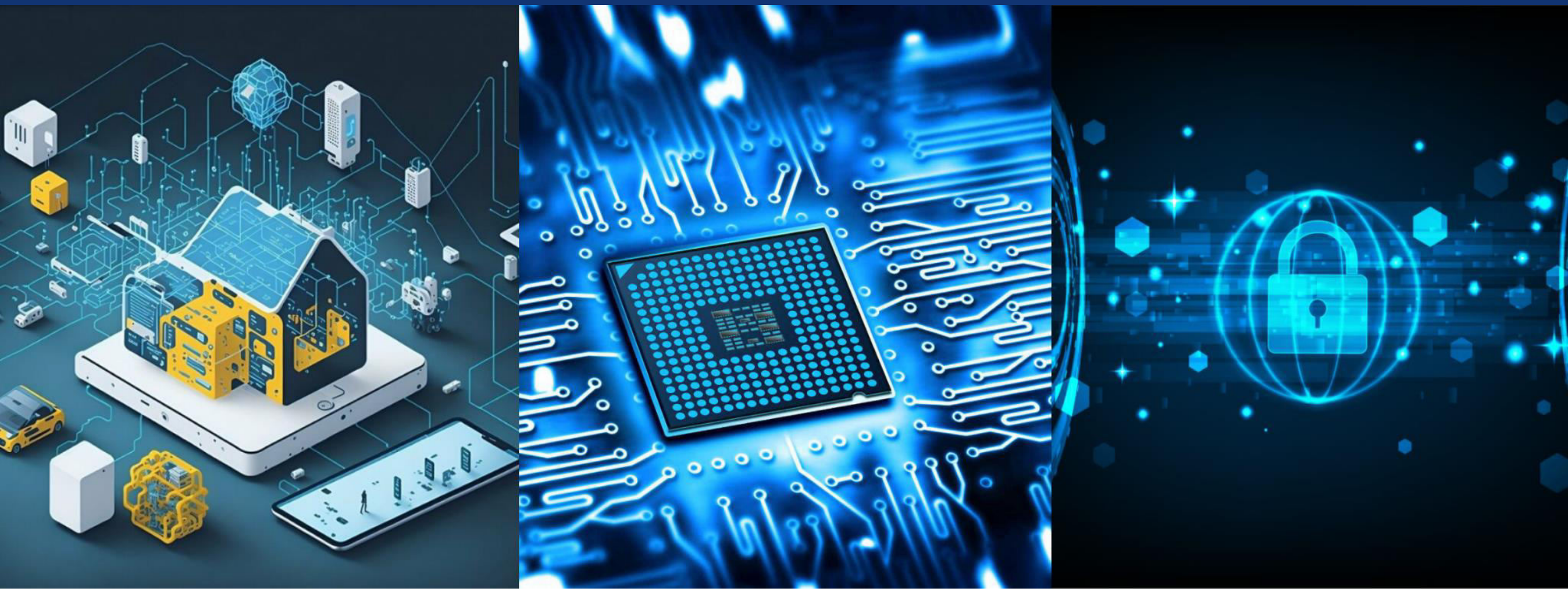
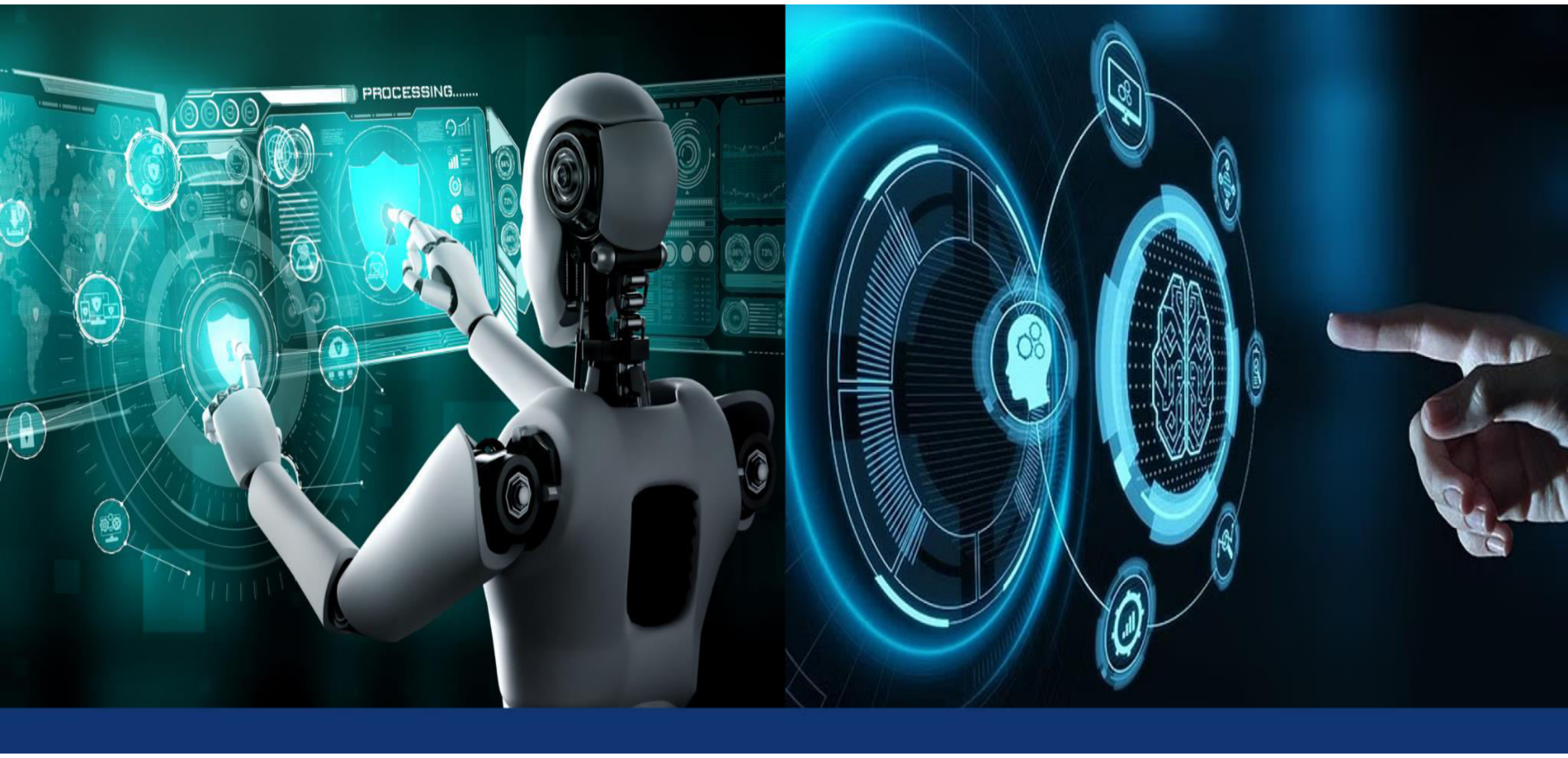
IJIRCCE

