key information during file transfer. It logs the server's listening status, the file's SHA-256

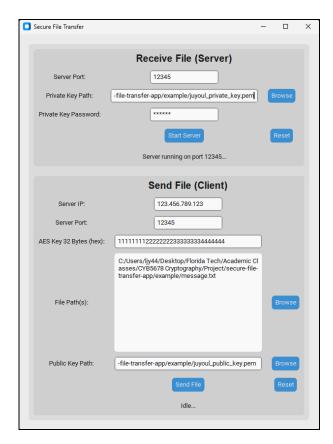


Fig. 2. GUI for Sending and Receiving Files

hash calculation, and AES encryption details. Additionally, it shows the lengths and previews of the encrypted file and AES key, along with connection details. After decryption, the terminal verifies the file's integrity by comparing hashes and confirms successful receipt and decryption. This ensures transparency and traceability throughout the process.

```
C:\Users\ljy44\Desktop\Florida Tech\Academic Classes\CYB5678 Cryptography\Project
Calculated file hash (SHA-256): 4eea8918a3030f334eaae080ff9cbo
File 'message.txt' encrypted. Encrypted file length: 74 bytes
First 50 bytes of encrypted file: b'\xd1\x9c\xd1G\x1b\xfc\x8d'
xd2\xc9\x81\x87\xde\x035\x01\xbb%G\xe1!\x8f\xaa\x0e\xa6\xa5\x
Encrypted AES key length: 256 bytes.
First 50 bytes of encrypted AES key: b"r\xc7'te\xcb\x14^ \x86'
c)\xc26\xa3\xd6\xb6;\xbbK\xcaT\xbdu$n\xed\x16B!"
    nected to server at 127.0.0.1:12345
nection from ('127.0.0.1', 21185) accepted.
Received filename: message.txt
Sending filename: message.txt, length: 11 bytes.
Sending file hash: 4eea8918a3030f334eaae080ff9cbdc29c11f566cd1
Received file hash: 4eea8918a3030f334eaae080ff9cbdc29c11f566c
Server received encrypted AES key length: 256 bytes
Sending encrypted AES key, length: 256 bytes.
Sending encrypted file, length: 74 bytes.
Decrypted AES key: 12341234123412341234123412341234...
Received encrypted data, length: 74 bytes.
Decrypted data preview: b'this is the message\r\n\r\npassword
Calculated hash: 4eea8918a3030f334eaae080ff9cbdc29c11f566cd1e
            ication successful. File integrity confi
                        received and decrypte
```