ISSN(O): 2320-9801
ISSN(P): 2320-9798



International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.771

Volume 13, Issue 4, April 2025

www.ijircce.com ijircce@gmail.com +91-9940572462 +91 63819 07438 www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

DropX: A Secure Peer-to-Peer File Transfer Application for Mobile Devices

Dr. Preeti Gupta, Muna Yadav, Harsh Vardhan Pandey, Bibhanshu Lal Karn, Rupam K. Sarangi, Ishan Avjad P Majeed

Assistant Professor, Dept. of CSE-CTMA, FET - Jain University, Bengaluru, Karnataka, India

UG Students [B.Tech], Dept. of CSE-CTMA, FET - Jain University, Bengaluru, Karnataka, India

ABSTRACT: This research paper presents a comprehensive analysis of DropX, a secure peer-to-peer file transfer system for Android implemented with React Native. DropX supports peer-to-peer devices to share files directly with one another over local networks. The application uses TCP/TLS sockets for autonomous end-to-end encryption, and certificate-based authentication for encryption and security. The application successfully addresses several common issues facing academic and many business users of mobile file sharing, such as security exposure and risks, cross-platform transferability, and usability challenges. DropX implements file transfers in "chunks", real-time updates to the end-user on transfer progress, automated recovery from connection losses. DropX enhances the user-friendliness and security of file sharing by creating peer-initiated connections utilizing QR codes and hashed data. We provide an in-depth analysis of the overall system architecture, local security features, file transfer protocols, and performance aspects of DropX, and compare it to other existing methods that either depend on the cloud or require a server on the mobile platform. In the end, we demonstrated that DropX is a much more reliable file transfer interface along with the highest security potential when compared to the cloud, as it eliminates the need to transfer files via a third-party server altogether, while maintaining user privacy.

I. INTRODUCTION

As the world continues to grow more interconnected, safe and efficient file sharing has become paramount. Traditional file sharing mechanisms have generally resided on cloud servers or third-party services, all of which have their own security vulnerabilities and privacy concerns. Most have offered little choice but to trade either convenience or security in exchange for the other. This trade-off becomes even more problematic when users simply want to quickly transfer or share files locally between their devices, especially when time is of the essence.

DropX addresses these problems by creating a device-to-device file transfer experience, prioritizing security and ease of use. DropX is a peer-to-peer app built on the React Native framework, allowing users to transfer files locally, directly between their devices, without sending their data through a remote, external server or third-party service, minimizing the attack surface and protecting their privacy.

II. LITERATURE SURVEY

Peer-to-peer (P2P) file sharing has evolved as a prominent paradigm for decentralized data distribution, offering advantages such as scalability, fault tolerance, and direct data exchange without reliance on central servers. However, this model introduces unique challenges, particularly in the realms of security, trust, and user experience. The literature reflects a broad spectrum of research addressing these issues, which are directly relevant to the design and implementation of applications like DropX.

Security Issues in P2P File Sharing
Security remains a primary concern in P2P systems. Wallach (2002) provides a comprehensive survey of security challenges in P2P networks, highlighting vulnerabilities in routing protocols, fairness, and trust among peers. The study discusses the application of cryptography, network probing, and incentive mechanisms to mitigate these risks, emphasizing the need for robust authentication and data integrity measures in file sharing applications[5]. Similarly, Arshiya Sulthana (2018) underscores the importance of secure data transmission in P2P networks, noting that TCP is

commonly used for reliable file transfer, but security enhancements such as encryption and flow control are necessary to prevent unauthorized access and data leakage[4].

A more recent review by the IJCSNS (2020) identifies major security vulnerabilities, such as the misuse of well-known ports and unauthorized file access. The authors advocate for the use of signature-based intrusion detection systems (IDS) and policy enforcement frameworks to detect and limit undesirable traffic. They also stress the importance of user awareness and control over shared files to prevent accidental data exposure[1].

Trust Models and Reputation Systems
Given the open and dynamic nature of P2P networks, establishing trust is crucial. Traditional trust models often fall short due to the fuzzy, complex, and dynamic characteristics of trust in distributed environments. Liu et al. (2009) introduce the NatureTrust model, which employs linguistic terms and fuzzy inference rules to assess trustworthiness, incorporating both trust and risk factors. Their experiments demonstrate improved resistance to malicious peers compared to earlier models like XRep and PeerTrust[2]. Similarly, research by Vitri Tundjungsari et al. focuses on reputation-based trust management, using feedback history and peer performance evaluation to enhance the reliability and effectiveness of file sharing[6].

Comparative Analysis of File Sharing Applications
A comparative study by Irfan and Khalique (2016) analyzes popular mobile file sharing applications such as SHAREit, Zapya, and Xender. These applications primarily use Wi-Fi Direct or WLAN hotspots for fast, local transfers, offering ease of access and high throughput. However, the study notes that while these apps provide speed and convenience, security features such as end-to-end encryption and robust authentication are often limited or absent, exposing users to potential risks during file transfers[3].

Hybrid and Decentralized Approaches
Recent work by Namagiri (2021) explores hybrid P2P file sharing systems, which combine centralized and decentralized architectures to balance scalability and control. The study highlights the potential of such systems to overcome limitations of pure P2P or client-server models, but also points out technical challenges related to network architecture, peer discovery, and data management. The author emphasizes the need for innovative solutions to address these challenges, particularly in the context of security and efficient peer discovery[7].

Summary and Relevance to DropX

The surveyed literature underscores several key requirements for modern P2P file sharing applications:

- Implementation of strong encryption (such as TLS) for secure data transmission[1][4][5].
- Adoption of certificate-based authentication and trust management systems to prevent impersonation and malicious activity[2][6].
- Mechanisms for user control and awareness to minimize accidental data exposure[1].
- Efficient chunked transfer protocols and flow control to handle large files and ensure reliability[4].
- Enhanced usability features, such as Hashed QR code-based connections, to simplify secure device pairing[3].

DropX addresses these critical challenges by integrating TLS encryption, certificate-based authentication, chunked file transfers, and user-friendly connection mechanisms, positioning itself as a secure and efficient alternative to both traditional P2P and mainstream mobile file sharing solutions.

CURRENT CHALLENGES IN MOBILE SHARING

The landscape for mobile file sharing still has multiple challenges that include:

- Worries over security for cloud-based services allowing service providers access to the data
- Complications when establishing a direct connection between mobile devices
- Compatibility challenges between iOS and Android
- User experience challenges which make security options unappealing to the average user

- Performance limitations when transmitting large files.

DropX aims to solve these challenges by taking an innovative approach to secure peer-to-peer file transfers using secure technology with enhanced user experience.

EXISTING MOBILE FILE SHARING SOLUTIONS

There are many file sharing apps that exist in the mobile ecosystem that each have different approaches and cons.