Fig. 3. Terminal for Debugging

No.																													
		Time		Sour	ce			Des	stinatio	n		Pro	tocol	Le	ngth	Inf	0												
Г	1	0.0000	100	140.	82.	112.	25	192	2.168.	1.2	38	TL	Sv1.2	2	101	. Ap	plica	ition	Dat	a									
L	2	0.0008	85	192.	168	.1.2	38	140	82.1	12.	25	TL	5v1.2	2	105	Ap	plica	tion	Dat	a									
	3	0.3983	03	192.	168	.1.2	38	140	82.1	12.	25	TL:	5v1.2	2	425	Ap	plica	ition	Dat	a									
		0.4320							2.168.				5v1.2					tion											
		60.396							2.168.				Sv1.2					ition											
		60.397							82.1				5v1.2	2				tion											
		119.83							2.168.			TCI						123											
		119.83							2.168.			TCI						123											
		119.84							2.168.			TCI						123											
-		119.84						_	2.168.	_	_		oSoc.					cet C											
		119.84							2.168. 2.168.			TCI						123											
		119.84						_	2.168. 2.168.	_		TCI						123											
		119.85							2.168.			TCI						123											
		120.39							2.168.				5v1.2					tion			Jeq		ACI						
		120.39							3.82.1				5v1.2					tion											
>		rame :																			52	hits	١.						
																				•									
		thern		-				_													_		2:/						
		ntern														•													
>	Ti	ransm:	iss	ion	Cor	ntro	ol I	Pro	tocol	., :	Src	Por	rt:	12	345,	D:	st P	ort:	12	345	, s	eq:	5,						
>	[2	2 Rea	ssei	mbl(ed 1	ТСР	Se	gmei	nts (13	by	tes): #	10	(2),	#:	11(1	1)]											
>	We	ebSoc	ket												[2 Reassembled TCP Segments (13 bytes): #10(2), #11(11)] WebSocket														
1	Co	ontin	ue:	6d	557	373	516	765	2e747	874	4																		
>				6de	557	373	516	765	2e747	874	4																		
	We	ebSoc	ket	6de	557	373	516	765	2e747	874	4																		
	We	ebSoc ebSoc	ket ket																										
>	We We	ebSoc ebSoc ontin	ket ket ue:									396	3386	439	9316	566:	1326	2333	8965	613	462	6366	333						
>	We We	ebSoc ebSoc	ket ket ue:									396	3386	439	9316	566:	1326	2333	3965	613	462	6366	33						
>	We We Co	ebSoc ebSoc ontin	ket ket ue: ket									396:	3386	439	9316	666	1326	2333	3965	613	462	6366	333						
>	We We Co	ebSoci ebSoci ontin ebSoci ebSoci	ket ket ue: ket ket		5439	933	3364	431	65383	03(938		3386							6134 9·#									
> >	We Co We We	ebSoc ebSoc ontin ebSoc ebSoc	ket ket ue: ket ket	610	643 <u>9</u>	30	3364	431 f9	23	6036 f6	25	d5		7e	c2	50	19		090		-%	Q~·	p.						
> > > > > 00	We Co We We	ebSociontino ebSociontino ebSocione ebSocione 0 01	ket ue: ket ket ee ff	610	39 c3	933: 30 00	3364 39 00	4310 f9 6d	23 65	f6 73	25 73	d5 61	51	7e 65	c2 2e	50 74	19 78		090	9 · #	·%	Q~·	p. tx						
> > 00 00	We We We We We	ebSociontino ebSociontino ebSocione ebSocione 0 01 0 00 74	ket ket ue: ket ket ee ff 00	610 30 f9	39 c3 00	30 00 40	3364 39 00 61	4310 f9 6d 64	23 65 39	f6 73 33	25 73 33	d5 61 64	51 67	7e 65 65	c2 2e 38	50 74 30	19 78 30	 t.	090 	9·# ·me	·% ssa 330	Q~· age. i1e8	P· tx						