- **Dropzone** is a free LAN file sharing, chat, and remote terminal app. Dropzone allows people to share files with everyone who is online within their local area network, or with specific contacts. Dropzone, unlike DropX, focuses primarily on desktop environments and does not take a mobile-first approach.
- **FILARE** is an Android based secure file sharing platform that uses AES and DES cryptographic algorithms to secure encrypted files and use Firebase for cloud storage and authentication. FILARE has superior encryption abilities compared to DropX, but relies on cloud storage instead of file transfers peer-to-peer, as is done by DropX.
- **Drop** is an end-to-end encrypted personal file sharing service that is self-hosted and designed to be simple and secure. Drop is focused on allowing people to upload encrypted files to a self-hosted server, while DropX allows for device-to-device transfers.
- **drop.lol** is a file sharing web app based on webRTC that allows for end-to-end encryption with encypted files threw a file upload or drag-and-drop interface without requiring any log in credentials. drop.lol is like DropX in that it is focused on user privacy and ease-of-use, but instead of being native on a mobile platform, it is a web app.

P2P TECHNOLOGIES IN MOBILE ENVIRONMENTS

Due to the network, device and platform limitations of mobile environments, peer-to-peer (P2P) technologies are faced with special challenges including:

- NAT traversal and firewall constraints,
- Battery usage,
- Intermittent connectivity
- Limitations to background processes imposed by the platform,
- Variable device hardware.

DropX has built its own implementation of TCP/TLS sockets and connection management specifically designed for mobile, that address the above challenges.

SYSTEM ARCHITECTURE

DropX is built on a modular framework that consists of three components, which work together on their own to allow for secure and efficient file transfers. In this section, we will detail each of these components, as well as how they interact with one another.

HIGH LEVEL ARCHITECTURE:

The application follows a layered architecture pattern, organized around each layer with clear separation of concerns as follows:

- **Presentation Layer:** Provided with React Native and React Native components, and handling the user interactions and display
- **Business Logic Layer**: Handle the application state and orchestration using Zustand
- **Network Layer:** Handle secure connections and file system through TCP/TLS sockets
- **Storage Layer**: Provided the file system interface through React Native FS

This structure allows us to have clean organization between the UI concerns and the underlying file transfer concerns, which promotes future readability and testing.

CORE COMPONENT: TCP PROVIDER

The TCP Provider is the DropX network implementation; dealing with all aspects of secure socket communication. The Provider does the following:

- Creates and manages secure TCP/TLS connections between devices
- Implements connection state management based on the events of the connection use-case
- Provides context for the application to leverage network functionality
- Provides error-handling and recovery

The TCP Provider abstracts away the underlying details of network communication and provides higher level components with a simple interface to allow them to request a file transfer operation and get a response. This is a crucial abstraction, because if there was a lot of direct socket programming in the code it would reduce the quality of code and hinder the evolution of the application without completely rewriting the networking code.

CORE COMPONENT: NAVIGATION SYSTEM

DropX is designed with a comprehensive navigation system that intends to lead the user through the very simple and straightforward file sharing process, with user friendly screen flows. The navigation system includes several primary screens:

- SplashScreen: Enters the application with a visual identity and initial loading
- HomeScreen: Where the user comes to choose between sending or receiving files
- ConnectionScreen: Confirms the desired connection to another device
- SendScreen: User interface to select and send files
- ReceiveScreen: User interface to see incoming file transfer progress
- ReceivedFileScreen: Displays successfully received files providing ability to share or open them(id).

The simplified flow of this interface is designed to smooth the intrinsic friction involved in completing file transfers, so that even if the user is not technically-savvy, they can complete the process easily.

CORE COMPONENT: FILE TRANSFER PROTOCOL

The file transfer protocol in DropX is also a custom file transfer protocol designed for peer-to-peer secure transfers on mobile devices. The protocol follows three different phases:

Connection Phase

- TLS handshake to initiate a secure channel
- Exchange of device information for identification
- Connection acknowledgment to confirm connection

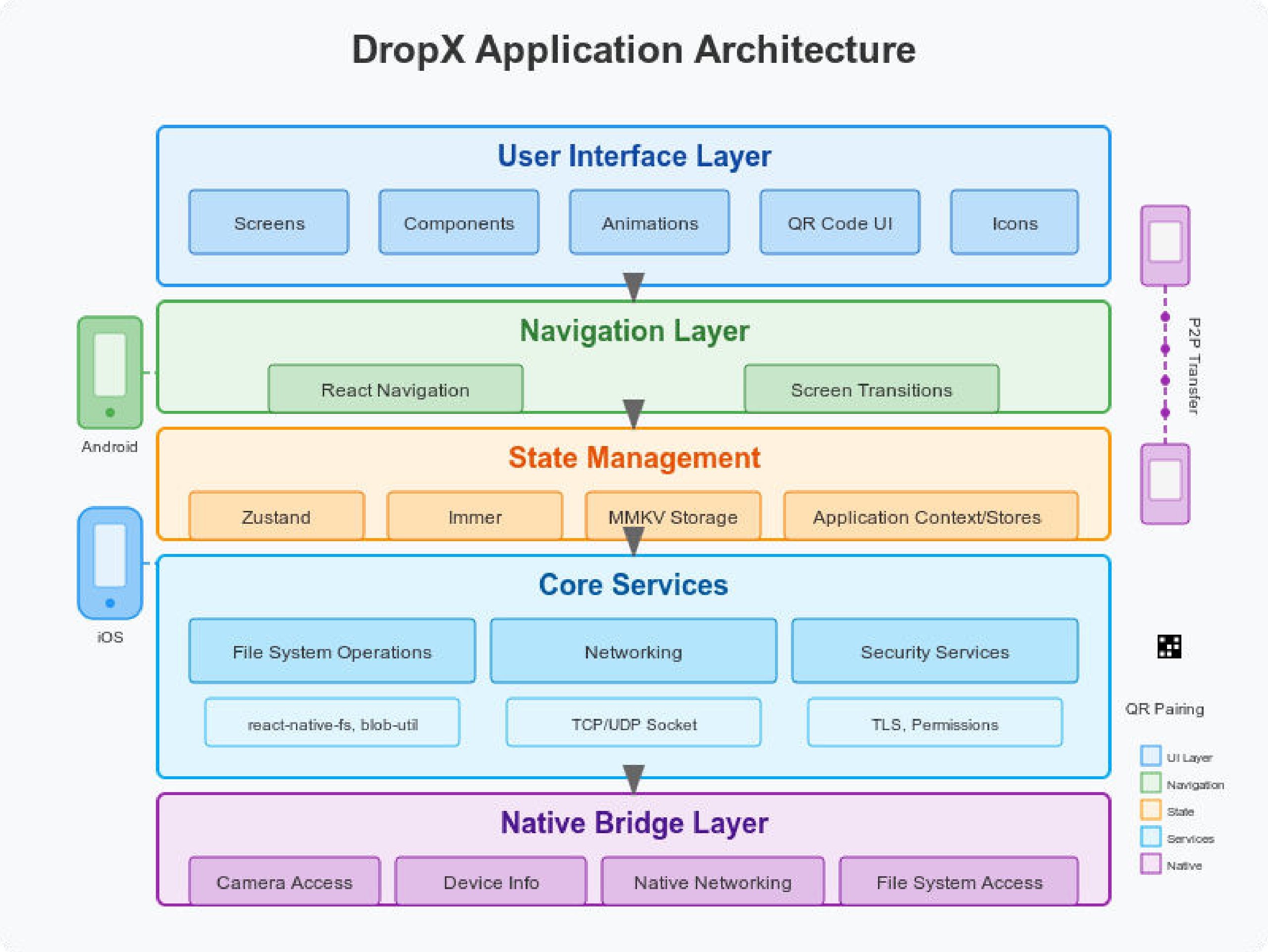
Transfer Phase

- File metadata exchange with name & size & type
- Chunked data transfer to allow for larger files
- Informed progress tracking to provide user feedback
- Chunk acknowledgement system to ensure delivery completion

Completion Phase

- File receiver integrity check with checksum
- File reconstruction from received chunks
- Success to both sender and receiver

This protocol has mechanisms for ensuring file security and reliability while also offering features to handle failures at every phase.



III.IMPLEMENTATION

Tech Stack Analysis

DropX leverages a modern, cross-platform technical stack to deliver a consistent experience across both Android and iOS devices:

- **React Native (v0.78.0)**: Provides the foundation for cross-platform mobile development, enabling code sharing between platforms while maintaining native performance.
- **Zustand**: A lightweight state management solution that simplifies state handling compared to alternatives like Redux, reducing boilerplate code and improving development efficiency.
- **TCP with TLS encryption:** Forms the backbone of the secure communication channel, ensuring that all data transferred between devices remains encrypted.
- **React Native FS:** Enables efficient file system operations, allowing the application to read, write, and manage files on the device.

Additional libraries enhance specific functionalities:

- react-native-tcp-socket: Provides low-level socket implementation
- react-native-fs: Handles file system operations
- immer: Simplifies immutable state updates
- react-native-device-info: Retrieves device information for identification

- react-native-vision-camera: Enables QR code scanning for easy connections
- react-native-qrcode-svg: Generates QR codes for sharing connection details

Frontend Implementation

The frontend of DropX is built using React Native components, following modern best practices for mobile application development. Key aspects of the implementation include:

- Component-Based Architecture: UI elements are separated into reusable components
- Responsive Design: Layouts adapt to different screen sizes and orientations
- Platform-Specific Adaptations: Where necessary, code branches handle platform differences
- Minimalist UI: Clean interfaces reduce cognitive load and improve usability

State Management with Zustand

Zustand provides several advantages over more complex state management solutions:

- Reduced Boilerplate: Minimizes the code required to manage application state
- Intuitive API: Simple store creation and state updates
- Performance Optimizations: Automatic memoization and selective rendering
- Middleware Support: Allows for logging, persistence, and other enhancements

This approach to state management keeps the codebase lean and maintainable while providing sufficient power to handle complex application states.