> > 00 00 00	We We We 326	ebSoci	ket ue: ket ket ee ff 00 39	30 f9 00 63	39 c3 00 38	30 00 40 64	39 00 61 39	f9 6d 64 31	23 65 39 66	f6 73 33 61	25 73 33 32	d5 61 64 62	51 67 31	7e 65 65 39	c2 2e 38 65	50 74 30 61	19 78 30 34	t ·	090 @	9·# ·me ad9	-%- ssa 330 a2l	Q~ ige. i1e8	P· tx 00						
> > 00 00 00 00	We We We We 1930 1930 1930 1930 1930 1930 1930 1930	ebSociontino ebSociontino ebSocioni 2001 2000 740 380 62	ket ue: ket ket ee ff 00 39 63	30 f9 00 63	39 c3 00 38 33	30 00 40 64 38	39 00 61 39 61	f9 6d 64 31 62	23 65 39 66 62	f6 73 33 61 37	25 73 33 32 37	d5 61 64 62 61	51 67 31 33	7e 65 65 39 62	c2 2e 38 65 38	50 74 30 61 39	19 78 30 34 32	t · · · · · · · · · · · · · · · · · · ·	090 e8d f38	9·# ·me ad9 91f	-% ssa 330 a2l 77a	Q~ ige. i1e8 i39e	P tx 00 a4 92						
> > 06 06 06 06 06 06	We We We 930 940 950 960 960 960 960 960 960 960 960 960 96	ebSociontino ebSociontino ebSocioni ebSocioni 2001 2002 74 2038 2062	ket ue: ket ket ee ff 00 39 63	30 f9 00 63 66	39 c3 00 38 33 65	30 00 40 64 38 38	39 00 61 39 61 39	f9 6d 64 31 62 32	23 65 39 66 62 62	f6 73 33 61 37 38	25 73 33 32 37	d5 61 64 62 61 37	51 67 31 33 37	7e 65 65 39 62 36	c2 2e 38 65 38 61	50 74 30 61 39 62	19 78 30 34 32 61	t · · · · · · · · · · · · · · · · · · ·	090 	9·# ·me ad9 91f abb	-% ssa 330 a21 77a 837	Q~ d1e8 d39e d39e d7b8	P · tx 00 a4 92 ba						
> > 06 06 06 06 06 06 06	We We We We 926 936 936 9376 9376 9376 9376 9376 9376	ebSociontino ebSociontino ebSocioni 2001 2002 740 380 620 660 660	ket ket ue: ket ee ff 00 39 63 37 61	30 f9 00 63 66 39 36	39 c3 00 38 33 65 37	30 00 40 64 38 38 33	39 00 61 39 61 39 00	f9 6d 64 31 62 32 00	23 65 39 66 62 62 01	f6 73 33 61 37 38 00	25 73 33 32 37 33 30	d5 61 64 62 61 37 b2	51 67 31 33 37 65	7e 65 65 39 62 36	c2 2e 38 65 38 61 47	50 74 30 61 39 62 c3	19 78 30 34 32 61 2c	t · · · · · · · · · · · · · · · · · · ·	090 c8d f38 9e8	9·# ·me ad9 91f abb 92b	-% ssa 330 a2l 77a 837	Q~ d1e8 o39e o7b8 7e6a	P · tx 00 a4 92 ba						
> > 06 06 06 06 06 06 06 06 06	We W	ebSociontine ebSociontine ebSocion 2001 2000 742 382 2062 2066 2066 400	ket ket ket ket 60 39 63 37 61 ef	30 f9 00 63 66 39 36	39 c3 00 38 33 65 37 8f	30 00 40 64 38 38 33 cb	39 00 61 39 61 39 00 ef	f9 6d 64 31 62 32 00 7d	23 65 39 66 62 62 01 e4	f6 73 33 61 37 38 00 f7	25 73 33 32 37 33 30 f1	d5 61 64 62 61 37 b2 ba	51 67 31 33 37 65 36	7e 65 65 39 62 36 19 3b	c2 2e 38 65 38 61 47 7b	50 74 30 61 39 62 c3 88	19 78 30 34 32 61 2c 1a	t · · · · · · · · · · · · · · · · · · ·	090 @ c8d f38 9e8	9·# ·me ad9 91f abb 92b ···	-% ssa 330 a21 77a 837	Q~ d1e8 d39e d7b8 7e6a 6 G	P· tx 00 a4 92 ba						