Network Protocol Implementation

The TCP/TLS implementation in DropX involves several technical considerations:

- Socket Configuration: Proper socket initialization with appropriate timeouts and buffer sizes
- TLS Certificate Management: Generation and validation of certificates for secure connections
- Error Handling: Robust error detection and recovery mechanisms
- Flow Control: Implementation of high-water mark to prevent buffer overflows

These implementations ensure that the connection between devices remains secure throughout the file transfer process, protecting against potential eavesdropping or man-in-the-middle attacks.

IV. SECURITY

DropX employs several key elements to ensure the protection and security of file transfers:

- **TLS Encryption:** All file transfers on DropX utilize Transport Layer Security (TLS). This ensures the data is encrypted in transit between devices. This means that even if someone was listening and trying to steal the files, they wouldn't be able to read them.

- **Certificate-Based Authentication**: Devices authenticate each other using digital certificates, not passwords. This ensures that only the designated devices trust each other and can connect and exchange files.

- **Secure Socket Configuration:** The app uses secure TCP/TLS sockets to maintain integrity and confidentiality of data. The TCP/TLS sockets used in the app are configured with strong parameters.

- **Chunked File Transfer with Checks:** Files are broken down into chunks before being transferred. Each snack is transferred securely and integrity checked to ensure the data is not corrupted or intercepted.

- **Hashed QR Codes:** Device details are exchanged in the form of QR codes. The QR codes have been hashed to make them safer and less visible when exchanging in the device pairing stage.

With all these elements taken together, we can provide, without question, that DropX provides the best and most complete peer-to-peer end-to-end secured file sharing available, thus eliminating the possibility of file exchanges with unauthorized devices, even while those exchanges are in complete encrypted transmission.

Compared to other file sharing software, DropX has certain security advantages:

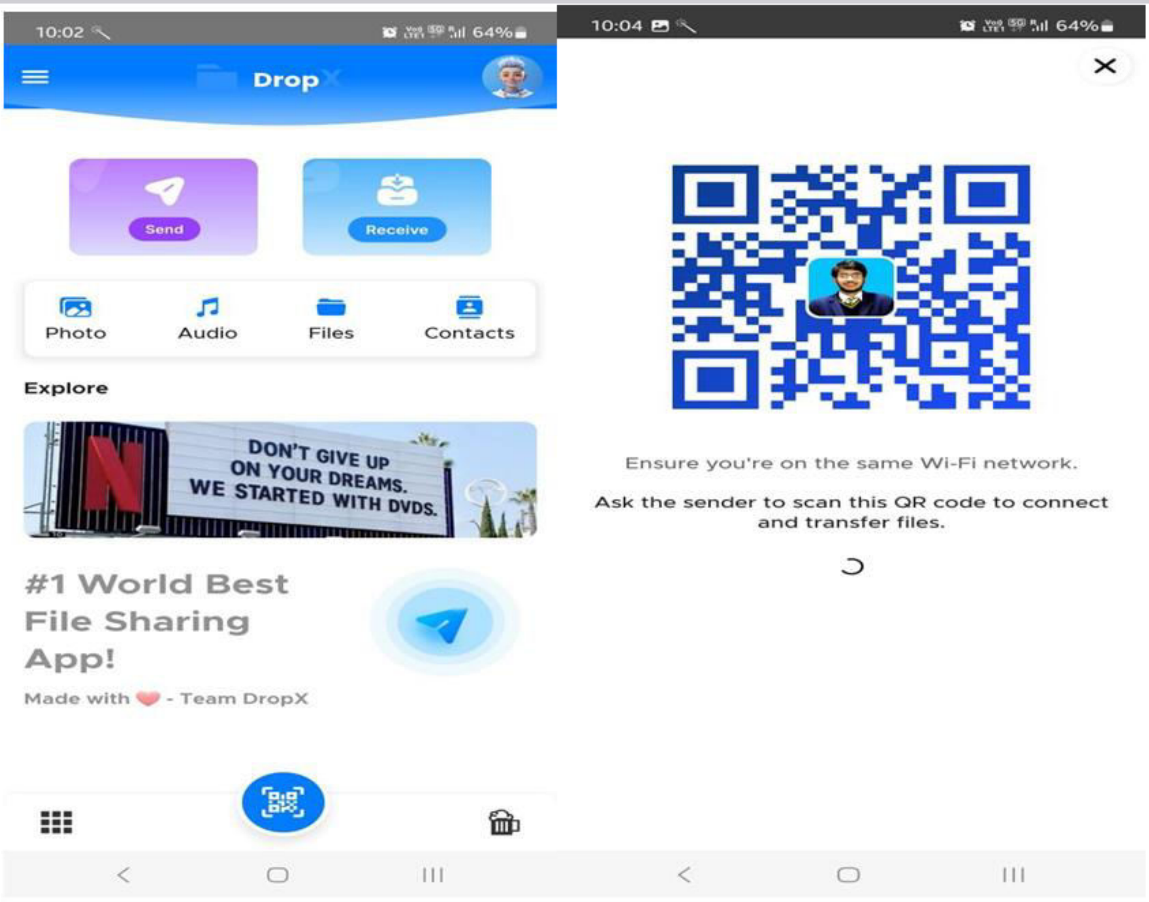
- **Versus FILARE**: While both are encryption-based, DropX cuts out cloud storage risk since the transfers are always direct to the receiving device.

- **Versus Dropzone:** In addition to the encryption for the file itself, DropX uses TLS for transport security, with certificate-based authentication, providing a stronger security guarantee than basic LAN file sharing.

- **Versus drop.lol**: Both offer end-to-end encryption, but DropX's native implementations may provide better security assurances based on the vulnerabilities associated with browser-based clients.

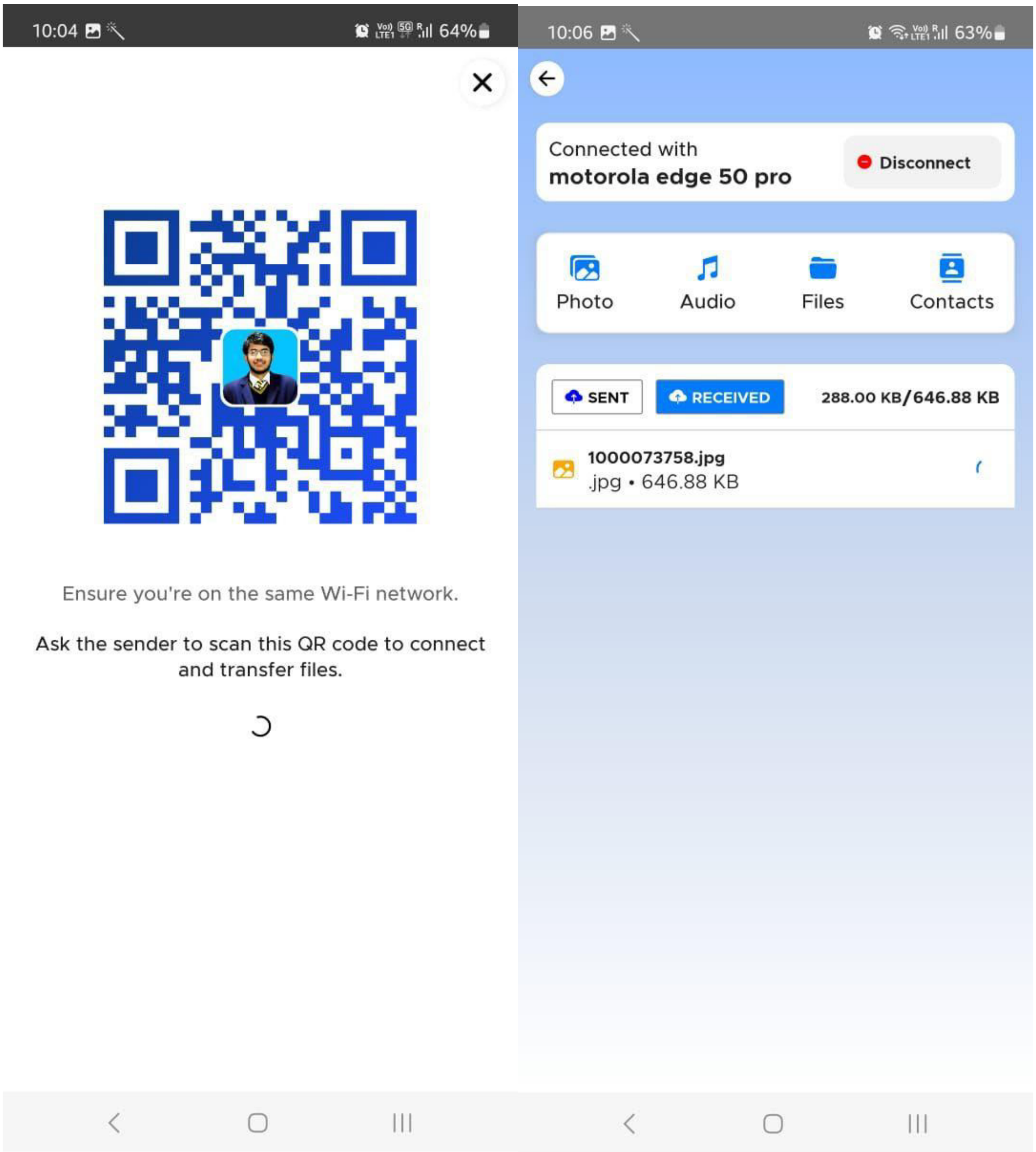
- **Versus Cloud Storage:** Cloud storage solutions must always store your files on third-party servers, meaning DropX has already greatly reduced the attack surface and removed a major privacy concern, as files are not hosted on servers.