> > 00 00 00 00 00 00 00 00 00 00	We W	ebSociontine ebSociontine ebSocion ebSocion 2001 2002 74 2036 2066 2066 2066 2066 2066	ket ket ket ee ff 00 39 63 37 61 ef 61	30 f9 00 63 66 39 36 01 ce	39 c3 00 38 33 65 37 8f 63	30 00 40 64 38 38 33 cb f5	39 00 61 39 61 39 00 ef	f9 6d 64 31 62 32 00 7d ed	23 65 39 66 62 62 01 e4 5d	f6 73 33 61 37 38 00 f7 db	25 73 33 32 37 33 30 f1 80	d5 61 64 62 61 37 b2 ba 14	51 67 31 33 37 65 36 45 8e	7e 65 65 39 62 36 19 3b 1f	c2 2e 38 65 38 61 47 7b 41	50 74 30 61 39 62 c3 88 8e	19 78 30 34 32 61 2c 1a c9	t· 89 bc f7 fa	090 c8d f38 9e8 673	9·# ·me ad9 91f abb 92b ···	-% ssa 330 a2l 77a 837	Q~ d1e8 39e 7e6a 6 G	P tx 000 a4 92 ba						
> > 000 000 000 000 000 000 000 000 000	We W	ebSociontino ebSociontino ebSociontino ebSocioni 2001 2002 2013 2014 2015 2016 2016 2016 2016 2016 2016 2016 2016	ket ket ket ee ff 00 39 63 37 61 ef 61 c6	30 f9 00 63 66 39 36 01	39 c3 00 38 33 65 37 8f 63 e5	30 00 40 64 38 33 cb f5 39	39 00 61 39 61 39 61 92 af	f9 6d 64 31 62 32 00 7d ed 08	23 65 39 66 62 62 62 01 e4 5d 82	f6 73 33 61 37 38 00 f7 db 3b	25 73 33 32 37 33 30 f1 80 ed	d5 61 64 62 61 37 b2 ba 14 ab	51 67 31 33 37 65 36 45 8e 84	7e 65 65 39 62 36 19 3b 1f a3	c2 2e 38 65 38 61 47 7b 41 97	50 74 30 61 39 62 c3 88 8e aa	19 78 30 34 32 61 2c 1a c9	t· 89 bc f7 fa	090 c8d f38 9e8 673	9·# ·me ad9 91f abb 92b ···	-% ssa 330 a2b 77a 837	Q~ d1e8 39e 7b8 7e6a 6 G	P. tx 000 a4 92 ba						
> > 06 06 06 06 06 06 06 06 06 06 06 06 06	We W	ebSociontino ebSociontino ebSociontino ebSocioni 2001 2002 2013 2014 2016 2016 2016 2016 2016 2016 2017 2017 2017 2017 2017 2017 2017 2017	ket ket ue: ket ee ff 00 39 63 37 61 ef 61 c6 4d	30 f9 00 63 66 39 36 01 ce 2c 77	39 c3 00 38 33 65 37 8f 63 e5	30 00 40 64 38 33 cb f5 39 31	39 00 61 39 61 39 61 39 ef 92 af	f9 6d 64 31 62 32 00 7d ed 08 47	23 65 39 66 62 62 62 01 e4 5d 82 7c	f6 73 33 61 37 38 00 f7 db 3b eb	25 73 33 32 37 33 30 f1 80 ed 8f	d5 61 64 62 61 37 b2 ba 14 ab	51 67 31 33 37 65 36 45 8e 84 24	7e 65 65 39 62 36 19 3b 1f a3 a0	c2 2e 38 65 38 61 47 7b 41 97 ba	50 74 30 61 39 62 c3 88 8e aa 53	19 78 30 34 32 61 2c 1a c9 11	t· 89 bc f7 fa 	090 c8d f38 9e8 673 	9·# •me ad9 91f abb 92b ••• •} ••[-% ssa 330 a2t 77a 837 -0	Q~ d1e8 d1e8 d7b8 7e6a 6 G E;{	P · tx 000 a4 92 ba						
> > 06 06 06 06 06 06 06 06 06 06 06 06 06	We W	ebSociontino ebSociontino ebSociontino ebSocioni ebSocio	ket ket ue: ket ee ff 00 39 63 37 61 ef 61 c6 4d 76	30 f9 00 63 66 39 36 01 ce 2c	39 c3 00 38 33 65 37 8f 63 e5 13 ae	30 00 40 64 38 33 cb f5 39 31 6e	39 00 61 39 61 39 61 92 af 19 8d	f9 6d 64 31 62 32 00 7d ed 08 47 c8	23 65 39 66 62 62 62 91 e4 5d 82 7c 28	f6 73 33 61 37 38 00 f7 db 3b eb 5f	25 73 33 32 37 33 30 f1 80 ed 8f a7	d5 61 64 62 61 37 b2 ba 14 ab 98 31	51 67 31 33 37 65 36 45 8e 84	7e 65 65 39 62 36 19 3b 1f a3 a0 9d	c2 2e 38 65 38 61 47 7b 41 97 ba a5	50 74 30 61 39 62 c3 88 8e aa 53 a2	19 78 30 34 32 61 2c 1a c9 11 27 b0	t· 89 bc f7 fa ·· ,a M·	090 028d f38 9e8 673 , , 9w.1	9·# ·me ad9 91f abb 92b ···	-% ssa 336 a2b 77a 837 -0	Q~ d1e8 d39e d7e8 7e6a 6 G E;{	P tx 000 a4 92 ba						