V. IMPLEMENTATION



Ensure you're on the same Wi-Fi network.

Ask the sender to scan this QR code to connect and transfer files.



Ensure you're on the same Wi-Fi network.

Ask the sender to scan this QR code to connect and transfer files.

Connected with
motorola edge 50 pro

Disconnect

Photo Audio Files Contacts

SENT RECEIVED 288.00 KB/646.88 KB

1000073758.jpg
.jpg • 646.88 KB

VI. CONCLUSION

DropX is a very important step forward in the development of secure peer-to-peer file sharing applications on mobile devices. While many file sharing applications incorporate a variety of security features, DropX utilizes a simple and elegant user interface to provide both usability and security features. A simple and intuitive design has been the challenge in addressing an apparent tension between security and usability features.

In the case of DropX, the application architecture consisting of the TCP Provider, Navigation System, and File Transfer Protocol is solid enough to support file transfers in a secure and efficient manner. As discussed in the analysis of encryption features, DropX takes significant steps to ensure files are protected from transfer over the entire transfer path with TLS encryption, certificate-based authentication, and chunked transfers.

DropX has also made performance optimizations via chunked transfers and connection recovery mechanisms to successfully transfer and recover connections for large files while managing potentially problematic network environments. In particular, the user experience for DropX features a QR code connection workflow that aids adoption and alleviates the challenges faced by some non-technical users by providing a clear connection pathway.

With increasing media attention on digital privacy, applications like DropX that prioritize security with minimal usability compromises will only rise in importance. The retention of files between peer-to-peer and direct connections and subsequent use of encryption during data transfers provides users with a respect for privacy and the opportunity to avoid file sharing through cloud services. It enables network nodes to adaptively regulate their communication strategies according to dynamically changing network environment.

With future growth, DropX could grow its functionality to other platforms and additional use cases, further increasing its value as a secure file sharing system. However, currently DropX is already a meaningful contribution to the field of secure mobile file sharing, addressing a real need to satisfy growing concerns with privacy and sensitivity in the digital environment that many work in today.

REFERENCES

[1] http://paper.ijcsns.org/07_book/202011/20201110.pdf
[2] https://www.scirp.org/journal/paperinformation?paperid=472
[3] https://ijcaonline.org/archives/volume156/number12/irfan-2016-ijca-912582.pdf
[4] https://www.ijnrd.org/papers/IJNRD1803003.pdf
[5] https://www.cs.rice.edu/~dwallach/pub/tokyo-p2p2002.pdf
[6] https://www.ijrti.org/papers/IJRTI2106008.pdf
[7] https://dr.lib.iastate.edu/bitstreams/39cf6aa1-8a35-40e5-89c1-b3ad00835469/download
[8] https://www.sciencedirect.com/topics/computer-science/peer-to-peer-file-sharing
[9] https://ijritcc.org/index.php/ijritcc/article/view/7920
[10] https://www.hivenet.com/post/top-peer-to-peer-sharing-solutions-for-effortless-file-transfer
[11] https://ijcaonline.org/archives/volume156/number12/26765-2016912582/
[12] https://www.sciencedirect.com/topics/computer-science/transport-layer-security
[13] https://www.spiceworks.com/tech/networking/articles/what-is-peer-to-peer/
[14] https://www.caplinked.com/blog/is-peer-to-peer-file-sharing-safe/
[15] https://www.ijres.org/papers/Volume-10/Issue-5/Ser-15/1005173185.pdf
[16] https://www.geeksforgeeks.org/transport-layer-security-tls/
[17] https://www.ftc.gov/system/files/documents/plain-language/bus46-peer-peer-file-sharing-guide-business.pdf
[18] https://www.irjet.net/archives/V8/i5/IRJET-V8I5836.pdf
[19] https://datatracker.ietf.org/doc/rfc8922/
[20] https://github.com/Zohair-Khan/TLS-File-Transfer

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

☎ 9940 572 462 6381 907 438 ✉ ijircce@gmail.com

www.ijircce.com

Scan to save the contact details