Fig. 4. Packet Capture on Wireshark

V. KEY LEARNINGS

This project provided valuable insights into the practical application of cryptographic techniques. Implementing AES and RSA encryption deepened our understanding of symmetric and asymmetric encryption methods, especially in the context of secure file transfer. Working with SHA-256 hashing also highlighted the importance of data integrity in maintaining secure communication.

The project exposed the complexities of public key distribution. In this setting, the sender already has the receiver's public key, but in a real-world application, secure distribution using Certificate Authorities (CAs) would be essential. Additionally, performance optimization was a key learning point, emphasizing the need to balance the encryption algorithms—AES for file encryption and RSA for key exchange—to ensure both security and optimal performance during file transfers.

VI. CHALLENGES

During the development of the Secure File Transfer Application, several challenges were encountered. Ensuring data integrity and ensuring the receiver gets the exact data sent by the sender was initially problematic, with issues of broken data due to misconfigurations. These were resolved by refining the transmission process. Another challenge was managing the transfer of multiple files in a single session instead of transferring them individually. This was also solved by implementing multi-file handling capabilities.

A critical challenge that remains unsolved is the implementation of a feature for the receiver to validate the file after receiving its metadata. This would allow the receiver to accept or reject a file based on security concerns regarding potentially malicious files.

VII. FUTURE RECOMMENDATIONS

To further enhance the Secure File Transfer Application, several improvements can be made:

- Advanced Algorithms: Incorporate Diffie-Hellman key exchange or TLS for more secure key exchange mechanisms, and consider integrating Post-Quantum Cryptography to future-proof the application against emerging threats.
- Post-Transfer Verification: Implement a feature allowing the receiver to accept or deny file transfers after validating the file metadata to prevent the acceptance of malicious files.
- Multi-Platform Compatibility: Extend support for Linux, macOS, and mobile devices through crossplatform frameworks, ensuring broader accessibility.
- Server-Client Authentication: Integrate user authentication with role-based access control (RBAC) and encrypted metadata exchange to ensure secure interactions between users and systems.
- PKI for Key Distribution: Implement Public Key Infrastructure (PKI) to securely manage and distribute public keys using digital certificates.

VIII. CONCLUSION

The Secure File Transfer Application successfully integrates AES, RSA, and SHA-256 to provide a robust solution for secure file sharing. The project not only addresses the confidentiality and integrity requirements of secure file transfer but also considers the importance of performance optimization. While some challenges remain, such as implementing file validation and secure public key distribution, the application provides a strong foundation for further enhancements. The recommendations for integrating advanced cryptographic techniques, improving platform compatibility, and enhancing security features will help future-proof the application for real-world use.

REFERENCES

- "Symmetric encryption," Cryptography, Accessed: Dec. 6, 2024. [Online]. Available: https://cryptography.io/en/latest/hazmat/primitives/symmetric-encryption/.
- [2] "RSA Cryptography documentation," The Cryptography Library, Accessed: Dec. 06, 2024. [Online]. Available: https://cryptography.io/en/latest/hazmat/primitives/asymmetric/rsa/.
- [3] F. Khan, "Modern GUI using Tkinter," Medium, Sep. 4, 2022. [Online]. Available: https://medium.com/@fareedkhandev/modern-gui-using-tkinter-12da0b983e22